Manuel d'utilisateur et de référence

Altova RaptorXML+XBRL Server 2018
Manuel d'utilisateur et de référence

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

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

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

Published: 2018

© 2018 Altova GmbH

Table des matières

1 À propos de RaptorXML+XBRL Server 3

1.1 Éditions et Interfaces ... 5
1.2 Exigences de système ... 7
1.3 Fonctions ... 8
1.4 Spécifications prises en charge ... 10

2 Configurer RaptorXML 14

2.1 Configuration sur Windows ... 15
2.1.1 Installation sur Windows ... 16
2.1.2 Obtention d'une licence sur Windows ... 18
2.2 Configuration sur Linux ... 22
2.2.1 Installation sur Linux ... 23
2.2.2 Obtention d'une licence sur Linux ... 27
2.2.3 Notes à propos de l'environnement ... 30
2.3 Configuration sur macOS ... 31
2.3.1 Installation sur macOS ... 32
2.3.2 Obtention d'une licence sur macOS ... 35
2.3.3 Notes à propos de l'environnement ... 38
2.4 Catalogues XML ... 39
2.4.1 Le fonctionnement des catalogues ... 40
2.4.2 Le mécanisme de Catalogue XML d'Altova ... 42
2.4.3 Variables pour les emplacements de Système ... 45
2.5 Ressources globales ... 47
2.6 Problèmes de sécurité ... 49

3 Interface de ligne de commande (CLI) 52

3.1 Commande de validation XML, DTD, XSD ... 56
3.1.1 valxml-withdtd (xml) ... 57
3.1.2 valxml-withxsd (xsi) ... 61
3.1.3 valdtd (dtd) ... 68
3.1.4 valxsd (xsd) ... 72
3.2 Commandes de vérification de la bonne formation ... 78

3.2.1 wfxml ... 79
3.2.2 wfdtd ... 83
3.2.3 wfany ... 87
3.3 Commandes de validation XBRL ... 88
3.3.1 valxbrl (xbrl) ... 89
3.3.2 valinlinexbrl (ixbrl) ... 103
3.3.3 valxbrltaxonomy (dts) ... 118
3.3.4 valtaxonomypackage (taxpkg) ... 127
3.3.5 valxbrlver ... 132
3.4 Commandes XSLT ... 137
3.4.1 xslt ... 138
3.4.2 valxslt ... 146
3.5 Commandes XQuery ... 152
3.5.1 xquery ... 153
3.5.2 xqueryupdate ... 161
3.5.3 valxquery ... 169
3.5.4 valxqueryupdate ... 175
3.6 Commandes JSON/Avro ... 181
3.6.1 avroextractschema ... 182
3.6.2 valavro (avro) ... 186
3.6.3 valavrojson (avrojson) ... 190
3.6.4 valavroschema (avroschema) ... 194
3.6.5 valjsonschema (jsonschema) ... 198
3.6.6 valjson (json) ... 202
3.6.7 wfjson ... 206
3.7 Commandes Signature XML ... 210
3.7.1 xmlsignature-sign ... 211
3.7.2 xmlsignature-verify ... 216
3.7.3 xmlsignature-update ... 219
3.7.4 xmlsignature-remove ... 222
3.8 Commande Valany ... 224
3.9 Commande Script ... 226
3.10 Commandes Aide et Licence ... 227
3.10.1 help ... 228
3.10.2 licenseserver ... 230
3.10.3 assignlicense ... 232
3.10.4 verifylicense ... 234
3.11 Commandes de localisation ... 236
3.11.1 exportresourcestrings ... 237
3.11.2 setdeflang ... 238

3.12 Options ... 239
3.12.1 Catalogues, Ressources globales, fichiers ZIP ... 240
3.12.2 Messages, Erreurs, Aide, Timeout, Version ... 241
3.12.3 Traitement ... 243
3.12.4 XBRL ... 245
3.12.5 XML ... 253
3.12.6 XSD ... 255
3.12.7 XQuery ... 257
3.12.8 XSLT ... 260
3.12.9 JSON/Avro ... 263

4 Interface HTTP 266

4.1 Configuration de serveur ... 268
4.1.1 Démarrer le serveur ... 269
4.1.2 Tester la connexion ... 271
4.1.3 Configurer le serveur ... 272
4.1.4 Paramètres HTTPS ... 278
4.1.5 Configurer le cryptage SSL ... 279
4.2 Requêtes Client ... 283
4.2.1 Initier les tâches avec POST ... 285
Exemple-1 (avec légendes): valider XML ... 289
Exemple-2: utiliser un catalogue pour trouver le schéma ... 291
Exemple-3 : utiliser des archives ZIP ... 292
Tester avec CURL ... 294
4.2.2 Réponse de serveur à une requête POST ... 297
4.2.3 Obtenir le document de résultat ... 300
4.2.4 Obtenir les documents Erreur/Message/Sortie ... 304
4.2.5 Libérer des ressources de serveur après le traitement ... 306

5 API Python et .NET 308

5.1 API Python ... 310
5.1.1 Versions d'API Python ... 312
5.1.2 RaptorXML+XBRL Server en tant que package Python ... 314
5.2 API .NET Framework ... 317

6 Interface Java 320

6.1 Projet d'exemple Java ... 322
6.2 Interfaces RaptorXML pour Java ... 325

6.2.1 RaptorXMLFactory ... 326
6.2.2 XMLValidator ... 331
6.2.3 XSLT ... 337
6.2.4 XQuery ... 343
6.2.5 XBRL ... 349
6.2.6 RaptorXMLException ... 358
6.3 Énumérations RaptorXML pour Java ... 359
6.3.1 RaptorXMLFactory ... 360
6.3.2 Validation XML ... 361
6.3.3 XSLT et XQuery ... 365
6.3.4 XBRL ... 367

7 Interfaces COM et .NET 370

7.1 À propos de l'interface COM ... 371
7.2 À propos de l'interface .NET ... 372
7.3 Langages de programmation ... 374
7.3.1 Exemple COM : VBScript ... 376
7.3.2 Exemple .NET : C# ... 379
7.3.3 Exemple .NET : Visual Basic .NET ... 382
7.4 Référence API ... 385
7.4.1 Interfaces ... 386
IServer ... 387
IXMLValidator ... 391
IXSLT ... 396
IXQuery ... 402
IXBRL ... 408
7.4.2 Énumérations ... 417
ENUMAssessmentMode ... 418
ENUMErrorFormat ... 419
ENUMLoadSchemalocation ... 420
ENUMQueryVersion ... 421
ENUMSchemaImports ... 422
ENUMSchemaMapping ... 424
ENUMTableOutputFormat ... 425
ENUMValidationType ... 426
ENUMWellformedCheckType ... 428
ENUMXBRLValidationType ... 429
ENUMIXBRLVersion ... 430
ENUMIXBRLUriStrategy ... 431

ENUMXMLValidationMode ... *432*
ENUMXQueryVersion ... *433*
ENUMXQueryUpdatedXML ... *434*
ENUMXSDVersion ... *435*
ENUMXSLTVersion ... *436*

8 Information supplémentaire 438

8.1 Codes de sortie ... 439
8.2 Indices d'emplacement de schéma ... 440
8.3 Paramètres de formule XBRL ... 441
8.3.1 Formats de paramètre de formule ... 442
8.3.2 Utiliser les paramètres de formule ... 444

9 Informations concernant le moteur XSLT et XQuery 450

9.1 XSLT 1.0 ... 451
9.2 XSLT 2.0 ... 452
9.3 XSLT 3.0 ... 455
9.4 XQuery 1.0 ... 456
9.5 XQuery 3.1 ... 460

10 Fonctions XSLT et XPath/XQuery 462

10.1 Fonctions d'extension Altova ... 464
10.1.1 Fonctions XSLT ... 466
10.1.2 Fonctions XPath/XQuery : Date et heure ... 470
10.1.3 Fonctions XPath/XQuery : Géolocalisation ... 485
10.1.4 Fonctions XPath/XQuery : Relatives aux images ... 494
10.1.5 Fonctions XPath/XQuery : Numérique ... 499
10.1.6 Fonctions XPath/XQuery : Séquence ... 502
10.1.7 Fonctions XPath/XQuery : Chaîne ... 510
10.1.8 Fonctions XPath/XQuery : Divers ... 516
10.1.9 Fonctions de graphique ... 517
Structure XML des données de graphique ... *522*
Exemple : fonctions de graphique ... *528*
10.1.10 Fonctions de code-barres ... 532
10.2 Fonctions d'extension diverses ... 535
10.2.1 Fonctions d'extension Java ... 536

Fichiers de classe définis par l'utilisateur ... *538*
Fichiers Jar définis par l'utilisateur ... *541*
Java: Constructeurs ... *542*
Java: Méthodes statiques et Champs statiques ... *543*
Java: Méthodes d'instance et Champs d'instance ... *544*
Types de données : XPath/XQuery en Java ... *545*
Types de données : Java en XPath/XQuery ... *547*
10.2.2 Fonctions d'extension .NET ... 548
.NET: Constructeurs ... *551*
.NET: Méthodes statiques et Champs statiques ... *552*
.NET: Méthodes d'instance et Champs d'instance ... *553*
Types de données : XPath/XQuery en .NET ... *554*
Types de données : .NET en XPath/XQuery ... *556*
10.2.3 Fonctions XBRL pour XSLT ... 557
10.2.4 Scripts MSXSL pour XSLT ... 558

11 Altova LicenseServer 562

11.1 Informations de réseau ... 564
11.2 Mise à jour du LicenseServer ... 566
11.3 Installation (Windows) ... 567
11.4 Installation (Linux) ... 569
11.5 Installation (macOS) ... 572
11.6 Altova ServiceController ... 574
11.7 Comment attribuer des licences ... 575
11.7.1 Lancer le LicenseServer ... 577
11.7.2 Ouvrir la page de Config de LicenseServer (Windows) ... 579
11.7.3 Ouvrir la page de Config de LicenseServer (Linux) ... 582
11.7.4 Ouvrir la page de Config de LicenseServer (macOS) ... 585
11.7.5 Charger des licences sur LicenseServer ... 588
11.7.6 Enregistrer un/des produits ... 592
Enregistrer les produits de bureau Altova ... *593*
Enregistrer FlowForce Server ... *594*
Enregistrer MapForce Server ... *599*
Enregistrer MobileTogether Server ... *601*
Enregistrer RaptorXML(+XBRL) Server ... *602*
Enregistrer StyleVision Server ... *604*
11.7.7 Attribuer des licences à des produits enregistrés ... 606
11.8 Référence de la page de Configuration ... 612
11.8.1 Pool de licences ... 613

11.8.2 Gestion des clients 620
11.8.3 Surveillance des clients 626
11.8.4 Paramètres 627
11.8.5 Messages, Log Out 635
11.9 Réinitialisation du mot de passe 636

Index **637**

Chapitre 1

À propos de RaptorXML+XBRL Server

1 À propos de RaptorXML+XBRL Server

Altova RaptorXML+XBRL Server (ci-après nommé RaptorXML) est le moteur de traitement XML et XBRL* haute vitesse de troisième génération Altova. Il a été optimisé pour s'adapter aux tout derniers standards et environnement de calcul parallèles. Conçu pour un usage multiplateforme, le moteur profite de l'omniprésence actuelle des ordinateurs multi-cœurs pour un traitement hyper-rapide des données XML et XBRL.

*** Note :** le traitement XBRL est uniquement disponible dans RaptorXML+XBRL Server, pas dans RaptorXML Server.

Éditions et systèmes d'exploitation

RaptorXML est disponible en deux éditions, chacune convenant à des exigences différentes. Ces éditions sont décrites dans la section Éditions et Interfaces. RaptorXML est disponible pour Windows, Linux et macOS. Pour plus de détails concernant la prise en charge du système, voir la section Exigences du système.

Fonctions et spécifications prises en charge

RaptorXML fournit une validation XML et XBRL, des transformations XSLT, et des exécutions XQuery, chacune comportant une palette étendue d'options intéressantes. Voir la section Fonctions pour une liste des fonctions disponibles. La section Spécifications prises en charge fournit une liste détaillée des spécifications auxquelles RaptorXML se conforme. Pour plus d'informations, consulter la page RaptorXML dans le site web Altova.

Certification XBRL

XMLSpy a été certifié pour XBRL par XBRL International. Pour plus d'informations à propos de la certification XBRL, voir XBRL Software Certification.

Cette documentation

Cette documentation est fournie avec l'application et est aussi disponible en ligne sur le site web Altova. Veuillez noter que le navigateur Chrome dispose d'une limite qui empêche les entrées dans le volet Table des matières (TOC) de s'agrandir lorsque la documentation est ouverte localement. La TOC dans Chrome fonctionne correctement, néanmoins, lorsque la documentation est ouverte depuis un serveur web.

Cette documentation est organisée dans les fonctions suivantes :

- À propos de RaptorXML (cette section)
- Configurer RaptorXML
- Interface de ligne de commande
- Interface HTTP
- API Python et .NET
- Interface Java
- Interface COM/.NET

- Information Moteur XSLT et XQuery
- Fonctions XSLT et XPath/XQuery
- Altova LicenseServer

Dernière mise à jour : 28 mai 2018

 © 2018 Altova GmbH

1.1 Éditions et Interfaces

RaptorXML est disponible dans les éditions suivantes :

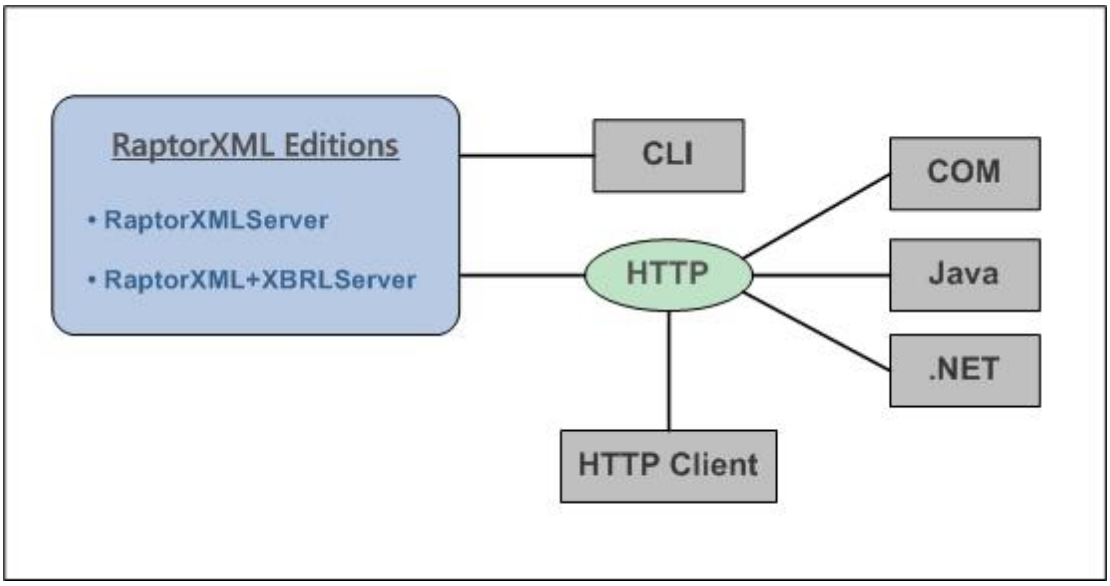
- *RaptorXML Server* est un moteur de traitement XML très rapide avec une prise en charge pour XML, Schéma XML, XSLT, XPath, XQuery, etc.
- *RaptorXML+XBRL Server* prend en charge toutes les fonctions de RaptorXML Server avec la capacité en outre de traiter et valider la famille XBRL des standards.

Interfaces

RaptorXML est accédé par les interfaces suivantes :

- Une interface par ligne de commande (CLI)
- Une interface COM sur les systèmes Windows
- Une interface .NET sur les systèmes Windows
- Une interface Java sur les systèmes Windows, Linux, et MacOS
- Une interface HTTP qui peut être accédée par un client HTTP
- Une interface Python avec laquelle les scripts Python peut accéder et traiter les parties de document par le biais des API Python de RaptorXML. Les scripts peuvent être soumis par CLI ou HTTP. De plus, la fonction RaptorXML est disponible en tant que pack Python ; cela permet d'utiliser la fonction RaptorXML à être utilisée dans le cadre du code Python ensemble avec des packs de tiers
- Un API Framework .NET

Le diagramme ci-dessous montre comment RaptorXML est accédé par le biais de ses interfaces.



Veuillez noter que les interfaces COM, Java et .NET utilisent un protocole HTTP pour se connecter à des éditions de serveur. Les scripts Python peuvent être soumis aux éditions de serveur par le biais de la ligne de commande et les interfaces HTTP.

Interface de ligne de commande (CLI)
Permet l'utilisation de la ligne de commande pour la validation XML (et autre document), la transformation XSLT et l'exécution XQuery. Voir la section Ligne de commande pour l'information d'utilisation.

Interface HTTP
Toutes les fonctions des éditions serveur peuvent être accédées par le biais d'une interface HTTP. Les requêtes client sont effectuées dans un format JSON. Chaque requête est assignée à un répertoire de tâche sur le serveur, dans lequel les fichiers de sortie sont enregistrés, les réponses Server au client comprennent toutes les informations pertinentes à propos du la tâche. Voir la section Interface HTTP.

API Python
Ensemble avec une commande CLI ou la requête HTTP, un script Python peut être soumis qui accède aux documents spécifiés dans la commande ou la requête. L'accès au document est fourni par les API Python pour XML, XSD et XBRL. Voir la section API Python pour une description des utilisations et des API Python variés. La fonction RaptorXML est aussi disponible en tant que pack Python ; cela permet d'utiliser la fonction RaptorXML dans le code Python ensemble avec d'autres pack de tiers.

API .NET Framework
L'API .NET Framework vous permet d'intégrer RaptorXML+XBRL Server dans une application qui prend en charge le .NET Framework. Voir la section API .NET Framework pour plus d'informations à propos de l'API.

RaptorXML peut être utilisée par le biais de l'interface COM, et peut donc être utilisée par les applications et les langues de script qui prennent en charge COM. La prise en charge de l'interface COM est mis en place pour les interfaces Raw et Dispatch. Les données d'entrée peuvent être fournies en tant que fichiers ou en tant que strings de texte dans les scripts et dans les données d'application.

Interface Java
La fonction RaptorXML est disponible en tant que classes Java qui peuvent être utilisée dans des programmes Java. Par exemple, il y a des classes Java qui fournissent une validation XML, une transformation XSLT et des fonctions d'exécution XQuery.

Interface .NET
Un fichier DLL est construit en tant que wrapper autour RaptorXML et permet aux utilisateurs .NET de se connecter aux fonctions RaptorXML. RaptorXML fournit l'assembly interop primaire signé par Altova. Les données d'entrée peuvent être fournies en tant que fichiers ou en tant que strings de texte dans les scripts et dans les données d'application.

 © 2018 Altova GmbH

1.2 Exigences de système

RaptorXML+XBRL Server est pris en charge sur les systèmes d'exploitation suivants :

- *Windows*

  Windows 7 SP1 avec mise à jour de la plateforme, Windows 8, Windows 10

- *Windows Server*

  Windows Server 2008 R2 SP1 avec mise à jour de la plateforme ou plus récent

- *Linux*

  - CentOS 6 ou plus récent
  - RedHat 6 ou plus récent
  - Debian 7 ou plus récent
  - Ubuntu 12.04 ou plus récent

  Les bibliothèques suivantes sont exigées en tant que condition préalable pour installer et exécuter l'application. Si les paquets ci-dessous ne sont pas encore disponibles sur votre appareil Linux, vous pourrez exécuter la commande command `yum` (ou `apt-get` le cas échéant) pour les installer.

Exigé par	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 ou plus récent

RaptorXML est disponible aussi bien pour les appareils 32-bit et 64-bit. En particulier, il s'agit de cœurs basés sur des ensembles d'instruction x86 et amd64 (x86-64) : Intel Core i5, i7, XEON E5. Pour utiliser RaptorXML par le biais d'une interface COM, les utilisateurs doivent avoir des privilèges pour utiliser l'interface COM, c'est à dire, d'enregistrer l'application et d'exécuter les applications pertinentes et/ou les scripts.

1.3 Fonctions

RaptorXML propose les fonctions listées ci-dessous. La plupart des fonctions peut être utilisée par le biais de la commande de ligne et l'interface COM. L'une des principales différences est que l'utilisation de l'interface COM sur Windows permet de construire des documents à partir des strings de texte par le biais de l'application ou de code de script (au lieu de référencer des fichiers XML, XBRL, DTD, Schéma XML, XSLT ou XQuery).

Validation XML et XBRL

- Valide le document XML ou XBRL fourni par rapport à des DTD internes ou externes ou des Schémas XML
- Vérifie la bonne formation de XML, DTD, schéma XML, XSLT, et les documents XQuery
- Valide des taxonomies XBRL, et des documents XBRL par rapport à des taxonomies XBRL
- Exécution de Formules XBRL et des Assertions de Validation
- Rendu de tables XBRL
- Prise en charge des spécifications XBRL 2.1, Dimensions 1.0, et Formula 1.0 et le Table Linkbase 1.0
- Prise en charge de Inline XBRL
- Prise en charge des packs de taxonomie XBRL

Transformations XSLT

- Transforme XML à l'aide du document XSLT 1.0, 2.0 ou 3.0 fourni
- Les documents XML et XSLT peuvent être fournis en tant qu'un fichier (par le biais d'une URL) ou, dans le cas de l'utilisation COM, en tant que string de texte
- La sortie est retournée en tant que fichier (sous un emplacement nommé) ou, dans le cas de l'utilisation COM, en tant que string de texte
- Les paramètres XSLT peuvent être fournis par le biais de la ligne de commande et par le biais de l'interface COM
- Les fonctions d'extension Altova, ainsi que des fonction d'extension XBRL, Java et .NET, permettent un traitement spécialisé. Cela permet, par exemple la création de fonctions comme des graphiques et des codes-barres dans les documents de sortie

Exécution XQuery

- Exécute des documents XQuery 1.0 et 3.0
- Les documents XQuery et XML peuvent être fournis en tant qu'un fichier (par le biais d'une URL) ou, dans le cas d'une utilisation COM, en tant que string de texte
- La sortie est retournée en tant que fichier (sous un emplacement nommé) ou, dans le cas de l'utilisation COM, en tant que string texte
- Des variables XQuery externes peuvent être fournies par le biais de la ligne de commande et par le biais de l'interface COM
- Les options de sérialisation comprennent : encodage de sortie, méthode de sortie (c'est à dire, que la sortie soit XML, XHTML, HTML ou texte), en omettant la déclaration XML et le retrait

 © 2018 Altova GmbH

Validation/Conversion JSON et Avro

- Validation de schéma JSON et de documents de schéma Avro
- Validation d'instances JSON par rapport aux schémas JSON et des schémas Avro
- Validation de binaires Avro
- Conversion de binaires Avro en schéma Avro et de données Avro en format JSON
- Conversion de données JSON Avro en binaire Avro

Fonctions d'hyper-performance

- Optimisations de code de très haute performance
  - Implémentations de sets d'instructions natives
  - Version 32-bit et 64-bit
- Empreinte mémoire très basse
  - Représentation in-memory extrêmement compacte de l'ensemble d'information XML
  - Validation d'instance de streaming
- Capacités multiplateforme
- Code hautement évolutif pour l'informatique multi-CPU/multi-cœur/parallèle
- Chargement, validation et traitement parallèle sur conception

Fonctions de développeur

- Capacités de rapport d'erreur supérieur
- Mode serveur Windows et mode daemon Unix (par le biais des options de ligne de commande)
- Interpréteur Python 3.x pour scripting est inclus
- Fonction RaptorXML dans un pack Python permet l'importation de la fonction en tant que bibliothèque Python
- API Framework .NET permet l'accès au modèle de données XML et XBRL sous-jacent
- Plateforme Windows sur API COM
- API Java partout
- Fonctions d'extension XPath Java, .NET, XBRL, etc.
- Sérialisation de streaming
- Serveur HTTP intégré avec l'API de validation REST

Pour plus d'informations, voir la section Spécifications prises en charge et le site web Altova.

1.4 Spécifications prises en charge

RaptorXML prend en charge les spécifications suivantes.

Recommandations W3C

Site web : World Wide Web Consortium (W3C)

- Extensible Markup Language (XML) 1.0 (cinquième édition)
- Extensible Markup Language (XML) 1.1 (deuxième édition)
- Espaces de noms dans XML 1.0 (troisième édition)
- Espaces de noms dans XML 1.1 (deuxième édition)
- Ensemble d'information XML (deuxième édition)
- Base XML (deuxième édition)
- Inclusions XML (XInclude) Version 1.0 (deuxième édition)
- XML Linking Language (XLink) Version 1.0
- Schéma XML Partie 1: Structures Deuxième édition
- Schéma XML Partie 2: Datatypes Deuxième édition
- Langage de définition de Schéma XML W3C (XSD) 1.1 Partie 1: Structures
- Langage de définition de Schéma XML W3C (XSD) 1.1 Partie 2: Types de données
- XPointer Framework
- XPointer xmlns() Scheme
- XPointer element() Scheme
- Langage de chemin XML (XPath) Version 1.0
- Transformations XSL (XSLT) Version 1.0
- Langage de chemin XML (XPath) 2.0 (deuxième édition)
- Transformations XSL (XSLT) Version 2.0
- XQuery 1.0: un langage de requête XML (deuxième édition)
- Fonctions XQuery 1.0 et XPath 2.0 et Opérateurs (deuxième édition)
- Sérialisation XSLT 2.0 et XQuery 1.0 (deuxième édition)
- Langage de chemin XML (XPath) 3.0
- Langage de chemin XML (XPath) 3.1
- XQuery 3.0: un langage de requête XML
- XQuery Update Facility 1.0
- Fonctions XPath et XQuery et Opérateurs 3.0
- Sérialisation XSLT et XQuery 3.0

W3C Ébauche de travail & Recommandations des candidats

Site web : World Wide Web Consortium (W3C)

- Transformations XSL (XSLT) Version 3.0 (sous-ensemble)
- XQuery 3.1 : un langage de requête XML
- Fonctions XPath et XQuery et Opérateurs 3.1
- XQuery Update Facility 3.0
- Sérialisation XSLT et XQuery 3.1

Standards OASIS

Site web : Standards OASIS

 © 2018 Altova GmbH

- Catalogues XML V 1.1 - Standard OASIS V1.1

Standards JSON/Avro

Sites web : Schéma JSON et Apache Avro

- JSON Schema Draft 4
- Schéma Apache Avro

Recommandations XBRL

Site web : Extensible Business Reporting Language (XBRL)

- XBRL 2.1
- Dimensions 1.0
- Extensible Enumerations 1.0 (et la recommandation proposée obsolète 1.1)
- Spécifications de Formula 1.0
  - Filtres Aspect Cover
  - Sévérité Assertion 1.0
  - Filtres Booléenne
  - Filtres Concept
  - Filtres Relation Concept
  - Consistency Assertions
  - Custom Function Implementation
  - Filtres Dimension
  - Filtres Entité
  - Existence Assertions
  - Formula
  - Registre Function
  - Filtres Général
  - Generic Messages
  - Filtres Implicite
  - Filtres Correspondance
  - Filtres Période
  - Filtres Relative
  - Filtres Scénario de Segment
  - Filtres Tuple
  - Filtres Unité
  - Validation
  - Messages de Validation
  - Assertions de Valeur
  - Filtres Valeur
  - Variables
- Base de lien de Table 1.0
- Registre de fonction 1.0
- Liens génériques 1.0
  - Références génériques
  - Libellés génériques
- Registres d'unités 1.0
- Inline XBRL 1.0 et 1.1

- Packs de Taxonomie 1.0
- Version XBRL 1.0

© 2018 Altova GmbH

Chapitre 2

Configurer RaptorXML

2 Configurer RaptorXML

Cette section décrit les procédures pour la configuration de RaptorXML+XBRL Server. Elle décrit les éléments suivants :

- Installation et mise sous licence de RaptorXML sur les systèmes Windows, Linux et macOS.
- Comment utiliser les Catalogues XML.
- Comment travailler avec les ressources globales Altova.
- Les problèmes de sécurité liés à RaptorXML.

RaptorXML a des options spéciales qui prennent en compte les Catalogues XML et les ressources globales Altova, tous deux augmentent la portabilité et la modularité. Vous pouvez donc mettre à profit l'utilisation de ces fonctions dans votre environnement.

Note : Vous trouverez une description des problèmes de sécurité et de la manière dont configurer les solutions de sécurité dans la section problèmes de sécurité.

 © 2018 Altova GmbH

2.1 Configuration sur Windows

Cette section décrit l'installation et l'obtention de la licence de RaptorXML+XBRL Server sur les systèmes Windows.

Installation sur Windows

- Exigences du système
- Installer RaptorXML+XBRL Server
- Altova LicenseServer
- Versions du LicenseServer
- Licence d'essai
- Emplacement du dossier d'application

Obtention de la licence sur Windows

- Lancer ServiceController
- Lancer LicenseServer
- Lancer RaptorXML+XBRL Server
- Enregistrer RaptorXML+XBRL Server
- Attribuer une licence

Installation sur Windows

RaptorXML+XBRL Server est disponible pour une installation sur les systèmes Windows. La procédure d'installation et de configuration est décrite ci-dessous.

- Exigences du système
  - *Windows*

    Windows 7 SP1 avec mise à jour de la plateforme, Windows 8, Windows 10

  - *Windows Server*

    Windows Server 2008 R2 SP1 avec mise à jour de la plateforme ou plus récent

- InstallerRaptorXML+XBRL Server

  RaptorXML+XBRL Server peut être installé sur les systèmes Windows comme suit :

  - En tant que produit de serveur autonome séparé appelé RaptorXML+XBRL Server. Pour installer RaptorXML+XBRL Server, télécharger et exécuter le programme d'installation RaptorXML+XBRL Server. Suivre les instructions sur l'écran.
  - En tant que partie intégrante du paquet d'installation FlowForce Server. Pour installer RaptorXML+XBRL Server en tant que partie intégrante du paquet FlowForce Server, télécharger et exécuter le programme d'installation FlowForce Server. Suivre les instructions sur l'écran et veiller à cocher l'option de l'installation de RaptorXML +XBRL Server.

  Les programmes d'installation de RaptorXML+XBRL Server et de FlowForce Server sont disponibles sur le Centre de téléchargement Altova (http://www.altova.com/download.html).

  Une fois l'installation effectuée, le programme d'exécution de RaptorXML+XBRL Server sera situé par défaut sous :

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

  Tous les enregistrements nécessaires pour utiliser RaptorXML+XBRL Server via une interface COM, en tant qu'interface Java et dans l'environnement .NET seront effectués par le programme d'installation. Cela inclut l'enregistrement du programme d'exécution de RaptorXML+XBRL Server en tant qu'objet de serveur COM, en installant `RaptorXMLLib.dll` (pour l'utilisation de l'interface Java) dans le répertoire `WINDIR\system32\` et l'ajout du fichier `Altova.RaptorXML.dll` à la bibliothèque de référence .NET.

- Cliquer pour agrandir/réduire
  - Pour que RaptorXML+XBRL Server puisse fonctionner, vous devez obtenir une licence pour votre réseau par le biais d'Altova LicenseServer.
  - Lors de l'installation de RaptorXML+XBRL Server ou de FlowForce Server sur des systèmes Windows, une option est disponible vous permettant de télécharger et d'installer Altova LicenseServer avec RaptorXML+XBRL Server ou le FlowForce Server.
  - Si Altova LicenseServer est déjà installé sur votre réseau, il ne sera pas nécessaire

 © 2018 Altova GmbH

d'en installer un autre, sauf si une version plus récente d'Altova LicenseServer est nécessaire. (*Voir point suivant, Versions de LicenseServer*.)
- Au cours du processus d'installation de RaptorXML+XBRL Server ou de FlowForce Server, cocher ou décocher l'option d'installation d'Altova LicenseServer selon le cas.

Voir la section Obtention de la licence sur Windows pour plus d'informations à propos de l'enregistrement de la licence RaptorXML+XBRL Server avec l'Altova LicenseServer.

Versions du LicenseServer

- Une licence doit être délivrée pour les produits de serveur Altova, soit avec la version LicenseServer convenant à la version du RaptorXML+XBRL Server installée, soit avec une version ultérieure de LicenseServer.
- La version LicenseServer qui convient pour une version particulière de RaptorXML +XBRL Server est affichée pendant l'installation de RaptorXML+XBRL Server. Vous pouvez installer cette version de LicenseServer avec RaptorXML+XBRL Server, ou bien vous pouvez installer LicenseServer séparément.
- Avant d'installer une version plus récente de LicenseServer, toute version plus ancienne doit être désinstallée. Le programme d'installation LicenseServer effectuera cette procédure automatiquement si elle détecte une version plus ancienne.
- Les versions LicenseServer sont rétrocompatibles. Elles fonctionneront avec des versions plus anciennes de RaptorXML+XBRL Server.
- Si vous installez une nouvelle version de RaptorXML+XBRL Server et si votre version installée de LicenseServer est plus ancienne que le LicenseServer approprié, installer la dernière version disponible du site web Altova.
- Au moment de la désinstallation de LicenseServer, toutes les informations d'enregistrement et d'obtention de la licence contenues dans la version plus ancienne de LicenseServer seront enregistrées dans une base de données sur la machine de votre serveur. Ces données seront importées automatiquement dans la version la plus récente lorsque la version la plus récente est installée.
- Le numéro de version du LicenseServer actuellement installé est indiqué au bas de la page de configuration de LicenseServer (tous les onglets).

  Version actuelle : 2.7

Licence d'essai

Au cours du processus d'installation, vous serez invité à demander une licence d'essai de 30 jours pour RaptorXML+XBRL Server. Une fois la demande renvoyée, la licence d'essai vous sera envoyée sur l'adresse électronique que vous aurez indiquée pendant l'inscription.

Emplacement du dossier d'application

L'application sera installée dans le dossier suivant :

Windows 7, 8, 10	`C:\Program Files\Altova\`
Version 32 bit sur SE 64-bit	`C:\Program Files (x86)\Altova\`

Obtention d'une licence sur Windows

Une licence pour RaptorXML+XBRL Server doit être délivrée avec un Altova LicenseServer pour qu'il puisse fonctionner. L'obtention de la licence est un processus en deux étapes :

1. **Enregistrer RaptorXML+XBRL Server** avec LicenseServer. L'enregistrement s'effectue depuis RaptorXML+XBRL Server.
2. **Attribuer une licence** à RaptorXML+XBRL Server. L'attribution de licence est effectuée depuis LicenseServer.

Les étapes à suivre sont indiquées ci-dessous.

▼ Lancer ServiceController

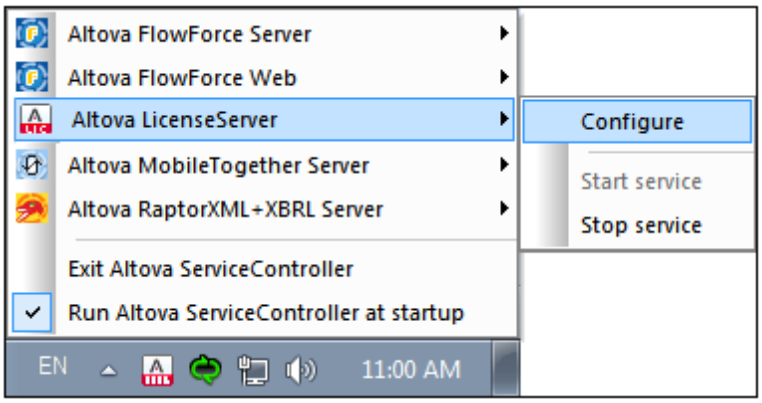
Altova ServiceController est démarré pour lancer Altova LicenseServer et Altova RaptorXML +XBRL Server.

Altova ServiceController (ServiceController en abrégé) est une application permettant de lancer, d'arrêter et de configurer confortablement les services Altova **sur les systèmes Windows**.

ServiceController est installé avec l'Altova LicenseServer et avec les *produits de serveur Altova installés en tant que services* (FlowForce Server, RaptorXML(+XBRL) Server, and Mobile Together Server). Il peut être démarré en cliquant sur **Start | Altova LicenseServer | Altova ServiceController**. (Cette commande est également disponible dans les dossiers de menu **Start** des *produits de serveur Altova qui sont installés en tant que services* (FlowForce Server, RaptorXML(+XBRL) Server, and Mobile Together Server).) Une fois que ServiceController a été démarré, il peut être accédé depuis la zone de notification (*capture d'écran ci-dessous*).

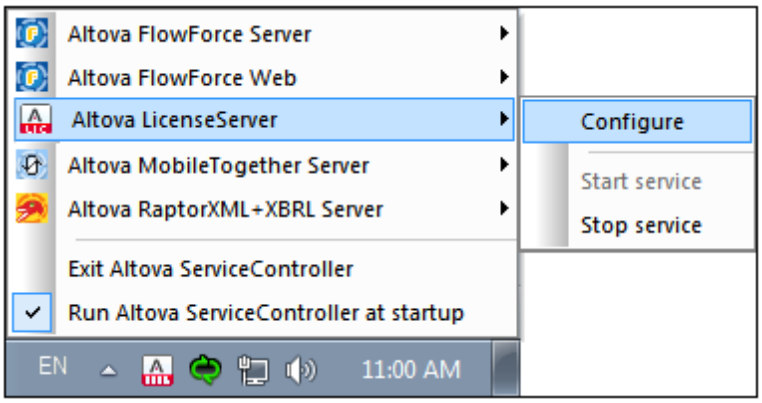
Pour spécifier que ServiceController démarre automatiquement lors de la connexion au système, cliquer sur l'icône **ServiceController** dans la zone de notification pour afficher le menu **ServiceController** (*capture d'écran ci-dessous*) et basculer sur la commande **Run Altova ServiceController at Startup**. (Cette commande est activée par défaut.) Pour quitter ServiceController, cliquer sur l'icône **ServiceController** dans la zone de notification et, dans le menu qui apparaît, (*voir capture d'écran ci-dessous*), cliquer sur **Exit Altova ServiceController**.

 © 2018 Altova GmbH



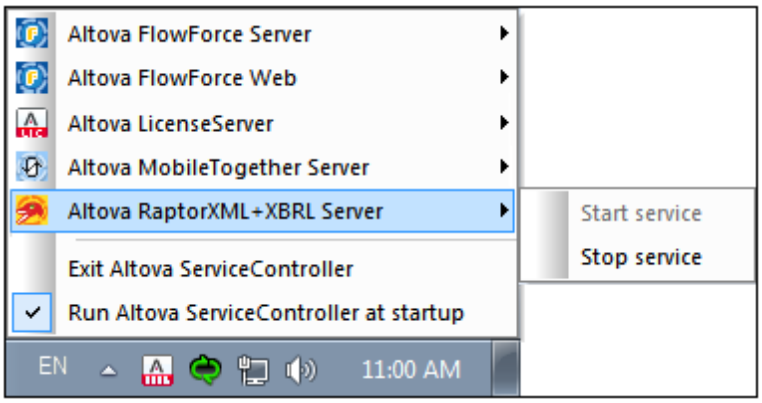
- Lancer LicenseServer

  Pour démarrer le LicenseServer, cliquer sur l'icône **ServiceController** dans la zone de notification, survoler au-dessus d'**Altova LicenseServer** dans le menu qui s'ouvre (*voir capture d'écran ci-dessous*) et choisir **Start Service** depuis le sous-menu LicenseServer. Si LicenseServer est déjà en cours d'exécution, l'option *Start Service* sera désactivée.



- Lancer RaptorXML+XBRL Server

  Pour lancer RaptorXML+XBRL Server, cliquer sur l'icône **ServiceController** dans la zone de notification, planer au-dessus d'**Altova RaptorXML+XBRL Server** dans le menu qui s'ouvre (*voir capture d'écran ci-dessous*) et choisir **Start Service** depuis le sous-menu RaptorXML +XBRL Server. Si RaptorXML+XBRL Server est déjà en cours d'exécution, l'option *Start Service* sera désactivée.



Note : Si une licence a été délivrée pour RaptorXML+XBRL Server pour n'effectuer que des exécutions à traitement unique (généralement parce que votre machine a plusieurs cœurs, mais votre licence n'a qu'un seul cœur), vous ne pourrez utiliser qu'une seule instance de RaptorXML+XBRL Server à la fois : soit en tant que service, soit depuis la ligne de commande. Cela est dû au fait que la licence à cœur unique est attribuée automatiquement à la première instance démarrée et actuellement en cours ; une seconde instance ne peut pas être démarrée avant que la première instance ne s'arrête.

- Si vous souhaitez utiliser RaptorXML+XBRL Server depuis la ligne de commande mais que le service est en cours d'exécution, vous devrez interrompre le service avant d'utiliser la ligne de commande.
- Si vous souhaitez lancer RaptorXML+XBRL Server en tant que service, veuillez vous assurer qu'aucune action de ligne de commande n'est actuellement exécutée. Dans le cas contraire, vous ne pourrez pas lancer le service.

Enregistrer RaptorXML+XBRL Server

Pour enregistrer RaptorXML+XBRL Server depuis l'interface de ligne de commande, utiliser la commande `licenseserver` :

```
RaptorXMLXBRL licenseserver [options] ServerName-Or-IP-Address
```

Par exemple, si `localhost` est le nom du serveur sur lequel LicenseServer est installé :

```
RaptorXMLXBRL licenseserver localhost
```

Si RaptorXML+XBRL Server a été installé dans le cadre d'une installation FlowForce Server, l'enregistrement du FlowForce Server avec LicenseServer entraînera automatiquement aussi l'enregistrement de RaptorXML+XBRL Server. En principe : (i) Lancer Altova FlowForce Web en tant que service via ServiceController (*voir point précédent*) ; (ii) Saisir votre mot de passe pour accéder à la page de Configuration ; (iii) Choisir le nom ou l'adresse du LicenseServer et cliquer sur **Register with LicenseServer**. Pour plus d'informations, voir Enregistrer FlowForce Server.

Une fois l'enregistrement effectué avec succès, rendez-vous à l'onglet Server Management de la page de configuration de LicenseServer pour attribuer une licence à RaptorXML+XBRL Server.

 © 2018 Altova GmbH

Attribuer une licence

Une fois l'enregistrement de RaptorXML+XBRL Server effectué avec succès, il sera compris dans l'onglet Server Management de la page de configuration du LicenseServer. Rendez-vous dans cette section et attribuez une licence à RaptorXML+XBRL Server.

Note sur les cœurs et les licences
L'obtention de la licence des produits de serveur Altova, est basée sur le nombre de cœurs de processeurs disponibles sur la machine de produit. Par exemple, un processeur double cœur a deux cœurs, un processeur quadricœur a quatre cœurs, un processeur hexacœurs a six cœurs, etc. Le nombre de cœurs pour lesquels une licence a été délivrée pour un produit doit être supérieur ou égal au nombre de cœurs disponibles sur cette machine de serveur, que ce serveur soit une machine physique ou virtuelle. Par exemple, si un serveur a huit cœurs (un processeur octacœur), vous devrez acheter au moins une licence octacœur. Vous pouvez aussi additionner les licences pour obtenir le nombre souhaité de cœurs. Ainsi, deux licences de quadricœurs peuvent être utilisées pour un serveur octacœur au lieu d'acheter une licence octacœur.

Si vous utilisez un serveur d'ordinateur avec un grand nombre de cœurs CPU, mais ne disposez que d'un faible volume à traiter, vous pouvez aussi créer une machine virtuelle qui disposera d'un plus petit nombre de cœurs et acheter une licence pour ce nombre de cœurs. Il va de soi que la vitesse de traitement d'un tel déploiement sera moins rapide que si tous les cœurs disponibles sur le serveur étaient utilisés.

Note: Chaque licence de produit de serveur Altova peut être utilisée pour une seule machine client à la fois (l'appareil sur lequel le produit de serveur Altova est installé), même si la licence a une capacité de licence n'est pas utilisée. Par exemple, si une licence de 10-cœurs est utilisée pour une machine client qui détient 6 cœurs CPU, les 4 cœurs restants de la capacité de licence ne pourront pas être utilisés simultanément pour une autre machine client.

Note : En raison des fonctions de ses services, MobileTogether Server Advanced Edition ne sera exécuté que sur des machines comportant **deux ou plus de cœurs**.

Les licences MobileTogether Server
Les licences de MobileTogether Server sont basées sur le nombre des cœurs CPU dans l'appareil MobileTogether Server. Les licences cœur permettent à un nombre illimité d'appareils MobileTogether Client de se connecter au serveur. Néanmoins, si vous cochez la case *Limiter à une seule exécution thread*, un seul appareil mobile pourra se connecter au MobileTogether Server à tout moment. Cela est utile pour l'évaluation et les tests à petite échelle. Veuillez noter que, dans ce cas, un second appareil se connectera à MobileTogether Server, et prendra la licence. Le premier appareil ne pourra plus se connecter et recevra un message d'erreur.

2.2 Configuration sur Linux

Cette section décrit l'installation et l'obtention de la licence de RaptorXML+XBRL Server sur des systèmes Linux (Debian, Ubuntu, CentOS, RedHat).

Installation sur Linux

- Exigences du système
- Note concernant l'utilisateur root
- Désinstaller les anciennes versions des produits de serveurs Altova
- Télécharger le paquet Linux
- Installer RaptorXML+XBRL Server
- Altova LicenseServer
- Versions de LicenseServer
- Licence d'essai

Obtention de la licence sur Linux

- Note concernant l'utilisateur root
- Lancer LicenseServer
- Lancer RaptorXML+XBRL Server
- Enregistrer RaptorXML+XBRL Server
- Attribuer une licence

 © 2018 Altova GmbH

Installation sur Linux

RaptorXML+XBRL Server est disponible pour une installation sur les systèmes Linux. La procédure d'installation et de configuration est décrite ci-dessous.

- Exigences du système
  - *Linux*
    - CentOS 6 ou plus récent
    - RedHat 6 ou plus récent
    - Debian 7 ou plus récent
    - Ubuntu 12.04 ou plus récent

    Les bibliothèques suivantes sont exigées en tant que condition préalable pour installer et exécuter l'application. Si les paquets ci-dessous ne sont pas encore disponibles sur votre appareil Linux, vous pourrez exécuter la commande command `yum` (ou `apt-get` le cas échéant) pour les installer.

Exigé par	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

- Intégration du serveur FlowForce

  Si vous installez RaptorXML+XBRL Server avec FlowForce Server, il est recommandé d'installer FlowForce Server d'abord. Sinon, après avoir installé RaptorXML+XBRL Server et FlowForce Server, exécuter la commande suivante :

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

  Cette commande copie le fichier **.tool** depuis le répertoire **/etc** de RaptorXML+XBRL Server vers le répertoire **/tools** de FlowForce Server. Le fichier **.tool** est nécessité par FlowForce Server. Il contient le chemin vers le programme exécutable RaptorXML+XBRL Server. Si vous installez FlowForce Server avant d'installer RaptorXML+XBRL Server, vous n'aurez pas besoin d'exécuter cette commande.

- Note à propos de l'utilisateur root

  Vous devez disposer de privilèges administrateur (root) pour pouvoir installer RaptorXML +XBRL Server. L'installation doit donc être effectuée en tant que l'utilisateur root. Si vous êtes connecté sous `root`, vous pouvez retirer le mot-clé `sudo` des commandes recensées ci-dessous.

Désinstaller les anciennes versions des produits de serveurs Altova

Suivez les étapes suivantes pour procéder à une désinstallation. Dans l'interface de ligne de commande Linux (CLI), vous pouvez vérifier quels sont les produits de serveur Altova installés avec les commandes suivantes :

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

Si RaptorXML+XBRL Server n'est pas installé, poursuivez l'installation comme documenté ci-dessous dans *Installer RaptorXML+XBRL Server*.

Si RaptorXML+XBRL Server est installé et vous souhaitez installer une version plus récente de RaptorXML+XBRL Server, désinstaller l'ancienne version avec la commande ::

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

Si vous devez désinstaller une ancienne version de Altova LicenseServer, procédez comme suit :

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

Télécharger le paquet Linux

Les paquets d'installation RaptorXML+XBRL Server pour les systèmes Linux suivants sont disponibles sur le Site web Altova .

Distribution	Extension du paquet
Debian 7 et plus récent	`.deb`
Ubuntu12.04 et plus récent	`.deb`
CentOS 6 et plus récent	`.rpm`
RedHat 6 et plus récent	`.rpm`

Après le téléchargement du paquet Linux, le copier dans un répertoire du système Linux. Puisqu'il vous faudra Altova LicenseServer pour pouvoir exécuter RaptorXML+XBRL Server, vous pouvez éventuellement télécharger LicenseServer depuis le site web Altova au moment de télécharger RaptorXML+XBRL Server, au lieu de le télécharger plus tard.

Installer RaptorXML+XBRL Server

Dans une fenêtre de terminal, passez au répertoire dans lequel vous avez copié le paquet Linux. Par exemple, si vous l'avez copié dans un répertoire d'utilisateur appelé `MyAltova` (situé par exemple dans le répertoire `/home/User`), passez à ce répertoire comme suit :

```
cd /home/User/MyAltova
```

Installer RaptorXML+XBRL Server avec la commande suivante :

```
[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
```

 © 2018 Altova GmbH

```
[RedHat]:   sudo rpm -ivh raptorxmlxbrlserver-2018-1.x86_64.rpm
```

Le paquet RaptorXML+XBRL Server sera installé dans le dossier :
/opt/Altova/RaptorXMLXBRLServer2018

Altova LicenseServer

Pour faire fonctionner tout produit Altova Server, y compris RaptorXML+XBRL Server, ce produit de serveur doit posséder une licence via Altova LicenseServer sur votre réseau.

Sur des systèmes Linux, Altova LicenseServer devra être installé séparément. Télécharger LicenseServer depuis le site web Altova et copier le paquet dans un répertoire sur le système Linux. L'installer de la même manière que RaptorXML+XBRL Server (*voir étape précédente*).

```
[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
```

Le paquet LicenseServer sera installé dans :
/opt/Altova/LicenseServer

Pour plus d'informations concernant l'enregistrement de RaptorXML+XBRL Server avec Altova LicenseServer et l'obtention d'une licence, voir la section, Obtention de la licence sur Linux. Voir aussi la documentation LicenseServer pour plus de détails

Versions de LicenseServer

- Une licence doit être délivrée pour les produits de serveur Altova, soit avec la version LicenseServer convenant à la version du RaptorXML+XBRL Server installée, soit avec une version ultérieure de LicenseServer.
- La version LicenseServer qui convient pour une version particulière de RaptorXML +XBRL Server est affichée pendant l'installation de RaptorXML+XBRL Server. Vous pouvez installer cette version de LicenseServer avec RaptorXML+XBRL Server, ou bien vous pouvez installer LicenseServer séparément.
- Avant d'installer une version plus récente de LicenseServer, toute version plus ancienne doit être désinstallée. Le programme d'installation LicenseServer effectuera cette procédure automatiquement si elle détecte une version plus ancienne.
- Les versions LicenseServer sont rétrocompatibles. Elles fonctionneront avec des versions plus anciennes de RaptorXML+XBRL Server.
- Si vous installez une nouvelle version de RaptorXML+XBRL Server et si votre version installée de LicenseServer est plus ancienne que le LicenseServer approprié, installer la dernière version disponible du site web Altova.
- Au moment de la désinstallation de LicenseServer, toutes les informations d'enregistrement et d'obtention de la licence contenues dans la version plus ancienne de LicenseServer seront enregistrées dans une base de données sur la machine de votre serveur. Ces données seront importées automatiquement dans la version la plus récente lorsque la version la plus récente est installée.
- Le numéro de version du LicenseServer actuellement installé est indiqué au bas de la page de configuration de LicenseServer (tous les onglets).

Version actuelle : 2.7

- Licence d'essai

  Pendant le processus d'installation, vous serez invité à demander une licence d'essai de 30 jours pour RaptorXML+XBRL Server. Une fois la demande renvoyée, une licence d'essai vous sera envoyée sur l'adresse électronique que vous aurez indiquée pendant l'inscription.

© 2018 Altova GmbH

Obtention d'une licence sur Linux

Une licence doit être délivrée avec un Altova LicenseServer pour pouvoir exécuter RaptorXML +XBRL Server. L'obtention de la licence est un processus en deux étapes :

1. **Enregistrer RaptorXML+XBRL Server** avec LicenseServer. L'enregistrement est effectué depuis RaptorXML+XBRL Server.
2. **Attribuer une licence** à RaptorXML+XBRL Server. L'attribution de licence est effectuée depuis LicenseServer.

Les étapes à suivre sont indiquées ci-dessous.

Note à propos de l'utilisateur root

Vous devez disposer de privilèges administrateur (root) pour pouvoir installer RaptorXML +XBRL Server. L'installation doit donc être effectuée en tant que l'utilisateur root. Si vous être connecté sous `root`, vous pouvez abandonner le mot-clé `sudo` des commandes recensées ci-dessous.

Lancer LicenseServer

Pour enregistrer et délivrer correctement une licence pour RaptorXML+XBRL Server avec LicenseServer, celui-ci doit fonctionner en tant que démon sur le réseau. Lancer LicenseServer en tant que démon avec la commande suivante :

[< 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, à tout moment, vous souhaitez arrêter le LicenseServer, remplacer **`start`** avec **`stop`** dans la commande ci-dessus. Par exemple :

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

Lancer RaptorXML+XBRL Server

Lancer RaptorXML+XBRL Server LicenseServer en tant que démon avec la commande suivante :

[< 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

Enregistrer RaptorXML+XBRL Server

Pour enregistrer RaptorXML+XBRL Server depuis l'interface à ligne de commande, utiliser la commande `licenseserver` :

```
sudo /opt/Altova/RaptorXMLXBRLServer2018/bin/raptorxmlxbrl
licenseserver [options] ServerName-Or-IP-Address
```

Par exemple, si `localhost` est le nom du serveur sur lequel LicenseServer est installé :

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

Dans la commande ci-dessus, `localhost` est le nom du serveur sur lequel LicenseServer est installé. Veuillez aussi noter que l'emplacement du programme d'exécution RaptorXML +XBRL Server est :

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

Une fois l'enregistrement effectué avec succès, rendez-vous sur l'onglet Server Management de la page de configuration de LicenseServer pour attribuer une licence à RaptorXML+XBRL Server.

Attribuer une licence

Une fois l'enregistrement de RaptorXML+XBRL Server effectué avec succès, il sera compris dans l'onglet Server Management de la page de configuration du LicenseServer. Rendez-vous dans cette section et attribuez une licence à RaptorXML+XBRL Server.

Note sur les cœurs et les licences

L'obtention de la licence des produits de serveur Altova, est basée sur le nombre de cœurs de processeurs disponibles sur la machine de produit. Par exemple, un processeur double cœur a deux cœurs, un processeur quadricœur a quatre cœurs, un processeur hexacœurs a six cœurs, etc. Le nombre de cœurs pour lesquels une licence a été délivrée pour un produit doit être supérieur ou égal au nombre de cœurs disponibles sur cette machine de serveur, que ce serveur soit une machine physique ou virtuelle. Par exemple, si un serveur a huit cœurs (un processeur octacœur), vous devrez acheter au moins une licence octacœur. Vous pouvez aussi additionner les licences pour obtenir le nombre souhaité de cœurs. Ainsi, deux licences de quadricœurs peuvent être utilisées pour un serveur octacœur au lieu d'acheter une licence octacœur.

Si vous utilisez un serveur d'ordinateur avec un grand nombre de cœurs CPU, mais ne disposez que d'un faible volume à traiter, vous pouvez aussi créer une machine virtuelle qui disposera d'un plus petit nombre de cœurs et acheter une licence pour ce nombre de cœurs. Il va de soi que la vitesse de traitement d'un tel déploiement sera moins rapide que si tous

les cœurs disponibles sur le serveur étaient utilisés.

Note: Chaque licence de produit de serveur Altova peut être utilisée pour une seule machine client à la fois (l'appareil sur lequel le produit de serveur Altova est installé), même si la licence a une capacité de licence n'est pas utilisée. Par exemple, si une licence de 10-cœurs est utilisée pour une machine client qui détient 6 cœurs CPU, les 4 cœurs restants de la capacité de licence ne pourront pas être utilisés simultanément pour une autre machine client.

Note : En raison des fonctions de ses services, MobileTogether Server Advanced Edition ne sera exécuté que sur des machines comportant **deux ou plus de cœurs**.

Les licences MobileTogether Server
Les licences de MobileTogether Server sont basées sur le nombre des cœurs CPU dans l'appareil MobileTogether Server. Les licences cœur permettent à un nombre illimité d'appareils MobileTogether Client de se connecter au serveur. Néanmoins, si vous cochez la case *Limiter à une seule exécution thread*, un seul appareil mobile pourra se connecter au MobileTogether Server à tout moment. Cela est utile pour l'évaluation et les tests à petite échelle. Veuillez noter que, dans ce cas, un second appareil se connectera à MobileTogether Server, et prendra la licence. Le premier appareil ne pourra plus se connecter et recevra un message d'erreur.

Notes à propos de l'environnement

Dossiers

Ci-dessous, vous trouverez une liste des dossiers importants dans votre configuration RaptorXML +XBRL Server.

Connexions JDBC

Veuillez noter les points suivants :

- Java Runtime Environment ou SDK doit être installé.
- Les pilotes JDBC de la base de données cible doivent être installés.
- Les variables d'environnement suivantes doivent être configurées correctement pour votre environnement:
  - `CLASSPATH`: pour trouver les fichiers jar
  - `PATH`: pour trouver le JRE, mais n'est pas forcément nécessaire, dépend de l'installation
  - `JAVA_HOME`: si nécessaire, selon l'installation.

> **Note**
> Sur les serveurs Linux, les seules connexions de bases de données prises en charge sont JDBC.

 © 2018 Altova GmbH

2.3 Configuration sur macOS

Cette section décrit l'installation et l'obtention de la licence de RaptorXML+XBRL Server sur les systèmes macOS.

Installation sur macOS

- Exigences du système
- Note concernant l'utilisateur root
- Désinstaller les anciennes versions des produits de serveurs Altova
- Télécharger le paquet macOS
- Installer RaptorXML+XBRL Server
- Altova LicenseServer
- Versions de LicenseServer
- Licence d'essai

Obtention de la licence sur macOS

- Note concernant l'utilisateur root
- Lancer LicenseServer
- Lancer RaptorXML+XBRL Server
- Enregistrer RaptorXML+XBRL Server
- Attribuer une licence

Installation sur macOS

RaptorXML+XBRL Server est disponible pour une installation sur les systèmes macOS. La procédure d'installation et de configuration est décrite ci-dessous.

- Exigences du système
  - (*Mac) OS X, macOS*

    OS X 10.10 ou plus récent

- Intégration du serveur FlowForce

  Si vous installez RaptorXML+XBRL Server avec FlowForce Server, il est recommandé d'installer FlowForce Server d'abord. Si vous installez RaptorXML+XBRL Server avant d'installer FlowForce Server, après avoir installé RaptorXML+XBRL Server et FlowForce Server, exécuter la commande suivante :

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

  Cette commande copie le fichier **.tool** depuis le répertoire **/etc** de RaptorXML+XBRL Server vers le répertoire **/tools** de FlowForce Server. Le fichier **.tool** est requis par FlowForce Server ; il contient le chemin vers le programme exécutable RaptorXML+XBRL Server. Il n'est pas nécessaire d'exécuter cette commande si vous installez FlowForce Server avant d'installer RaptorXML+XBRL Server.

- Note à propos de l'utilisateur root

  Vous devez disposer de privilèges administrateur (root) pour pouvoir installer RaptorXML +XBRL Server. L'installation doit donc être effectuée en tant que l'utilisateur root. Si vous être connecté sous `root`, vous pouvez retirer le mot-clé `sudo` des commandes recensées ci-dessous.

- Désinstaller les anciennes versions des produits de serveurs Altova

  Avant de désinstaller RaptorXML+XBRL Server, terminer le service avec la commande suivante :

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

  Pour vérifier si le service a été arrêté, ouvrez le Moniteur d'activité dans le Finder et assurez-vous que RaptorXML+XBRL Server ne se trouve pas dans la liste. Dans le dossier d'Application dans le Finder, cliquez avec le bouton de droite sur l'icône de RaptorXML +XBRL Server et choisissez **Déplacer dans la corbeille**. L'application sera déplacée dans la Corbeille. Néanmoins, vous devrez supprimer l'application du dossier `usr`. Utilisez la commande :

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

  Si vous souhaitez désinstaller une ancienne version d'Altova LicenseServer, vous devrez d'abord stopper son exécution en tant que service. Utiliser la commande suivante :

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

 © 2018 Altova GmbH

com.altova.LicenseServer.plist

Pour vérifier si le service a été interrompu, ouvrir le Moniteur d'activité dans Finder et s'assurer que LicenseServer ne se trouve pas dans la liste. Ensuite poursuivre le processus de désinstallation tel que décrit ci-dessus pour RaptorXML+XBRL Server.

- Télécharger le fichier image disque

  Télécharger le fichier image disque (.dmg) depuis le site Internet Altova (http://www.altova.com/download.html).

- Installer RaptorXML+XBRL Server

  Cliquer pour ouvrir l'image disque (.dmg) téléchargée. Le programme d'installation de RaptorXML+XBRL Server apparaît en tant que nouveau disque virtuel sur votre ordinateur. Sur ce nouveau disque virtuel, double-cliquer le paquet d'installation (.pkg). Suivre les étapes de l'assistant d'installation. Les explications sont simples à comprendre et contiennent une étape au cours de laquelle vous devrez confirmer les accords de licence avant de pouvoir continuer. Pour éjecter le disque après l'installation, cliquer avec le bouton de droite et sélectionner **Éjecter**.

  Le paquet RaptorXML+XBRL Server sera installé dans le dossier :
  /usr/local/Altova/RaptorXMLXBRLServer2018 (fichiers binaires)
  /var/Altova/RaptorXMLXBRLServer (fichiers de données : base de données et journaux)

  Le démon de serveur de RaptorXML+XBRL Server démarre automatiquement après l'installation et effectue un redémarrage de la machine. Vous pouvez lancer RaptorXML+XBRL Server en tant que démon avec la commande suivante :

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

- Altova LicenseServer

  Pour faire fonctionner tout produit Altova Server, y compris RaptorXML+XBRL Server, ce produit de serveur doit posséder une licence via un Altova LicenseServer sur votre réseau.

  Le paquet d'installation Altova LicenseServer est disponible sur le disque virtuel que vous avez monté dans l'étape précédente. Pour installer Altova LicenseServer, double-cliquer sur le paquet d'installation compris dans le disque virtuel et suivre les instructions sur l'écran. Vous devrez accepter l'accord de licence pour poursuivre l'installation.

  Altova LicenseServer peut également être téléchargé et installé séparément depuis le site Internet Altova (http://www.altova.com/download.html).

  Le paquet LicenseServer sera installé dans le dossier :
  /usr/local/Altova/LicenseServer

  Pour plus d'informations concernant l'enregistrement de RaptorXML+XBRL Server avec Altova LicenseServer et l'obtention d'une licence, voir la section, Obtention d'une licence sur macOS.

LicenseServer versions

- Une licence doit être délivrée pour les produits de serveur Altova, soit avec la version LicenseServer convenant à la version du RaptorXML+XBRL Server installée, soit avec une version ultérieure de LicenseServer.
- La version LicenseServer qui convient pour une version particulière de RaptorXML +XBRL Server est affichée pendant l'installation de RaptorXML+XBRL Server. Vous pouvez installer cette version de LicenseServer avec RaptorXML+XBRL Server, ou bien vous pouvez installer LicenseServer séparément.
- Avant d'installer une version plus récente de LicenseServer, toute version plus ancienne doit être désinstallée. Le programme d'installation LicenseServer effectuera cette procédure automatiquement si elle détecte une version plus ancienne.
- Les versions LicenseServer sont rétrocompatibles. Elles fonctionneront avec des versions plus anciennes de RaptorXML+XBRL Server.
- Si vous installez une nouvelle version de RaptorXML+XBRL Server et si votre version installée de LicenseServer est plus ancienne que le LicenseServer approprié, installer la dernière version disponible du site web Altova.
- Au moment de la désinstallation de LicenseServer, toutes les informations d'enregistrement et d'obtention de la licence contenues dans la version plus ancienne de LicenseServer seront enregistrées dans une base de données sur la machine de votre serveur. Ces données seront importées automatiquement dans la version la plus récente lorsque la version la plus récente est installée.
- Le numéro de version du LicenseServer actuellement installé est indiqué au bas de la page de configuration de LicenseServer (tous les onglets).

  Version actuelle : 2.7

Licence d'essai

Pendant le processus d'installation, vous serez invité à demander une licence d'essai de 30 jours pour RaptorXML+XBRL Server. Une fois la demande renvoyée, une licence d'essai vous sera envoyée sur l'adresse électronique que vous aurez indiquée pendant l'inscription.

 © 2018 Altova GmbH

Obtention d'une licence sur macOS

Une licence doit être délivrée avec un Altova LicenseServer pour pouvoir exécuter RaptorXML +XBRL Server. L'obtention de la licence est un processus en deux étapes:

1. **Enregistrer RaptorXML+XBRL Server** avec LicenseServer. L'enregistrement est effectué depuis RaptorXML+XBRL Server.
2. **Attribuer une licence** à RaptorXML+XBRL Server. L'attribution de licence est effectuée depuis LicenseServer.

Les étapes à suivre sont indiquées ci-dessous.

- Note à propos de l'utilisateur root

  Vous devez disposer de privilèges administrateur (root) pour pouvoir installer RaptorXML +XBRL Server. L'installation doit donc être effectuée en tant que l'utilisateur root. Si vous être connecté sous `root`, vous pouvez retirer le mot-clé `sudo` des commandes recensées ci-dessous.

- Lancer LicenseServer

  Pour enregistrer et délivrer correctement une licence pour RaptorXML+XBRL Server avec LicenseServer, celui-ci doit fonctionner en tant que démon. Lancer LicenseServer en tant que démon avec la commande suivante :

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

  Si, à tout moment, vous souhaitez arrêter le LicenseServer, remplacer **`load`** avec **`unload`** dans la commande ci-dessus :

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

- Lancer RaptorXML+XBRL Server

  Le démon de serveur de RaptorXML+XBRL Server démarre automatiquement après l'installation et effectue un redémarrage de la machine. Vous pouvez lancer RaptorXML +XBRL Server en tant que démon avec la commande suivante :

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

  Si, à tout moment, vous souhaitez arrêter RaptorXML+XBRL Server, utiliser :

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

- Enregistrer RaptorXML+XBRL Server

  Pour enregistrer RaptorXML+XBRL Server depuis l'interface à ligne de commande, utiliser la commande `licenseserver` :

  ```
  sudo /usr/local/Altova/RaptorXMLXBRLServer2018/bin/RaptorXMLXBRL
  licenseserver [options] ServerName-Or-IP-Address
  ```

  Par exemple, si `localhost` est le nom du serveur sur lequel LicenseServer est installé :

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

```
licenseserver localhost
```

Dans la commande ci-dessus, `localhost` est le nom du serveur sur lequel LicenseServer est installé. Veuillez aussi noter que l'emplacement du programme d'exécution RaptorXML +XBRL Server est :

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

Une fois l'enregistrement effectué avec succès, rendez-vous sur l'onglet Server Management de la page de configuration de LicenseServer pour attribuer une licence à RaptorXML+XBRL Server.

Attribuer une licence

Une fois l'enregistrement de RaptorXML+XBRL Server, effectué avec succès, il sera compris dans l'onglet Server Management de la page de configuration du LicenseServer. Rendez-vous dans cette section et attribuez une licence à RaptorXML+XBRL Server.

Note sur les cœurs et les licences
L'obtention de la licence des produits de serveur Altova, est basée sur le nombre de cœurs de processeurs disponibles sur la machine de produit. Par exemple, un processeur double cœur a deux cœurs, un processeur quadricœur a quatre cœurs, un processeur hexacœurs a six cœurs, etc. Le nombre de cœurs pour lesquels une licence a été délivrée pour un produit doit être supérieur ou égal au nombre de cœurs disponibles sur cette machine de serveur, que ce serveur soit une machine physique ou virtuelle. Par exemple, si un serveur a huit cœurs (un processeur octacœur), vous devrez acheter au moins une licence octacœur. Vous pouvez aussi additionner les licences pour obtenir le nombre souhaité de cœurs. Ainsi, deux licences de quadricœurs peuvent être utilisées pour un serveur octacœur au lieu d'acheter une licence octacœur.

Si vous utilisez un serveur d'ordinateur avec un grand nombre de cœurs CPU, mais ne disposez que d'un faible volume à traiter, vous pouvez aussi créer une machine virtuelle qui disposera d'un plus petit nombre de cœurs et acheter une licence pour ce nombre de cœurs. Il va de soi que la vitesse de traitement d'un tel déploiement sera moins rapide que si tous les cœurs disponibles sur le serveur étaient utilisés.

Note: Chaque licence de produit de serveur Altova peut être utilisée pour une seule machine client à la fois (l'appareil sur lequel le produit de serveur Altova est installé), même si la licence a une capacité de licence n'est pas utilisée. Par exemple, si une licence de 10-cœurs est utilisée pour une machine client qui détient 6 cœurs CPU, les 4 cœurs restants de la capacité de licence ne pourront pas être utilisés simultanément pour une autre machine client.

Note : En raison des fonctions de ses services, MobileTogether Server Advanced Edition ne sera exécuté que sur des machines comportant **deux ou plus de cœurs**.

Les licences MobileTogether Server
Les licences de MobileTogether Server sont basées sur le nombre des cœurs CPU dans l'appareil MobileTogether Server. Les licences cœur permettent à un nombre illimité d'appareils MobileTogether Client de se connecter au serveur. Néanmoins, si vous cochez la case *Limiter à une seule exécution thread*, un seul appareil mobile pourra se connecter au MobileTogether Server à tout moment. Cela est utile pour l'évaluation et les tests à petite

échelle. Veuillez noter que, dans ce cas, un second appareil se connectera à MobileTogether Server, et prendra la licence. Le premier appareil ne pourra plus se connecter et recevra un message d'erreur.

Notes à propos de l'environnement

Dossiers

Ci-dessous, vous trouverez une liste des dossiers importants dans votre configuration RaptorXML +XBRL Server.

Connexions JDBC

Veuillez noter les points suivants :

- Java Runtime Environment ou SDK doit être installé.
- Les pilotes JDBC de la base de données cible doivent être installés.
- Les variables d'environnement suivantes doivent être configurées correctement pour votre environnement:
  - `CLASSPATH`: pour trouver les fichiers jar
  - `PATH`: pour trouver le JRE, mais n'est pas forcément nécessaire, dépend de l'installation
  - `JAVA_HOME`: si nécessaire, selon l'installation

> **Note**
> Sur les serveurs macOS, les seules connexions de bases de données prises en charge sont JDBC.

 © 2018 Altova GmbH

2.4 Catalogues XML

Le mécanisme de catalogue XML permet d'extraire des fichiers depuis des dossiers locaux, accroissant ainsi la vitesse de traitement générale, ainsi qu'en améliorant la portabilité des documents—puisque seules les URI de fichier de catalogue nécessitent un changement. Voir la section Le fonctionnement des catalogues pour plus de détails.

Les produits XML d'Altova utilisent un mécanisme de catalogue pour accéder et charger rapidement des fichiers utilisés communément, comme des DTDs et des Schémas XML. Ce mécanisme de catalogue peut être personnalisé et étendu par l'utilisateur et est décrit dans la section Le mécanisme de Catalogue XML d'Altova. La section Variables pour les emplacements de Système rassemble les variables Windows pour les emplacement de système communs. Ces variables peuvent être utilisées dans les fichiers de catalogue pour situer des dossiers utilisés communément.

Cette section est organisée dans les sous-sections suivantes :

- Le fonctionnement des catalogues
- Le mécanisme de Catalogue XML d'Altova
- Variables pour les emplacements de Système

Pour plus d'informations concernant les catalogues, voir la spécification de catalogues XML.

Le fonctionnement des catalogues

Cette section :

- Mappage d'identifiants publics et de système dans des URL locales
- Mappage de chemins de fichier, URL Web et de noms dans des URL locales

Les catalogues servent à rediriger les appels vers des ressources à distance vers une URL locale. Cela est effectué par le biais de mappage, dans le fichier de catalogue, les identifiants publics ou de système, les URI ou des parties d'identifiants, ou des URIs de l'URL locale requise.

Mappage d'identifiants publics et de système dans des URL locales

Lorsque la déclaration `DOCTYPE` d'un DTD dans un fichier XML est lue, l'identifiant public ou système de la déclaration localise la ressource requise. Si l'identifiant sélectionne une ressource à distance ou si l'identifiant n'est pas un localisateur, il peut toujours être mappé par le biais d'une entrée de catalogue dans une ressource locale.

Considérons par exemple le fichier SVG suivant :

```
<?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>
```

Son identifiant public est : **`-//W3C//DTD SVG 1.1//EN`**
Son identifiant système est : `http://www.w3.org/Graphics/SVG/1.1/DTD/svg11.dtd`

Une entrée de catalogue pourrait mapper l'identifiant public dans une URL locale comme l'exemple suivant :

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

Ou bien, une entrée de catalogue pourrait mapper l'identifiant de système dans une URL locale comme l'exemple suivant :

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

En cas de correspondance avec l'identifiant public ou système dans le catalogue, l'URL à laquelle il est mappé est utilisée. (Les chemins relatifs sont résolus avec une référence à un attribut `xml:base` dans l'élément de catalogue de redirection ; L'URL de base de fallback est l'URL du fichier de catalogue.) Si aucune correspondance ne peut être établie pour l'identifiant public ou système dans le catalogue, l'URL dans le document XML sera réutilisée (dans l'exemple ci-dessus : `http://www.w3.org/Graphics/SVG/1.1/DTD/svg11.dtd`).

 © 2018 Altova GmbH

Mappage de chemins de fichier relatifs ou absolus, URL Web ou de noms dans des URL locales

L'élément **uri** peut être utilisé pour mapper un chemin de fichier relatif ou absolu ou une URL Web, un bien un simple nom vers une URL local, comme suit:

- `<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"/>`

Lorsque la valeur `name` est rencontrée, elle est mappée à la ressource spécifiée dans l'attribut `uri`. Avec un différent catalogue, le même nom peut être mappé vers une ressource différente. Par exemple, si vous avez :

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

Normalement, la part d'URI de la valeur de l'attribut (en gras dans l'exemple ci-dessus) est un chemin vers l'emplacement de schéma. Néanmoins, si le schéma est référencé par le biais d'un catalogue, la partie d'URI ne doit pas obligatoirement pointer vers un Schéma XML réel, mais elle doit exister de manière à ce que la validité lexicale de l'attribut `xsi:schemaLocation` soit maintenue. Une valeur de `foo`, par exemple, serait suffisante pour la part URI de la valeur d'attribut de `xsi:schemaLocation` (au lieu de `Orgchart.xsd`). Le schéma est situé dans le catalogue au moyen de la partie espace de noms de la valeur d'attribut `xsi:schemaLocation`. Dans l'exemple ci-dessus, la partie espace de noms est `http://www.altova.com/schemas/orgchart`.

Dans le catalogue, l'entrée suivante situerait le schéma sur la base de cette partie d'espace de noms.

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

Pour plus d'informations concernant ces éléments, voir la spécification de Catalogues XML.

Le mécanisme de Catalogue XML d'Altova

Cette section :

- Le fichier racine de catalogue, `RootCatalog.xml`, contient les fichiers de catalogue que RaptorXML consultera.
- Les fichiers d'extension de catalogue Altova : `CoreCatalog.xml`, `CustomCatalog.xml`, et `Catalog.xml`.
- Sous ensemble de catalogue pris en charge.

RootCatalog.xml

Par défaut, RaptorXML consultera le fichier `RootCatalog.xml` (*listé ci-dessous*) pour la liste des fichiers de catalogue à utiliser. `RootCatalog.xml` est situé dans le dossier :

```
<ProgramFilesFolder>\Altova\RaptorXMLXBRLServer2018\etc
```

Pour utiliser un autre fichier que le catalogue root, utiliser l'option `--catalog` dans la ligne de commande, la méthode `setCatalog` de l'interface Java, ou la méthode `Catalog` de l'interface 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"/>

  <!-- Include all catalogs under common schemas folder on the first directory
level -->
  <nextCatalog spy:recurseFrom="%AltovaCommonFolder%/Schemas"
catalog="catalog.xml" spy:depth="1"/>

  <!-- Include all catalogs under common XBRL folder on the first directory
level -->
  <nextCatalog spy:recurseFrom="%AltovaCommonFolder%/XBRL" catalog="catalog.xml"
spy:depth="1"/>
</catalog>
```

Les fichiers de catalogue supplémentaires à consulter sont tous listés dans un élément `nextCatalog` et vous pouvez en ajouter autant que vous voulez. Chaque fichier de catalogue est consulté et les mappages qu'ils contiennent sont résolus.

Dans la liste ci-dessus, veuillez noter que deux catalogues sont directement référencés : `CoreCatalog.xml` et `CustomCatalog.xml`. De plus, les catalogues nommés `catalog.xml` qui se trouvent dans le premier niveau des sous-dossiers des dossiers `Schemas` et `XBRL` sont aussi référencés. (La valeur de la variable `%AltovaCommonFolder%` est donnée dans la section,

Variables pour Localisations de système.)

Les fichiers de catalogue situés dans le dossier Altova Common Folder mappent les identifiants public et système prédéfinies des schémas utilisés communément (comme le Schéma XML et XHTML) vers des URI qui pointent vers les copies locales des schémas respectifs. Ces schémas sont installés dans le dossier Altova Common Folder lorsque RaptorXML est installé.

CoreCatalog.xml, CustomCatalog.xml, et Catalog.xml

Les fichiers de catalogue `CoreCatalog.xml` et `CustomCatalog.xml` sont listés dans `RootCatalog.xml` pour une consultation :

- `CoreCatalog.xml` contient certains mappages spécifiques à Altova pour localiser les schémas dans le dossier Altova Common Folder.
- `CustomCatalog.xml` est un fichier squelette dans lequel vous pouvez créer vos propres mappages. Vous pouvez ajouter des mappages dans `CustomCatalog.xml` pour tout schéma que vous nécessitez mais qui n'est pas adressé par les fichiers de catalogue dans le dossier Altova Common Folder. Pour ce faire utiliser les éléments pris en charge du mécanisme de catalogue OASIS (*voir ci-dessous*).
- Il existe un certain nombre de fichiers `Catalog.xml` contenus dans le dossier d'un schéma spécifique ou d'une taxonomie XBRL dans le dossier Altova Common Folder, et chacun mappe les identifiants public et/ou system vers les URI qui pointent vers les copies enregistrées localement des schémas respectifs.

`CoreCatalog.xml` et `CustomCatalog.xml` se trouvent dans le dossier, `<ProgramFilesFolder>\Altova\RaptorXMLXBRLServer2018\etc`. Les fichiers `catalog.xml` se trouvent chacun dans un dossier de schéma spécifique. Ces dossiers de schéma se trouvent dans les dossiers : `%AltovaCommonFolder%\Schemas` et `%AltovaCommonFolder%\XBRL`.

Sous-ensemble de catalogue pris en charge

Lors de la création des entrées dans un fichier de catalogue que RaptorXML utilisera, utiliser uniquement les éléments suivants de la spécification de catalogue OASIS. Chacun des éléments ci-dessous est listé avec une explication de la valeur de ses attributs. Pour une explication plus détaillée, voir la spécification de Catalogues XML.

- `<public publicId="PublicID of Resource" uri="URL of local file"/>`
- `<system systemId="SystemID of Resource" uri="URL of local file"/>`
- `<uri name="filename" uri="URL of file identified by filename"/>`
- `<rewriteURI uriStartString="StartString of URI to rewrite" rewritePrefix="String to replace StartString"/>`
- `<rewriteSystem systemIdStartString="StartString of SystemID" rewritePrefix="Replacement string to locate resource locally"/>`

Dans les cas où il n'y a pas d'identifiant public, l'identifiant system peut être mappé directement dans une URL par le biais de l'élément `system`. De plus, un URI peut être mappé à un autre URI avec l'élément `uri`. Les éléments `rewriteURI` et `rewriteSystem` permettent la réécriture de la partie de début d'un URI ou d'un identifiant system, respectivement. Cela permet de remplacer le début d'un chemin de fichier et donc de permettre le ciblage vers un autre répertoire.

Note : Chaque élément peut prendre l'attribut `xml:base`, qui est utilisé pour spécifier l'URI de

base de cet élément. Si aucun élément `xml:base` n'est présent, l'URI de base sera l'URI du fichier de catalogue.

Pour plus d'informations concernant ces éléments, voir la spécification de Catalogues XML.

 © 2018 Altova GmbH

Variables pour les emplacements de Système

Les variables d'environnement shell peuvent être utilisées dans les fichiers de catalogue pour spécifier le chemin vers les différents emplacements de système Windows. Les variables suivantes sont prises en charge :

`%AltovaCommonFolder%`	C:\Program Files\Altova\Common2018
`%DesktopFolder%`	Chemin complet vers le dossier Desktop pour l'utilisateur actuel.
`%ProgramMenuFolder%`	Chemin complet vers le dossier Program Menu pour l'utilisateur actuel.
`%StartMenuFolder%`	Chemin complet vers le dossier Start Menu pour l'utilisateur actuel.
`%StartUpFolder%`	Chemin complet vers le dossier Start Up pour l'utilisateur actuel.
`%TemplateFolder%`	Chemin complet vers le dossier Template pour l'utilisateur actuel.
`%AdminToolsFolder%`	Chemin complet vers le répertoire de système de fichier qui stocke les outils administratifs pour l'utilisateur actuel.
`%AppDataFolder%`	Chemin complet vers le dossier Application Data pour l'utilisateur actuel.
`%CommonAppDataFolder%`	Chemin complet vers le répertoire de fichier contenant les données d'application pour tous les utilisateurs.
`%FavoritesFolder%`	Chemin complet vers le dossier Favorites pour l'utilisateur actuel.
`%PersonalFolder%`	Chemin complet vers le dossier Personal pour l'utilisateur actuel.
`%SendToFolder%`	Chemin complet vers le dossier SendTo pour l'utilisateur actuel.
`%FontsFolder%`	Chemin complet vers le dossier System Fonts.
`%ProgramFilesFolder%`	Chemin complet vers le dossier Program Files pour l'utilisateur actuel.
`%CommonFilesFolder%`	Chemin complet vers le dossier Common Files pour l'utilisateur actuel.
`%WindowsFolder%`	Chemin complet vers le dossier Windows pour l'utilisateur actuel.

`%SystemFolder%`	Chemin complet vers le dossier System pour l'utilisateur actuel.
`%LocalAppDataFolder%`	Chemin complet vers le répertoire de système de fichier qui sert en tant qu'archivage de données pour les applications locales (non-roaming).
`%MyPicturesFolder%`	Chemin complet vers le dossier MyPictures.

© 2018 Altova GmbH

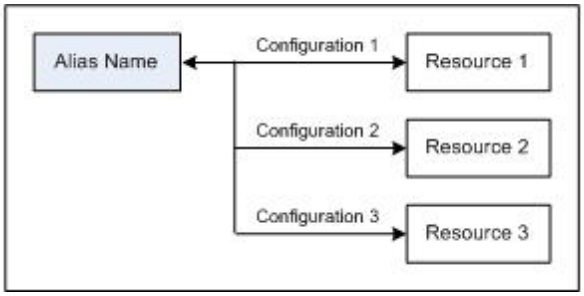
2.5 Ressources globales

Cette section :

- À propos des Ressources globales
- Utiliser les Ressources globales

À propos des Ressources globales

Un fichier de ressource globale Altova mappe un alias vers des ressources multiples par le biais de configurations différentes, comme indiqué dans le diagramme ci-dessous. Un alias peut donc être changé pour accéder à une ressource différence en changeant sa configuration.



Les ressources globales sont définies dans des produits Altova, comme Altova XMLSpy, et sont enregistrées dans un fichier XML de ressources globales. RaptorXML est en mesure d'utiliser les ressources globales en tant qu'entrées. Pour ce faire, il nécessite le nom et l'emplacement des fichiers de ressources globales et l'alias et la configuration à utiliser.

L'avantage d'utiliser des ressources globales est que la ressource peut être modifiée simplement en changeant le nom de la configuration. Lorsque vous utilisez RaptorXML, cela signifie qu'en fournissant une valeur différente de l'option `--globalresourcesconfig | --gc`, une ressource différente peut être utilisée. (*Voir l'exemple ci-dessous.*)

Utiliser les Ressources globales avec RaptorXML

Pour spécifier une ressource globale en tant qu'une entrée pour une commande RaptorXML, les paramètres suivants sont exigés :

- Le fichier XML de ressources globales (spécifié dans la CLI avec l'option `--globalresourcesfile | --gr`)
- La configuration exigée (spécifiée dans la CLI avec l'option `--globalresourcesconfig | --gc`)
- L'alias. Cela peut être spécifié directement dans la CLI où un nom de fichier est requis, ou il peut se trouver à un emplacement dans un fichier XML où RaptorXML cherche le nom d'un fichier (comme dans un attribut `xsi:schemaLocation`).

Par exemple, si vous souhaitez transformer `input.xml` avec `transform.xslt` en `output.html`, cela s'effectuera généralement sur la CLI avec la commande suivante qui utilise les noms de fichier :

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

Si, néanmoins, vous détenez une définition de ressource globale qui correspond à l'alias `MyInput` à la ressource de fichier `FirstInput.xml` comme une configuration appelée `FirstConfig`, alors vous pourriez utiliser l'alias `MyInput` sur la CLI comme suit :

```
raptorxmlxbrl xslt --input=altova://file_resource/MyInput --gr=C:
\MyGlobalResources.xml --gc=FirstConfig --output=Output.html transform.xslt
```

Maintenant, si vous avez une autre ressource de fichier, comme par exemple `SecondInput.xml`, qui correspond à l'alias `MyInput` par le biais d'une configuration appelée `SecondConfig`, alors cette ressource peut être utilisée en ne changeant que l'option `--gc` de la commande précédente :

```
raptorxmlxbrl xslt --input=altova://file_resource/MyInput --gr=C:
\MyGlobalResources.xml --gc=SecondConfig --output=Output.html transform.xslt
```

Note : Dans l'exemple ci-dessus, une ressource de fichier a été utilisée ; une ressource de fichier doit être préfixée par **`altova://file_resource/`**. Vous pouvez aussi utiliser des ressources globales qui sont des dossiers. Pour identifier une ressource de dossier, utiliser : **`altova://folder_resource/AliasName`**. Veuillez noter que, sur la CLI, vous pouvez aussi utiliser des ressources de dossier en tant que partie d'un chemin de fichier. Par exemple : **`altova://folder_resource/AliasName/input.xml`**.

 © 2018 Altova GmbH

2.6 Problèmes de sécurité

Cette section :

- Problèmes de sécurité liés à l'interface HTTP
- Rendre les scripts Python sûrs

Certaines fonctions d'interface de RaptorXML+XBRL Server posent des problèmes de sécurité. Ceux-ci sont décrits ci-dessous avec leurs solutions.

Problèmes de sécurité liés à l'interface HTTP

L'interface HTTP, par défaut, permet aux documents de résultat d'être rédigés sur tout emplacement spécifié par le client (qui est accessible avec le protocole HTTP). C'est pourquoi il est important de considérer ces aspects de sécurité lors de la configuration de RaptorXML+XBRL Server.

Si vous estimez que la sécurité ait pu être compromise ou que l'interface ait pu être utilisée à mauvais escient, le serveur peut être configuré pour écrire les documents de résultat dans un répertoire de sortie réservé sur le serveur lui-même. Cela est spécifié en configurant l'option `server.unrestricted-filesystem-access` du fichier de configuration du serveur sur `false`. Lorsque l'accès est limité de cette manière, le client peut télécharger les documents de résultat depuis le répertoire de sortie réservé avec des requêtes `GET`. En alternative, un administrateur peut copier/charger les fichiers de document de résultat depuis le serveur vers l'emplacement cible.

Rendre les scripts Python sûrs

Lorsqu'un script Python est spécifié dans une commande via HTTP pour RaptorXML+XBRL Server, le script fonctionnera uniquement s'il est situé dans le répertoire de confiance. Le script est exécuté depuis le répertoire de confiance. La spécification d'un script Python provenant d'un autre répertoire quelconque résultera en une erreur. Le répertoire de confiance est spécifié dans le paramètre `server.script-root-dir` du fichier de configuration de serveur, et un répertoire de confiance **doit** être spécifié si vous souhaitez utiliser des scripts Python. Veuillez vous assurer que tous les scripts Python à utiliser soient sauvegardés dans ce répertoire.

Bien que toutes les sorties générées par le serveur pour les requêtes de tâche HTTP soient écrites dans le répertoire de sortie de tâche (qui est un sous-répertoire de `output-root-directory`), cette limitation ne s'applique pas aux scripts Python, qui peuvent écrire dans n'importe quel emplacement. L'administrateur de serveur doit revoir les scripts Python dans le répertoire de confiance en ce qui concerne des problèmes de vulnérabilité potentiels.

Chapitre 3

Interface de ligne de commande (CLI)

3 Interface de ligne de commande (CLI)

L'exécutable RaptorXML+XBRL Server à utiliser avec l'interface de commande de ligne (CLI) est situé par défaut sous :

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

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

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

Casse et lignes obliques sur la ligne de commande

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

* Veuillez noter que les caractères minuscules (`raptorxmlxbrl`) fonctionnent sur toutes les plates-formes (Windows, Linux et Mac), alors qu'une écriture en majuscule/minuscule (`RaptorXMLXBRL`) ne fonctionne que sur Windows et Mac.
* Utiliser les barres obliques penchées vers l'avant pour Linux et Mac, et les barres obliques penchées vers l'arrière sur Windows.

Utilisation

La syntaxe de ligne de commande est :

Windows **`RaptorXMLXBRL --h | --help | --version | <command> [options] [arguments]`**

Linux **`raptorxmlxbrl --h | --help | --version | <command> [options] [arguments]`**

Mac **`raptorxmlxbrl --h | --help | --version | <command> [options] [arguments]`**

`RaptorXMLXBRL`	Appelle l'application des plateformes Windows.
`raptorxmlxbrl`	Appelle l'application sur les plateformes Unix (Linux et Mac).
`--h | --help`	Affiche le texte help.
`--version`	Affiche le numéro de version d'application.
`<command>`	La commande à exécuter. Voir la liste ci-dessous. Chaque commande est décrite en détail, avec ses options et arguments, dans des sous-sections de cette section.

 © 2018 Altova GmbH

[options]	Les options d'une commande. Elles sont rassemblées avec leurs commandes respectives et sont décrites en détail dans la section Options.
[arguments]	Les arguments d'une commande. Elles sont rassemblées et décrites avec leurs commandes respectives.

Commandes CLI

Les commandes CLI disponibles sont rassemblées ci-dessous, organisées par leurs fonctions. Elles sont expliquées en détail dans les sous-sections de cette section. (Veuillez noter que certaines commandes de validation apparaissent dans plus d'un groupe dans la liste ci-dessous.)

Toutes les commandes de validation

valdtd | dtd	Valide un document DTD.
valxsd | xsd	Valide un document W3C de Schéma XML.
valxml-withdtd | xml	Valide un document XML par rapport à un DTD.
valxml-withxsd | xsi	Valide un document XML par rapport au Schéma XML.
valxslt	Valide un document XSLT.
valxquery	Valide un document XQuery.
valxbrl | xbrl	Valide un document d'instance XBRL (extension .xbrl).
valinlinexbrl | ixbrl	Valide un document Inline XBRL (iXBRL).
valxbrltaxonomy | dts	Valide un document de taxonomie XBRL (schéma) (extension .xsd).
valtaxonomypackage	Valide un pack de taxonomie XBRL.
valjson	Valide un document JSON.
valjsonschema	Valide un document de schéma JSON par rapport à la spécification de schéma.
valavroschema	Valide un schéma Avro par rapport à la spécification de schéma Avro.
valavrojson	Valide un fichier de données JSON par rapport à un schéma Avro.
valavro	Valide les données dans un binaire Avro par rapport à un son schéma Avro.
valany	Valide tout document d'un type validé par les commandes précédentes. Le type de document est détecté automatiquement.

Commandes de vérification de la bonne formation

wfjson	Vérifie la bonne formation des documents JSON.
wfxml	Vérifie la bonne formation d'un document XML.
wfdtd	Vérifie la bonne formation d'un document DTD.
wfany	Vérifie la bonne formation de n'importe quel document XML ou DTD.

Commandes de validation XBRL

valxbrl | xbrl	Valide un document d'instance XBRL (extension `.xbrl`).
valinlinexbrl | ixbrl	Valide un document XBRL (iXBRL) Inline.
valxbrltaxonomy | dts	Valide un document de taxonomie XBRL (schéma) (extension `.xsd`).
valtaxonomypackage	Valide un pack de taxonomie XBRL.
valxbrlver	Valide un ou plusieurs rapports de version XBRL.

Commandes XSLT

xslt	Effectue une transformation en utilisant le fichier XSLT fourni par l'argument.
valxslt	Valide un document XSLT.

Commandes XQuery

xquery	Exécute un XQuery utilisant le fichier XQuery fourni par l'argument.
valxquery	Valide un document XQuery.

Commandes de Signature XML

xmlsignature-sign	Crée un document de sortie de signature XML depuis un document d'entrée
xmlsignature-verify	Vérifie un document de signature XML
xmlsignature-update	Met à jour la signature d'un document XML (modifié)
xmlsignature-remove	Supprime la signature d'un document XML

Commandes JSON/Avro

avroextractschema	Extrait le schéma Avro depuis un fichier binaire Avro
valavro	Valide les données dans un ou plusieurs binaire Avro par rapport au

 © 2018 Altova GmbH

	schéma Avro respectif de chaque binaire
valavrojson	Valide un ou plusieurs fichiers de données JSON par rapport à un schéma Avro
valavroschema	Valide un schéma Avro par rapport à la spécification de schéma Avro
valjsonschema	Contrôle la validité des documents de schéma JSON.
valjson	Contrôle la validité des documents JSON.
wfjson	Contrôle la bonne formation de JSON.

3.1 Commande de validation XML, DTD, XSD

Les commandes de validation XML peuvent être utilisées pour valider les types de document suivantes :

- *XML:* Valide des documents d'instance XML par rapport à un DTD (`valxml-withdtd | xml`) ou un Schéma XML 1.0/1.1 (`valxml-withxsd | xsi`).
- *DTD:* Vérifier qu'un DTD est bien formé et ne contienne aucune erreur (`valdtd | dtd`).
- *XSD:* Valide un document de Schéma XML (XSD) W3C par rapport aux règles de la spécification de Schéma XML (`valxsd | xsd`).

Les commandes de validation XML sont décrites en détail dans les sous-sections de cette section :

`valxml-withdtd | xml`	Valide un document XML par rapport à un DTD.
`valxml-withxsd | xsi`	Valide un document d'instance XML par rapport au Schéma XML.
`valdtd | dtd`	Valide un document DTD.
`valxsd | xsd`	Valide un document de Schéma (XSD) XML W3C.

Note : Instance XBRL, taxonomie XBRL, les documents XSLT, XQuery, JSON, et Avro peuvent aussi être validés. Ces commandes de validation sont décrits dans leurs sections respectives : Commandes de validation XBRL, Commandes XSLT, Commandes XQuery, Commandes JSON/Avro.

 © 2018 Altova GmbH

valxml-withdtd (xml)

La commande **valxml-withdtd | xml** valide un ou plusieurs documents d'instance XML par rapport à un DTD.

Windows	**RaptorXMLXBRL valxml-withdtd | xml [options] *InputFile***
Linux	**raptorxmlxbrl valxml-withdtd | xml [options] *InputFile***
Mac	**raptorxmlxbrl valxml-withdtd | xml [options] *InputFile***

L'argument *InputFile* est le document XML à valider. Si une référence à un DTD existe dans le document XML, l'option `--dtd` n'est pas requise.

Pour valider plusieurs documents, vous pouvez soit : (i) lister les fichiers à valider sur la CLI, chaque fichier étant séparé de l'autre par un espace ; ou (ii) lister les fichiers à valider dans un fichier de texte (fichier `.txt`), avec un nom de fichier par ligne, et fournir ce fichier de texte en tant que l'argument *InputFile* avec l'option `--listfile` définie sur `true` (*voir la liste des Options ci-dessous*).

Exemples

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

Casse et lignes obliques sur la ligne de commande

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

* Veuillez noter que les caractères minuscules (`raptorxmlxbrl`) fonctionnent sur toutes les plates-formes (Windows, Linux et Mac), alors qu'une écriture en majuscule/minuscule (`RaptorXMLXBRL`) ne fonctionne que sur Windows et Mac.
* Utiliser les barres obliques penchées vers l'avant pour Linux et Mac, et les barres obliques penchées vers l'arrière sur Windows.

Options

Les options de la commande sont listées ci-dessous, et organisées en groupes. Les valeurs peuvent être spécifiées sans guillemets sauf dans deux cas : (i) lorsque le string de valeur contient des espaces, ou (ii) si la présence de guillemets est explicitement requise dans la description de l'option.

Validation et traitement

- dtd

  `--dtd =` *`FILE`*
  Spécifie le document DTD externe à utiliser pour la validation. Si une référence à un DTD externe est présente dans le document XM, alors l'option CLI contourne la référence externe.

- listfile

  `--listfile =` `true|false`
  Si `true`, traiter l'argument *`InputFile`* de la commande en tant que fichier de texte contenant un nom de fichier par ligne. La valeur par défaut est `false`. (Une alternative est de lister les fichiers sur la CLI avec un espace en tant que séparateur. Veuillez noter, néanmoins, que les CLI ont une limitation de caractères maximum.) Veuillez noter que l'option `--listfile` s'applique uniquement aux arguments, et non pas aux options.
  Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- namespaces

  `--namespaces =` `true|false`
  Permet le traitement sensible aux espaces de noms. Cela est utile pour contrôler l'instance XML pour des erreurs dues à des espaces de noms incorrectes. La valeur par défaut est `false`.
  Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- recurse

  `--recurse =` `true|false`
  Utilisé pour sélectionner des fichiers dans le cadre de sous-répertoires, y compris des archives ZIP. Si `true`, l'argument *`InputFile`* de la commande sélectionnera aussi le fichier spécifié dans les sous-répertoires. Par exemple : `test.zip|zip\test.xml` sélectionnera des fichiers nommés `test.xml` à tous les niveaux de dossier du dossier zip. Les caractères génériques `*` et `?` peuvent être utilisés. Donc, `*.xml` sélectionnera tous les fichiers `.xml` dans le dossier (zip). La valeur par défaut de l'option est `false`.
  Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- streaming

  `--streaming =` `true|false`
  Permet la validation de streaming. La valeur par défaut est `true`. Dans le mode streaming, les données stockées dans la mémoire sont minimisées et le traitement est plus rapide. L'inconvénient est que l'information que peut être requise par conséquent, par exemple, un modèle de données du document d'instance XML —ne sera pas disponible. Dans des situations où cela est significatif, le mode de streaming devra être éteint (en donnant à `--streaming` une valeur de `false`). En utilisant l'option `--script` avec la commande `valxml-withxsd`, désactiver le streaming. Veuillez noter que l'option `--streaming` est ignorée si`--parallel-assessment` est défini sur `true`.
  Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

 © 2018 Altova GmbH

- Catalogues et ressources globales
  - catalog

    --catalog = *FILE*
    Spécifie le chemin absolu vers un fichier de catalogue root qui n'est pas le fichier catalogue root installé. La valeur par défaut est le chemin absolu vers le fichier de catalogue root installé (`<installation-folder>\Altova\RaptorXMLXBRLServer2018\etc\RootCatalog.xml`). Voir la section, Catalogues XML, pour plus d'informations à propos du travail avec les catalogues.

  - user-catalog

    --user-catalog = *FILE*
    Spécifie le chemin absolu vers un catalogue XML à utiliser en plus du catalogue root. Voir la section, Catalogues XML, pour plus d'informations concernant le travail avec des catalogues.

  - enable-globalresources

    --enable-globalresources = `true|false`
    Active les ressources globales. La valeur par défaut est `false`.
    Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

  - globalresourceconfig [gc]

    --gc | --globalresourceconfig = *VALUE*
    Spécifie la configuration active de la ressource globale (et active les ressources globales).

  - globalresourcefile [gr]

    --gr | --globalresourcefile = *FILE*
    Spécifie le fichier de ressource globale (et active les ressources globales).

- Messages, erreurs, aide, timeout, version
  - error-format

    --error-format = `text|shortxml|longxml`
    Spécifie le format de la sortie d'erreur. La valeur par défaut est `text`. Les autres options génèrent des formats XML, avec `longxml` générant plus de détails.

  - error-limit

    --error-limit = *N*
    Spécifie la limite d'erreur avec une plage de valeur de `1` à `999`. La valeur par défaut est `100`. Le traitement s'arrête lorsque la limite d'erreur est atteinte. Utile pour limiter l'utilisation du processeur pendant la validation/transformation.

  - help

    --help

Affiche le texte d'aide pour la commande. Par exemple, `valany --h`. (en alternative, la commande `help` peut être utilisée avec un argument. Par exemple : `help valany`.)

- log-output

  `--log-output =` *`FILE`*
  Écrit la sortie de log dans l'URL de fichier spécifié. Veuillez vous assurer que la CLI a une permission d'écriture dans l'emplacement de la sortie.

- network-timeout

  `--network-timeout =` *`VALUE`*
  Spécifie le timeout en secondes pour les opérations I/O à distance. La valeur par défaut est : `40`.

- verbose

  `--verbose =` `true|false`
  Une valeur de `true` permet la sortie d'informations supplémentaires pendant la validation. La valeur par défaut est `false`.
  Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- verbose-output

  `--verbose-output =` *`FILE`*
  Écrit la sortie verbeuse sur *`FILE`*.

- version

  `--version`
  Affiche la version de RaptorXML+XBRL Server. Si utilisé avec une commande, placer `--version` avant la commande.

- warning-limit

  `--warning-limit =` *`VALUE`*
  Spécifie la limite d'avertissement dans la plage `1-65535`. Le traitement se poursuit si cette limite a été atteinte, mais d'autres avertissements ne sont pas rapportés. La valeur par défaut est `100`.

 © 2018 Altova GmbH

valxml-withxsd (xsi)

La commande **valxml-withdtd | xsi** valide un ou plusieurs documents d'instance XML par rapport aux spécifications W3C XML Schema Definition Language (XSD) 1.0 et 1.1.

Windows	**RaptorXMLXBRL valxml-withxsd | xsi [options] *InputFile***
Linux	**raptorxmlxbrl valxml-withxsd | xsi [options] *InputFile***
Mac	**raptorxmlxbrl valxml-withxsd | xsi [options] *InputFile***

L'argument *InputFile* est le document XML à valider. `--schemalocation-hints=true|false` indique si la référence XSD dans le document XML doit être utilisée ou pas, la valeur par défaut étant `true` (l'emplacement est utilisé). L'option `--xsd=FILE` spécifie les schémas à utiliser.

Pour valider plusieurs documents, vous pouvez soit : (i) lister les fichiers à valider sur la CLI, chaque fichier étant séparé de l'autre par un espace ; ou (ii) lister les fichiers à valider dans un fichier de texte (fichier `.txt`), avec un nom de fichier par ligne, et fournir ce fichier de texte en tant que l'argument *InputFile* avec l'option `--listfile` définie sur `true` (*voir la liste des Options ci-dessous*).

Note : Si vous utilisez l'option `--script` pour exécuter des scripts Python, veuillez vous assurer de spécifier `--streaming=false`.

Exemples

- **raptorxmlxbrl** `valxml-withxsd --schemalocation-hints=false --xsd=c:\MyXSD.xsd c:\HasNoXSDRef.xml`
- **raptorxmlxbrl** `xsi c:\HasXSDRef.xml`
- **raptorxmlxbrl** `xsi --xsd-version=1.1 --listfile=true c:\FileList.txt`

Casse et lignes obliques sur la ligne de commande

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

* Veuillez noter que les caractères minuscules (`raptorxmlxbrl`) fonctionnent sur toutes les plates-formes (Windows, Linux et Mac), alors qu'une écriture en majuscule/minuscule (`RaptorXMLXBRL`) ne fonctionne que sur Windows et Mac.
* Utiliser les barres obliques penchées vers l'avant pour Linux et Mac, et les barres obliques penchées vers l'arrière sur Windows.

Options

Les options de la commande sont listées ci-dessous, et organisées en groupes. Les valeurs peuvent être spécifiées sans guillemets sauf dans deux cas : (i) lorsque le string de valeur contient des espaces, ou (ii) si la présence de guillemets est explicitement requise dans la

description de l'option.

- Validation et traitement
  - assessment-mode

    `--assessment-mode =` `lax|strict`
    Spécifie le mode d'évaluation de validité de schéma tel que défini dans les spécifications XSD. La valeur par défaut est `strict`. Le document d'instance XML sera validé conformément au mode spécifié avec cette option.

  - ct-restrict-mode

    `--ct-restrict-mode =` `1.0|1.1|default`
    Spécifie comment contrôler les restrictions de type complexes. Une valeur de **`1.0`** contrôles les restrictions de type complexes telles que définies dans la spécification XSD 1.0 — même dans le mode de validation XSD 1.1. Une valeur de **`1.1`** contrôles les restrictions de type complexe telles que définies dans la spécification XSD 1.1 — même dans le mode de validation XSD 1.0. Une valeur de **`default`** contrôle les restrictions de type complexes telles que définies dans la spécification XSD du mode de validation actuel (1.0 ou 1.1). La valeur par défaut est **`default`**.

  - listfile

    `--listfile =` `true|false`
    Si `true`, traiter l'argument *`InputFile`* de la commande en tant que fichier de texte contenant un nom de fichier par ligne. La valeur par défaut est `false`. (Une alternative est de lister les fichiers sur la CLI avec un espace en tant que séparateur. Veuillez noter, néanmoins, que les CLI ont une limitation de caractères maximum.) Veuillez noter que l'option `--listfile` s'applique uniquement aux arguments, et non pas aux options.
    Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

  - parallel-assessment [pa]

    `--pa | --parallel-assessment =` `true|false`
    Si configuré sur `true`, l'évaluation de validité de schéma est effectuée en parallèle. Cela signifie qu'il y a plus de 128 éléments à tout niveau, ces éléments sont traités en parallèle, en utilisant plusieurs threads. Des fichiers XML très volumineux peuvent donc être traités plus rapidement si cette option est activée. L'évaluation parallèle a lieu un niveau hiérarchique à la fois, mais peut se produire à plusieurs niveaux différents dans une seule infobulle. Veuillez noter que l'évaluation parallèle ne fonctionne pas en mode streaming. C'est pour cette raison que l'option `--streaming` est ignorée si l'évaluation `--parallel-` est définie sur `true`. De même, les besoins en mémoire sont plus élevés lorsque l'option `--parallel-assessment` est utilisée. Le paramètre par défaut est `false`. La forme abrégée pour l'option est `--pa`.
    Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

  - recurse

    `--recurse =` `true|false`
    Utilisé pour sélectionner des fichiers dans le cadre de sous-répertoires, y compris des archives ZIP. Si `true`, l'argument *`InputFile`* de la commande sélectionnera aussi le

fichier spécifié dans les sous-répertoires. Par exemple : `test.zip|zip\test.xml` sélectionnera des fichiers nommés `test.xml` à tous les niveaux de dossier du dossier zip. Les caractères génériques `*` et `?` peuvent être utilisés. Donc, `*.xml` sélectionnera tous les fichiers `.xml` dans le dossier (zip). La valeur par défaut de l'option est `false`.
Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

schema-imports

`--schema-imports =` `load-by-schemalocation | load-preferring-schemalocation | load-by-namespace | load-combining-both | license-namespace-only`
Spécifie le comportement des éléments `xs:import`, chacun d'entre eux ayant un attribut `namespace` optionnel et un attribut `schemaLocation` optionnel : `<import namespace="someNS" schemaLocation="someURL">`. L'option spécifie s'il faut charger un document de schéma ou juste mettre un espace de noms sous licence, et, si un document de schéma doit être chargé, quelle information doit être utilisée pour le trouver. Défaut : `load-preferring-schemalocation`.
Le comportement est le suivant :

- `load-by-schemalocation`: La valeur de l'attribut `schemaLocation` est utilisée pour situer le schéma, en prenant en compte les mappages de catalogue. Si l'attribut de l'espace de nom est présent, l'espace de noms est importé (licencé).
- `load-preferring-schemalocation`: Si l'attribut `schemaLocation` est présent, il est utilisé en prenant en compte les mappages de catalogue. Si l'attribut `schemaLocation` est présent, la valeur de l'attribut `namespace` est utilisée via un mappage de catalogue. C'est la **valeur par défaut**.
- `load-by-namespace`: La valeur de l'attribut `namespace` est utilisée pour situer le schéma via un mappage de catalogue.
- `load-combining-both`: Si soit l'Attribut `namespace` ou l'attribut `schemaLocation` a un mappage de catalogue, le mappage est utilisé. Si les deux ont des mappages de catalogue, alors la valeur de l'option `--schema-mapping` (option XBRL et option XML/XSD) décide de quel mappage utiliser. Si aucun mappage de catalogue n'est présent, l'attribut `schemaLocation` est utilisé.
- `license-namespace-only`: Le nom d'espace est importé. Aucun document de schéma n'est importé.

schemalocation-hints

`--schemalocation-hints =` `load-by-schemalocation | load-by-namespace | load-combining-both | ignore`
Spécifie le comportement des attributs `xsi:schemaLocation` et `xsi:noNamespaceSchemaLocation` : S'il faut charger un document de schéma et, si oui, quelle information doit être utilisée pour la trouver. Défaut : `load-by-schemalocation`.

- La valeur `load-by-schemalocation` utilise l'URL de l'emplacement de schéma dans les attributs `xsi:schemaLocation` et `xsi:noNamespaceSchemaLocation` dans les documents d'instance XML ou XBRL. Il s'agit de la **valeur par défaut**.
- La valeur `load-by-namespace` prend la part d'espace de nom `xsi:schemaLocation` et un string vide dans le cas de `xsi:noNamespaceSchemaLocation` et localise le schéma par le biais d'un mappage de catalogue.

- Si `load-combining-both` est utilisé et si soit la partie espace de noms ou la partie URL a un mappage de catalogue, alors le mappage de catalogue est utilisé. Si tous deux ont des mappages de catalogue, alors la valeur de l'option de `--schema-mapping` (option XBRL et option XML/XSD) de quel mappage utiliser. Si ni l'espace de noms ni l'URL n'a un mappage de catalogue, l'URL est utilisée.
- Si la valeur de l'option est `ignore`, les attributs `xsi:schemaLocation` et `xsi:noNamespaceSchemaLocation` seront ignorés tous les deux.

schema-mapping

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

Si l'emplacement de schéma et l'espace de noms sont tous les deux utilisés pour trouver un document de schéma, spécifie lequel des deux doit être utilisé pendant la consultation du catalogue. (Si soit l'option `--schemalocation-hints` ou l'option `--schema-imports` a une valeur de `load-combining-both`, et si les parties d'espace de noms et d'URL impliquées ont toutes les deux des mappages de catalogue, alors la valeur de cette option spécifie lequel des deux mappages utiliser (mappage d'espace de noms ou mappage URL ; la valeur `prefer-schemalocation` réfère au mappage URL).) Défaut : `prefer-schemalocation`.

script

`--script =` *`FILE`*

Exécute le script Python dans le fichier soumis une fois que la validation a été terminée. Ajouter l'option plusieurs fois pour spécifier plus d'un seul script.

script-api-version

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

Spécifie la version API Python à utiliser pour le script. La valeur par défaut est la dernière version, actuellement **`2.4`**. Au lieu des valeurs `1` et `2`, vous pouvez aussi utiliser les valeurs `1.0` et `2.0`, respectivement.

script-output

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

Rédige la sortie standard du script dans le fichier nommé dans *`FILE`*.

script-param

`--script-param =` *`KEY:VALUE`*

Des paramètres spécifiés par l'utilisateur supplémentaires qui peuvent être accédés au cours de l'exécution des scripts Python. Ajouter l'option plusieurs fois pour spécifier plus d'un paramètre de script.

streaming

`--streaming =` `true|false`

Permet la validation de streaming. La valeur par défaut est `true`. Dans le mode streaming, les données stockées dans la mémoire sont minimisées et le traitement est plus rapide. L'inconvénient est que l'information que peut être requise par conséquent, par exemple, un modèle de données du document d'instance XML —ne sera pas disponible. Dans des situations où cela est significatif, le mode de streaming devra être éteint (en donnant à `--streaming` une valeur de `false`). En utilisant l'option `--script`

 © 2018 Altova GmbH

avec la commande `valxml-withxsd`, désactiver le streaming. Veuillez noter que l'option `--streaming` est ignorée si`--parallel-assessment` est défini sur `true`.
Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- xinclude

  `--xinclude =` `true|false`
  Active la prise en charge de XML Inclusions (XInclude). La valeur par défaut est `false`. Si `false`, les élément `include` d'XInclude sont ignorés.
  Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- xml-mode

  `--xml-mode =` `wf|id|valid`
  Spécifie le mode de traitement XML à utiliser : `wf`=wellformed check (vérification de la bonne forme); `id`=bien formé avec des contrôles ID/IDREF ; `valid`=validation. La valeur par défaut est `wf`. Veuillez noter qu'une valeur de `valid` exige que chaque document de schéma/instance chargé depuis les traitements référence un DTD. Si aucun DTD n'existe, une erreur est rapportée.

- xml-mode-for-schemas

  `--xml-mode-for-schemas =` `wf|id|valid`
  Spécifie le mode de traitement XML à utiliser pour les documents de schéma XML : `wf`=vérification de la bonne formation ; `id`=contrôles de la bonne formation avec ID/IDREF ; `valid`=validation. La valeur par défaut est `wf`. Veuillez noter qu'une valeur de `valid` exige que chaque document de schéma chargé pendant le traitement référence un DTD. Si aucun DTD n'existe, une erreur est rapportée.

- xsd

  `--xsd =` *`FILE`*
  Spécifie un ou plusieurs documents de Schéma XML à utiliser pour la validation des documents d'instance XML. Ajouter l'option plusieurs fois pour spécifier plus d'un document de schéma.

- xsd-version

  `--xsd-version =` `1.0|1.1|detect`
  Spécifie la version du langage de définition de Schéma W3C (XSD) à utiliser. Le défaut est `1.0`. Cette option peut être utile pour découvrir comment un schéma compatible à 1.0 n'est pas compatible à 1.1. L'option `detect` est une fonction spécifique à Altova. Elle permet la détection de la version du document de Schéma XML (`1.0` ou `1.1`) en lisant la valeur de l'attribut `vc:minVersion` de l'élément du document `<xs:schema>`. Si la valeur de l'attribut `@vc:minVersion` est `1.1`, le schéma est détecté comme étant de version `1.1`. Pour toute autre valeur, ou si l'attribut `@vc:minVersion` est absent, le schéma est détecté comme étant de version `1.0`.

- Catalogues et ressources globales
  - catalog

--catalog = *FILE*
Spécifie le chemin absolu vers un fichier de catalogue root qui n'est pas le fichier catalogue root installé. La valeur par défaut est le chemin absolu vers le fichier de catalogue root installé (`<installation-folder>\Altova\RaptorXMLXBRLServer2018\etc\RootCatalog.xml`). Voir la section, Catalogues XML, pour plus d'informations à propos du travail avec les catalogues.

- user-catalog

--user-catalog = *FILE*
Spécifie le chemin absolu vers un catalogue XML à utiliser en plus du catalogue root. Voir la section, Catalogues XML, pour plus d'informations concernant le travail avec des catalogues.

- enable-globalresources

--enable-globalresources = `true|false`
Active les ressources globales. La valeur par défaut est `false`.
Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- globalresourceconfig [gc]

--gc | --globalresourceconfig = *VALUE*
Spécifie la configuration active de la ressource globale (et active les ressources globales).

- globalresourcefile [gr]

--gr | --globalresourcefile = *FILE*
Spécifie le fichier de ressource globale (et active les ressources globales).

- Messages, erreurs, aide, timeout, version
  - error-format

--error-format = `text|shortxml|longxml`
Spécifie le format de la sortie d'erreur. La valeur par défaut est `text`. Les autres options génèrent des formats XML, avec `longxml` générant plus de détails.

  - error-limit

--error-limit = *N*
Spécifie la limite d'erreur avec une plage de valeur de `1` à `999`. La valeur par défaut est `100`. Le traitement s'arrête lorsque la limite d'erreur est atteinte. Utile pour limiter l'utilisation du processeur pendant la validation/transformation.

  - help

--help
Affiche le texte d'aide pour la commande. Par exemple, `valany --h`. (en alternative, la commande `help` peut être utilisée avec un argument. Par exemple : `help valany`.)

  - log-output

--log-output = *FILE*
Écrit la sortie de log dans l'URL de fichier spécifié. Veuillez vous assurer que la CLI a une permission d'écriture dans l'emplacement de la sortie.

- network-timeout

--network-timeout = *VALUE*
Spécifie le timeout en secondes pour les opérations I/O à distance. La valeur par défaut est : `40`.

- verbose

--verbose = `true|false`
Une valeur de `true` permet la sortie d'informations supplémentaires pendant la validation. La valeur par défaut est `false`.
Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- verbose-output

--verbose-output = *FILE*
Écrit la sortie verbeuse sur *FILE*.

- version

--version
Affiche la version de RaptorXML+XBRL Server. Si utilisé avec une commande, placer `--version` avant la commande.

- warning-limit

--warning-limit = *VALUE*
Spécifie la limite d'avertissement dans la plage `1-65535`. Le traitement se poursuit si cette limite a été atteinte, mais d'autres avertissements ne sont pas rapportés. La valeur par défaut est `100`.

valdtd (dtd)

La commande **valdtd | dtd** valide un ou plusieurs documents DTD conformément à la spécification XML 1.0 ou XML 1.1.

Windows	**RaptorXMLXBRL valdtd | dtd [options] *InputFile***
Linux	**raptorxmlxbrl valdtd | dtd [options] *InputFile***
Mac	**raptorxmlxbrl valdtd | dtd [options] *InputFile***

L'argument *InputFile* est le document DTD à valider. Pour valider plusieurs documents, vous devez soit : (i) lister les fichiers à valider sur la CLI, chaque fichier étant séparé du suivant par un espace ; ou (ii) lister les fichiers à valider dans un fichier texte (fichier `.txt`), avec un nom de fichier par ligne, et fournir ce fichier de texte en tant que l'argument *InputFile* avec l'option `--listfile` définie sur `true` (*voir la liste des Options ci-dessous*).

Exemples

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

Casse et lignes obliques sur la ligne de commande

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

* Veuillez noter que les caractères minuscules (`raptorxmlxbrl`) fonctionnent sur toutes les plates-formes (Windows, Linux et Mac), alors qu'une écriture en majuscule/minuscule (`RaptorXMLXBRL`) ne fonctionne que sur Windows et Mac.
* Utiliser les barres obliques penchées vers l'avant pour Linux et Mac, et les barres obliques penchées vers l'arrière sur Windows.

Options

Les options de la commande sont listées ci-dessous, et organisées en groupes. Les valeurs peuvent être spécifiées sans guillemets sauf dans deux cas : (i) lorsque le string de valeur contient des espaces, ou (ii) si la présence de guillemets est explicitement requise dans la description de l'option.

Validation et traitement

listfile

--listfile = `true|false`
Si `true`, traiter l'argument *InputFile* de la commande en tant que fichier de texte

 © 2018 Altova GmbH

contenant un nom de fichier par ligne. La valeur par défaut est `false`. (Une alternative est de lister les fichiers sur la CLI avec un espace en tant que séparateur. Veuillez noter, néanmoins, que les CLI ont une limitation de caractères maximum.) Veuillez noter que l'option `--listfile` s'applique uniquement aux arguments, et non pas aux options.
Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- recurse

  `--recurse =` `true|false`
  Utilisé pour sélectionner des fichiers dans le cadre de sous-répertoires, y compris des archives ZIP. Si `true`, l'argument *`InputFile`* de la commande sélectionnera aussi le fichier spécifié dans les sous-répertoires. Par exemple : `test.zip|zip\test.xml` sélectionnera des fichiers nommés `test.xml` à tous les niveaux de dossier du dossier zip. Les caractères génériques `*` et `?` peuvent être utilisés. Donc, `*.xml` sélectionnera tous les fichiers `.xml` dans le dossier (zip). La valeur par défaut de l'option est `false`.
  Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- script

  `--script =` *`FILE`*
  Exécute le script Python dans le fichier soumis une fois que la validation a été terminée. Ajouter l'option plusieurs fois pour spécifier plus d'un seul script.

- script-api-version

  `--api, --script-api-version =` `1|2|2.1|2.2|2.3|2.4`
  Spécifie la version API Python à utiliser pour le script. La valeur par défaut est la dernière version, actuellement **`2.4`**. Au lieu des valeurs `1` et `2`, vous pouvez aussi utiliser les valeurs `1.0` et `2.0`, respectivement.

- script-output

  `--script-output =` *`FILE`*
  Rédige la sortie standard du script dans le fichier nommé dans *`FILE`*.

- script-param

  `--script-param =` *`KEY:VALUE`*
  Des paramètres spécifiés par l'utilisateur supplémentaires qui peuvent être accédés au cours de l'exécution des scripts Python. Ajouter l'option plusieurs fois pour spécifier plus d'un paramètre de script.

- Catalogues et ressources globales
  - catalog

    `--catalog =` *`FILE`*
    Spécifie le chemin absolu vers un fichier de catalogue root qui n'est pas le fichier catalogue root installé. La valeur par défaut est le chemin absolu vers le fichier de catalogue root installé (`<installation-folder>\Altova\RaptorXMLXBRLServer2018`

`\etc\RootCatalog.xml`). Voir la section, Catalogues XML, pour plus d'informations à propos du travail avec les catalogues.

- user-catalog

  `--user-catalog =` *`FILE`*
  Spécifie le chemin absolu vers un catalogue XML à utiliser en plus du catalogue root. Voir la section, Catalogues XML, pour plus d'informations concernant le travail avec des catalogues.

- enable-globalresources

  `--enable-globalresources =` `true|false`
  Active les ressources globales. La valeur par défaut est `false`.
  Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- globalresourceconfig [gc]

  `--gc | --globalresourceconfig =` *`VALUE`*
  Spécifie la configuration active de la ressource globale (et active les ressources globales).

- globalresourcefile [gr]

  `--gr | --globalresourcefile =` *`FILE`*
  Spécifie le fichier de ressource globale (et active les ressources globales).

- Messages, erreurs, aide, timeout, version
  - error-format

    `--error-format =` `text|shortxml|longxml`
    Spécifie le format de la sortie d'erreur. La valeur par défaut est `text`. Les autres options génèrent des formats XML, avec `longxml` générant plus de détails.

  - error-limit

    `--error-limit =` *`N`*
    Spécifie la limite d'erreur avec une plage de valeur de `1` à `999`. La valeur par défaut est `100`. Le traitement s'arrête lorsque la limite d'erreur est atteinte. Utile pour limiter l'utilisation du processeur pendant la validation/transformation.

  - help

    `--help`
    Affiche le texte d'aide pour la commande. Par exemple, `valany --h`. (en alternative, la commande `help` peut être utilisée avec un argument. Par exemple : `help valany`.)

  - log-output

    `--log-output =` *`FILE`*
    Écrit la sortie de log dans l'URL de fichier spécifié. Veuillez vous assurer que la CLI a une permission d'écriture dans l'emplacement de la sortie.

 © 2018 Altova GmbH

- network-timeout

  `--network-timeout =` *`VALUE`*
  Spécifie le timeout en secondes pour les opérations I/O à distance. La valeur par défaut est : `40`.

- verbose

  `--verbose =` `true|false`
  Une valeur de `true` permet la sortie d'informations supplémentaires pendant la validation. La valeur par défaut est `false`.
  Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- verbose-output

  `--verbose-output =` *`FILE`*
  Écrit la sortie verbeuse sur *`FILE`*.

- version

  `--version`
  Affiche la version de RaptorXML+XBRL Server. Si utilisé avec une commande, placer `--version` avant la commande.

- warning-limit

  `--warning-limit =` *`VALUE`*
  Spécifie la limite d'avertissement dans la plage `1-65535`. Le traitement se poursuit si cette limite a été atteinte, mais d'autres avertissements ne sont pas rapportés. La valeur par défaut est `100`.

valxsd (xsd)

La commande **valxsd | xsd** valide un ou plusieurs documents de Schéma XML (documents XSD) conformément à la spécification de langue de définition de Schéma XML W3C (XSD) 1.0 ou 1.1. Veuillez noter que c'est le schéma lui-même qui est validé par rapport à la spécification de Schéma XML, et non pas un document d'instance XML par rapport à un Schéma XML.

Windows	**RaptorXMLXBRL valxsd | xsd [options] *InputFile***
Linux	**raptorxmlxbrl valxsd | xsd [options] *InputFile***
Mac	**raptorxmlxbrl valxsd | xsd [options] *InputFile***

L'argument *InputFile* est le document de Schéma XML à valider. L'option --xsd-version=1.0|1.1|detect spécifie la version XSD par rapport à laquelle valider, le défaut étant 1.0.

Pour valider plusieurs documents, il faut soit : (i) lister les fichiers à valider sur la CLI, chaque fichier étant séparé du suivant par un espace ; ou (ii) lister les fichiers à valider dans un fichier texte (fichier .txt), avec un nom de fichier par ligne, et fournir ce fichier de texte en tant que l'argument *InputFile* avec l'option --listfile définie sur true (*voir la liste des Options ci-dessous*).

Exemples

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

Casse et lignes obliques sur la ligne de commande

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

* Veuillez noter que les caractères minuscules (raptorxmlxbrl) fonctionnent sur toutes les plates-formes (Windows, Linux et Mac), alors qu'une écriture en majuscule/minuscule (RaptorXMLXBRL) ne fonctionne que sur Windows et Mac.
* Utiliser les barres obliques penchées vers l'avant pour Linux et Mac, et les barres obliques penchées vers l'arrière sur Windows.

Options

Les options de la commande sont listées ci-dessous, et organisées en groupes. Les valeurs peuvent être spécifiées sans guillemets sauf dans deux cas : (i) lorsque le string de valeur contient des espaces, ou (ii) si la présence de guillemets est explicitement requise dans la description de l'option.

Validation et traitement

- ct-restrict-mode

  `--ct-restrict-mode =` `1.0|1.1|default`
  Spécifie comment contrôler les restrictions de type complexes. Une valeur de **`1.0`** contrôles les restrictions de type complexes telles que définies dans la spécification XSD 1.0 — même dans le mode de validation XSD 1.1. Une valeur de **`1.1`** contrôles les restrictions de type complexe telles que définies dans la spécification XSD 1.1 — même dans le mode de validation XSD 1.0. Une valeur de **`default`** contrôle les restrictions de type complexes telles que définies dans la spécification XSD du mode de validation actuel (1.0 ou 1.1). La valeur par défaut est **`default`**.

- listfile

  `--listfile =` `true|false`
  Si `true`, traiter l'argument *`InputFile`* de la commande en tant que fichier de texte contenant un nom de fichier par ligne. La valeur par défaut est `false`. (Une alternative est de lister les fichiers sur la CLI avec un espace en tant que séparateur. Veuillez noter, néanmoins, que les CLI ont une limitation de caractères maximum.) Veuillez noter que l'option `--listfile` s'applique uniquement aux arguments, et non pas aux options.
  Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- recurse

  `--recurse =` `true|false`
  Utilisé pour sélectionner des fichiers dans le cadre de sous-répertoires, y compris des archives ZIP. Si `true`, l'argument *`InputFile`* de la commande sélectionnera aussi le fichier spécifié dans les sous-répertoires. Par exemple : `test.zip|zip\test.xml` sélectionnera des fichiers nommés `test.xml` à tous les niveaux de dossier du dossier zip. Les caractères génériques `*` et `?` peuvent être utilisés. Donc, `*.xml` sélectionnera tous les fichiers `.xml` dans le dossier (zip). La valeur par défaut de l'option est `false`.
  Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- schema-imports

  `--schema-imports =` `load-by-schemalocation | load-preferring-schemalocation | load-by-namespace | load-combining-both | license-namespace-only`
  Spécifie le comportement des éléments `xs:import`, chacun d'entre eux ayant un attribut `namespace` optionnel et un attribut `schemaLocation` optionnel : `<import namespace="someNS" schemaLocation="someURL">`. L'option spécifie s'il faut charger un document de schéma ou juste mettre un espace de noms sous licence, et, si un document de schéma doit être chargé, quelle information doit être utilisée pour le trouver. Défaut : `load-preferring-schemalocation`.
  Le comportement est le suivant :

  - `load-by-schemalocation`: La valeur de l'attribut `schemaLocation` est utilisée pour situer le schéma, en prenant en compte les mappages de catalogue. Si l'attribut de l'espace de nom est présent, l'espace de noms est importé (licencé).
  - `load-preferring-schemalocation`: Si l'attribut `schemaLocation` est présent, il est utilisé en prenant en compte les mappages de catalogue. Si l'attribut

schemaLocation est présent, la valeur de l'attribut namespace est utilisée via un mappage de catalogue. C'est la **valeur par défaut**.

- load-by-namespace: La valeur de l'attribut namespace est utilisée pour situer le schéma via un mappage de catalogue.
- load-combining-both: Si soit l'Attribut namespace ou l'attribut schemaLocation a un mappage de catalogue, le mappage est utilisé. Si les deux ont des mappages de catalogue, alors la valeur de l'option --schema-mapping (option XBRL et option XML/XSD) décide de quel mappage utiliser. Si aucun mappage de catalogue n'est présent, l'attribut schemaLocation est utilisé.
- license-namespace-only: Le nom d'espace est importé. Aucun document de schéma n'est importé.

schemalocation-hints

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

Spécifie le comportement des attributs xsi:schemaLocation et xsi:noNamespaceSchemaLocation : S'il faut charger un document de schéma et, si oui, quelle information doit être utilisée pour la trouver. Défaut : load-by-schemalocation.

- La valeur load-by-schemalocation utilise l'URL de l'emplacement de schéma dans les attributs xsi:schemaLocation et xsi:noNamespaceSchemaLocation dans les documents d'instance XML ou XBRL. Il s'agit de la **valeur par défaut**.
- La valeur load-by-namespace prend la part d'espace de nom xsi:schemaLocation et un string vide dans le cas de xsi:noNamespaceSchemaLocation et localise le schéma par le biais d'un mappage de catalogue.
- Si load-combining-both est utilisé et si soit la partie espace de noms ou la partie URL a un mappage de catalogue, alors le mappage de catalogue est utilisé. Si tous deux ont des mappages de catalogue, alors la valeur de l'option de --schema-mapping (option XBRL et option XML/XSD) de quel mappage utiliser. Si ni l'espace de noms ni l'URL n'a un mappage de catalogue, l'URL est utilisée.
- Si la valeur de l'option est ignore, les attributs xsi:schemaLocation et xsi:noNamespaceSchemaLocation seront ignorés tous les deux.

schema-mapping

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

Si l'emplacement de schéma et l'espace de noms sont tous les deux utilisés pour trouver un document de schéma, spécifie lequel des deux doit être utilisé pendant la consultation du catalogue. (Si soit l'option --schemalocation-hints ou l'option --schema-imports a une valeur de load-combining-both, et si les parties d'espace de noms et d'URL impliquées ont toutes les deux des mappages de catalogue, alors la valeur de cette option spécifie lequel des deux mappages utiliser (mappage d'espace de noms ou mappage URL ; la valeur prefer-schemalocation réfère au mappage URL).) Défaut : prefer-schemalocation.

script

--script = *FILE*

Exécute le script Python dans le fichier soumis une fois que la validation a été terminée. Ajouter l'option plusieurs fois pour spécifier plus d'un seul script.

 © 2018 Altova GmbH

- script-api-version

  --api, --script-api-version = 1|2|2.1|2.2|2.3|2.4
  Spécifie la version API Python à utiliser pour le script. La valeur par défaut est la dernière version, actuellement **2.4**. Au lieu des valeurs 1 et 2, vous pouvez aussi utiliser les valeurs 1.0 et 2.0, respectivement.

- script-api-version

  --api, --script-api-version = 1|2|2.1|2.2|2.3|2.4
  Spécifie la version API Python à utiliser pour le script. La valeur par défaut est la dernière version, actuellement **2.4**. Au lieu des valeurs 1 et 2, vous pouvez aussi utiliser les valeurs 1.0 et 2.0, respectivement.

- script-output

  --script-output = *FILE*
  Rédige la sortie standard du script dans le fichier nommé dans *FILE*.

- script-param

  --script-param = *KEY:VALUE*
  Des paramètres spécifiés par l'utilisateur supplémentaires qui peuvent être accédés au cours de l'exécution des scripts Python. Ajouter l'option plusieurs fois pour spécifier plus d'un paramètre de script.

- xinclude

  --xinclude = true|false
  Active la prise en charge de XML Inclusions (XInclude). La valeur par défaut est false. Si false, les élément include d'XInclude sont ignorés.
  Note : les valeurs d'option booléennes sont configurées sur true si l'option est spécifiée sans une valeur.

- xml-mode

  --xml-mode = wf|id|valid
  Spécifie le mode de traitement XML à utiliser : wf=wellformed check (vérification de la bonne forme); id=bien formé avec des contrôles ID/IDREF ; valid=validation. La valeur par défaut est wf. Veuillez noter qu'une valeur de valid exige que chaque document de schéma/instance chargé depuis les traitements référence un DTD. Si aucun DTD n'existe, une erreur est rapportée.

- xml-mode-for-schemas

  --xml-mode-for-schemas = wf|id|valid
  Spécifie le mode de traitement XML à utiliser pour les documents de schéma XML : wf=vérification de la bonne formation ; id=contrôles de la bonne formation avec ID/IDREF ; valid=validation. La valeur par défaut est wf. Veuillez noter qu'une valeur de valid exige que chaque document de schéma chargé pendant le traitement référence un DTD. Si aucun DTD n'existe, une erreur est rapportée.

- xsd-version

--xsd-version = `1.0|1.1|detect`
Spécifie la version du langage de définition de Schéma W3C (XSD) à utiliser. Le défaut est `1.0`. Cette option peut être utile pour découvrir comment un schéma compatible à 1.0 n'est pas compatible à 1.1. L'option `detect` est une fonction spécifique à Altova. Elle permet la détection de la version du document de Schéma XML (`1.0` ou `1.1`) en lisant la valeur de l'attribut `vc:minVersion` de l'élément du document `<xs:schema>`. Si la valeur de l'attribut `@vc:minVersion` est `1.1`, le schéma est détecté comme étant de version `1.1`. Pour toute autre valeur, ou si l'attribut `@vc:minVersion` est absent, le schéma est détecté comme étant de version `1.0`.

- Catalogues et ressources globales
  - catalog

    --catalog = *FILE*
    Spécifie le chemin absolu vers un fichier de catalogue root qui n'est pas le fichier catalogue root installé. La valeur par défaut est le chemin absolu vers le fichier de catalogue root installé (`<installation-folder>\Altova\RaptorXMLXBRLServer2018\etc\RootCatalog.xml`). Voir la section, Catalogues XML, pour plus d'informations à propos du travail avec les catalogues.

  - user-catalog

    --user-catalog = *FILE*
    Spécifie le chemin absolu vers un catalogue XML à utiliser en plus du catalogue root. Voir la section, Catalogues XML, pour plus d'informations concernant le travail avec des catalogues.

  - enable-globalresources

    --enable-globalresources = `true|false`
    Active les ressources globales. La valeur par défaut est `false`.
    Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

  - globalresourceconfig [gc]

    --gc | --globalresourceconfig = *VALUE*
    Spécifie la configuration active de la ressource globale (et active les ressources globales).

  - globalresourcefile [gr]

    --gr | --globalresourcefile = *FILE*
    Spécifie le fichier de ressource globale (et active les ressources globales).

- Messages, erreurs, aide, timeout, version
  - error-format

    --error-format = `text|shortxml|longxml`
    Spécifie le format de la sortie d'erreur. La valeur par défaut est `text`. Les autres options

génèrent des formats XML, avec `longxml` générant plus de détails.

- error-limit

  `--error-limit =` *`N`*
  Spécifie la limite d'erreur avec une plage de valeur de `1` à `999`. La valeur par défaut est `100`. Le traitement s'arrête lorsque la limite d'erreur est atteinte. Utile pour limiter l'utilisation du processeur pendant la validation/transformation.

- help

  `--help`
  Affiche le texte d'aide pour la commande. Par exemple, `valany --h`. (en alternative, la commande `help` peut être utilisée avec un argument. Par exemple : `help valany`.)

- log-output

  `--log-output =` *`FILE`*
  Écrit la sortie de log dans l'URL de fichier spécifié. Veuillez vous assurer que la CLI a une permission d'écriture dans l'emplacement de la sortie.

- network-timeout

  `--network-timeout =` *`VALUE`*
  Spécifie le timeout en secondes pour les opérations I/O à distance. La valeur par défaut est : `40`.

- verbose

  `--verbose =` `true|false`
  Une valeur de `true` permet la sortie d'informations supplémentaires pendant la validation. La valeur par défaut est `false`.
  Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- verbose-output

  `--verbose-output =` *`FILE`*
  Écrit la sortie verbeuse sur *`FILE`*.

- version

  `--version`
  Affiche la version de RaptorXML+XBRL Server. Si utilisé avec une commande, placer `--version` avant la commande.

- warning-limit

  `--warning-limit =` *`VALUE`*
  Spécifie la limite d'avertissement dans la plage `1-65535`. Le traitement se poursuit si cette limite a été atteinte, mais d'autres avertissements ne sont pas rapportés. La valeur par défaut est `100`.

3.2 Commandes de vérification de la bonne formation

Les commandes de vérification de la bonne formation peuvent être utilisées pour vérifier la bonne formation de documents XML et DTD. Ces commandes sont listées ci-dessous et sont décrites en détails dans les sous-sections de cette section :

wfxml	Vérifie la bonne formation de documents XML.
wfdtd	Vérifie la bonne formation de DTD.
wfany	Vérifie la bonne formation d'un document XML ou de DTD. Le type est détecté automatiquement.

 © 2018 Altova GmbH

wfxml

La commande **wfxml** contrôle la bonne formation d'un ou de plusieurs documents XML conformément à la spécification XML 1.0 ou XML 1.1.

Windows	**RaptorXMLXBRL wfxml [options] *InputFile***
Linux	**raptorxmlxbrl wfxml [options] *InputFile***
Mac	**raptorxmlxbrl wfxml [options] *InputFile***

L'argument *InputFile* est le document XML dont vous souhaitez vérifier la bonne formation. Si vous souhaitez contrôler plusieurs documents il faut soit : (i) lister les fichiers à contrôler sur la CLI, chaque fichier étant séparé du suivant par un espace ; ou (ii) lister les fichiers à contrôler dans un fichier texte (fichier `.txt`), avec un nom de fichier par ligne, et fournir ce fichier de texte en tant que l'argument *InputFile* avec l'option `--listfile` définie sur `true` (*voir la liste des Options ci-dessous*).

Exemples

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

Casse et lignes obliques sur la ligne de commande

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

* Veuillez noter que les caractères minuscules (`raptorxmlxbrl`) fonctionnent sur toutes les plates-formes (Windows, Linux et Mac), alors qu'une écriture en majuscule/minuscule (`RaptorXMLXBRL`) ne fonctionne que sur Windows et Mac.
* Utiliser les barres obliques penchées vers l'avant pour Linux et Mac, et les barres obliques penchées vers l'arrière sur Windows.

Options

Les options de la commande sont listées ci-dessous, et organisées en groupes. Les valeurs peuvent être spécifiées sans guillemets sauf dans deux cas : (i) lorsque le string de valeur contient des espaces, ou (ii) si la présence de guillemets est explicitement requise dans la description de l'option.

Validation et traitement

dtd

--dtd = *FILE*

Spécifie le document DTD externe à utiliser pour la validation. Si une référence à un DTD externe est présente dans le document XM, alors l'option CLI contourne la référence externe.

- listfile

  `--listfile =` `true|false`
  Si `true`, traiter l'argument *`InputFile`* de la commande en tant que fichier de texte contenant un nom de fichier par ligne. La valeur par défaut est `false`. (Une alternative est de lister les fichiers sur la CLI avec un espace en tant que séparateur. Veuillez noter, néanmoins, que les CLI ont une limitation de caractères maximum.) Veuillez noter que l'option `--listfile` s'applique uniquement aux arguments, et non pas aux options.
  Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- namespaces

  `--namespaces =` `true|false`
  Permet le traitement sensible aux espaces de noms. Cela est utile pour contrôler l'instance XML pour des erreurs dues à des espaces de noms incorrectes. La valeur par défaut est `false`.
  Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- recurse

  `--recurse =` `true|false`
  Utilisé pour sélectionner des fichiers dans le cadre de sous-répertoires, y compris des archives ZIP. Si `true`, l'argument *`InputFile`* de la commande sélectionnera aussi le fichier spécifié dans les sous-répertoires. Par exemple : `test.zip|zip\test.xml` sélectionnera des fichiers nommés `test.xml` à tous les niveaux de dossier du dossier zip. Les caractères génériques `*` et `?` peuvent être utilisés. Donc, `*.xml` sélectionnera tous les fichiers `.xml` dans le dossier (zip). La valeur par défaut de l'option est `false`.
  Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- streaming

  `--streaming =` `true|false`
  Permet la validation de streaming. La valeur par défaut est `true`. Dans le mode streaming, les données stockées dans la mémoire sont minimisées et le traitement est plus rapide. L'inconvénient est que l'information que peut être requise par conséquent, par exemple, un modèle de données du document d'instance XML —ne sera pas disponible. Dans des situations où cela est significatif, le mode de streaming devra être éteint (en donnant à `--streaming` une valeur de `false`). En utilisant l'option `--script` avec la commande `valxml-withxsd`, désactiver le streaming. Veuillez noter que l'option `--streaming` est ignorée si`--parallel-assessment` est défini sur `true`.
  Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- Catalogues et ressources globales

- catalog

  `--catalog =` *`FILE`*
  Spécifie le chemin absolu vers un fichier de catalogue root qui n'est pas le fichier catalogue root installé. La valeur par défaut est le chemin absolu vers le fichier de catalogue root installé (`<installation-folder>\Altova\RaptorXMLXBRLServer2018\etc\RootCatalog.xml`). Voir la section, Catalogues XML, pour plus d'informations à propos du travail avec les catalogues.

- user-catalog

  `--user-catalog =` *`FILE`*
  Spécifie le chemin absolu vers un catalogue XML à utiliser en plus du catalogue root. Voir la section, Catalogues XML, pour plus d'informations concernant le travail avec des catalogues.

- enable-globalresources

  `--enable-globalresources =` `true|false`
  Active les ressources globales. La valeur par défaut est `false`.
  Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- globalresourceconfig [gc]

  `--gc | --globalresourceconfig =` *`VALUE`*
  Spécifie la configuration active de la ressource globale (et active les ressources globales).

- globalresourcefile [gr]

  `--gr | --globalresourcefile =` *`FILE`*
  Spécifie le fichier de ressource globale (et active les ressources globales).

- Messages, erreurs, aide, timeout, version

  - error-format

    `--error-format =` `text|shortxml|longxml`
    Spécifie le format de la sortie d'erreur. La valeur par défaut est `text`. Les autres options génèrent des formats XML, avec `longxml` générant plus de détails.

  - error-limit

    `--error-limit =` *`N`*
    Spécifie la limite d'erreur avec une plage de valeur de `1` à `999`. La valeur par défaut est `100`. Le traitement s'arrête lorsque la limite d'erreur est atteinte. Utile pour limiter l'utilisation du processeur pendant la validation/transformation.

  - help

    `--help`
    Affiche le texte d'aide pour la commande. Par exemple, `valany --h`. (en alternative, la commande `help` peut être utilisée avec un argument. Par exemple : `help valany`.)

- log-output

  `--log-output =` *`FILE`*
  Écrit la sortie de log dans l'URL de fichier spécifié. Veuillez vous assurer que la CLI a une permission d'écriture dans l'emplacement de la sortie.

- network-timeout

  `--network-timeout =` *`VALUE`*
  Spécifie le timeout en secondes pour les opérations I/O à distance. La valeur par défaut est : `40`.

- verbose

  `--verbose =` `true|false`
  Une valeur de `true` permet la sortie d'informations supplémentaires pendant la validation. La valeur par défaut est `false`.
  Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- verbose-output

  `--verbose-output =` *`FILE`*
  Écrit la sortie verbeuse sur *`FILE`*.

- version

  `--version`
  Affiche la version de RaptorXML+XBRL Server. Si utilisé avec une commande, placer `--version` avant la commande.

- warning-limit

  `--warning-limit =` *`VALUE`*
  Spécifie la limite d'avertissement dans la plage `1-65535`. Le traitement se poursuit si cette limite a été atteinte, mais d'autres avertissements ne sont pas rapportés. La valeur par défaut est `100`.

 © 2018 Altova GmbH

wfdtd

La commande **wfdtd** contrôle la bonne formation d'un ou de plusieurs documents DTD conformément à la spécification XML 1.0 ou XML 1.1.

Windows	**RaptorXMLXBRL wfdtd [options] *InputFile***
Linux	**raptorxmlxbrl wfdtd [options] *InputFile***
Mac	**raptorxmlxbrl wfdtd [options] *InputFile***

L'argument *InputFile* est le document DTD dont vous souhaitez vérifier la bonne formation. Si vous souhaitez contrôler plusieurs documents il faut soit : (i) lister les fichiers à contrôler sur la CLI, chaque fichier étant séparé du suivant par un espace ; ou (ii) lister les fichiers à contrôler dans un fichier texte (fichier `.txt`), avec un nom de fichier par ligne, et fournir ce fichier de texte en tant que l'argument *InputFile* avec l'option `--listfile` définie sur `true` (*voir la liste des Options ci-dessous*).

Exemples

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

Casse et lignes obliques sur la ligne de commande

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

* Veuillez noter que les caractères minuscules (`raptorxmlxbrl`) fonctionnent sur toutes les plates-formes (Windows, Linux et Mac), alors qu'une écriture en majuscule/minuscule (`RaptorXMLXBRL`) ne fonctionne que sur Windows et Mac.
* Utiliser les barres obliques penchées vers l'avant pour Linux et Mac, et les barres obliques penchées vers l'arrière sur Windows.

Options

Les options de la commande sont listées ci-dessous, et organisées en groupes. Les valeurs peuvent être spécifiées sans guillemets sauf dans deux cas : (i) lorsque le string de valeur contient des espaces, ou (ii) si la présence de guillemets est explicitement requise dans la description de l'option.

Validation et traitement

listfile

--listfile = `true|false`

Si `true`, traiter l'argument *`InputFile`* de la commande en tant que fichier de texte contenant un nom de fichier par ligne. La valeur par défaut est `false`. (Une alternative est de lister les fichiers sur la CLI avec un espace en tant que séparateur. Veuillez noter, néanmoins, que les CLI ont une limitation de caractères maximum.) Veuillez noter que l'option `--listfile` s'applique uniquement aux arguments, et non pas aux options.
Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- recurse

  `--recurse =` `true|false`
  Utilisé pour sélectionner des fichiers dans le cadre de sous-répertoires, y compris des archives ZIP. Si `true`, l'argument *`InputFile`* de la commande sélectionnera aussi le fichier spécifié dans les sous-répertoires. Par exemple : `test.zip|zip\test.xml` sélectionnera des fichiers nommés `test.xml` à tous les niveaux de dossier du dossier zip. Les caractères génériques `*` et `?` peuvent être utilisés. Donc, `*.xml` sélectionnera tous les fichiers `.xml` dans le dossier (zip). La valeur par défaut de l'option est `false`.
  Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- Catalogues et ressources globales
  - catalog

    `--catalog =` *`FILE`*
    Spécifie le chemin absolu vers un fichier de catalogue root qui n'est pas le fichier catalogue root installé. La valeur par défaut est le chemin absolu vers le fichier de catalogue root installé (`<installation-folder>\Altova\RaptorXMLXBRLServer2018\etc\RootCatalog.xml`). Voir la section, Catalogues XML, pour plus d'informations à propos du travail avec les catalogues.

  - user-catalog

    `--user-catalog =` *`FILE`*
    Spécifie le chemin absolu vers un catalogue XML à utiliser en plus du catalogue root. Voir la section, Catalogues XML, pour plus d'informations concernant le travail avec des catalogues.

  - enable-globalresources

    `--enable-globalresources =` `true|false`
    Active les ressources globales. La valeur par défaut est `false`.
    Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

  - globalresourceconfig [gc]

    `--gc | --globalresourceconfig =` *`VALUE`*
    Spécifie la configuration active de la ressource globale (et active les ressources globales).

  - globalresourcefile [gr]

--gr | --globalresourcefile = *FILE*
Spécifie le fichier de ressource globale (et active les ressources globales).

- Messages, erreurs, aide, timeout, version
  - error-format

    --error-format = `text|shortxml|longxml`
    Spécifie le format de la sortie d'erreur. La valeur par défaut est `text`. Les autres options génèrent des formats XML, avec `longxml` générant plus de détails.

  - error-limit

    --error-limit = *N*
    Spécifie la limite d'erreur avec une plage de valeur de `1` à `999`. La valeur par défaut est `100`. Le traitement s'arrête lorsque la limite d'erreur est atteinte. Utile pour limiter l'utilisation du processeur pendant la validation/transformation.

  - help

    --help
    Affiche le texte d'aide pour la commande. Par exemple, `valany --h`. (en alternative, la commande `help` peut être utilisée avec un argument. Par exemple : `help valany`.)

  - log-output

    --log-output = *FILE*
    Écrit la sortie de log dans l'URL de fichier spécifié. Veuillez vous assurer que la CLI a une permission d'écriture dans l'emplacement de la sortie.

  - network-timeout

    --network-timeout = *VALUE*
    Spécifie le timeout en secondes pour les opérations I/O à distance. La valeur par défaut est : `40`.

  - verbose

    --verbose = `true|false`
    Une valeur de `true` permet la sortie d'informations supplémentaires pendant la validation. La valeur par défaut est `false`.
    Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

  - verbose-output

    --verbose-output = *FILE*
    Écrit la sortie verbeuse sur *FILE*.

  - version

    --version
    Affiche la version de RaptorXML+XBRL Server. Si utilisé avec une commande, placer `--version` avant la commande.

- warning-limit

  `--warning-limit =` *`VALUE`*

  Spécifie la limite d'avertissement dans la plage `1-65535`. Le traitement se poursuit si cette limite a été atteinte, mais d'autres avertissements ne sont pas rapportés. La valeur par défaut est `100`.

© 2018 Altova GmbH

wfany

La commande **wfany** contrôle la bonne formation d'un document XML ou DTD conformément à la spécification respective. Le type de document est détecté automatiquement.

Windows	**RaptorXMLXBRL wfany [options]** ***InputFile***
Linux	**raptorxmlxbrl wfany [options]** ***InputFile***
Mac	**raptorxmlxbrl wfany [options]** ***InputFile***

L'argument *InputFile* est le document dont vous souhaitez vérifier la bonne formation. Veuillez noter qu'un seul document peut être soumis en tant que l'argument de la commande. Le type du document soumis est détecté automatiquement.

Exemples

- **raptorxmlxbrl** wfany c:\Test.xml
- **raptorxmlxbrl** wfany --error-format=text c:\Test.xml

Casse et lignes obliques sur la ligne de commande

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

* Veuillez noter que les caractères minuscules (raptorxmlxbrl) fonctionnent sur toutes les plates-formes (Windows, Linux et Mac), alors qu'une écriture en majuscule/minuscule (RaptorXMLXBRL) ne fonctionne que sur Windows et Mac.
* Utiliser les barres obliques penchées vers l'avant pour Linux et Mac, et les barres obliques penchées vers l'arrière sur Windows.

Options

Les options de la commande sont listées ci-dessous, et organisées en groupes. Les valeurs peuvent être spécifiées sans guillemets sauf dans deux cas : (i) lorsque le string de valeur contient des espaces, ou (ii) si la présence de guillemets est explicitement requise dans la description de l'option.

3.3 Commandes de validation XBRL

Les commandes de validation peuvent être utilisées pour valider des documents d'instance XBRL et des taxonomies XBRL conformément aux spécifications XBRL 2.1, Dimensions 1.0 et Formula 1.0. Les commandes disponibles sont listées ci-dessous et sont décrites en détails dans les sous-sections de cette section :

valxbrl | xbrl	Valide un document d'instance XBRL (extension `.xbrl`).
valinlinexbrl (ixbrl)	Valide un ou plusieurs documents Inline XBRL (iXBRL) conformément à la spécification Inline XBRL 1.0 ou Inline XBRL 1.1
valxbrltaxonomy | dts	Valide un document de taxonomie XBRL (schéma) (extension `.xsd`).
valtaxonomypackage | taxpkg	Valide un ou plusieurs packs de taxonomie XBRL conformément à la spécification des packs de taxonomie 1.0.
valxbrlver	Valide un ou plusieurs rapports de version XBRL conformément à la spécification de version XBRL 1.0.

 © 2018 Altova GmbH

valxbrl (xbrl)

La commande **valxbrl | xbrl** valide un ou plusieurs documents d'instance XBRL conformément aux spécifications XBRL 2.1, Dimensions 1.0 et Formula 1.0.

Windows	**RaptorXMLXBRL valxbrl | xbrl [options] *InputFile***
Linux	**raptorxmlxbrl valxbrl | xbrl [options] *InputFile***
Mac	**raptorxmlxbrl valxbrl | xbrl [options] *InputFile***

L'argument *InputFile* est le document d'instance XBRL à valider. Pour valider plusieurs documents, il faut soit : (i) lister les fichiers à valider sur la CLI, chaque fichier étant séparé du suivant par un espace ; ou (ii) lister les fichiers à valider dans un fichier texte (fichier `.txt`), avec un nom de fichier par ligne, et fournir ce fichier de texte en tant que l'argument *InputFile* avec l'option `--listfile` définie sur `true` (*voir la liste des Options ci-dessous*).

Note : Le document d'instance XBRL ne doit pas être imbriqué dans un autre document XML et doit posséder l'élément `xbrl` en tant que son élément racine.
```
<xbrl xmlns="http://www.xbrl.org/2003/instance"> ... </xbrl>
```

Validation EDGAR

EDGAR (Electronic Data Gathering, Analysis, and Retrieval) est un système qui effectue une collecte, validation et indexage automatisé de déclarations financières déposées par des entreprises auprès de la United States SEC (Securities and Exchange Commission). raptorxmlxbrl prend en charge la validation EDGAR par le biais d'API Python de Raptor. Pour exécuter une validation EDGAR sur un fichier d'instance XBRL, utiliser l'option `--script` pour exécuter le script Python de validation EDGAR. Dans raptorxmlxbrl, ce script, **efm-validation.py**, se trouve dans le dossier `etc\scripts\sec-edgar-tools` du dossier d'application :

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

Exemples

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

▼ Casse et lignes obliques sur la ligne de commande

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

* Veuillez noter que les caractères minuscules (raptorxmlxbrl) fonctionnent sur toutes les plates-formes (Windows, Linux et Mac), alors qu'une écriture en majuscule/minuscule (RaptorXMLXBRL) ne fonctionne que sur Windows et Mac.
* Utiliser les barres obliques penchées vers l'avant pour Linux et Mac, et les barres obliques penchées vers l'arrière sur Windows.

Options

Les options de la commande sont listées ci-dessous, et organisées en groupes. Les valeurs peuvent être spécifiées sans guillemets sauf dans deux cas : (i) lorsque le string de valeur contient des espaces, ou (ii) si la présence de guillemets est explicitement requise dans la description de l'option.

- XBRL validation et traitement
  - additional-dts-entry-point

    --additional-dts-entry-point = *FILE*
    Force le fichier de point d'entrée spécifié à être inclus dans le processus de découverte de DTS. Ajouter l'option plusieurs fois pour spécifier plus d'un point d'entrée.

  - ct-restrict-mode

    --ct-restrict-mode = 1.0|1.1|default
    Spécifie comment contrôler les restrictions de type complexes. Une valeur de **1.0** contrôles les restrictions de type complexes telles que définies dans la spécification XSD 1.0 — même dans le mode de validation XSD 1.1. Une valeur de **1.1** contrôles les restrictions de type complexe telles que définies dans la spécification XSD 1.1 — même dans le mode de validation XSD 1.0. Une valeur de **default** contrôle les restrictions de type complexes telles que définies dans la spécification XSD du mode de validation actuel (1.0 ou 1.1). La valeur par défaut est **default**.

  - dimensions

    --dimensions = true|false
    Active les extensions XBRL Dimension 1.0. La valeur par défaut est true.
    Note : les valeurs d'option booléennes sont configurées sur true si l'option est spécifiée sans une valeur.

  - dts

    --dts = *FILE*
    Précharge le point d'entrée DTS indiqué dans *FILE*, et l'utilise pour la validation des fichiers d'instance. Ajouter l'option plusieurs fois pour spécifier plus d'un seul point d'entrée. Les points d'entrée dans les fichiers d'instance seront ignorés. La commande est utilisée pour valider en bloc plusieurs fichiers d'instance qui réfèrent tous à la même taxonomie. L'option --dts instruit le moteur Raptor à charger le DTS une seule fois — et non pas pour chaque instance dans la liste de fichier. (Une liste de fichier est donnée

soit dans la CLI, soit dans un fichier de texte qui est indiqué en tant que liste de fichier en plaçant l'option `--listfile` sur `true`.) Si un fichier d'instance dans la liste de fichier référence une taxonomie différente, un avertissement sera émis. Cette commande peut accélérer considérablement le temps de validation lors de la validation de nombreux petits fichiers d'instance ayant chacun un grand DTS de support ; elle n'apporte aucun avantage lors de la validation d'un fichier d'instance unique.

- essence-alias-checks

  `--essence-alias-checks =` `true|false`
  Active le rapport des incohérences essence-alias. La valeur par défaut est `true`.
  Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- extensible-enumerations

  `--extensible-enumerations =` `true|false`
  Si `true`, active l'extension XBRL Extensible Enumerations 1.0. La valeur par défaut est : `true`.

- inconsistencies-limit

  `--inconsistencies-limit =` *`VALUE`*
  Spécifie la limite d'inconsistances XBRL, avec des valeurs allant entre `1-65535`. Le traitement continue lorsque la limite est atteinte mais les autres inconsistances ne sont pas rapportées. La valeur par défaut est `100`.

- listfile

  `--listfile =` `true|false`
  Si `true`, traiter l'argument *`InputFile`* de la commande en tant que fichier de texte contenant un nom de fichier par ligne. La valeur par défaut est `false`. (Une alternative est de lister les fichiers sur la CLI avec un espace en tant que séparateur. Veuillez noter, néanmoins, que les CLI ont une limitation de caractères maximum.) Veuillez noter que l'option `--listfile` s'applique uniquement aux arguments, et non pas aux options.
  Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- parallel-assessment [pa]

  `--pa | --parallel-assessment =` `true|false`
  Si configuré sur `true`, l'évaluation de validité de schéma est effectuée en parallèle. Cela signifie qu'il y a plus de 128 éléments à tout niveau, ces éléments sont traités en parallèle, en utilisant plusieurs threads. Des fichiers XML très volumineux peuvent donc être traités plus rapidement si cette option est activée. L'évaluation parallèle a lieu un niveau hiérarchique à la fois, mais peut se produire à plusieurs niveaux différents dans une seule infobulle. Veuillez noter que l'évaluation parallèle ne fonctionne pas en mode streaming. C'est pour cette raison que l'option `--streaming` est ignorée si l'évaluation `--parallel-` est définie sur `true`. De même, les besoins en mémoire sont plus élevés lorsque l'option `--parallel-assessment` est utilisée. Le paramètre par défaut est `false`. La forme abrégée pour l'option est `--pa`.
  Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est

spécifiée sans une valeur.

- preload-xbrl-schemas

  `--preload-xbrl-schemas =` `true|false`
  Précharge les schémas de la spécification XBRL 2.1. La valeur par défaut est `true`.
  Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- recurse

  `--recurse =` `true|false`
  Utilisé pour sélectionner des fichiers dans le cadre de sous-répertoires, y compris des archives ZIP. Si `true`, l'argument *`InputFile`* de la commande sélectionnera aussi le fichier spécifié dans les sous-répertoires. Par exemple : `test.zip|zip\test.xml` sélectionnera des fichiers nommés `test.xml` à tous les niveaux de dossier du dossier zip. Les caractères génériques `*` et `?` peuvent être utilisés. Donc, `*.xml` sélectionnera tous les fichiers `.xml` dans le dossier (zip). La valeur par défaut de l'option est `false`.
  Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- requires-element-checks

  `--requires-element-checks =` `true|false`
  Active le rapport des incohérences requires-element. La valeur par défaut est `true`.
  Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- schema-imports

  `--schema-imports =` `load-by-schemalocation | load-preferring-schemalocation | load-by-namespace | load-combining-both | license-namespace-only`
  Spécifie le comportement des éléments `xs:import`, chacun d'entre eux ayant un attribut `namespace` optionnel et un attribut `schemaLocation` optionnel : `<import namespace="someNS" schemaLocation="someURL">`. L'option spécifie s'il faut charger un document de schéma ou juste mettre un espace de noms sous licence, et, si un document de schéma doit être chargé, quelle information doit être utilisée pour le trouver. Défaut : `load-preferring-schemalocation`.
  Le comportement est le suivant :

  - `load-by-schemalocation`: La valeur de l'attribut `schemaLocation` est utilisée pour situer le schéma, en prenant en compte les mappages de catalogue. Si l'attribut de l'espace de nom est présent, l'espace de noms est importé (licencé).
  - `load-preferring-schemalocation`: Si l'attribut `schemaLocation` est présent, il est utilisé en prenant en compte les mappages de catalogue. Si l'attribut `schemaLocation` est présent, la valeur de l'attribut `namespace` est utilisée via un mappage de catalogue. C'est la **valeur par défaut**.
  - `load-by-namespace`: La valeur de l'attribut `namespace` est utilisée pour situer le schéma via un mappage de catalogue.
  - `load-combining-both`: Si soit l'Attribut `namespace` ou l'attribut `schemaLocation` a un mappage de catalogue, le mappage est utilisé. Si les deux ont des mappages de catalogue, alors la valeur de l'option `--schema-mapping` (option XBRL et option

XML/XSD) décide de quel mappage utiliser. Si aucun mappage de catalogue n'est présent, l'attribut `schemaLocation` est utilisé.
- `license-namespace-only`: Le nom d'espace est importé. Aucun document de schéma n'est importé.

schema-mapping

`--schema-mapping =` `prefer-schemalocation | prefer-namespace`
Si l'emplacement de schéma et l'espace de noms sont tous les deux utilisés pour trouver un document de schéma, spécifie lequel des deux doit être utilisé pendant la consultation du catalogue. (Si soit l'option `--schemalocation-hints` ou l'option `--schema-imports` a une valeur de `load-combining-both`, et si les parties d'espace de noms et d'URL impliquées ont toutes les deux des mappages de catalogue, alors la valeur de cette option spécifie lequel des deux mappages utiliser (mappage d'espace de noms ou mappage URL ; la valeur `prefer-schemalocation` réfère au mappage URL).) Défaut : `prefer-schemalocation`.

schemalocation-hints

`--schemalocation-hints =` `load-by-schemalocation | load-by-namespace | load-combining-both | ignore`
Spécifie le comportement des attributs `xsi:schemaLocation` et `xsi:noNamespaceSchemaLocation` : S'il faut charger un document de schéma et, si oui, quelle information doit être utilisée pour la trouver. Défaut : `load-by-schemalocation`.

- La valeur `load-by-schemalocation` utilise l'URL de l'emplacement de schéma dans les attributs `xsi:schemaLocation` et `xsi:noNamespaceSchemaLocation` dans les documents d'instance XML ou XBRL. Il s'agit de la **valeur par défaut**.
- La valeur `load-by-namespace` prend la part d'espace de nom `xsi:schemaLocation` et un string vide dans le cas de `xsi:noNamespaceSchemaLocation` et localise le schéma par le biais d'un mappage de catalogue.
- Si `load-combining-both` est utilisé et si soit la partie espace de noms ou la partie URL a un mappage de catalogue, alors le mappage de catalogue est utilisé. Si tous deux ont des mappages de catalogue, alors la valeur de l'option de `--schema-mapping` (option XBRL et option XML/XSD) de quel mappage utiliser. Si ni l'espace de noms ni l'URL n'a un mappage de catalogue, l'URL est utilisée.
- Si la valeur de l'option est `ignore`, les attributs `xsi:schemaLocation` et `xsi:noNamespaceSchemaLocation` seront ignorés tous les deux.

script

`--script =` *`FILE`*
Exécute le script Python dans le fichier soumis une fois que la validation a été terminée. Ajouter l'option plusieurs fois pour spécifier plus d'un seul script.

script-api-version

`--api, --script-api-version =` `1|2|2.1|2.2|2.3|2.4`
Spécifie la version API Python à utiliser pour le script. La valeur par défaut est la dernière version, actuellement **`2.4`**. Au lieu des valeurs `1` et `2`, vous pouvez aussi utiliser les valeurs `1.0` et `2.0`, respectivement.

- script-output

  --script-output = *FILE*
  Rédige la sortie standard du script dans le fichier nommé dans *FILE*.

- script-param

  --script-param = *KEY:VALUE*
  Des paramètres spécifiés par l'utilisateur supplémentaires qui peuvent être accédés au cours de l'exécution des scripts Python. Ajouter l'option plusieurs fois pour spécifier plus d'un paramètre de script.

- summation-item-checks

  --summation-item-checks = `true|false`
  Active le rapport des incohérences summation-item. La valeur par défaut est `true`.
  Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- taxonomy-package

  --taxonomy-package = *FILE*
  Spécifie le chemin absolu vers un pack de taxonomie supplémentaire tel que décrit dans les ébauches de travail Taxonomy Package 1.0 et Taxonomy Packages 1.0. La valeur de FILE donne l'emplacement du pack de taxonomie. Ajouter l'option plusieurs fois pour spécifier plus d'un pack de taxonomie.

- taxonomy-packages-config-file

  --taxonomy-packages-config-file = *FILE*
  Spécifie le chemin vers le fichier **TaxonomyPackagesConfig.json**, qui est utilisé pour charger les packs de taxonomie XBRL. L'avantage du fichier est qu'il sert en tant que catalogue des packs de taxonomie. La structure des fichiers JSON est tel que dans l'exemple indiqué ci-dessous. La valeur de la clé **uri** donne l'emplacement du pack. La clé **active** (d'un ensemble de packs) bascule lors de l'utilisation de ces packs.

```
{"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`
  Entraîne l'échec de la validation XBRL si le fichier contient des inconsistances telles que définies par la spécification XBRL 2.1. La valeur par défaut est `false`.
  Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

 © 2018 Altova GmbH

- utr

  --utr = `true|false`
  Si `true`, active l'extension XBRL Unit Registry 1.0. La valeur par défaut est : `false`.

- utr-file

  --utr-file = *FILE*
  Spécifie le chemin menant au fichier Units Registry. Le chemin par défaut est : **http://www.xbrl.org/utr/utr.xml**.

- utr-status

  --utr-status = `PWD|CR|REC|NIE|PROPOSED|IWD|ACK|RR`
  Active la prise en charge des définitions d'unité avec le statut donné dans le Units Registry. Ajouter l'option plusieurs fois pour spécifier plus d'un statut d'unité. Le défaut est **REC**.

- validate-dts-only

  --validate-dts-only = `true|false`
  Le DTS est découvert en démarrant depuis le document d'instance XBRL. Tous les schémas de taxonomie référencés et bases de lien sont découverts et validés. Le reste du document d'instance XBRL est ignoré. La valeur par défaut est `false`.
  Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- xinclude

  --xinclude = `true|false`
  Active la prise en charge de XML Inclusions (XInclude). La valeur par défaut est `false`. Si `false`, les élément `include` d'XInclude sont ignorés.
  Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- XBRL formules et assertions
  - assertion-severity

    --assertion-severity = `true|false`
    Active l'extension Assertion Severity 1.0. La valeur par défaut est `true`.
    Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

  - assertions-output

    --assertions-output = *FILE*
    Écrit la sortie de l'évaluation d'assertion sur le *FILE* spécifié. Si défini, spécifie automatiquement `--formula-execution=true`.

  - assertions-output-format

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

Spécifie le format de sortie de l'évaluation d'assertion. La valeur par défaut est `json`.

- evaluate-referenced-parameters-only

  `--evaluate-referenced-parameters-only =` `true|false`
  Si `false`, force l'évaluation de tous les paramètres, même s'ils ne sont pas référencés par d'autres formules/assertions/tables. La valeur par défaut est : `true`.

- formula

  `--formula =` `true|false`
  Active l'extension XBRL Formula 1.0. La valeur par défaut est `true`.
  Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- formula-assertion-set [[DEPRECATED]]

  `--formula-assertion-set =` *`VALUE`*
  Limite l'exécution de la formule au seul ensemble d'assertion donné. Ajouter l'option plusieurs fois pour spécifier plus d'un ensemble d'assertion. La forme abrégée est `--as`. La ***`VALUE`*** est soit la valeur de l'attribut `@id`, soit une URI avec un fragment XPointer qui identifie la ressource. Les valeurs spéciales `##none` et `##all` peuvent aussi être utilisées.

- formula-execution

  `--formula-execution =` `true|false`
  Active l'extension des formules XBRL. La valeur par défaut est `true`. Si `true`, spécifie automatiquement `--formula=true`.
  Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- formula-output

  `--formula-output =` *`FILE`*
  Écrit la sortie de l'évaluation de la formule pour le *`FILE`* spécifié. Si défini, spécifie automatiquement `--formula-execution=true`.

- formula-parameters

  `--formula-parameters =` *`JSON-ARRAY`*
  Spécifie les paramètres pour l'évaluation de la formule XBRL en tant qu'un array de mappages JSON directement sur la CLI. Voir la section Paramètres de formule pour plus de détails.

- formula-parameters-file

  `--formula-parameters-file =` *`FILE`*
  Spécifie un *`FILE`* contenant les paramètres pour l'évaluation de formule XBRL. Le fichier peut être soit un fichier XML soit un fichier JSON. Voir la section Paramètres de formule.

- ignore-assertion

  `--ignore-assertion =` *`VALUE`*

Exclut l'assertion donnée de l'exécution. Ajouter cette option plusieurs fois pour spécifier plus d'une assertion.

- ignore-assertions-file

  `--ignore-assertions-file =` *`FILE`*
  Spécifie un chemin vers un fichier qui contient une liste de IDs/XPointers d'assertions à exclure du traitement. Dans le fichier, saisir une assertion par ligne.

- ignore-formula

  `--ignore-formula =` *`VALUE`*
  Exclut la formule donnée de l'exécution. Ajouter cette option plusieurs fois pour spécifier plus d'une formule.

- ignore-formulas-file

  `--ignore-formulas-file =` *`FILE`*
  Spécifie un chemin vers un fichier qui contient une liste de IDs/XPointers de formules à exclure du traitement. Dans le fichier, saisir une formule par ligne.

- message-lang

  `--message-lang =` *`VALUE`*
  Spécifie la langue à utiliser lors de l'affichage des messages de validation. La valeur par défaut est : `en`. Les autres valeurs acceptées sont `de`, `es`, `fr`, `ja`, pour, respectivement, l'allemand, l'espagnol, le français et le japonais.

- message-role

  `--message-role =` *`VALUE`*
  Spécifie le rôle de message préféré à utiliser lors de l'affichage des messages de validation. La valeur par défaut est : `http://www.xbrl.org/2010/role/message`.

- preload-formula-schemas

  `--preload-formula-schemas =` `true|false`
  Précharge les schémas de la spécification XBRL Formula 1.0. La valeur par défaut est `false`.
  Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- process-assertion [a]

  `--a | --process-assertion =` *`VALUE`*
  Limite l'exécution de formule à la seule assertion donnée. Ajouter l'option plusieurs fois pour spécifier plus d'une assertion. Forme abrégée est `--a`. La ***`VALUE`*** est soit la valeur de l'attribut `@id`, soit un URI avec un fragment XPointer qui identifie la ressource. Les valeurs spéciales `##none` et `##all` peuvent aussi être utilisées.

- process-assertion-set [as]

  `--as | --process-assertion-set =` *`VALUE`*
  Limite l'exécution de formule au seul ensemble d'assertion donné. Ajouter l'option plusieurs fois pour spécifier plus d'un ensemble d'assertion. Forme abrégée est `--as`. La ***`VALUE`*** est soit la valeur de l'attribut `@id`, soit un URI avec un fragment XPointer qui

identifie la ressource. Les valeurs spéciales `##none` et `##all` peuvent aussi être utilisées.

- process-assertions-file

  `--process-assertions-file =` *`FILE`*
  Spécifie un chemin vers un fichier qui contient une liste de IDs/XPointers d'assertions à exécuter. Dans le fichier, saisir une assertion par ligne.

- process-formula [f]

  `--f | --process-formula =` *`VALUE`*
  Limite l'exécution de formule à la seule formule donnée. Ajouter l'option plusieurs fois pour spécifier plus d'une formule. Forme abrégée est `--f`. La ***VALUE*** est soit la valeur de l'attribut `@id`, soit un URI avec un fragment XPointer qui identifie la ressource. Les valeurs spéciales `##none` et `##all` peuvent aussi être utilisées.

- process-formulas-file

  `--process-formulas-file =` *`FILE`*
  Spécifie un chemin vers un fichier qui contient une liste de IDs/XPointers de formules à exécuter. Dans le fichier, saisir une formule par ligne.

- report-unsatisfied-assertion-evaluations

  `--report-unsatisfied-assertion-evaluations =` `true|false`
  Rapporte l'évaluation de l'assertion non satisfaite en tant qu'erreurs ou avertissements selon le degré de sévérité de l'assertion attribuée. La valeur par défaut est `false`.
  Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- variableset-execution-timeout

  `--variableset-execution-timeout =` *`VALUE`*
  Appliqué lors de l'exécution de formules (**`--formula-execution=true`**). Spécifie le temps maximum autorisé pour l'exécution d'un ensemble de variables unique (une formule ou une valeur, ou une existence ou une assertion de consistance). Le temps est spécifié en minutes et doit être un nombre positif. La valeur par défaut est 30min. Si un ensemble de variables particulier ne termine pas l'exécution avant que le timeout n'ait été atteint, il sera abandonné. Un message d'erreur est affiché (et saisi dans le journal verbeux). Veuillez noter, néanmoins, que le contrôle de timeout est uniquement effectué après l'évaluation de chaque ensemble de variable ; et non pas pendant l'exécution des expressions XPath individuelles. Donc, si une seule expression XPath prend beaucoup de temps à être exécutée, la limite de timeout peut avoir été dépassée. L'exécution d'un ensemble de variable est abandonnée uniquement une fois que l'évaluation du set de variable complet ait été exécuté.

- Tables XBRL

  - concept-label-linkrole

    `--concept-label-linkrole =` *`VALUE`*
    Spécifie le rôle de lien étendu préféré à utiliser lors du rendu de libellés de concept.

- concept-label-role

  --cconcept-label-role = *VALUE*
  Spécifie le rôle de lien préféré à utiliser lors du rendu de libellés de concept. Par défaut : `http://www.xbrl.org/2003/role/label`.

- evaluate-referenced-parameters-only

  --evaluate-referenced-parameters-only = `true|false`
  Si `false`, force l'évaluation de tous les paramètres, même s'ils ne sont pas référencés par d'autres formules/assertions/tables. La valeur par défaut est : `true`.

- generic-label-linkrole

  --generic-label-linkrole = *VALUE*
  Spécifie le rôle de lien étendu préféré à utiliser lors du rendu des libellés génériques.

- generic-label-role

  --generic-label-role = *VALUE*
  Spécifie le rôle de libellé préféré à utiliser lors du rendu des libellés génériques. La valeur par défaut est : http://www.xbrl.org/2003/role/label.

- label-lang

  --label-lang = *VALUE*
  Spécifie la langue préférée à utiliser lors du rendu des libellés. La valeur par défaut est : `en`.

- preload-table-schemas

  --preload-table-schemas = `true|false`
  Précharge les schémas de la spécification XBRL Table 1.0. La valeur par défaut est `false`.
  Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- process-table [t]

  --t | --process-table = *VALUE*
  Limite l'exécution de formule à la seule table donnée. Ajouter l'option plusieurs fois pour spécifier plus d'une table. Forme abrégée est `--t`. La ***VALUE*** est soit la valeur de l'attribut `@id`, soit un URI avec un fragment XPointer qui identifie la ressource. Les valeurs spéciales `##none` et `##all` peuvent aussi être utilisées.

- table

  --table = `true|false`
  Active l'extension XBRL Table 1.0. La valeur par défaut est `true`. Si `true`, spécifie automatiquement `--formula=true` et `--dimensions=true`.
  Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- table-elimination

--table-elimination = true|false
Active l'élimination de lignes/colonnes vides dans la sortie HTML. La valeur par défaut est true.
Note : les valeurs d'option booléennes sont configurées sur true si l'option est spécifiée sans une valeur.

- table-execution

--table-execution = true|false
Active l'évaluation des tables XBRL. La valeur par défaut est false. Sera configuré sur true si --table-output est spécifié. Si true, spécifie automatiquement --table=true.
Note : les valeurs d'option booléennes sont configurées sur true si l'option est spécifiée sans une valeur.

- 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

Active le chargement des bases de lien de table rédigées par le biais d'une spécification d'ébauche précédente. Néanmoins, la validation, la résolution et la mise en page de la base de lien de table sont toujours effectués conformément à la Recommandation Table Linkbase 1.0 du 18 mars 2014. Utiliser ##detect pour activer la détection automatique.

- table-output

--table-output = *FILE*
Écrit la sortie de table dans le *FILE* spécifié. Si défini, spécifie automatiquement --table-execution=true.

- table-output-format

--table-output-format = xml|html
Spécifie le format de la sortie de table. La valeur par défaut est xml.

- Catalogues et ressources globales
  - catalog

--catalog = *FILE*
Spécifie le chemin absolu vers un fichier de catalogue root qui n'est pas le fichier catalogue root installé. La valeur par défaut est le chemin absolu vers le fichier de catalogue root installé (<installation-folder>\Altova\RaptorXMLXBRLServer2018\etc\RootCatalog.xml). Voir la section, Catalogues XML, pour plus d'informations à propos du travail avec les catalogues.

  - user-catalog

--user-catalog = *FILE*
Spécifie le chemin absolu vers un catalogue XML à utiliser en plus du catalogue root. Voir la section, Catalogues XML, pour plus d'informations concernant le travail avec des catalogues.

- enable-globalresources

--enable-globalresources = `true|false`
Active les ressources globales. La valeur par défaut est `false`.
Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- globalresourceconfig [gc]

--gc | --globalresourceconfig = *VALUE*
Spécifie la configuration active de la ressource globale (et active les ressources globales).

- globalresourcefile [gr]

--gr | --globalresourcefile = *FILE*
Spécifie le fichier de ressource globale (et active les ressources globales).

- Messages, erreurs, aide, timeout, version
  - error-format

--error-format = `text|shortxml|longxml`
Spécifie le format de la sortie d'erreur. La valeur par défaut est `text`. Les autres options génèrent des formats XML, avec `longxml` générant plus de détails.

  - error-limit

--error-limit = *N*
Spécifie la limite d'erreur avec une plage de valeur de `1` à `999`. La valeur par défaut est `100`. Le traitement s'arrête lorsque la limite d'erreur est atteinte. Utile pour limiter l'utilisation du processeur pendant la validation/transformation.

  - help

--help
Affiche le texte d'aide pour la commande. Par exemple, `valany --h`. (en alternative, la commande `help` peut être utilisée avec un argument. Par exemple : `help valany`.)

  - log-output

--log-output = *FILE*
Écrit la sortie de log dans l'URL de fichier spécifié. Veuillez vous assurer que la CLI a une permission d'écriture dans l'emplacement de la sortie.

  - network-timeout

--network-timeout = *VALUE*
Spécifie le timeout en secondes pour les opérations I/O à distance. La valeur par défaut

est : `40`.

- verbose

  `--verbose =` `true|false`
  Une valeur de `true` permet la sortie d'informations supplémentaires pendant la validation. La valeur par défaut est `false`.
  Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- verbose-output

  `--verbose-output =` *`FILE`*
  Écrit la sortie verbeuse sur *`FILE`*.

- version

  `--version`
  Affiche la version de RaptorXML+XBRL Server. Si utilisé avec une commande, placer `--version` avant la commande.

- warning-limit

  `--warning-limit =` *`VALUE`*
  Spécifie la limite d'avertissement dans la plage `1-65535`. Le traitement se poursuit si cette limite a été atteinte, mais d'autres avertissements ne sont pas rapportés. La valeur par défaut est `100`.

 © 2018 Altova GmbH

valinlinexbrl (ixbrl)

La commande **valinlinexbrl | ixbrl** valide un ou plusieurs documents Inline XBRL (iXBRL) conformément à la spécification Inline XBRL 1.0 ou Inline XBRL 1.1. Un fichier XBRL est généré depuis l'Inline XBRL et peut être enregistré.

Windows	**RaptorXMLXBRL valxbrl | xbrl [options] *InputFile***
Linux	**raptorxmlxbrl valxbrl | xbrl [options] *InputFile***
Mac	**raptorxmlxbrl valxbrl | xbrl [options] *InputFile***

L'argument *InputFile* spécifie le document Inline XBRL à valider (généralement, un document XHTML). Dans la première étape de la validation, l'exécution de commande transforme le XHTML en XBRL en extrayant l'Inline XBRL. Si aucune erreur n'est générée au cours de cette première étape et si l'option `--validate-xbrl` est définie sur `true` (*voir la liste des Options ci-dessous*) le XBRL généré est validé par rapport à la taxonomie spécifiée dans l'Inline XBRL.

Pour valider plusieurs documents, il faut soit : (i) lister les fichiers à valider sur la CLI, chaque fichier étant séparé du suivant par un espace ; ou (ii) lister les fichiers à valider dans un fichier texte (fichier `.txt`), avec un nom de fichier par ligne, et fournir ce fichier de texte en tant que l'argument *InputFile* avec l'option `--listfile` définie sur `true` (*voir la liste des Options ci-dessous*). Veuillez noter que même si plusieurs fichiers d'entrée Inline XBRL sont traités, seul un document XBRL est généré.

Exemples

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

Casse et lignes obliques sur la ligne de commande

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

* Veuillez noter que les caractères minuscules (`raptorxmlxbrl`) fonctionnent sur toutes les plates-formes (Windows, Linux et Mac), alors qu'une écriture en majuscule/minuscule (`RaptorXMLXBRL`) ne fonctionne que sur Windows et Mac.
* Utiliser les barres obliques penchées vers l'avant pour Linux et Mac, et les barres obliques penchées vers l'arrière sur Windows.

Options

Les options de la commande sont listées ci-dessous, et organisées en groupes. Les valeurs peuvent être spécifiées sans guillemets sauf dans deux cas : (i) lorsque le string de valeur contient des espaces, ou (ii) si la présence de guillemets est explicitement requise dans la description de l'option.

- XBRL validation et traitement
  - additional-dts-entry-point

    `--additional-dts-entry-point =` *`FILE`*

    Force le fichier de point d'entrée spécifié à être inclus dans le processus de découverte de DTS. Ajouter l'option plusieurs fois pour spécifier plus d'un point d'entrée.

  - ct-restrict-mode

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

    Spécifie comment contrôler les restrictions de type complexes. Une valeur de **`1.0`** contrôles les restrictions de type complexes telles que définies dans la spécification XSD 1.0 — même dans le mode de validation XSD 1.1. Une valeur de **`1.1`** contrôles les restrictions de type complexe telles que définies dans la spécification XSD 1.1 — même dans le mode de validation XSD 1.0. Une valeur de **`default`** contrôle les restrictions de type complexes telles que définies dans la spécification XSD du mode de validation actuel (1.0 ou 1.1). La valeur par défaut est **`default`**.

  - dimensions

    `--dimensions =` `true|false`

    Active les extensions XBRL Dimension 1.0. La valeur par défaut est `true`.
    Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

  - dts

    `--dts =` *`FILE`*

    Précharge le point d'entrée DTS indiqué dans *`FILE`*, et l'utilise pour la validation des fichiers d'instance. Ajouter l'option plusieurs fois pour spécifier plus d'un seul point d'entrée. Les points d'entrée dans les fichiers d'instance seront ignorés. La commande est utilisée pour valider en bloc plusieurs fichiers d'instance qui réfèrent tous à la même taxonomie. L'option `--dts` instruit le moteur Raptor à charger le DTS une seule fois — et non pas pour chaque instance dans la liste de fichier. (Une liste de fichier est donnée soit dans la CLI, soit dans un fichier de texte qui est indiqué en tant que liste de fichier en plaçant l'option `--listfile` sur `true`.) Si un fichier d'instance dans la liste de fichier référence une taxonomie différente, un avertissement sera émis. Cette commande peut accélérer considérablement le temps de validation lors de la validation de nombreux petits fichiers d'instance ayant chacun un grand DTS de support ; elle n'apporte aucun avantage lors de la validation d'un fichier d'instance unique.

  - essence-alias-checks

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

    Active le rapport des incohérences essence-alias. La valeur par défaut est `true`.

 © 2018 Altova GmbH

Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- extensible-enumerations

  `--extensible-enumerations =` `true|false`
  Si `true`, active l'extension XBRL Extensible Enumerations 1.0. La valeur par défaut est : `true`.

- inconsistencies-limit

  `--inconsistencies-limit =` *`VALUE`*
  Spécifie la limite d'inconsistances XBRL, avec des valeurs allant entre `1-65535`. Le traitement continue lorsque la limite est atteinte mais les autres inconsistances ne sont pas rapportées. La valeur par défaut est `100`.

- listfile

  `--listfile =` `true|false`
  Si `true`, traiter l'argument *`InputFile`* de la commande en tant que fichier de texte contenant un nom de fichier par ligne. La valeur par défaut est `false`. (Une alternative est de lister les fichiers sur la CLI avec un espace en tant que séparateur. Veuillez noter, néanmoins, que les CLI ont une limitation de caractères maximum.) Veuillez noter que l'option `--listfile` s'applique uniquement aux arguments, et non pas aux options.
  Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- parallel-assessment [pa]

  `--pa | --parallel-assessment =` `true|false`
  Si configuré sur `true`, l'évaluation de validité de schéma est effectuée en parallèle. Cela signifie qu'il y a plus de 128 éléments à tout niveau, ces éléments sont traités en parallèle, en utilisant plusieurs threads. Des fichiers XML très volumineux peuvent donc être traités plus rapidement si cette option est activée. L'évaluation parallèle a lieu un niveau hiérarchique à la fois, mais peut se produire à plusieurs niveaux différents dans une seule infobulle. Veuillez noter que l'évaluation parallèle ne fonctionne pas en mode streaming. C'est pour cette raison que l'option `--streaming` est ignorée si l'évaluation `--parallel-` est définie sur `true`. De même, les besoins en mémoire sont plus élevés lorsque l'option `--parallel-assessment` est utilisée. Le paramètre par défaut est `false`. La forme abrégée pour l'option est `--pa`.
  Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- preload-xbrl-schemas

  `--preload-xbrl-schemas =` `true|false`
  Précharge les schémas de la spécification XBRL 2.1. La valeur par défaut est `true`.
  Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- recurse

  `--recurse =` `true|false`

Utilisé pour sélectionner des fichiers dans le cadre de sous-répertoires, y compris des archives ZIP. Si `true`, l'argument *`InputFile`* de la commande sélectionnera aussi le fichier spécifié dans les sous-répertoires. Par exemple : `test.zip|zip\test.xml` sélectionnera des fichiers nommés `test.xml` à tous les niveaux de dossier du dossier zip. Les caractères génériques `*` et `?` peuvent être utilisés. Donc, `*.xml` sélectionnera tous les fichiers `.xml` dans le dossier (zip). La valeur par défaut de l'option est `false`.
Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- requires-element-checks

`--requires-element-checks =` `true|false`
Active le rapport des incohérences requires-element. La valeur par défaut est `true`.
Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- schema-imports

`--schema-imports =` `load-by-schemalocation | load-preferring-schemalocation | load-by-namespace | load-combining-both | license-namespace-only`
Spécifie le comportement des éléments `xs:import`, chacun d'entre eux ayant un attribut `namespace` optionnel et un attribut `schemaLocation` optionnel : `<import namespace="someNS" schemaLocation="someURL">`. L'option spécifie s'il faut charger un document de schéma ou juste mettre un espace de noms sous licence, et, si un document de schéma doit être chargé, quelle information doit être utilisée pour le trouver. Défaut : `load-preferring-schemalocation`.
Le comportement est le suivant :

- `load-by-schemalocation`: La valeur de l'attribut `schemaLocation` est utilisée pour situer le schéma, en prenant en compte les mappages de catalogue. Si l'attribut de l'espace de nom est présent, l'espace de noms est importé (licencé).
- `load-preferring-schemalocation`: Si l'attribut `schemaLocation` est présent, il est utilisé en prenant en compte les mappages de catalogue. Si l'attribut `schemaLocation` est présent, la valeur de l'attribut `namespace` est utilisée via un mappage de catalogue. C'est la **valeur par défaut**.
- `load-by-namespace`: La valeur de l'attribut `namespace` est utilisée pour situer le schéma via un mappage de catalogue.
- `load-combining-both`: Si soit l'Attribut `namespace` ou l'attribut `schemaLocation` a un mappage de catalogue, le mappage est utilisé. Si les deux ont des mappages de catalogue, alors la valeur de l'option `--schema-mapping` (option XBRL et option XML/XSD) décide de quel mappage utiliser. Si aucun mappage de catalogue n'est présent, l'attribut `schemaLocation` est utilisé.
- `license-namespace-only`: Le nom d'espace est importé. Aucun document de schéma n'est importé.

- schema-mapping

`--schema-mapping =` `prefer-schemalocation | prefer-namespace`
Si l'emplacement de schéma et l'espace de noms sont tous les deux utilisés pour trouver un document de schéma, spécifie lequel des deux doit être utilisé pendant la consultation du catalogue. (Si soit l'option `--schemalocation-hints` ou l'option `--schema-imports` a une valeur de `load-combining-both`, et si les parties d'espace de

noms et d'URL impliquées ont toutes les deux des mappages de catalogue, alors la valeur de cette option spécifie lequel des deux mappages utiliser (mappage d'espace de noms ou mappage URL ; la valeur `prefer-schemalocation` réfère au mappage URL).) Défaut : `prefer-schemalocation`.

- schemalocation-hints

  `--schemalocation-hints =` `load-by-schemalocation | load-by-namespace | load-combining-both | ignore`
  Spécifie le comportement des attributs `xsi:schemaLocation` et `xsi:noNamespaceSchemaLocation` : S'il faut charger un document de schéma et, si oui, quelle information doit être utilisée pour la trouver. Défaut : `load-by-schemalocation`.

  - La valeur `load-by-schemalocation` utilise l'URL de l'emplacement de schéma dans les attributs `xsi:schemaLocation` et `xsi:noNamespaceSchemaLocation` dans les documents d'instance XML ou XBRL. Il s'agit de la **valeur par défaut**.
  - La valeur `load-by-namespace` prend la part d'espace de nom `xsi:schemaLocation` et un string vide dans le cas de `xsi:noNamespaceSchemaLocation` et localise le schéma par le biais d'un mappage de catalogue.
  - Si `load-combining-both` est utilisé et si soit la partie espace de noms ou la partie URL a un mappage de catalogue, alors le mappage de catalogue est utilisé. Si tous deux ont des mappages de catalogue, alors la valeur de l'option de `--schema-mapping` (option XBRL et option XML/XSD) de quel mappage utiliser. Si ni l'espace de noms ni l'URL n'a un mappage de catalogue, l'URL est utilisée.
  - Si la valeur de l'option est `ignore`, les attributs `xsi:schemaLocation` et `xsi:noNamespaceSchemaLocation` seront ignorés tous les deux.

- script

  `--script =` *`FILE`*
  Exécute le script Python dans le fichier soumis une fois que la validation a été terminée. Ajouter l'option plusieurs fois pour spécifier plus d'un seul script.

- script-api-version

  `--api, --script-api-version =` `1|2|2.1|2.2|2.3|2.4`
  Spécifie la version API Python à utiliser pour le script. La valeur par défaut est la dernière version, actuellement **`2.4`**. Au lieu des valeurs `1` et `2`, vous pouvez aussi utiliser les valeurs `1.0` et `2.0`, respectivement.

- script-output

  `--script-output =` *`FILE`*
  Rédige la sortie standard du script dans le fichier nommé dans *`FILE`*.

- script-param

  `--script-param =` *`KEY:VALUE`*
  Des paramètres spécifiés par l'utilisateur supplémentaires qui peuvent être accédés au cours de l'exécution des scripts Python. Ajouter l'option plusieurs fois pour spécifier plus d'un paramètre de script.

- summation-item-checks

  --summation-item-checks = `true|false`
  Active le rapport des incohérences summation-item. La valeur par défaut est `true`.
  Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- taxonomy-package

  --taxonomy-package = *FILE*
  Spécifie le chemin absolu vers un pack de taxonomie supplémentaire tel que décrit dans les ébauches de travail Taxonomy Package 1.0 et Taxonomy Packages 1.0. La valeur de FILE donne l'emplacement du pack de taxonomie. Ajouter l'option plusieurs fois pour spécifier plus d'un pack de taxonomie.

- taxonomy-packages-config-file

  --taxonomy-packages-config-file = *FILE*
  Spécifie le chemin vers le fichier **TaxonomyPackagesConfig.json**, qui est utilisé pour charger les packs de taxonomie XBRL. L'avantage du fichier est qu'il sert en tant que catalogue des packs de taxonomie. La structure des fichiers JSON est tel que dans l'exemple indiqué ci-dessous. La valeur de la clé **uri** donne l'emplacement du pack. La clé **active** (d'un ensemble de packs) bascule lors de l'utilisation de ces packs.

```
{"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`
  Entraîne l'échec de la validation XBRL si le fichier contient des inconsistances telles que définies par la spécification XBRL 2.1. La valeur par défaut est `false`.
  Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- utr

  --utr = `true|false`
  Si `true`, active l'extension XBRL Unit Registry 1.0. La valeur par défaut est : `false`.

- utr-file

  --utr-file = *FILE*
  Spécifie le chemin menant au fichier Units Registry. Le chemin par défaut est : **http://www.xbrl.org/utr/utr.xml**.

 © 2018 Altova GmbH

- utr-status

  --utr-status = PWD|CR|REC|NIE|PROPOSED|IWD|ACK|RR
  Active la prise en charge des définitions d'unité avec le statut donné dans le Units Registry. Ajouter l'option plusieurs fois pour spécifier plus d'un statut d'unité. Le défaut est **REC**.

- validate-dts-only

  --validate-dts-only = true|false
  Le DTS est découvert en démarrant depuis le document d'instance XBRL. Tous les schémas de taxonomie référencés et bases de lien sont découverts et validés. Le reste du document d'instance XBRL est ignoré. La valeur par défaut est `false`.
  Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- xinclude

  --xinclude = true|false
  Active la prise en charge de XML Inclusions (XInclude). La valeur par défaut est `false`. Si `false`, les élément `include` d'XInclude sont ignorés.
  Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- Inline XBRL
  - document-set

    --document-set = true|false
    Si réglé sur `true`, tous les fichiers soumis (les documents XBRL Inline) seront traités en tant qu'Ensemble de Document XBRL Inline. La valeur par défaut est `false`.
    Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

  - ixbrl-version

    --ixbrl-version = 1.0|1.1|detect
    Spécifie la version de la spécification XBRL Inline à utiliser pour la validation. La valeur par défaut est `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

    Dans Inline XBRL, les types de date et numérique peuvent être rendus dans une variété de formats lexicaux, mais dans XBRL ces types ont des formats spécifiques. Par exemple, une date dans l'Inline XBRL peut être donnée en tant que `01 January 2017`, mais dans XBRL, le format de cette date serait `2017-01-01`. Les Inline XBRL Transformation Registries spécifient les règles de conversion à suivre lorsque les

valeurs de date et numériques sont converties depuis Inline XBRL en XBRL. L'option `transformation-registry` limite le registre qui sera utilisé à celui que vous spécifiez. Ajouter l'option plusieurs fois pour spécifier plus qu'un registre de transformation. La valeur par défaut est `#all`, elle sélectionne tous les registres de transformation disponibles. (*Pour la liste des registres de transformation disponibles dans votre version de RaptorXML+XBRL Server, vérifiez la description de la commande sur la CLI.*)

- uri-transformation

  `--uri-transformation =` `none|make-absolute|make-relative|keep-relative`
  Spécifie comment les URI doivent être écrites sur le document XBRL généré.

  - `none`: Copier l'URI tel quel dans le document cible
  - `make-absolute`: rendre absolu tout URIs relatif en le résolvant par rapport à l'URI de base in-scope dans l'élément respectif dans le document d'entrée. Par exemple : si le fichier d'entrée est `c:\test\inlinexbrl.xhtml`, et qu'il contient une référence relative à un schéma, `schemas\schema.xsd`, alors la référence relative sera résolue à : `c:\test\schemas\schema.xsd`. Veuillez noter que les attributs `xml:base` dans le document d'entrée peuvent changer l'URI de base.
  - `make-relative`: Si possible, rendre relatif tout URI absolu et relatif par rapport au document de sortie (sinon écrire l'URI absolu relatif résout)
  - `keep-relative`: Si possible, rendre relatif uniquement les URI relatifs par rapport au document de sortie (et copier tout URI absolu)

- validate-xbrl

  `--validate-xbrl =` `true|false`
  Si défini sur `true`, active la validation du document XBRL qui est généré depuis le document Inline XBRL. Si défini sur `false`, la validation stoppe après la première étape qui génère le XBRL. La valeur par défaut est `true`.
  Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- xbrl-output

  `--xbrl-output =` *`FILE`*
  Écrit la sortie XBRL générée dans l'emplacement du fichier spécifié dans cette option. Veuillez noter que même lorsque des fichiers d'entrée XBRL multiples sont traités, seul un document XBRL est généré.

- XBRL formules et assertions
  - assertion-severity

    `--assertion-severity =` `true|false`
    Active l'extension Assertion Severity 1.0. La valeur par défaut est `true`.
    Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

  - assertions-output

--assertions-output = *FILE*
Écrit la sortie de l'évaluation d'assertion sur le *FILE* spécifié. Si défini, spécifie automatiquement --formula-execution=true.

- assertions-output-format

--assertions-output-format = `json|xml`
Spécifie le format de sortie de l'évaluation d'assertion. La valeur par défaut est `json`.

- evaluate-referenced-parameters-only

--evaluate-referenced-parameters-only = `true|false`
Si `false`, force l'évaluation de tous les paramètres, même s'ils ne sont pas référencés par d'autres formules/assertions/tables. La valeur par défaut est : `true`.

- formula

--formula = `true|false`
Active l'extension XBRL Formula 1.0. La valeur par défaut est `true`.
Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- formula-assertion-set [[DEPRECATED]]

--formula-assertion-set = *VALUE*
Limite l'exécution de la formule au seul ensemble d'assertion donné. Ajouter l'option plusieurs fois pour spécifier plus d'un ensemble d'assertion. La forme abrégée est `--as`. La ***VALUE*** est soit la valeur de l'attribut `@id`, soit une URI avec un fragment XPointer qui identifie la ressource. Les valeurs spéciales `##none` et `##all` peuvent aussi être utilisées.

- formula-execution

--formula-execution = `true|false`
Active l'extension des formules XBRL. La valeur par défaut est `true`. Si `true`, spécifie automatiquement --formula=true.
Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- formula-output

--formula-output = *FILE*
Écrit la sortie de l'évaluation de la formule pour le *FILE* spécifié. Si défini, spécifie automatiquement --formula-execution=true.

- formula-parameters

--formula-parameters = *JSON-ARRAY*
Spécifie les paramètres pour l'évaluation de la formule XBRL en tant qu'un array de mappages JSON directement sur la CLI. Voir la section Paramètres de formule pour plus de détails.

- formula-parameters-file

--formula-parameters-file = *FILE*

Spécifie un *FILE* contenant les paramètres pour l'évaluation de formule XBRL. Le fichier peut être soit un fichier XML soit un fichier JSON. Voir la section Paramètres de formule.

- ignore-assertion

  --ignore-assertion = *VALUE*
  Exclut l'assertion donnée de l'exécution. Ajouter cette option plusieurs fois pour spécifier plus d'une assertion.

- ignore-assertions-file

  --ignore-assertions-file = *FILE*
  Spécifie un chemin vers un fichier qui contient une liste de IDs/XPointers d'assertions à exclure du traitement. Dans le fichier, saisir une assertion par ligne.

- ignore-formula

  --ignore-formula = *VALUE*
  Exclut la formule donnée de l'exécution. Ajouter cette option plusieurs fois pour spécifier plus d'une formule.

- ignore-formulas-file

  --ignore-formulas-file = *FILE*
  Spécifie un chemin vers un fichier qui contient une liste de IDs/XPointers de formules à exclure du traitement. Dans le fichier, saisir une formule par ligne.

- message-lang

  --message-lang = *VALUE*
  Spécifie la langue à utiliser lors de l'affichage des messages de validation. La valeur par défaut est : `en`. Les autres valeurs acceptées sont `de`, `es`, `fr`, `ja`, pour, respectivement, l'allemand, l'espagnol, le français et le japonais.

- message-role

  --message-role = *VALUE*
  Spécifie le rôle de message préféré à utiliser lors de l'affichage des messages de validation. La valeur par défaut est : `http://www.xbrl.org/2010/role/message`.

- preload-formula-schemas

  --preload-formula-schemas = `true|false`
  Précharge les schémas de la spécification XBRL Formula 1.0. La valeur par défaut est `false`.
  Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- process-assertion [a]

  --a | --process-assertion = *VALUE*
  Limite l'exécution de formule à la seule assertion donnée. Ajouter l'option plusieurs fois pour spécifier plus d'une assertion. Forme abrégée est `--a`. La ***VALUE*** est soit la valeur de l'attribut `@id`, soit un URI avec un fragment XPointer qui identifie la ressource. Les valeurs spéciales `##none` et `##all` peuvent aussi être utilisées.

 © 2018 Altova GmbH

- process-assertion-set [as]

  --as | --process-assertion-set = *VALUE*
  Limite l'exécution de formule au seul ensemble d'assertion donné. Ajouter l'option plusieurs fois pour spécifier plus d'un ensemble d'assertion. Forme abrégée est `--as`. La ***VALUE*** est soit la valeur de l'attribut `@id`, soit un URI avec un fragment XPointer qui identifie la ressource. Les valeurs spéciales `##none` et `##all` peuvent aussi être utilisées.

- process-assertions-file

  --process-assertions-file = *FILE*
  Spécifie un chemin vers un fichier qui contient une liste de IDs/XPointers d'assertions à exécuter. Dans le fichier, saisir une assertion par ligne.

- process-formula [f]

  --f | --process-formula = *VALUE*
  Limite l'exécution de formule à la seule formule donnée. Ajouter l'option plusieurs fois pour spécifier plus d'une formule. Forme abrégée est `--f`. La ***VALUE*** est soit la valeur de l'attribut `@id`, soit un URI avec un fragment XPointer qui identifie la ressource. Les valeurs spéciales `##none` et `##all` peuvent aussi être utilisées.

- process-formulas-file

  --process-formulas-file = *FILE*
  Spécifie un chemin vers un fichier qui contient une liste de IDs/XPointers de formules à exécuter. Dans le fichier, saisir une formule par ligne.

- report-unsatisfied-assertion-evaluations

  --report-unsatisfied-assertion-evaluations = `true|false`
  Rapporte l'évaluation de l'assertion non satisfaite en tant qu'erreurs ou avertissements selon le degré de sévérité de l'assertion attribuée. La valeur par défaut est `false`.
  Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- variableset-execution-timeout

  --variableset-execution-timeout = *VALUE*
  Appliqué lors de l'exécution de formules (**`--formula-execution=true`**). Spécifie le temps maximum autorisé pour l'exécution d'un ensemble de variables unique (une formule ou une valeur, ou une existence ou une assertion de consistance). Le temps est spécifié en minutes et doit être un nombre positif. La valeur par défaut est 30min. Si un ensemble de variables particulier ne termine pas l'exécution avant que le timeout n'ait été atteint, il sera abandonné. Un message d'erreur est affiché (et saisi dans le journal verbeux). Veuillez noter, néanmoins, que le contrôle de timeout est uniquement effectué après l'évaluation de chaque ensemble de variable ; et non pas pendant l'exécution des expressions XPath individuelles. Donc, si une seule expression XPath prend beaucoup de temps à être exécutée, la limite de timeout peut avoir été dépassée. L'exécution d'un ensemble de variable est abandonnée uniquement une fois que l'évaluation du set de variable complet ait été exécuté.

- Tables XBRL
  - concept-label-linkrole

    --concept-label-linkrole = *VALUE*
    Spécifie le rôle de lien étendu préféré à utiliser lors du rendu de libellés de concept.

  - concept-label-role

    --cconcept-label-role = *VALUE*
    Spécifie le rôle de lien préféré à utiliser lors du rendu de libellés de concept. Par défaut : `http://www.xbrl.org/2003/role/label`.

  - generic-label-linkrole

    --generic-label-linkrole = *VALUE*
    Spécifie le rôle de lien étendu préféré à utiliser lors du rendu des libellés génériques.

  - generic-label-role

    --generic-label-role = *VALUE*
    Spécifie le rôle de libellé préféré à utiliser lors du rendu des libellés génériques. La valeur par défaut est : http://www.xbrl.org/2003/role/label.

  - label-lang

    --label-lang = *VALUE*
    Spécifie la langue préférée à utiliser lors du rendu des libellés. La valeur par défaut est : `en`.

  - preload-table-schemas

    --preload-table-schemas = `true|false`
    Précharge les schémas de la spécification XBRL Table 1.0. La valeur par défaut est `false`.
    Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

  - process-table [t]

    --t | --process-table = *VALUE*
    Limite l'exécution de formule à la seule table donnée. Ajouter l'option plusieurs fois pour spécifier plus d'une table. Forme abrégée est `--t`. La ***VALUE*** est soit la valeur de l'attribut `@id`, soit un URI avec un fragment XPointer qui identifie la ressource. Les valeurs spéciales `##none` et `##all` peuvent aussi être utilisées.

  - table

    --table = `true|false`
    Active l'extension XBRL Table 1.0. La valeur par défaut est `true`. Si `true`, spécifie automatiquement `--formula=true` et `--dimensions=true`.
    Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

  - table-elimination

    --table-elimination = `true|false`

 © 2018 Altova GmbH

Active l'élimination de lignes/colonnes vides dans la sortie HTML. La valeur par défaut est `true`.
Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- table-execution

  `--table-execution =` `true|false`
  Active l'évaluation des tables XBRL. La valeur par défaut est `false`. Sera configuré sur `true` si `--table-output` est spécifié. Si `true`, spécifie automatiquement `--table=true`.
  Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- 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`

  Active le chargement des bases de lien de table rédigées par le biais d'une spécification d'ébauche précédente. Néanmoins, la validation, la résolution et la mise en page de la base de lien de table sont toujours effectués conformément à la Recommandation Table Linkbase 1.0 du 18 mars 2014. Utiliser `##detect` pour activer la détection automatique.

- table-output

  `--table-output =` *`FILE`*
  Écrit la sortie de table dans le *`FILE`* spécifié. Si défini, spécifie automatiquement `--table-execution=true`.

- table-output-format

  `--table-output-format =` `xml|html`
  Spécifie le format de la sortie de table. La valeur par défaut est `xml`.

- Catalogues et ressources globales
  - catalog

    `--catalog =` *`FILE`*
    Spécifie le chemin absolu vers un fichier de catalogue root qui n'est pas le fichier catalogue root installé. La valeur par défaut est le chemin absolu vers le fichier de catalogue root installé (`<installation-folder>\Altova\RaptorXMLXBRLServer2018\etc\RootCatalog.xml`). Voir la section, Catalogues XML, pour plus d'informations à propos du travail avec les catalogues.

  - user-catalog

--user-catalog = *FILE*
Spécifie le chemin absolu vers un catalogue XML à utiliser en plus du catalogue root. Voir la section, Catalogues XML, pour plus d'informations concernant le travail avec des catalogues.

- enable-globalresources

--enable-globalresources = `true|false`
Active les ressources globales. La valeur par défaut est `false`.
Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- globalresourceconfig [gc]

--gc | --globalresourceconfig = *VALUE*
Spécifie la configuration active de la ressource globale (et active les ressources globales).

- globalresourcefile [gr]

--gr | --globalresourcefile = *FILE*
Spécifie le fichier de ressource globale (et active les ressources globales).

- Messages, erreurs, aide, timeout, version
  - error-format

--error-format = `text|shortxml|longxml`
Spécifie le format de la sortie d'erreur. La valeur par défaut est `text`. Les autres options génèrent des formats XML, avec `longxml` générant plus de détails.

  - error-limit

--error-limit = *N*
Spécifie la limite d'erreur avec une plage de valeur de `1` à `999`. La valeur par défaut est `100`. Le traitement s'arrête lorsque la limite d'erreur est atteinte. Utile pour limiter l'utilisation du processeur pendant la validation/transformation.

  - help

--help
Affiche le texte d'aide pour la commande. Par exemple, `valany --h`. (en alternative, la commande `help` peut être utilisée avec un argument. Par exemple : `help valany`.)

  - log-output

--log-output = *FILE*
Écrit la sortie de log dans l'URL de fichier spécifié. Veuillez vous assurer que la CLI a une permission d'écriture dans l'emplacement de la sortie.

  - network-timeout

--network-timeout = *VALUE*
Spécifie le timeout en secondes pour les opérations I/O à distance. La valeur par défaut

 © 2018 Altova GmbH

est : `40`.

- verbose

  `--verbose =` `true|false`
  Une valeur de `true` permet la sortie d'informations supplémentaires pendant la validation. La valeur par défaut est `false`.
  Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- verbose-output

  `--verbose-output =` *`FILE`*
  Écrit la sortie verbeuse sur *`FILE`*.

- version

  `--version`
  Affiche la version de RaptorXML+XBRL Server. Si utilisé avec une commande, placer `--version` avant la commande.

- warning-limit

  `--warning-limit =` *`VALUE`*
  Spécifie la limite d'avertissement dans la plage `1-65535`. Le traitement se poursuit si cette limite a été atteinte, mais d'autres avertissements ne sont pas rapportés. La valeur par défaut est `100`.

valxbrltaxonomy (dts)

La commande **valxbrltaxonomy | dts** valide une ou plusieurs taxonomies XBRL (schémas) conformément aux spécifications XBRL 2.1, Dimensions 1.0 et Formula 1.0.

Windows	**RaptorXMLXBRL valxbrltaxonomy | dts [options] *InputFile***
Linux	**raptorxmlxbrl valxbrltaxonomy | dts [options] *InputFile***
Mac	**raptorxmlxbrl valxbrltaxonomy | dts [options] *InputFile***

L'argument *InputFile* est la taxonomie XBRL à valider. Pour valider plusieurs documents, il faut soit : (i) lister les fichiers à valider sur la CLI, chaque fichier étant séparé du suivant par un espace ; ou (ii) lister les fichiers à valider dans un fichier texte (fichier `.txt`), avec un nom de fichier par ligne, et fournir ce fichier de texte en tant que l'argument *InputFile* avec l'option `--listfile` définie sur `true` (*voir la liste des Options ci-dessous*).

Exemples

- **raptorxmlxbrl** valxbrltaxonomy c:\Test.xsd
- **raptorxmlxbrl** dts --listfile c:\FileList.txt

Casse et lignes obliques sur la ligne de commande

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

* Veuillez noter que les caractères minuscules (`raptorxmlxbrl`) fonctionnent sur toutes les plates-formes (Windows, Linux et Mac), alors qu'une écriture en majuscule/minuscule (`RaptorXMLXBRL`) ne fonctionne que sur Windows et Mac.
* Utiliser les barres obliques penchées vers l'avant pour Linux et Mac, et les barres obliques penchées vers l'arrière sur Windows.

Options

Les options de la commande sont listées ci-dessous, et organisées en groupes. Les valeurs peuvent être spécifiées sans guillemets sauf dans deux cas : (i) lorsque le string de valeur contient des espaces, ou (ii) si la présence de guillemets est explicitement requise dans la description de l'option.

XBRL validation et traitement

ct-restrict-mode

--ct-restrict-mode = `1.0|1.1|default`
Spécifie comment contrôler les restrictions de type complexes. Une valeur de **1.0** contrôles les restrictions de type complexes telles que définies dans la spécification

XSD 1.0 — même dans le mode de validation XSD 1.1. Une valeur de **`1.1`** contrôles les restrictions de type complexe telles que définies dans la spécification XSD 1.1 — même dans le mode de validation XSD 1.0. Une valeur de **`default`** contrôle les restrictions de type complexes telles que définies dans la spécification XSD du mode de validation actuel (1.0 ou 1.1). La valeur par défaut est **`default`**.

- assertion-severity

  `--assertion-severity =` `true|false`
  Active l'extension Assertion Severity 1.0. La valeur par défaut est `true`.
  Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- dimensions

  `--dimensions =` `true|false`
  Active les extensions XBRL Dimension 1.0. La valeur par défaut est `true`.
  Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- evaluate-referenced-parameters-only

  `--evaluate-referenced-parameters-only =` `true|false`
  Si `false`, force l'évaluation de tous les paramètres, même s'ils ne sont pas référencés par d'autres formules/assertions/tables. La valeur par défaut est : `true`.

- extensible-enumerations

  `--extensible-enumerations =` `true|false`
  Si `true`, active l'extension XBRL Extensible Enumerations 1.0. La valeur par défaut est : `true`.

- inconsistencies-limit

  `--inconsistencies-limit =` *`VALUE`*
  Spécifie la limite d'inconsistances XBRL, avec des valeurs allant entre `1-65535`. Le traitement continue lorsque la limite est atteinte mais les autres inconsistances ne sont pas rapportées. La valeur par défaut est `100`.

- formula

  `--formula =` `true|false`
  Active l'extension XBRL Formula 1.0. La valeur par défaut est `true`.
  Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- formula-parameters

  `--formula-parameters =` *`JSON-ARRAY`*
  Spécifie les paramètres pour l'évaluation de la formule XBRL en tant qu'un array de mappages JSON directement sur la CLI. Voir la section Paramètres de formule pour plus de détails.

- formula-parameters-file

--formula-parameters-file = *FILE*
Spécifie un *FILE* contenant les paramètres pour l'évaluation de formule XBRL. Le fichier peut être soit un fichier XML soit un fichier JSON. Voir la section Paramètres de formule.

- listfile

--listfile = `true|false`
Si `true`, traiter l'argument *InputFile* de la commande en tant que fichier de texte contenant un nom de fichier par ligne. La valeur par défaut est `false`. (Une alternative est de lister les fichiers sur la CLI avec un espace en tant que séparateur. Veuillez noter, néanmoins, que les CLI ont une limitation de caractères maximum.) Veuillez noter que l'option `--listfile` s'applique uniquement aux arguments, et non pas aux options.
Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- preload-formula-schemas

--preload-formula-schemas = `true|false`
Précharge les schémas de la spécification XBRL Formula 1.0. La valeur par défaut est `false`.
Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- preload-xbrl-schemas

--preload-xbrl-schemas = `true|false`
Précharge les schémas de la spécification XBRL 2.1. La valeur par défaut est `true`.
Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- recurse

--recurse = `true|false`
Utilisé pour sélectionner des fichiers dans le cadre de sous-répertoires, y compris des archives ZIP. Si `true`, l'argument *InputFile* de la commande sélectionnera aussi le fichier spécifié dans les sous-répertoires. Par exemple : `test.zip|zip\test.xml` sélectionnera des fichiers nommés `test.xml` à tous les niveaux de dossier du dossier zip. Les caractères génériques `*` et `?` peuvent être utilisés. Donc, `*.xml` sélectionnera tous les fichiers `.xml` dans le dossier (zip). La valeur par défaut de l'option est `false`.
Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- schema-imports

--schema-imports = `load-by-schemalocation | load-preferring-schemalocation | load-by-namespace | load-combining-both | license-namespace-only`
Spécifie le comportement des éléments `xs:import`, chacun d'entre eux ayant un attribut `namespace` optionnel et un attribut `schemaLocation` optionnel : `<import namespace="someNS" schemaLocation="someURL">`. L'option spécifie s'il faut charger un document de schéma ou juste mettre un espace de noms sous licence, et, si un document de schéma doit être chargé, quelle information doit être utilisée pour le

trouver. Défaut : `load-preferring-schemalocation`.
Le comportement est le suivant :

- `load-by-schemalocation`: La valeur de l'attribut `schemaLocation` est utilisée pour situer le schéma, en prenant en compte les mappages de catalogue. Si l'attribut de l'espace de nom est présent, l'espace de noms est importé (licencé).
- `load-preferring-schemalocation`: Si l'attribut `schemaLocation` est présent, il est utilisé en prenant en compte les mappages de catalogue. Si l'attribut `schemaLocation` est présent, la valeur de l'attribut `namespace` est utilisée via un mappage de catalogue. C'est la **valeur par défaut**.
- `load-by-namespace`: La valeur de l'attribut `namespace` est utilisée pour situer le schéma via un mappage de catalogue.
- `load-combining-both`: Si soit l'Attribut `namespace` ou l'attribut `schemaLocation` a un mappage de catalogue, le mappage est utilisé. Si les deux ont des mappages de catalogue, alors la valeur de l'option `--schema-mapping` (option XBRL et option XML/XSD) décide de quel mappage utiliser. Si aucun mappage de catalogue n'est présent, l'attribut `schemaLocation` est utilisé.
- `license-namespace-only`: Le nom d'espace est importé. Aucun document de schéma n'est importé.

schema-mapping

`--schema-mapping =` `prefer-schemalocation | prefer-namespace`
Si l'emplacement de schéma et l'espace de noms sont tous les deux utilisés pour trouver un document de schéma, spécifie lequel des deux doit être utilisé pendant la consultation du catalogue. (Si soit l'option `--schemalocation-hints` ou l'option `--schema-imports` a une valeur de `load-combining-both`, et si les parties d'espace de noms et d'URL impliquées ont toutes les deux des mappages de catalogue, alors la valeur de cette option spécifie lequel des deux mappages utiliser (mappage d'espace de noms ou mappage URL ; la valeur `prefer-schemalocation` réfère au mappage URL).)
Défaut : `prefer-schemalocation`.

schemalocation-hints

`--schemalocation-hints =` `load-by-schemalocation | load-by-namespace | load-combining-both | ignore`
Spécifie le comportement des attributs `xsi:schemaLocation` et `xsi:noNamespaceSchemaLocation` : S'il faut charger un document de schéma et, si oui, quelle information doit être utilisée pour la trouver. Défaut : `load-by-schemalocation`.

- La valeur `load-by-schemalocation` utilise l'URL de l'emplacement de schéma dans les attributs `xsi:schemaLocation` et `xsi:noNamespaceSchemaLocation` dans les documents d'instance XML ou XBRL. Il s'agit de la **valeur par défaut**.
- La valeur `load-by-namespace` prend la part d'espace de nom `xsi:schemaLocation` et un string vide dans le cas de `xsi:noNamespaceSchemaLocation` et localise le schéma par le biais d'un mappage de catalogue.
- Si `load-combining-both` est utilisé et si soit la partie espace de noms ou la partie URL a un mappage de catalogue, alors le mappage de catalogue est utilisé. Si tous deux ont des mappages de catalogue, alors la valeur de l'option de `--schema-mapping` (option XBRL et option XML/XSD) de quel mappage utiliser. Si ni

l'espace de noms ni l'URL n'a un mappage de catalogue, l'URL est utilisée.
- Si la valeur de l'option est `ignore`, les attributs `xsi:schemaLocation` et `xsi:noNamespaceSchemaLocation` seront ignorés tous les deux.

script

`--script =` *`FILE`*
Exécute le script Python dans le fichier soumis une fois que la validation a été terminée. Ajouter l'option plusieurs fois pour spécifier plus d'un seul script.

script-api-version

`--api, --script-api-version =` `1|2|2.1|2.2|2.3|2.4`
Spécifie la version API Python à utiliser pour le script. La valeur par défaut est la dernière version, actuellement **`2.4`**. Au lieu des valeurs `1` et `2`, vous pouvez aussi utiliser les valeurs `1.0` et `2.0`, respectivement.

script-output

`--script-output =` *`FILE`*
Rédige la sortie standard du script dans le fichier nommé dans *`FILE`*.

script-param

`--script-param =` *`KEY:VALUE`*
Des paramètres spécifiés par l'utilisateur supplémentaires qui peuvent être accédés au cours de l'exécution des scripts Python. Ajouter l'option plusieurs fois pour spécifier plus d'un paramètre de script.

taxonomy-package

`--taxonomy-package =` *`FILE`*
Spécifie le chemin absolu vers un pack de taxonomie supplémentaire tel que décrit dans les ébauches de travail Taxonomy Package 1.0 et Taxonomy Packages 1.0. La valeur de FILE donne l'emplacement du pack de taxonomie. Ajouter l'option plusieurs fois pour spécifier plus d'un pack de taxonomie.

taxonomy-packages-config-file

`--taxonomy-packages-config-file =` *`FILE`*
Spécifie le chemin vers le fichier **`TaxonomyPackagesConfig.json`**, qui est utilisé pour charger les packs de taxonomie XBRL. L'avantage du fichier est qu'il sert en tant que catalogue des packs de taxonomie. La structure des fichiers JSON est tel que dans l'exemple indiqué ci-dessous. La valeur de la clé **`uri`** donne l'emplacement du pack. La clé **`active`** (d'un ensemble de packs) bascule lors de l'utilisation de ces packs.

```
{"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`
  Entraîne l'échec de la validation XBRL si le fichier contient des inconsistances telles que définies par la spécification XBRL 2.1. La valeur par défaut est `false`.
  Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- xinclude

  --xinclude = `true|false`
  Active la prise en charge de XML Inclusions (XInclude). La valeur par défaut est `false`. Si `false`, les élément `include` d'XInclude sont ignorés.
  Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- Tables XBRL
  - aspect-node-order

    --aspect-node-order = `lexical|instance`
    Spécifie l'ordre de mise en page d'un aspectNode ouvert. Le défaut est **`lexical`**.

  - concept-label-linkrole

    --concept-label-linkrole = *VALUE*
    Spécifie le rôle de lien étendu préféré à utiliser lors du rendu de libellés de concept.

  - concept-label-role

    --cconcept-label-role = *VALUE*
    Spécifie le rôle de lien préféré à utiliser lors du rendu de libellés de concept. Par défaut : `http://www.xbrl.org/2003/role/label`.

  - evaluate-referenced-parameters-only

    --evaluate-referenced-parameters-only = `true|false`
    Si `false`, force l'évaluation de tous les paramètres, même s'ils ne sont pas référencés par d'autres formules/assertions/tables. La valeur par défaut est : `true`.

  - generic-label-linkrole

    --generic-label-linkrole = *VALUE*
    Spécifie le rôle de lien étendu préféré à utiliser lors du rendu des libellés génériques.

  - generic-label-role

    --generic-label-role = *VALUE*
    Spécifie le rôle de libellé préféré à utiliser lors du rendu des libellés génériques. La valeur par défaut est : http://www.xbrl.org/2003/role/label.

- label-lang

  --label-lang = *VALUE*
  Spécifie la langue préférée à utiliser lors du rendu des libellés. La valeur par défaut est : `en`.

- preload-table-schemas

  --preload-table-schemas = `true|false`
  Précharge les schémas de la spécification XBRL Table 1.0. La valeur par défaut est `false`.
  Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- process-table [t]

  --t | --process-table = *VALUE*
  Limite l'exécution de formule à la seule table donnée. Ajouter l'option plusieurs fois pour spécifier plus d'une table. Forme abrégée est `--t`. La ***VALUE*** est soit la valeur de l'attribut `@id`, soit un URI avec un fragment XPointer qui identifie la ressource. Les valeurs spéciales `##none` et `##all` peuvent aussi être utilisées.

- table

  --table = `true|false`
  Active l'extension XBRL Table 1.0. La valeur par défaut est `true`. Si `true`, spécifie automatiquement `--formula=true` et `--dimensions=true`.
  Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- table-execution

  --table-execution = `true|false`
  Active l'évaluation des tables XBRL. La valeur par défaut est `false`. Sera configuré sur `true` si `--table-output` est spécifié. Si `true`, spécifie automatiquement `--table=true`.
  Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- 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`

  Active le chargement des bases de lien de table rédigées par le biais d'une spécification d'ébauche précédente. Néanmoins, la validation, la résolution et la mise en page de la base de lien de table sont toujours effectués conformément à la Recommandation Table Linkbase 1.0 du 18 mars 2014. Utiliser `##detect` pour activer la détection automatique.

 © 2018 Altova GmbH

- table-output

  --table-output = *FILE*
  Écrit la sortie de table dans le *FILE* spécifié. Si défini, spécifie automatiquement --table-execution=true.

- table-output-format

  --table-output-format = xml|html
  Spécifie le format de la sortie de table. La valeur par défaut est xml.

- Catalogues et ressources globales
  - catalog

    --catalog = *FILE*
    Spécifie le chemin absolu vers un fichier de catalogue root qui n'est pas le fichier catalogue root installé. La valeur par défaut est le chemin absolu vers le fichier de catalogue root installé (<installation-folder>\Altova\RaptorXMLXBRLServer2018\etc\RootCatalog.xml). Voir la section, Catalogues XML, pour plus d'informations à propos du travail avec les catalogues.

  - user-catalog

    --user-catalog = *FILE*
    Spécifie le chemin absolu vers un catalogue XML à utiliser en plus du catalogue root. Voir la section, Catalogues XML, pour plus d'informations concernant le travail avec des catalogues.

  - enable-globalresources

    --enable-globalresources = true|false
    Active les ressources globales. La valeur par défaut est false.
    Note : les valeurs d'option booléennes sont configurées sur true si l'option est spécifiée sans une valeur.

  - globalresourceconfig [gc]

    --gc | --globalresourceconfig = *VALUE*
    Spécifie la configuration active de la ressource globale (et active les ressources globales).

  - globalresourcefile [gr]

    --gr | --globalresourcefile = *FILE*
    Spécifie le fichier de ressource globale (et active les ressources globales).

- Messages, erreurs, aide, timeout, version
  - error-format

    --error-format = text|shortxml|longxml
    Spécifie le format de la sortie d'erreur. La valeur par défaut est text. Les autres options génèrent des formats XML, avec longxml générant plus de détails.

- error-limit

  `--error-limit =` *`N`*
  Spécifie la limite d'erreur avec une plage de valeur de `1` à `999`. La valeur par défaut est `100`. Le traitement s'arrête lorsque la limite d'erreur est atteinte. Utile pour limiter l'utilisation du processeur pendant la validation/transformation.

- help

  `--help`
  Affiche le texte d'aide pour la commande. Par exemple, `valany --h`. (en alternative, la commande `help` peut être utilisée avec un argument. Par exemple : `help valany`.)

- log-output

  `--log-output =` *`FILE`*
  Écrit la sortie de log dans l'URL de fichier spécifié. Veuillez vous assurer que la CLI a une permission d'écriture dans l'emplacement de la sortie.

- network-timeout

  `--network-timeout =` *`VALUE`*
  Spécifie le timeout en secondes pour les opérations I/O à distance. La valeur par défaut est : `40`.

- verbose

  `--verbose =` `true|false`
  Une valeur de `true` permet la sortie d'informations supplémentaires pendant la validation. La valeur par défaut est `false`.
  Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- verbose-output

  `--verbose-output =` *`FILE`*
  Écrit la sortie verbeuse sur *`FILE`*.

- version

  `--version`
  Affiche la version de RaptorXML+XBRL Server. Si utilisé avec une commande, placer `--version` avant la commande.

- warning-limit

  `--warning-limit =` *`VALUE`*
  Spécifie la limite d'avertissement dans la plage `1-65535`. Le traitement se poursuit si cette limite a été atteinte, mais d'autres avertissements ne sont pas rapportés. La valeur par défaut est `100`.

 © 2018 Altova GmbH

valtaxonomypackage (taxpkg)

La commande **`valtaxonomypackage | taxpkg`** valide un ou plusieurs packs de taxonomies XBRL conformément à la spécification Taxonomy 1.0 Packages.

Windows **`RaptorXMLXBRL valtaxonomypackage | taxpkg [options] TaxonomyPackage`**

Linux **`raptorxmlxbrl valtaxonomypackage | taxpkg [options] TaxonomyPackage`**

Mac **`raptorxmlxbrl valtaxonomypackage | taxpkg [options] TaxonomyPackage`**

L'argument *`TaxonomyPackage`* est le pack de taxonomie XBRL à valider. Le pack de taxonomie est généralement un fichier ZIP. Pour valider plusieurs packs de taxonomie, il faut soit : (i) lister les fichiers à valider sur la CLI, chaque pack étant séparé du suivant par un espace ; ou (ii) lister les packs à valider dans un fichier texte (fichier `.txt`), avec un nom de fichier par ligne, et fournir ce fichier de texte en tant que l'argument *`TaxonomyPackage`* avec l'option `--listfile` définie sur `true` (*voir la liste des Options ci-dessous*).

Exemples

- **`raptorxmlxbrl`** `valxtaxonomypackage c:\Test.zip`
- **`raptorxmlxbrl`** `taxpkg --listfile=true c:\TaxonomyPackageList.txt`
- **`raptorxmlxbrl`** `taxpkg --listfile c:\TaxonomyPackageList.txt`

Casse et lignes obliques sur la ligne de commande

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

* Veuillez noter que les caractères minuscules (`raptorxmlxbrl`) fonctionnent sur toutes les plates-formes (Windows, Linux et Mac), alors qu'une écriture en majuscule/minuscule (`RaptorXMLXBRL`) ne fonctionne que sur Windows et Mac.
* Utiliser les barres obliques penchées vers l'avant pour Linux et Mac, et les barres obliques penchées vers l'arrière sur Windows.

Options

Les options de la commande sont listées ci-dessous, et organisées en groupes. Les valeurs peuvent être spécifiées sans guillemets sauf dans deux cas : (i) lorsque le string de valeur contient des espaces, ou (ii) si la présence de guillemets est explicitement requise dans la description de l'option.

Traitement

- listfile

  --listfile = `true|false`
  Si `true`, traiter l'argument *`InputFile`* de la commande en tant que fichier de texte contenant un nom de fichier par ligne. La valeur par défaut est `false`. (Une alternative est de lister les fichiers sur la CLI avec un espace en tant que séparateur. Veuillez noter, néanmoins, que les CLI ont une limitation de caractères maximum.) Veuillez noter que l'option `--listfile` s'applique uniquement aux arguments, et non pas aux options.
  Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- recurse

  --recurse = `true|false`
  Utilisé pour sélectionner des fichiers dans le cadre de sous-répertoires, y compris des archives ZIP. Si `true`, l'argument *`InputFile`* de la commande sélectionnera aussi le fichier spécifié dans les sous-répertoires. Par exemple : `test.zip|zip\test.xml` sélectionnera des fichiers nommés `test.xml` à tous les niveaux de dossier du dossier zip. Les caractères génériques `*` et `?` peuvent être utilisés. Donc, `*.xml` sélectionnera tous les fichiers `.xml` dans le dossier (zip). La valeur par défaut de l'option est `false`.
  Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- schema-imports

  --schema-imports = `load-by-schemalocation | load-preferring-schemalocation | load-by-namespace | load-combining-both | license-namespace-only`
  Spécifie le comportement des éléments `xs:import`, chacun d'entre eux ayant un attribut `namespace` optionnel et un attribut `schemaLocation` optionnel : `<import namespace="someNS" schemaLocation="someURL">`. L'option spécifie s'il faut charger un document de schéma ou juste mettre un espace de noms sous licence, et, si un document de schéma doit être chargé, quelle information doit être utilisée pour le trouver. Défaut : `load-preferring-schemalocation`.
  Le comportement est le suivant :

  - `load-by-schemalocation`: La valeur de l'attribut `schemaLocation` est utilisée pour situer le schéma, en prenant en compte les mappages de catalogue. Si l'attribut de l'espace de nom est présent, l'espace de noms est importé (licencé).
  - `load-preferring-schemalocation`: Si l'attribut `schemaLocation` est présent, il est utilisé en prenant en compte les mappages de catalogue. Si l'attribut `schemaLocation` est présent, la valeur de l'attribut `namespace` est utilisée via un mappage de catalogue. C'est la **valeur par défaut**.
  - `load-by-namespace`: La valeur de l'attribut `namespace` est utilisée pour situer le schéma via un mappage de catalogue.
  - `load-combining-both`: Si soit l'Attribut `namespace` ou l'attribut `schemaLocation` a un mappage de catalogue, le mappage est utilisé. Si les deux ont des mappages de catalogue, alors la valeur de l'option `--schema-mapping` (option XBRL et option XML/XSD) décide de quel mappage utiliser. Si aucun mappage de catalogue n'est présent, l'attribut `schemaLocation` est utilisé.
  - `license-namespace-only`: Le nom d'espace est importé. Aucun document de schéma n'est importé.

 © 2018 Altova GmbH

schema-mapping

--schema-mapping = `prefer-schemalocation | prefer-namespace`
Si l'emplacement de schéma et l'espace de noms sont tous les deux utilisés pour trouver un document de schéma, spécifie lequel des deux doit être utilisé pendant la consultation du catalogue. (Si soit l'option `--schemalocation-hints` ou l'option `--schema-imports` a une valeur de `load-combining-both`, et si les parties d'espace de noms et d'URL impliquées ont toutes les deux des mappages de catalogue, alors la valeur de cette option spécifie lequel des deux mappages utiliser (mappage d'espace de noms ou mappage URL ; la valeur `prefer-schemalocation` réfère au mappage URL).) Défaut : `prefer-schemalocation`.

schemalocation-hints

--schemalocation-hints = `load-by-schemalocation | load-by-namespace | load-combining-both | ignore`
Spécifie le comportement des attributs `xsi:schemaLocation` et `xsi:noNamespaceSchemaLocation` : S'il faut charger un document de schéma et, si oui, quelle information doit être utilisée pour la trouver. Défaut : `load-by-schemalocation`.

- La valeur `load-by-schemalocation` utilise l'URL de l'emplacement de schéma dans les attributs `xsi:schemaLocation` et `xsi:noNamespaceSchemaLocation` dans les documents d'instance XML ou XBRL. Il s'agit de la **valeur par défaut**.
- La valeur `load-by-namespace` prend la part d'espace de nom `xsi:schemaLocation` et un string vide dans le cas de `xsi:noNamespaceSchemaLocation` et localise le schéma par le biais d'un mappage de catalogue.
- Si `load-combining-both` est utilisé et si soit la partie espace de noms ou la partie URL a un mappage de catalogue, alors le mappage de catalogue est utilisé. Si tous deux ont des mappages de catalogue, alors la valeur de l'option de `--schema-mapping` (option XBRL et option XML/XSD) de quel mappage utiliser. Si ni l'espace de noms ni l'URL n'a un mappage de catalogue, l'URL est utilisée.
- Si la valeur de l'option est `ignore`, les attributs `xsi:schemaLocation` et `xsi:noNamespaceSchemaLocation` seront ignorés tous les deux.

script

--script = *FILE*
Exécute le script Python dans le fichier soumis une fois que la validation a été terminée. Ajouter l'option plusieurs fois pour spécifier plus d'un seul script.

script-api-version

--api, --script-api-version = `1|2|2.1|2.2|2.3|2.4`
Spécifie la version API Python à utiliser pour le script. La valeur par défaut est la dernière version, actuellement **`2.4`**. Au lieu des valeurs `1` et `2`, vous pouvez aussi utiliser les valeurs `1.0` et `2.0`, respectivement.

script-output

--script-output = *FILE*

Rédige la sortie standard du script dans le fichier nommé dans *FILE*.

- script-param

  --script-param = *KEY:VALUE*
  Des paramètres spécifiés par l'utilisateur supplémentaires qui peuvent être accédés au cours de l'exécution des scripts Python. Ajouter l'option plusieurs fois pour spécifier plus d'un paramètre de script.

- Catalogues et ressources globales
  - catalog

    --catalog = *FILE*
    Spécifie le chemin absolu vers un fichier de catalogue root qui n'est pas le fichier catalogue root installé. La valeur par défaut est le chemin absolu vers le fichier de catalogue root installé (`<installation-folder>\Altova\RaptorXMLXBRLServer2018\etc\RootCatalog.xml`). Voir la section, Catalogues XML, pour plus d'informations à propos du travail avec les catalogues.

  - user-catalog

    --user-catalog = *FILE*
    Spécifie le chemin absolu vers un catalogue XML à utiliser en plus du catalogue root. Voir la section, Catalogues XML, pour plus d'informations concernant le travail avec des catalogues.

  - enable-globalresources

    --enable-globalresources = `true|false`
    Active les ressources globales. La valeur par défaut est `false`.
    Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

  - globalresourceconfig [gc]

    --gc | --globalresourceconfig = *VALUE*
    Spécifie la configuration active de la ressource globale (et active les ressources globales).

  - globalresourcefile [gr]

    --gr | --globalresourcefile = *FILE*
    Spécifie le fichier de ressource globale (et active les ressources globales).

- Messages, erreurs, aide, timeout, version
  - error-format

    --error-format = `text|shortxml|longxml`
    Spécifie le format de la sortie d'erreur. La valeur par défaut est `text`. Les autres options génèrent des formats XML, avec `longxml` générant plus de détails.

 © 2018 Altova GmbH

- error-limit

  `--error-limit =` *`N`*
  Spécifie la limite d'erreur avec une plage de valeur de `1` à `999`. La valeur par défaut est `100`. Le traitement s'arrête lorsque la limite d'erreur est atteinte. Utile pour limiter l'utilisation du processeur pendant la validation/transformation.

- help

  `--help`
  Affiche le texte d'aide pour la commande. Par exemple, `valany --h`. (en alternative, la commande `help` peut être utilisée avec un argument. Par exemple : `help valany`.)

- log-output

  `--log-output =` *`FILE`*
  Écrit la sortie de log dans l'URL de fichier spécifié. Veuillez vous assurer que la CLI a une permission d'écriture dans l'emplacement de la sortie.

- network-timeout

  `--network-timeout =` *`VALUE`*
  Spécifie le timeout en secondes pour les opérations I/O à distance. La valeur par défaut est : `40`.

- verbose

  `--verbose =` `true|false`
  Une valeur de `true` permet la sortie d'informations supplémentaires pendant la validation. La valeur par défaut est `false`.
  Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- verbose-output

  `--verbose-output =` *`FILE`*
  Écrit la sortie verbeuse sur *`FILE`*.

- version

  `--version`
  Affiche la version de RaptorXML+XBRL Server. Si utilisé avec une commande, placer `--version` avant la commande.

- warning-limit

  `--warning-limit =` *`VALUE`*
  Spécifie la limite d'avertissement dans la plage `1-65535`. Le traitement se poursuit si cette limite a été atteinte, mais d'autres avertissements ne sont pas rapportés. La valeur par défaut est `100`.

valxbrlver

La commande **valxbrlver** valide un ou plusieurs rapports de version XBRL conformément à la spécification XBRL Versioning 1.0 .

Windows	**RaptorXMLXBRL valxbrlver [options] *InputFile***
Linux	**raptorxmlxbrl valxbrlver [options] *InputFile***
Mac	**raptorxmlxbrl valxbrlver [options] *InputFile***

L'argument *InputFile* est le rapport de version XBRL qui est un document XML (avec une extension `.xml`). Pour valider plusieurs documents, il faut soit : (i) lister les fichiers à valider sur la CLI, chaque fichier étant séparé du suivant par un espace ; ou (ii) lister les fichiers à valider dans un fichier texte (fichier `.txt`), avec un nom de fichier par ligne, et fournir ce fichier de texte en tant que l'argument *InputFile* avec l'option `--listfile` définie sur `true` (*voir la liste des Options ci-dessous*).

Exemples

- **raptorxmlxbrl** valxbrlver c:\MyXBRLVersioningReport.xml
- **raptorxmlxbrl** valxbrlver c:\MyXBRLVersioningReports.txt

Casse et lignes obliques sur la ligne de commande

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

* Veuillez noter que les caractères minuscules (`raptorxmlxbrl`) fonctionnent sur toutes les plates-formes (Windows, Linux et Mac), alors qu'une écriture en majuscule/minuscule (`RaptorXMLXBRL`) ne fonctionne que sur Windows et Mac.
* Utiliser les barres obliques penchées vers l'avant pour Linux et Mac, et les barres obliques penchées vers l'arrière sur Windows.

Options

Les options de la commande sont listées ci-dessous, et organisées en groupes. Les valeurs peuvent être spécifiées sans guillemets sauf dans deux cas : (i) lorsque le string de valeur contient des espaces, ou (ii) si la présence de guillemets est explicitement requise dans la description de l'option.

Messages, erreurs, aide, timeout, version

error-format

--error-format = `text|shortxml|longxml`
Spécifie le format de la sortie d'erreur. La valeur par défaut est `text`. Les autres options

génèrent des formats XML, avec `longxml` générant plus de détails.

- error-limit

  `--error-limit =` *`N`*
  Spécifie la limite d'erreur avec une plage de valeur de `1` à `999`. La valeur par défaut est `100`. Le traitement s'arrête lorsque la limite d'erreur est atteinte. Utile pour limiter l'utilisation du processeur pendant la validation/transformation.

- help

  `--help`
  Affiche le texte d'aide pour la commande. Par exemple, `valany --h`. (en alternative, la commande `help` peut être utilisée avec un argument. Par exemple : `help valany`.)

- log-output

  `--log-output =` *`FILE`*
  Écrit la sortie de log dans l'URL de fichier spécifié. Veuillez vous assurer que la CLI a une permission d'écriture dans l'emplacement de la sortie.

- network-timeout

  `--network-timeout =` *`VALUE`*
  Spécifie le timeout en secondes pour les opérations I/O à distance. La valeur par défaut est : `40`.

- verbose

  `--verbose =` `true|false`
  Une valeur de `true` permet la sortie d'informations supplémentaires pendant la validation. La valeur par défaut est `false`.
  Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- verbose-output

  `--verbose-output =` *`FILE`*
  Écrit la sortie verbeuse sur *`FILE`*.

- version

  `--version`
  Affiche la version de RaptorXML+XBRL Server. Si utilisé avec une commande, placer `--version` avant la commande.

- warning-limit

  `--warning-limit =` *`VALUE`*
  Spécifie la limite d'avertissement dans la plage `1-65535`. Le traitement se poursuit si cette limite a été atteinte, mais d'autres avertissements ne sont pas rapportés. La valeur par défaut est `100`.

- Catalogues et ressources globales

- catalog

  --catalog = *FILE*
  Spécifie le chemin absolu vers un fichier de catalogue root qui n'est pas le fichier catalogue root installé. La valeur par défaut est le chemin absolu vers le fichier de catalogue root installé (`<installation-folder>\Altova\RaptorXMLXBRLServer2018\etc\RootCatalog.xml`). Voir la section, Catalogues XML, pour plus d'informations à propos du travail avec les catalogues.

- user-catalog

  --user-catalog = *FILE*
  Spécifie le chemin absolu vers un catalogue XML à utiliser en plus du catalogue root. Voir la section, Catalogues XML, pour plus d'informations concernant le travail avec des catalogues.

- enable-globalresources

  --enable-globalresources = `true|false`
  Active les ressources globales. La valeur par défaut est `false`.
  Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- globalresourceconfig [gc]

  --gc | --globalresourceconfig = *VALUE*
  Spécifie la configuration active de la ressource globale (et active les ressources globales).

- globalresourcefile [gr]

  --gr | --globalresourcefile = *FILE*
  Spécifie le fichier de ressource globale (et active les ressources globales).

- Schéma XML

  - schema-imports

    --schema-imports = `load-by-schemalocation | load-preferring-schemalocation | load-by-namespace | load-combining-both | license-namespace-only`
    Spécifie le comportement des éléments `xs:import`, chacun d'entre eux ayant un attribut `namespace` optionnel et un attribut `schemaLocation` optionnel : `<import namespace="someNS" schemaLocation="someURL">`. L'option spécifie s'il faut charger un document de schéma ou juste mettre un espace de noms sous licence, et, si un document de schéma doit être chargé, quelle information doit être utilisée pour le trouver. Défaut : `load-preferring-schemalocation`.
    Le comportement est le suivant :

    - `load-by-schemalocation`: La valeur de l'attribut `schemaLocation` est utilisée pour situer le schéma, en prenant en compte les mappages de catalogue. Si l'attribut de l'espace de nom est présent, l'espace de noms est importé (licencé).
    - `load-preferring-schemalocation`: Si l'attribut `schemaLocation` est présent, il est utilisé en prenant en compte les mappages de catalogue. Si l'attribut

 © 2018 Altova GmbH

schemaLocation est présent, la valeur de l'attribut namespace est utilisée via un mappage de catalogue. C'est la **valeur par défaut**.

- load-by-namespace: La valeur de l'attribut namespace est utilisée pour situer le schéma via un mappage de catalogue.
- load-combining-both: Si soit l'Attribut namespace ou l'attribut schemaLocation a un mappage de catalogue, le mappage est utilisé. Si les deux ont des mappages de catalogue, alors la valeur de l'option --schema-mapping (option XBRL et option XML/XSD) décide de quel mappage utiliser. Si aucun mappage de catalogue n'est présent, l'attribut schemaLocation est utilisé.
- license-namespace-only: Le nom d'espace est importé. Aucun document de schéma n'est importé.

schema-mapping

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

Si l'emplacement de schéma et l'espace de noms sont tous les deux utilisés pour trouver un document de schéma, spécifie lequel des deux doit être utilisé pendant la consultation du catalogue. (Si soit l'option --schemalocation-hints ou l'option --schema-imports a une valeur de load-combining-both, et si les parties d'espace de noms et d'URL impliquées ont toutes les deux des mappages de catalogue, alors la valeur de cette option spécifie lequel des deux mappages utiliser (mappage d'espace de noms ou mappage URL ; la valeur prefer-schemalocation réfère au mappage URL).) Défaut : prefer-schemalocation.

schemalocation-hints

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

Spécifie le comportement des attributs xsi:schemaLocation et xsi:noNamespaceSchemaLocation : S'il faut charger un document de schéma et, si oui, quelle information doit être utilisée pour la trouver. Défaut : load-by-schemalocation.

- La valeur load-by-schemalocation utilise l'URL de l'emplacement de schéma dans les attributs xsi:schemaLocation et xsi:noNamespaceSchemaLocation dans les documents d'instance XML ou XBRL. Il s'agit de la **valeur par défaut**.
- La valeur load-by-namespace prend la part d'espace de nom xsi:schemaLocation et un string vide dans le cas de xsi:noNamespaceSchemaLocation et localise le schéma par le biais d'un mappage de catalogue.
- Si load-combining-both est utilisé et si soit la partie espace de noms ou la partie URL a un mappage de catalogue, alors le mappage de catalogue est utilisé. Si tous deux ont des mappages de catalogue, alors la valeur de l'option de --schema-mapping (option XBRL et option XML/XSD) de quel mappage utiliser. Si ni l'espace de noms ni l'URL n'a un mappage de catalogue, l'URL est utilisée.
- Si la valeur de l'option est ignore, les attributs xsi:schemaLocation et xsi:noNamespaceSchemaLocation seront ignorés tous les deux.

Divers

listfile

--listfile = `true|false`
Si `true`, traiter l'argument *`InputFile`* de la commande en tant que fichier de texte contenant un nom de fichier par ligne. La valeur par défaut est `false`. (Une alternative est de lister les fichiers sur la CLI avec un espace en tant que séparateur. Veuillez noter, néanmoins, que les CLI ont une limitation de caractères maximum.) Veuillez noter que l'option `--listfile` s'applique uniquement aux arguments, et non pas aux options.
Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- xinclude

--xinclude = `true|false`
Active la prise en charge de XML Inclusions (XInclude). La valeur par défaut est `false`. Si `false`, les élément `include` d'XInclude sont ignorés.
Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

 © 2018 Altova GmbH

3.4 Commandes XSLT

Les commandes XSLT sont :

- xslt: pour transformer les documents XML avec un document XSLT
- valxslt: pour valiser des documents XSLT

Les arguments et les options pour chaque commande sont listés dans les sous-sections, xslt et valxslt.

xslt

La commande **xslt** prend un fichier XSLT en tant que son argument unique et l'utilise pour transformer un fichier XML d'entrée pour produire un fichier de sortie. Les fichiers d'entrée et de sortie sont spécifiés en tant qu'options.

Windows	**RaptorXMLXBRL xslt [options] *XSLT-File***
Linux	**raptorxmlxbrl xslt [options] *XSLT-File***
Mac	**raptorxmlxbrl xslt [options] *XSLT-File***

L'argument *XSLT-File* est le chemin et le nom du fichier XSLT à utiliser pour la transformation. Un fichier XML d'entrée (--input) ou un point d'entrée de modèle nommé (--template-entry-point) est nécessaire. Si aucune option --output n'est spécifiée, la sortie est écrite dans la sortie standard. Vous pouvez utiliser XSLT 1.0, 2.0 ou 3.0. XSLT 3.0 est la spécification utilisée par défaut.

Exemples

- **raptorxmlxbrl** xslt --input=c:\Test.xml --output=c:\Output.xml c:\Test.xslt
- **raptorxmlxbrl** xslt --template-entry-point=StartTemplate --output=c:\Output.xml c:\Test.xslt
- **raptorxmlxbrl** xslt --input=c:\Test.xml --output=c:\Output.xml --param=date://node[1]/@att1 --p=title:'stringwithoutspace' --param=title:"'string with spaces'" --p=amount:456 c:\Test.xslt

▼ Casse et lignes obliques sur la ligne de commande

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

* Veuillez noter que les caractères minuscules (raptorxmlxbrl) fonctionnent sur toutes les plates-formes (Windows, Linux et Mac), alors qu'une écriture en majuscule/minuscule (RaptorXMLXBRL) ne fonctionne que sur Windows et Mac.
* Utiliser les barres obliques penchées vers l'avant pour Linux et Mac, et les barres obliques penchées vers l'arrière sur Windows.

Options

Les options de la commande sont listées ci-dessous, et organisées en groupes. Les valeurs peuvent être spécifiées sans guillemets sauf dans deux cas : (i) lorsque le string de valeur contient des espaces, ou (ii) si la présence de guillemets est explicitement requise dans la description de l'option.

▼ Traitement XSLT

 © 2018 Altova GmbH

- indent-characters

  --indent-characters = *VALUE*

  Spécifie le string de caractère à utiliser en tant que retrait.

- input

  --input = *FILE*

  L'URL du fichier XML à transformer.

- output, xsltoutput

  output = *FILE*, **xsltoutput =** *FILE*

  L'URL du fichier sortie-primaire. Par exemple, dans le cas d'une sortie HTML de fichier multiple, le fichier de sortie primaire sera l'emplacement du fichier HTML du point d'entrée. Les fichiers de sortie supplémentaires, comme des fichiers d'image générée, sont rapportés en tant que `xslt-additional-output-files`. Si aucune option `--output` ou `--xsltoutput` n'est spécifiée, la sortie est écrite sur la sortie standard.

- param [p]

  --p | --param = *KEY:VALUE*

  - ***XQuery***

    Spécifie la valeur d'un paramètre externe. Un paramètre externe est déclaré dans le document XQuery avec la déclaration `declare variable` suivi par un nom de variable puis le mot-clé `external` suivi par le point-virgule de fin de ligne. Par exemple :

    ```
    declare variable $foo as xs:string external;
    ```

    En raison du mot-clé `external`, `$foo` devient un paramètre externe, dont la valeur est passée pendant le runtime depuis une source externe. Le paramètre externe reçoit une value avec la commande CLI. Par exemple :

    ```
    --param=foo:'MyName'
    ```

    Dans l'instruction de description ci-dessus, *KEY* est le nom de paramètre externe, *VALUE* est la valeur du paramètre externe, donné en tant qu'une expression XPath. Les noms de paramètres utilisés sur la CLI doivent être déclarés dans le document XQuery. Si plusieurs paramètres externes sont des valeurs passées sur la CLI, chacun doit être recevoir une option séparée `--param`. Les doubles guillemets doivent être utilisés si l'expression XPath contient des espaces.

  - ***XSLT***

    Spécifie paramètre de feuille de style global. *KEY* est le nom de paramètre, *VALUE* est une expression XPath qui fournit la valeur de paramètre. Les noms de paramètre utilisés sur la CLI doivent être déclarés dans la feuille de style. Si plusieurs paramètres multiples sont utilisés, le changement `--param` doit être utilisé avant chaque paramètre. Les double guillemets doivent être utilisés autour de l'expression XPath si elle contient un espace ; que l'espace se trouve dans l'expression XPath elle-même ou dans un littéral de string dans l'expression. Par exemple :

    ```
    raptorxmlxbrl xslt --input=c:\Test.xml --output=c:\Output.xml --
    param=date://node[1]/@att1 --p=title:'stringwithoutspace' --
    param=title:"'string with spaces'" --p=amount:456 c:\Test.xslt
    ```

- streaming-serialization-enabled

  --streaming-serialization-enabled = `true|false`
  Active la sérialisation de streaming. La valeur par défaut est `true`.
  Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- initial-template, template-entry-point

  --template-entry-point = *VALUE*
  Donne le nom d'un modèle nommé dans la feuille de style XSLT qui est le point d'entrée de la transformation.

- initial-mode, template-mode

  --template-mode = *VALUE*
  Spécifie le mode de modèle à utiliser pour la transformation.

- xslt-version

  --xslt-version = `1|1.0|2|2.0|3|3.0|3.1`
  Spécifie si le processeur XSLT devrait utiliser XSLT 1.0, XSLT 2.0 ou XSLT 3.0. La valeur par défaut est `3`.

- Schéma XML et instance XML
  - load-xml-with-psvi

    --load-xml-with-psvi = `true|false`
    Active la validation des fichiers XML d'entrée et génère des informations de post-schema-validation pour les fichiers. La valeur par défaut est : `true`.

  - schema-imports

    --schema-imports = `load-by-schemalocation | load-preferring-schemalocation | load-by-namespace | load-combining-both | license-namespace-only`
    Spécifie le comportement des éléments `xs:import`, chacun d'entre eux ayant un attribut `namespace` optionnel et un attribut `schemaLocation` optionnel : `<import namespace="someNS" schemaLocation="someURL">`. L'option spécifie s'il faut charger un document de schéma ou juste mettre un espace de noms sous licence, et, si un document de schéma doit être chargé, quelle information doit être utilisée pour le trouver. Défaut : `load-preferring-schemalocation`.
    Le comportement est le suivant :

    - `load-by-schemalocation`: La valeur de l'attribut `schemaLocation` est utilisée pour situer le schéma, en prenant en compte les mappages de catalogue. Si l'attribut de l'espace de nom est présent, l'espace de noms est importé (licencé).
    - `load-preferring-schemalocation`: Si l'attribut `schemaLocation` est présent, il est utilisé en prenant en compte les mappages de catalogue. Si l'attribut `schemaLocation` est présent, la valeur de l'attribut `namespace` est utilisée via un mappage de catalogue. C'est la **valeur par défaut**.

- `load-by-namespace`: La valeur de l'attribut `namespace` est utilisée pour situer le schéma via un mappage de catalogue.
- `load-combining-both`: Si soit l'Attribut `namespace` ou l'attribut `schemaLocation` a un mappage de catalogue, le mappage est utilisé. Si les deux ont des mappages de catalogue, alors la valeur de l'option `--schema-mapping` (option XBRL et option XML/XSD) décide de quel mappage utiliser. Si aucun mappage de catalogue n'est présent, l'attribut `schemaLocation` est utilisé.
- `license-namespace-only`: Le nom d'espace est importé. Aucun document de schéma n'est importé.

schemalocation-hints

`--schemalocation-hints =` `load-by-schemalocation | load-by-namespace | load-combining-both | ignore`
Spécifie le comportement des attributs `xsi:schemaLocation` et `xsi:noNamespaceSchemaLocation` : S'il faut charger un document de schéma et, si oui, quelle information doit être utilisée pour la trouver. Défaut : `load-by-schemalocation`.

- La valeur `load-by-schemalocation` utilise l'URL de l'emplacement de schéma dans les attributs `xsi:schemaLocation` et `xsi:noNamespaceSchemaLocation` dans les documents d'instance XML ou XBRL. Il s'agit de la **valeur par défaut**.
- La valeur `load-by-namespace` prend la part d'espace de nom `xsi:schemaLocation` et un string vide dans le cas de `xsi:noNamespaceSchemaLocation` et localise le schéma par le biais d'un mappage de catalogue.
- Si `load-combining-both` est utilisé et si soit la partie espace de noms ou la partie URL a un mappage de catalogue, alors le mappage de catalogue est utilisé. Si tous deux ont des mappages de catalogue, alors la valeur de l'option de `--schema-mapping` (option XBRL et option XML/XSD) de quel mappage utiliser. Si ni l'espace de noms ni l'URL n'a un mappage de catalogue, l'URL est utilisée.
- Si la valeur de l'option est `ignore`, les attributs `xsi:schemaLocation` et `xsi:noNamespaceSchemaLocation` seront ignorés tous les deux.

schema-mapping

`--schema-mapping =` `prefer-schemalocation | prefer-namespace`
Si l'emplacement de schéma et l'espace de noms sont tous les deux utilisés pour trouver un document de schéma, spécifie lequel des deux doit être utilisé pendant la consultation du catalogue. (Si soit l'option `--schemalocation-hints` ou l'option `--schema-imports` a une valeur de `load-combining-both`, et si les parties d'espace de noms et d'URL impliquées ont toutes les deux des mappages de catalogue, alors la valeur de cette option spécifie lequel des deux mappages utiliser (mappage d'espace de noms ou mappage URL ; la valeur `prefer-schemalocation` réfère au mappage URL).) Défaut : `prefer-schemalocation`.

xinclude

`--xinclude =` `true|false`
Active la prise en charge de XML Inclusions (XInclude). La valeur par défaut est `false`. Si `false`, les élément `include` d'XInclude sont ignorés.
Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- xml-mode

  --xml-mode = wf|id|valid

  Spécifie le mode de traitement XML à utiliser : wf=wellformed check (vérification de la bonne forme); id=bien formé avec des contrôles ID/IDREF ; valid=validation. La valeur par défaut est wf. Veuillez noter qu'une valeur de valid exige que chaque document de schéma/instance chargé depuis les traitements référence un DTD. Si aucun DTD n'existe, une erreur est rapportée.

- xml-mode-for-schemas

  --xml-mode-for-schemas = wf|id|valid

  Spécifie le mode de traitement XML à utiliser pour les documents de schéma XML : wf=vérification de la bonne formation ; id=contrôles de la bonne formation avec ID/IDREF ; valid=validation. La valeur par défaut est wf. Veuillez noter qu'une valeur de valid exige que chaque document de schéma chargé pendant le traitement référence un DTD. Si aucun DTD n'existe, une erreur est rapportée.

- xml-validation-error-as-warning

  --xml-validation-error-as-warning = true|false

  Si true, traite les erreurs de validation en tant qu'avertissements. Si les erreurs sont traitées en tant qu'avertissements, les traitements supplémentaires, comme des transformations XSLT, continueront quelles que soient les erreurs. La valeur par défaut est false.

- xsd

  --xsd = *FILE*

  Spécifie un ou plusieurs documents de Schéma XML à utiliser pour la validation des documents d'instance XML. Ajouter l'option plusieurs fois pour spécifier plus d'un document de schéma.

- xsd-version

  --xsd-version = 1.0|1.1|detect

  Spécifie la version du langage de définition de Schéma W3C (XSD) à utiliser. Le défaut est 1.0. Cette option peut être utile pour découvrir comment un schéma compatible à 1.0 n'est pas compatible à 1.1. L'option detect est une fonction spécifique à Altova. Elle permet la détection de la version du document de Schéma XML (1.0 ou 1.1) en lisant la valeur de l'attribut vc:minVersion de l'élément du document <xs:schema>. Si la valeur de l'attribut @vc:minVersion est 1.1, le schéma est détecté comme étant de version 1.1. Pour toute autre valeur, ou si l'attribut @vc:minVersion est absent, le schéma est détecté comme étant de version 1.0.

- Catalogues et ressources globales
  - catalog

    --catalog = *FILE*

    Spécifie le chemin absolu vers un fichier de catalogue root qui n'est pas le fichier catalogue root installé. La valeur par défaut est le chemin absolu vers le fichier de

catalogue root installé (`<installation-folder>\Altova\RaptorXMLXBRLServer2018\etc\RootCatalog.xml`). Voir la section, Catalogues XML, pour plus d'informations à propos du travail avec les catalogues.

- user-catalog

  `--user-catalog =` *`FILE`*
  Spécifie le chemin absolu vers un catalogue XML à utiliser en plus du catalogue root. Voir la section, Catalogues XML, pour plus d'informations concernant le travail avec des catalogues.

- enable-globalresources

  `--enable-globalresources =` `true|false`
  Active les ressources globales. La valeur par défaut est `false`.
  Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- globalresourceconfig [gc]

  `--gc | --globalresourceconfig =` *`VALUE`*
  Spécifie la configuration active de la ressource globale (et active les ressources globales).

- globalresourcefile [gr]

  `--gr | --globalresourcefile =` *`FILE`*
  Spécifie le fichier de ressource globale (et active les ressources globales).

- Extensions

  Ces options définissent la gestion des fonctions d'extension spéciales disponibles dans un certain nombre de produits Altova de niveau d'entreprise (comme XMLSpy Enterprise Edition). Leur utilisation est décrite dans les manuels d'utilisateur de ces produits.

  - chartext-disable

    `--chartext-disable =` `true|false`
    Désactive les extensions de graphique. La valeur par défaut est `false`.
    Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

  - dotnetext-disable

    `--dotnetext-disable =` `true|false`
    Désactive les extensions .NET. La valeur par défaut est `false`.
    Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

  - jvm-location

    `--jvm-location =` *`FILE`*
    `FILE` spécifie l'emplacement de la Java Virtual Machine (DLL sur Windows, objet partagé sur Linux). La JVM est nécessaire si vous utilisez les fonctions d'extension Java dans votre code XSLT/XQuery. La valeur par défaut est `false`.

- javaext-barcode-location

  --javaext-barcode-location = *FILE*
  Spécifie le chemin vers le dossier qui contient le fichier d'extension du code-barres `AltovaBarcodeExtension.jar`. Le chemin doit être donné dans une des formes suivantes :

  - Un URI de fichier, par exemple : `--javaext-barcode-location="file:///C:/Program Files/Altova/RaptorXMLXBRLServer2018/etc/jar/"`
  - Un chemin Windows avec un échappement par barres obliques inversées, par exemple : `--javaext-barcode-location="C:\\Program Files\\Altova\\RaptorXMLXBRLServer2018\\etc\\jar\\"`

- javaext-disable

  --javaext-disable = `true|false`
  Désactive les extensions Java. La valeur par défaut est `false`.
  Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- Messages, erreurs, aide, timeout, version
  - error-format

    --error-format = `text|shortxml|longxml`
    Spécifie le format de la sortie d'erreur. La valeur par défaut est `text`. Les autres options génèrent des formats XML, avec `longxml` générant plus de détails.

  - error-limit

    --error-limit = *N*
    Spécifie la limite d'erreur avec une plage de valeur de `1` à `999`. La valeur par défaut est `100`. Le traitement s'arrête lorsque la limite d'erreur est atteinte. Utile pour limiter l'utilisation du processeur pendant la validation/transformation.

  - help

    --help
    Affiche le texte d'aide pour la commande. Par exemple, `valany --h`. (en alternative, la commande `help` peut être utilisée avec un argument. Par exemple : `help valany`.)

  - network-timeout

    --network-timeout = *VALUE*
    Spécifie le timeout en secondes pour les opérations I/O à distance. La valeur par défaut est : `40`.

  - verbose

    --verbose = `true|false`
    Une valeur de `true` permet la sortie d'informations supplémentaires pendant la validation. La valeur par défaut est `false`.
    Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est

spécifiée sans une valeur.

- verbose-output

  `--verbose-output =` *`FILE`*
  Écrit la sortie verbeuse sur *`FILE`*.

- version

  `--version`
  Affiche la version de RaptorXML+XBRL Server. Si utilisé avec une commande, placer `--version` avant la commande.

valxslt

La commande **valxslt** prend un fichier XSLT en tant que son argument unique et le valide.

Windows	**RaptorXMLXBRL valxslt [options]** ***XSLT-File***
Linux	**raptorxmlxbrl valxslt [options]** ***XSLT-File***
Mac	**raptorxmlxbrl valxslt [options]** ***XSLT-File***

L'argument *XSLT-File* est le chemin et le nom du fichier XSLT à utiliser pour la validation. La validation peut s'effectuer conformément à la spécification XSLT 1.0, 2.0 ou 3.0. XSLT 3.0 est la spécification utilisée par défaut.

Exemples

- **raptorxmlxbrl** valxslt c:\Test.xslt
- **raptorxmlxbrl** valxslt --xslt-version=2 c:\Test.xslt

Casse et lignes obliques sur la ligne de commande

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

* Veuillez noter que les caractères minuscules (raptorxmlxbrl) fonctionnent sur toutes les plates-formes (Windows, Linux et Mac), alors qu'une écriture en majuscule/minuscule (RaptorXMLXBRL) ne fonctionne que sur Windows et Mac.
* Utiliser les barres obliques penchées vers l'avant pour Linux et Mac, et les barres obliques penchées vers l'arrière sur Windows.

Options

Les options de la commande sont listées ci-dessous, et organisées en groupes. Les valeurs peuvent être spécifiées sans guillemets sauf dans deux cas : (i) lorsque le string de valeur contient des espaces, ou (ii) si la présence de guillemets est explicitement requise dans la description de l'option.

Traitement XSLT

initial-mode, template-mode

--template-mode = *VALUE*
Spécifie le mode de modèle à utiliser pour la transformation.

initial-template, template-entry-point

--template-entry-point = *VALUE*
Donne le nom d'un modèle nommé dans la feuille de style XSLT qui est le point d'entrée

 © 2018 Altova GmbH

de la transformation.

- xslt-version

  --xslt-version = 1|1.0|2|2.0|3|3.0|3.1
  Spécifie si le processeur XSLT devrait utiliser XSLT 1.0, XSLT 2.0 ou XSLT 3.0. La valeur par défaut est 3.

- Schéma XML et instance XML
  - load-xml-with-psvi

    --load-xml-with-psvi = true|false
    Active la validation des fichiers XML d'entrée et génère des informations de post-schema-validation pour les fichiers. La valeur par défaut est : true.

  - schema-imports

    --schema-imports = load-by-schemalocation | load-preferring-schemalocation | load-by-namespace | load-combining-both | license-namespace-only
    Spécifie le comportement des éléments xs:import, chacun d'entre eux ayant un attribut namespace optionnel et un attribut schemaLocation optionnel : <import namespace="someNS" schemaLocation="someURL">. L'option spécifie s'il faut charger un document de schéma ou juste mettre un espace de noms sous licence, et, si un document de schéma doit être chargé, quelle information doit être utilisée pour le trouver. Défaut : load-preferring-schemalocation.
    Le comportement est le suivant :

    - load-by-schemalocation: La valeur de l'attribut schemaLocation est utilisée pour situer le schéma, en prenant en compte les mappages de catalogue. Si l'attribut de l'espace de nom est présent, l'espace de noms est importé (licencé).
    - load-preferring-schemalocation: Si l'attribut schemaLocation est présent, il est utilisé en prenant en compte les mappages de catalogue. Si l'attribut schemaLocation est présent, la valeur de l'attribut namespace est utilisée via un mappage de catalogue. C'est la **valeur par défaut**.
    - load-by-namespace: La valeur de l'attribut namespace est utilisée pour situer le schéma via un mappage de catalogue.
    - load-combining-both: Si soit l'Attribut namespace ou l'attribut schemaLocation a un mappage de catalogue, le mappage est utilisé. Si les deux ont des mappages de catalogue, alors la valeur de l'option --schema-mapping (option XBRL et option XML/XSD) décide de quel mappage utiliser. Si aucun mappage de catalogue n'est présent, l'attribut schemaLocation est utilisé.
    - license-namespace-only: Le nom d'espace est importé. Aucun document de schéma n'est importé.

  - schemalocation-hints

    --schemalocation-hints = load-by-schemalocation | load-by-namespace | load-combining-both | ignore
    Spécifie le comportement des attributs xsi:schemaLocation et xsi:noNamespaceSchemaLocation : S'il faut charger un document de schéma et, si oui, quelle information doit être utilisée pour la trouver. Défaut : load-by-

schemalocation.

- La valeur load-by-schemalocation utilise l'URL de l'emplacement de schéma dans les attributs xsi:schemaLocation et xsi:noNamespaceSchemaLocation dans les documents d'instance XML ou XBRL. Il s'agit de la **valeur par défaut**.
- La valeur load-by-namespace prend la part d'espace de nom xsi:schemaLocation et un string vide dans le cas de xsi:noNamespaceSchemaLocation et localise le schéma par le biais d'un mappage de catalogue.
- Si load-combining-both est utilisé et si soit la partie espace de noms ou la partie URL a un mappage de catalogue, alors le mappage de catalogue est utilisé. Si tous deux ont des mappages de catalogue, alors la valeur de l'option de --schema-mapping (option XBRL et option XML/XSD) de quel mappage utiliser. Si ni l'espace de noms ni l'URL n'a un mappage de catalogue, l'URL est utilisée.
- Si la valeur de l'option est ignore, les attributs xsi:schemaLocation et xsi:noNamespaceSchemaLocation seront ignorés tous les deux.

schema-mapping

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

Si l'emplacement de schéma et l'espace de noms sont tous les deux utilisés pour trouver un document de schéma, spécifie lequel des deux doit être utilisé pendant la consultation du catalogue. (Si soit l'option --schemalocation-hints ou l'option --schema-imports a une valeur de load-combining-both, et si les parties d'espace de noms et d'URL impliquées ont toutes les deux des mappages de catalogue, alors la valeur de cette option spécifie lequel des deux mappages utiliser (mappage d'espace de noms ou mappage URL ; la valeur prefer-schemalocation réfère au mappage URL).)
Défaut : prefer-schemalocation.

xinclude

--xinclude = true|false

Active la prise en charge de XML Inclusions (XInclude). La valeur par défaut est false. Si false, les élément include d'XInclude sont ignorés.
Note : les valeurs d'option booléennes sont configurées sur true si l'option est spécifiée sans une valeur.

xml-mode

--xml-mode = wf|id|valid

Spécifie le mode de traitement XML à utiliser : wf=wellformed check (vérification de la bonne forme); id=bien formé avec des contrôles ID/IDREF ; valid=validation. La valeur par défaut est wf. Veuillez noter qu'une valeur de valid exige que chaque document de schéma/instance chargé depuis les traitements référence un DTD. Si aucun DTD n'existe, une erreur est rapportée.

xml-mode-for-schemas

--xml-mode-for-schemas = wf|id|valid

Spécifie le mode de traitement XML à utiliser pour les documents de schéma XML : wf=vérification de la bonne formation ; id=contrôles de la bonne formation avec ID/IDREF ; valid=validation. La valeur par défaut est wf. Veuillez noter qu'une valeur de valid exige que chaque document de schéma chargé pendant le traitement référence

un DTD. Si aucun DTD n'existe, une erreur est rapportée.

- xsd-version

  --xsd-version = `1.0|1.1|detect`
  Spécifie la version du langage de définition de Schéma W3C (XSD) à utiliser. Le défaut est `1.0`. Cette option peut être utile pour découvrir comment un schéma compatible à 1.0 n'est pas compatible à 1.1. L'option `detect` est une fonction spécifique à Altova. Elle permet la détection de la version du document de Schéma XML (`1.0` ou `1.1`) en lisant la valeur de l'attribut `vc:minVersion` de l'élément du document `<xs:schema>`. Si la valeur de l'attribut `@vc:minVersion` est `1.1`, le schéma est détecté comme étant de version `1.1`. Pour toute autre valeur, ou si l'attribut `@vc:minVersion` est absent, le schéma est détecté comme étant de version `1.0`.

- Catalogues et ressources globales
  - catalog

    --catalog = *FILE*
    Spécifie le chemin absolu vers un fichier de catalogue root qui n'est pas le fichier catalogue root installé. La valeur par défaut est le chemin absolu vers le fichier de catalogue root installé (`<installation-folder>\Altova\RaptorXMLXBRLServer2018\etc\RootCatalog.xml`). Voir la section, Catalogues XML, pour plus d'informations à propos du travail avec les catalogues.

  - user-catalog

    --user-catalog = *FILE*
    Spécifie le chemin absolu vers un catalogue XML à utiliser en plus du catalogue root. Voir la section, Catalogues XML, pour plus d'informations concernant le travail avec des catalogues.

  - enable-globalresources

    --enable-globalresources = `true|false`
    Active les ressources globales. La valeur par défaut est `false`.
    Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

  - globalresourceconfig [gc]

    --gc | --globalresourceconfig = *VALUE*
    Spécifie la configuration active de la ressource globale (et active les ressources globales).

  - globalresourcefile [gr]

    --gr | --globalresourcefile = *FILE*
    Spécifie le fichier de ressource globale (et active les ressources globales).

- Extensions

  Ces options définissent la gestion des fonctions d'extension spéciales disponibles dans un

certain nombre de produits Altova de niveau d'entreprise (comme XMLSpy Enterprise Edition). Leur utilisation est décrite dans les manuels d'utilisateur de ces produits.

- chartext-disable

  `--chartext-disable =` `true|false`
  Désactive les extensions de graphique. La valeur par défaut est `false`.
  Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- dotnetext-disable

  `--dotnetext-disable =` `true|false`
  Désactive les extensions .NET. La valeur par défaut est `false`.
  Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- jvm-location

  `--jvm-location =` *`FILE`*
  `FILE` spécifie l'emplacement de la Java Virtual Machine (DLL sur Windows, objet partagé sur Linux). La JVM est nécessaire si vous utilisez les fonctions d'extension Java dans votre code XSLT/XQuery. La valeur par défaut est `false`.

- javaext-barcode-location

  `--javaext-barcode-location =` *`FILE`*
  Spécifie le chemin vers le dossier qui contient le fichier d'extension du code-barres `AltovaBarcodeExtension.jar`. Le chemin doit être donné dans une des formes suivantes :

  - Un URI de fichier, par exemple : `--javaext-barcode-location="file:///C:/Program Files/Altova/RaptorXMLXBRLServer2018/etc/jar/"`
  - Un chemin Windows avec un échappement par barres obliques inversées, par exemple : `--javaext-barcode-location="C:\\Program Files\\Altova\\RaptorXMLXBRLServer2018\\etc\\jar\\"`

- javaext-disable

  `--javaext-disable =` `true|false`
  Désactive les extensions Java. La valeur par défaut est `false`.
  Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

Messages, erreurs, aide, timeout, version

- error-format

  `--error-format =` `text|shortxml|longxml`
  Spécifie le format de la sortie d'erreur. La valeur par défaut est `text`. Les autres options génèrent des formats XML, avec `longxml` générant plus de détails.

- error-limit

  `--error-limit =` *`N`*

 © 2018 Altova GmbH

Spécifie la limite d'erreur avec une plage de valeur de `1` à `999`. La valeur par défaut est `100`. Le traitement s'arrête lorsque la limite d'erreur est atteinte. Utile pour limiter l'utilisation du processeur pendant la validation/transformation.

- help

  `--help`
  Affiche le texte d'aide pour la commande. Par exemple, `valany --h`. (en alternative, la commande `help` peut être utilisée avec un argument. Par exemple : `help valany`.)

- network-timeout

  `--network-timeout =` *`VALUE`*
  Spécifie le timeout en secondes pour les opérations I/O à distance. La valeur par défaut est : `40`.

- verbose

  `--verbose =` `true|false`
  Une valeur de `true` permet la sortie d'informations supplémentaires pendant la validation. La valeur par défaut est `false`.
  Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- verbose-output

  `--verbose-output =` *`FILE`*
  Écrit la sortie verbeuse sur *`FILE`*.

- version

  `--version`
  Affiche la version de RaptorXML+XBRL Server. Si utilisé avec une commande, placer `--version` avant la commande.

3.5 Commandes XQuery

Les commandes XQuery sont :

- xquery: pour exécuter des documents XQuery documents, en option, avec un document d'entrée
- xqueryupdate: pour exécuter un document XQuery update, utiliser un document XQuery et, en option, le document XML d'entrée à mettre à jour
- valxquery: pour valider des documents XQuery
- valxqueryupdate: pour valider un document XQuery (update)

Les arguments et les options pour chaque commande sont listés dans les sous-sections, xquery et valxslt.

 © 2018 Altova GmbH

xquery

La commande **xquery** prend un fichier XQuery en tant que son argument unique et l'exécute avec un fichier d'entrée optionnel pour produire un fichier de sortie. Les fichiers d'entrée et de sortie sont spécifiés en tant qu'options.

Windows	**RaptorXMLXBRL xquery [options] *XQuery-File***
Linux	**raptorxmlxbrl xquery [options] *XQuery-File***
Mac	**raptorxmlxbrl xquery [options] *XQuery-File***

L'argument ***XQuery-File*** est le chemin et le nom du fichier XQuery à exécuter. Vous pouvez utiliser XQuery 1.0 ou 3.0. XQuery 3.0 est utilisé par défaut.

Exemples

- **raptorxmlxbrl** xquery --output=c:\Output.xml c:\TestQuery.xq
- **raptorxmlxbrl** xquery --input=c:\Input.xml --output=c:\Output.xml --param=company:"Altova" --p=date:"2006-01-01" c:\TestQuery.xq
- **raptorxmlxbrl** xquery --input=c:\Input.xml --output=c:\Output.xml --param=source:" doc('c:\test\books.xml')//book "
- **raptorxmlxbrl** xquery --output=c:\Output.xml --omit-xml-declaration=false --output-encoding=ASCII c:\TestQuery.xq

Casse et lignes obliques sur la ligne de commande

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

* Veuillez noter que les caractères minuscules (raptorxmlxbrl) fonctionnent sur toutes les plates-formes (Windows, Linux et Mac), alors qu'une écriture en majuscule/minuscule (RaptorXMLXBRL) ne fonctionne que sur Windows et Mac.
* Utiliser les barres obliques penchées vers l'avant pour Linux et Mac, et les barres obliques penchées vers l'arrière sur Windows.

Options

Les options de la commande sont listées ci-dessous, et organisées en groupes. Les valeurs peuvent être spécifiées sans guillemets sauf dans deux cas : (i) lorsque le string de valeur contient des espaces, ou (ii) si la présence de guillemets est explicitement requise dans la description de l'option.

Traitement XQuery

indent-characters

--indent-characters = *VALUE*

Spécifie le string de caractère à utiliser en tant que retrait.

- input

  --input = *FILE*
  L'URL du fichier XML à transformer.

- omit-xml-declaration

  --omit-xml-declaration = `true|false`
  Option de sérialisation pour spécifier si la déclaration XML doit être omise de la sortie ou pas. Si `true`, il n'y aura pas de déclaration XML dans le document de sortie. Si `false`, une déclaration XML sera incluse. La valeur par défaut est `false`.
  Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- output, xsltoutput

  output = *FILE,* **xsltoutput =** *FILE*
  L'URL du fichier sortie-primaire. Par exemple, dans le cas d'une sortie HTML de fichier multiple, le fichier de sortie primaire sera l'emplacement du fichier HTML du point d'entrée. Les fichiers de sortie supplémentaires, comme des fichiers d'image générée, sont rapportés en tant que `xslt-additional-output-files`. Si aucune option `--output` ou `--xsltoutput` n'est spécifiée, la sortie est écrite sur la sortie standard.

- output-encoding

  --output-encoding = *VALUE*
  La valeur de l'attribut d'encodage dans le document de sortie. Les valeurs valides sont des noms dans le registre d'ensemble de caractères IANA. La valeur par défaut est `UTF-8`.

- output-indent

  --output-indent = `true|false`
  Si `true`, la sortie sera en retrait conformément à sa structure hiérarchique. Si `false`, il n'y aura pas de retrait hiérarchique. La valeur par défaut est `false`.
  Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- output-method

  --output-method = `xml|html|xhtml|text`
  Spécifie le format de sortie. La valeur par défaut est `xml`.

- param [p]

  --p | --param = *KEY:VALUE*

  XQuery

  Spécifie la valeur d'un paramètre externe. Un paramètre externe est déclaré dans le document XQuery avec la déclaration `declare variable` suivi par un nom de variable puis le mot-clé `external` suivi par le point-virgule de fin de ligne. Par exemple :

  ```
  declare variable $foo as xs:string external;
  ```

 © 2018 Altova GmbH

En raison du mot-clé `external`, `$foo` devient un paramètre externe, dont la valeur est passée pendant le runtime depuis une source externe. Le paramètre externe reçoit une value avec la commande CLI. Par exemple :

```
--param=foo:'MyName'
```

Dans l'instruction de description ci-dessus, *KEY* est le nom de paramètre externe, *VALUE* est la valeur du paramètre externe, donné en tant qu'une expression XPath. Les noms de paramètres utilisés sur la CLI doivent être déclarés dans le document XQuery. Si plusieurs paramètres externes sont des valeurs passées sur la CLI, chacun doit être recevoir une option séparée `--param`. Les doubles guillemets doivent être utilisés si l'expression XPath contient des espaces.

XSLT

Spécifie paramètre de feuille de style global. *KEY* est le nom de paramètre, *VALUE* est une expression XPath qui fournit la valeur de paramètre. Les noms de paramètre utilisés sur la CLI doivent être déclarés dans la feuille de style. Si plusieurs paramètres multiples sont utilisés, le changement `--param` doit être utilisé avant chaque paramètre. Les double guillemets doivent être utilisés autour de l'expression XPath si elle contient un espace ; que l'espace se trouve dans l'expression XPath elle-même ou dans un littéral de string dans l'expression. Par exemple :

```
raptorxmlxbrl xslt --input=c:\Test.xml --output=c:\Output.xml --
param=date://node[1]/@att1 --p=title:'stringwithoutspace' --
param=title:"'string with spaces'" --p=amount:456 c:\Test.xslt
```

- xquery-version

  `--xquery-version =` `1|1.0|3|3.0|3.1`

  Spécifie si le processeur XQuery devrait utiliser XSLT XQuery 1.0 ou XQuery 3.0. La valeur par défaut est 3.1.

- Schéma XML et instance XML
  - load-xml-with-psvi

    `--load-xml-with-psvi =` `true|false`

    Active la validation des fichiers XML d'entrée et génère des informations de post-schema-validation pour les fichiers. La valeur par défaut est : `true`.

  - schema-imports

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

    Spécifie le comportement des éléments `xs:import`, chacun d'entre eux ayant un attribut `namespace` optionnel et un attribut `schemaLocation` optionnel : `<import namespace="someNS" schemaLocation="someURL">`. L'option spécifie s'il faut charger un document de schéma ou juste mettre un espace de noms sous licence, et, si un document de schéma doit être chargé, quelle information doit être utilisée pour le trouver. Défaut : `load-preferring-schemalocation`.

    Le comportement est le suivant :

- `load-by-schemalocation`: La valeur de l'attribut `schemaLocation` est utilisée pour situer le schéma, en prenant en compte les mappages de catalogue. Si l'attribut de l'espace de nom est présent, l'espace de noms est importé (licencé).
- `load-preferring-schemalocation`: Si l'attribut `schemaLocation` est présent, il est utilisé en prenant en compte les mappages de catalogue. Si l'attribut `schemaLocation` est présent, la valeur de l'attribut `namespace` est utilisée via un mappage de catalogue. C'est la **valeur par défaut**.
- `load-by-namespace`: La valeur de l'attribut `namespace` est utilisée pour situer le schéma via un mappage de catalogue.
- `load-combining-both`: Si soit l'Attribut `namespace` ou l'attribut `schemaLocation` a un mappage de catalogue, le mappage est utilisé. Si les deux ont des mappages de catalogue, alors la valeur de l'option `--schema-mapping` (option XBRL et option XML/XSD) décide de quel mappage utiliser. Si aucun mappage de catalogue n'est présent, l'attribut `schemaLocation` est utilisé.
- `license-namespace-only`: Le nom d'espace est importé. Aucun document de schéma n'est importé.

schemalocation-hints

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

Spécifie le comportement des attributs `xsi:schemaLocation` et `xsi:noNamespaceSchemaLocation` : S'il faut charger un document de schéma et, si oui, quelle information doit être utilisée pour la trouver. Défaut : `load-by-schemalocation`.

- La valeur `load-by-schemalocation` utilise l'URL de l'emplacement de schéma dans les attributs `xsi:schemaLocation` et `xsi:noNamespaceSchemaLocation` dans les documents d'instance XML ou XBRL. Il s'agit de la **valeur par défaut**.
- La valeur `load-by-namespace` prend la part d'espace de nom `xsi:schemaLocation` et un string vide dans le cas de `xsi:noNamespaceSchemaLocation` et localise le schéma par le biais d'un mappage de catalogue.
- Si `load-combining-both` est utilisé et si soit la partie espace de noms ou la partie URL a un mappage de catalogue, alors le mappage de catalogue est utilisé. Si tous deux ont des mappages de catalogue, alors la valeur de l'option de `--schema-mapping` (option XBRL et option XML/XSD) de quel mappage utiliser. Si ni l'espace de noms ni l'URL n'a un mappage de catalogue, l'URL est utilisée.
- Si la valeur de l'option est `ignore`, les attributs `xsi:schemaLocation` et `xsi:noNamespaceSchemaLocation` seront ignorés tous les deux.

schema-mapping

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

Si l'emplacement de schéma et l'espace de noms sont tous les deux utilisés pour trouver un document de schéma, spécifie lequel des deux doit être utilisé pendant la consultation du catalogue. (Si soit l'option `--schemalocation-hints` ou l'option `--schema-imports` a une valeur de `load-combining-both`, et si les parties d'espace de noms et d'URL impliquées ont toutes les deux des mappages de catalogue, alors la valeur de cette option spécifie lequel des deux mappages utiliser (mappage d'espace de noms ou mappage URL ; la valeur `prefer-schemalocation` réfère au mappage URL).)

Défaut : `prefer-schemalocation`.

- xinclude

  `--xinclude =` `true|false`
  Active la prise en charge de XML Inclusions (XInclude). La valeur par défaut est `false`. Si `false`, les élément `include` d'XInclude sont ignorés.
  Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- xml-mode

  `--xml-mode =` `wf|id|valid`
  Spécifie le mode de traitement XML à utiliser : `wf`=wellformed check (vérification de la bonne forme); `id`=bien formé avec des contrôles ID/IDREF ; `valid`=validation. La valeur par défaut est `wf`. Veuillez noter qu'une valeur de `valid` exige que chaque document de schéma/instance chargé depuis les traitements référence un DTD. Si aucun DTD n'existe, une erreur est rapportée.

- xml-mode-for-schemas

  `--xml-mode-for-schemas =` `wf|id|valid`
  Spécifie le mode de traitement XML à utiliser pour les documents de schéma XML : `wf`=vérification de la bonne formation ; `id`=contrôles de la bonne formation avec ID/IDREF ; `valid`=validation. La valeur par défaut est `wf`. Veuillez noter qu'une valeur de `valid` exige que chaque document de schéma chargé pendant le traitement référence un DTD. Si aucun DTD n'existe, une erreur est rapportée.

- xml-validation-error-as-warning

  `--xml-validation-error-as-warning =` `true|false`
  Si `true`, traite les erreurs de validation en tant qu'avertissements. Si les erreurs sont traitées en tant qu'avertissements, les traitements supplémentaires, comme des transformations XSLT, continueront quelles que soient les erreurs. La valeur par défaut est `false`.

- xsd

  `--xsd =` *`FILE`*
  Spécifie un ou plusieurs documents de Schéma XML à utiliser pour la validation des documents d'instance XML. Ajouter l'option plusieurs fois pour spécifier plus d'un document de schéma.

- xsd-version

  `--xsd-version =` `1.0|1.1|detect`
  Spécifie la version du langage de définition de Schéma W3C (XSD) à utiliser. Le défaut est `1.0`. Cette option peut être utile pour découvrir comment un schéma compatible à 1.0 n'est pas compatible à 1.1. L'option `detect` est une fonction spécifique à Altova. Elle permet la détection de la version du document de Schéma XML (`1.0` ou `1.1`) en lisant la valeur de l'attribut `vc:minVersion` de l'élément du document `<xs:schema>`. Si la valeur de l'attribut `@vc:minVersion` est `1.1`, le schéma est détecté comme étant de version `1.1`. Pour toute autre valeur, ou si l'attribut `@vc:minVersion` est absent, le schéma est détecté comme étant de version `1.0`.

- Catalogues et ressources globales
  - catalog

    --catalog = *FILE*
    Spécifie le chemin absolu vers un fichier de catalogue root qui n'est pas le fichier catalogue root installé. La valeur par défaut est le chemin absolu vers le fichier de catalogue root installé (`<installation-folder>\Altova\RaptorXMLXBRLServer2018\etc\RootCatalog.xml`). Voir la section, Catalogues XML, pour plus d'informations à propos du travail avec les catalogues.

  - user-catalog

    --user-catalog = *FILE*
    Spécifie le chemin absolu vers un catalogue XML à utiliser en plus du catalogue root. Voir la section, Catalogues XML, pour plus d'informations concernant le travail avec des catalogues.

  - enable-globalresources

    --enable-globalresources = `true|false`
    Active les ressources globales. La valeur par défaut est `false`.
    Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

  - globalresourceconfig [gc]

    --gc | --globalresourceconfig = *VALUE*
    Spécifie la configuration active de la ressource globale (et active les ressources globales).

  - globalresourcefile [gr]

    --gr | --globalresourcefile = *FILE*
    Spécifie le fichier de ressource globale (et active les ressources globales).

- Extensions

  Ces options définissent la gestion des fonctions d'extension spéciales disponibles dans un certain nombre de produits Altova de niveau d'entreprise (comme XMLSpy Enterprise Edition). Leur utilisation est décrite dans les manuels d'utilisateur de ces produits.
  - chartext-disable

    --chartext-disable = `true|false`
    Désactive les extensions de graphique. La valeur par défaut est `false`.
    Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

  - dotnetext-disable

    --dotnetext-disable = `true|false`
    Désactive les extensions .NET. La valeur par défaut est `false`.

 © 2018 Altova GmbH

Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- jvm-location

  `--jvm-location =` *`FILE`*
  `FILE` spécifie l'emplacement de la Java Virtual Machine (DLL sur Windows, objet partagé sur Linux). La JVM est nécessaire si vous utilisez les fonctions d'extension Java dans votre code XSLT/XQuery. La valeur par défaut est `false`.

- javaext-barcode-location

  `--javaext-barcode-location =` *`FILE`*
  Spécifie le chemin vers le dossier qui contient le fichier d'extension du code-barres `AltovaBarcodeExtension.jar`. Le chemin doit être donné dans une des formes suivantes :

  - Un URI de fichier, par exemple : `--javaext-barcode-location="file:///C:/Program Files/Altova/RaptorXMLXBRLServer2018/etc/jar/"`
  - Un chemin Windows avec un échappement par barres obliques inversées, par exemple : `--javaext-barcode-location="C:\\Program Files\\Altova\\RaptorXMLXBRLServer2018\\etc\\jar\\"`

- javaext-disable

  `--javaext-disable =` `true|false`
  Désactive les extensions Java. La valeur par défaut est `false`.
  Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- Messages, erreurs, aide, timeout, version

  - error-format

    `--error-format =` `text|shortxml|longxml`
    Spécifie le format de la sortie d'erreur. La valeur par défaut est `text`. Les autres options génèrent des formats XML, avec `longxml` générant plus de détails.

  - error-limit

    `--error-limit =` *`N`*
    Spécifie la limite d'erreur avec une plage de valeur de `1` à `999`. La valeur par défaut est `100`. Le traitement s'arrête lorsque la limite d'erreur est atteinte. Utile pour limiter l'utilisation du processeur pendant la validation/transformation.

  - help

    `--help`
    Affiche le texte d'aide pour la commande. Par exemple, `valany --h`. (en alternative, la commande `help` peut être utilisée avec un argument. Par exemple : `help valany`.)

  - network-timeout

    `--network-timeout =` *`VALUE`*

Spécifie le timeout en secondes pour les opérations I/O à distance. La valeur par défaut est : `40`.

- verbose

  `--verbose =` `true|false`
  Une valeur de `true` permet la sortie d'informations supplémentaires pendant la validation. La valeur par défaut est `false`.
  Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- verbose-output

  `--verbose-output =` *`FILE`*
  Écrit la sortie verbeuse sur *`FILE`*.

- version

  `--version`
  Affiche la version de RaptorXML+XBRL Server. Si utilisé avec une commande, placer `--version` avant la commande.

 © 2018 Altova GmbH

xqueryupdate

La commande **xqueryupdate** prend un fichier XQuery en tant que son argument unique et l'exécute avec un fichier d'entrée optionnel pour produire un fichier de sortie mis à jour. Les fichiers d'entrée et de sortie sont spécifiés en tant qu'options.

Windows	**RaptorXMLXBRL xqueryupdate [options] *XQuery-File***
Linux	**raptorxmlxbrl xqueryupdate [options] *XQuery-File***
Mac	**raptorxmlxbrl xqueryupdate [options] *XQuery-File***

L'argument ***XQuery-File*** est le chemin et le nom du fichier XQuery à exécuter. Vous pouvez spécifier si vous préférez utiliser XQuery Update 1.0 ou 3.0. XQuery Update 3.0 est utilisé par défaut.

Exemples

- **raptorxmlxbrl** `xqueryupdate --output=c:\Output.xml c:\TestQuery.xq`
- **raptorxmlxbrl** `xqueryupdate --input=c:\Input.xml --output=c:\Output.xml --param=company:"Altova" --p=date:"2006-01-01" c:\TestQuery.xq`
- **raptorxmlxbrl** `xqueryupdate --input=c:\Input.xml --output=c:\Output.xml --param=source:" doc( 'c:\test\books.xml' )//book "`
- **raptorxmlxbrl** `xqueryupdate --output=c:\Output.xml --omit-xml-declaration=false --output-encoding=ASCII c:\TestQuery.xq`

Casse et lignes obliques sur la ligne de commande

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

* Veuillez noter que les caractères minuscules (`raptorxmlxbrl`) fonctionnent sur toutes les plates-formes (Windows, Linux et Mac), alors qu'une écriture en majuscule/minuscule (`RaptorXMLXBRL`) ne fonctionne que sur Windows et Mac.
* Utiliser les barres obliques penchées vers l'avant pour Linux et Mac, et les barres obliques penchées vers l'arrière sur Windows.

Options

Les options de la commande sont listées ci-dessous, et organisées en groupes. Les valeurs peuvent être spécifiées sans guillemets sauf dans deux cas : (i) lorsque le string de valeur contient des espaces, ou (ii) si la présence de guillemets est explicitement requise dans la description de l'option.

Traitement XQuery Update

- indent-characters

  --indent-characters = *VALUE*
  Spécifie le string de caractère à utiliser en tant que retrait.

- input

  --input = *FILE*
  L'URL du fichier XML à transformer.

- omit-xml-declaration

  --omit-xml-declaration = `true|false`
  Option de sérialisation pour spécifier si la déclaration XML doit être omise de la sortie ou pas. Si `true`, il n'y aura pas de déclaration XML dans le document de sortie. Si `false`, une déclaration XML sera incluse. La valeur par défaut est `false`.
  Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- output, xsltoutput

  output = *FILE,* **xsltoutput =** *FILE*
  L'URL du fichier sortie-primaire. Par exemple, dans le cas d'une sortie HTML de fichier multiple, le fichier de sortie primaire sera l'emplacement du fichier HTML du point d'entrée. Les fichiers de sortie supplémentaires, comme des fichiers d'image générée, sont rapportés en tant que `xslt-additional-output-files`. Si aucune option `--output` ou `--xsltoutput` n'est spécifiée, la sortie est écrite sur la sortie standard.

- output-encoding

  --output-encoding = *VALUE*
  La valeur de l'attribut d'encodage dans le document de sortie. Les valeurs valides sont des noms dans le registre d'ensemble de caractères IANA. La valeur par défaut est `UTF-8`.

- output-indent

  --output-indent = `true|false`
  Si `true`, la sortie sera en retrait conformément à sa structure hiérarchique. Si `false`, il n'y aura pas de retrait hiérarchique. La valeur par défaut est `false`.
  Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- output-method

  --output-method = `xml|html|xhtml|text`
  Spécifie le format de sortie. La valeur par défaut est `xml`.

- param [p]

  --p | --param = *KEY:VALUE*
  - ***XQuery***

    Spécifie la valeur d'un paramètre externe. Un paramètre externe est déclaré dans le document XQuery avec la déclaration `declare variable` suivi par un nom de

variable puis le mot-clé `external` suivi par le point-virgule de fin de ligne. Par exemple :

```
declare variable $foo as xs:string external;
```

En raison du mot-clé `external`, `$foo` devient un paramètre externe, dont la valeur est passée pendant le runtime depuis une source externe. Le paramètre externe reçoit une value avec la commande CLI. Par exemple :

```
--param=foo:'MyName'
```

Dans l'instruction de description ci-dessus, *KEY* est le nom de paramètre externe, *VALUE* est la valeur du paramètre externe, donné en tant qu'une expression XPath. Les noms de paramètres utilisés sur la CLI doivent être déclarés dans le document XQuery. Si plusieurs paramètres externes sont des valeurs passées sur la CLI, chacun doit être recevoir une option séparée `--param`. Les doubles guillemets doivent être utilisés si l'expression XPath contient des espaces.

XSLT

Spécifie paramètre de feuille de style global. *KEY* est le nom de paramètre, *VALUE* est une expression XPath qui fournit la valeur de paramètre. Les noms de paramètre utilisés sur la CLI doivent être déclarés dans la feuille de style. Si plusieurs paramètres multiples sont utilisés, le changement `--param` doit être utilisé avant chaque paramètre. Les double guillemets doivent être utilisés autour de l'expression XPath si elle contient un espace ; que l'espace se trouve dans l'expression XPath elle-même ou dans un littéral de string dans l'expression. Par exemple :

```
raptorxmlxbrl xslt --input=c:\Test.xml --output=c:\Output.xml --
param=date://node[1]/@att1 --p=title:'stringwithoutspace' --
param=title:"'string with spaces'" --p=amount:456 c:\Test.xslt
```

xquery-update-version

`--xquery-update-version =` `1|3`

Spécifie si le processeur XQuery devrait utiliser XQuery Update Facility 1.0 ou XQuery Update Facility 3.0. La valeur par défaut est `3`.

keep-formatting

`--keep-formatting =` `true|false`

Garde le formatage du document cible autant que possible. La valeur par défaut est : `true`.

updated-xml

`--updated-xml =` `discard|writeback|asmainresult`

Spécifie comment gérer le fichier XML mis à jour. Les mises à jour peuvent être soit :

- éliminées et pas écrites sur le fichier (`discard`)
- réécrites sur le fichier d'entrée XML qui est spécifié avec l'option `--input` (`writeback`)
- enregistrées soit sur une sortie standard soit sut l'emplacement spécifié dans l'option `--output` (si cela a été défini)

La valeur par défaut est : `discard`.

- Schéma XML et instance XML
  - load-xml-with-psvi

    `--load-xml-with-psvi =` `true|false`
    Active la validation des fichiers XML d'entrée et génère des informations de post-schema-validation pour les fichiers. La valeur par défaut est : `true`.

  - schema-imports

    `--schema-imports =` `load-by-schemalocation | load-preferring-schemalocation | load-by-namespace | load-combining-both | license-namespace-only`
    Spécifie le comportement des éléments `xs:import`, chacun d'entre eux ayant un attribut `namespace` optionnel et un attribut `schemaLocation` optionnel : `<import namespace="someNS" schemaLocation="someURL">`. L'option spécifie s'il faut charger un document de schéma ou juste mettre un espace de noms sous licence, et, si un document de schéma doit être chargé, quelle information doit être utilisée pour le trouver. Défaut : `load-preferring-schemalocation`.
    Le comportement est le suivant :

    - `load-by-schemalocation`: La valeur de l'attribut `schemaLocation` est utilisée pour situer le schéma, en prenant en compte les mappages de catalogue. Si l'attribut de l'espace de nom est présent, l'espace de noms est importé (licencé).
    - `load-preferring-schemalocation`: Si l'attribut `schemaLocation` est présent, il est utilisé en prenant en compte les mappages de catalogue. Si l'attribut `schemaLocation` est présent, la valeur de l'attribut `namespace` est utilisée via un mappage de catalogue. C'est la **valeur par défaut**.
    - `load-by-namespace`: La valeur de l'attribut `namespace` est utilisée pour situer le schéma via un mappage de catalogue.
    - `load-combining-both`: Si soit l'Attribut `namespace` ou l'attribut `schemaLocation` a un mappage de catalogue, le mappage est utilisé. Si les deux ont des mappages de catalogue, alors la valeur de l'option `--schema-mapping` (option XBRL et option XML/XSD) décide de quel mappage utiliser. Si aucun mappage de catalogue n'est présent, l'attribut `schemaLocation` est utilisé.
    - `license-namespace-only`: Le nom d'espace est importé. Aucun document de schéma n'est importé.

  - schemalocation-hints

    `--schemalocation-hints =` `load-by-schemalocation | load-by-namespace | load-combining-both | ignore`
    Spécifie le comportement des attributs `xsi:schemaLocation` et `xsi:noNamespaceSchemaLocation` : S'il faut charger un document de schéma et, si oui, quelle information doit être utilisée pour la trouver. Défaut : `load-by-schemalocation`.

    - La valeur `load-by-schemalocation` utilise l'URL de l'emplacement de schéma dans les attributs `xsi:schemaLocation` et `xsi:noNamespaceSchemaLocation` dans les documents d'instance XML ou XBRL. Il s'agit de la **valeur par défaut**.
    - La valeur `load-by-namespace` prend la part d'espace de nom

xsi:schemaLocation et un string vide dans le cas de xsi:noNamespaceSchemaLocation et localise le schéma par le biais d'un mappage de catalogue.

- Si load-combining-both est utilisé et si soit la partie espace de noms ou la partie URL a un mappage de catalogue, alors le mappage de catalogue est utilisé. Si tous deux ont des mappages de catalogue, alors la valeur de l'option de --schema-mapping (option XBRL et option XML/XSD) de quel mappage utiliser. Si ni l'espace de noms ni l'URL n'a un mappage de catalogue, l'URL est utilisée.
- Si la valeur de l'option est ignore, les attributs xsi:schemaLocation et xsi:noNamespaceSchemaLocation seront ignorés tous les deux.

schema-mapping

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

Si l'emplacement de schéma et l'espace de noms sont tous les deux utilisés pour trouver un document de schéma, spécifie lequel des deux doit être utilisé pendant la consultation du catalogue. (Si soit l'option --schemalocation-hints ou l'option --schema-imports a une valeur de load-combining-both, et si les parties d'espace de noms et d'URL impliquées ont toutes les deux des mappages de catalogue, alors la valeur de cette option spécifie lequel des deux mappages utiliser (mappage d'espace de noms ou mappage URL ; la valeur prefer-schemalocation réfère au mappage URL).) Défaut : prefer-schemalocation.

xinclude

--xinclude = true|false

Active la prise en charge de XML Inclusions (XInclude). La valeur par défaut est false. Si false, les élément include d'XInclude sont ignorés.
Note : les valeurs d'option booléennes sont configurées sur true si l'option est spécifiée sans une valeur.

xml-mode

--xml-mode = wf|id|valid

Spécifie le mode de traitement XML à utiliser : wf=wellformed check (vérification de la bonne forme); id=bien formé avec des contrôles ID/IDREF ; valid=validation. La valeur par défaut est wf. Veuillez noter qu'une valeur de valid exige que chaque document de schéma/instance chargé depuis les traitements référence un DTD. Si aucun DTD n'existe, une erreur est rapportée.

xml-mode-for-schemas

--xml-mode-for-schemas = wf|id|valid

Spécifie le mode de traitement XML à utiliser pour les documents de schéma XML : wf=vérification de la bonne formation ; id=contrôles de la bonne formation avec ID/IDREF ; valid=validation. La valeur par défaut est wf. Veuillez noter qu'une valeur de valid exige que chaque document de schéma chargé pendant le traitement référence un DTD. Si aucun DTD n'existe, une erreur est rapportée.

xml-validation-error-as-warning

--xml-validation-error-as-warning = true|false

Si true, traite les erreurs de validation en tant qu'avertissements. Si les erreurs sont traitées en tant qu'avertissements, les traitements supplémentaires, comme des

transformations XSLT, continueront quelles que soient les erreurs. La valeur par défaut est `false`.

- xsd

  `--xsd =` *`FILE`*
  Spécifie un ou plusieurs documents de Schéma XML à utiliser pour la validation des documents d'instance XML. Ajouter l'option plusieurs fois pour spécifier plus d'un document de schéma.

- xsd-version

  `--xsd-version =` `1.0|1.1|detect`
  Spécifie la version du langage de définition de Schéma W3C (XSD) à utiliser. Le défaut est `1.0`. Cette option peut être utile pour découvrir comment un schéma compatible à 1.0 n'est pas compatible à 1.1. L'option `detect` est une fonction spécifique à Altova. Elle permet la détection de la version du document de Schéma XML (`1.0` ou `1.1`) en lisant la valeur de l'attribut `vc:minVersion` de l'élément du document `<xs:schema>`. Si la valeur de l'attribut `@vc:minVersion` est `1.1`, le schéma est détecté comme étant de version `1.1`. Pour toute autre valeur, ou si l'attribut `@vc:minVersion` est absent, le schéma est détecté comme étant de version `1.0`.

- Catalogues et ressources globales
  - catalog

    `--catalog =` *`FILE`*
    Spécifie le chemin absolu vers un fichier de catalogue root qui n'est pas le fichier catalogue root installé. La valeur par défaut est le chemin absolu vers le fichier de catalogue root installé (`<installation-folder>\Altova\RaptorXMLXBRLServer2018\etc\RootCatalog.xml`). Voir la section, Catalogues XML, pour plus d'informations à propos du travail avec les catalogues.

  - user-catalog

    `--user-catalog =` *`FILE`*
    Spécifie le chemin absolu vers un catalogue XML à utiliser en plus du catalogue root. Voir la section, Catalogues XML, pour plus d'informations concernant le travail avec des catalogues.

  - enable-globalresources

    `--enable-globalresources =` `true|false`
    Active les ressources globales. La valeur par défaut est `false`.
    Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

  - globalresourceconfig [gc]

    `--gc | --globalresourceconfig =` *`VALUE`*
    Spécifie la configuration active de la ressource globale (et active les ressources globales).

 © 2018 Altova GmbH

- globalresourcefile [gr]

  --gr | --globalresourcefile = *FILE*
  Spécifie le fichier de ressource globale (et active les ressources globales).

- Extensions

  Ces options définissent la gestion des fonctions d'extension spéciales disponibles dans un certain nombre de produits Altova de niveau d'entreprise (comme XMLSpy Enterprise Edition). Leur utilisation est décrite dans les manuels d'utilisateur de ces produits.

  - chartext-disable

    --chartext-disable = `true|false`
    Désactive les extensions de graphique. La valeur par défaut est `false`.
    Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

  - dotnetext-disable

    --dotnetext-disable = `true|false`
    Désactive les extensions .NET. La valeur par défaut est `false`.
    Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

  - jvm-location

    --jvm-location = *FILE*
    FILE spécifie l'emplacement de la Java Virtual Machine (DLL sur Windows, objet partagé sur Linux). La JVM est nécessaire si vous utilisez les fonctions d'extension Java dans votre code XSLT/XQuery. La valeur par défaut est `false`.

  - javaext-barcode-location

    --javaext-barcode-location = *FILE*
    Spécifie le chemin vers le dossier qui contient le fichier d'extension du code-barres `AltovaBarcodeExtension.jar`. Le chemin doit être donné dans une des formes suivantes :

    - Un URI de fichier, par exemple : `--javaext-barcode-location="file:///C:/Program Files/Altova/RaptorXMLXBRLServer2018/etc/jar/"`
    - Un chemin Windows avec un échappement par barres obliques inversées, par exemple : `--javaext-barcode-location="C:\\Program Files\\Altova\\RaptorXMLXBRLServer2018\\etc\\jar\\"`

  - javaext-disable

    --javaext-disable = `true|false`
    Désactive les extensions Java. La valeur par défaut est `false`.
    Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- Messages, erreurs, aide, timeout, version

- error-format

  `--error-format =` `text|shortxml|longxml`
  Spécifie le format de la sortie d'erreur. La valeur par défaut est `text`. Les autres options génèrent des formats XML, avec `longxml` générant plus de détails.

- error-limit

  `--error-limit =` *`N`*
  Spécifie la limite d'erreur avec une plage de valeur de `1` à `999`. La valeur par défaut est `100`. Le traitement s'arrête lorsque la limite d'erreur est atteinte. Utile pour limiter l'utilisation du processeur pendant la validation/transformation.

- help

  `--help`
  Affiche le texte d'aide pour la commande. Par exemple, `valany --h`. (en alternative, la commande `help` peut être utilisée avec un argument. Par exemple : `help valany`.)

- network-timeout

  `--network-timeout =` *`VALUE`*
  Spécifie le timeout en secondes pour les opérations I/O à distance. La valeur par défaut est : `40`.

- verbose

  `--verbose =` `true|false`
  Une valeur de `true` permet la sortie d'informations supplémentaires pendant la validation. La valeur par défaut est `false`.
  Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- verbose-output

  `--verbose-output =` *`FILE`*
  Écrit la sortie verbeuse sur *`FILE`*.

- version

  `--version`
  Affiche la version de RaptorXML+XBRL Server. Si utilisé avec une commande, placer `--version` avant la commande.

 © 2018 Altova GmbH

valxquery

La commande **valxquery** prend un fichier XQuery en tant que son argument unique et le valide.

Windows	**RaptorXMLXBRL valxquery [options]** ***XQuery-File***
Linux	**raptorxmlxbrl valxquery [options]** ***XQuery-File***
Mac	**raptorxmlxbrl valxquery [options]** ***XQuery-File***

L'argument ***XQuery-File*** est le chemin et le nom du fichier XQuery à valider.

Exemples

- **raptorxmlxbrl** valxquery c:\Test.xquery
- **raptorxmlxbrl** valxquery --xquery-version=1 c:\Test.xquery

Casse et lignes obliques sur la ligne de commande

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

* Veuillez noter que les caractères minuscules (raptorxmlxbrl) fonctionnent sur toutes les plates-formes (Windows, Linux et Mac), alors qu'une écriture en majuscule/minuscule (RaptorXMLXBRL) ne fonctionne que sur Windows et Mac.
* Utiliser les barres obliques penchées vers l'avant pour Linux et Mac, et les barres obliques penchées vers l'arrière sur Windows.

Options

Les options de la commande sont listées ci-dessous, et organisées en groupes. Les valeurs peuvent être spécifiées sans guillemets sauf dans deux cas : (i) lorsque le string de valeur contient des espaces, ou (ii) si la présence de guillemets est explicitement requise dans la description de l'option.

Traitement XQuery

omit-xml-declaration

--omit-xml-declaration = true|false
Option de sérialisation pour spécifier si la déclaration XML doit être omise de la sortie ou pas. Si true, il n'y aura pas de déclaration XML dans le document de sortie. Si false, une déclaration XML sera incluse. La valeur par défaut est false.
Note : les valeurs d'option booléennes sont configurées sur true si l'option est spécifiée sans une valeur.

xquery-version

--xquery-version = `1|1.0|3|3.0|3.1`
Spécifie si le processeur XQuery devrait utiliser XSLT XQuery 1.0 ou XQuery 3.0. La valeur par défaut est 3.1.

- Schéma XML et instance XML
  - load-xml-with-psvi

    --load-xml-with-psvi = `true|false`
    Active la validation des fichiers XML d'entrée et génère des informations de post-schema-validation pour les fichiers. La valeur par défaut est : `true`.

  - schema-imports

    --schema-imports = `load-by-schemalocation | load-preferring-schemalocation | load-by-namespace | load-combining-both | license-namespace-only`
    Spécifie le comportement des éléments `xs:import`, chacun d'entre eux ayant un attribut `namespace` optionnel et un attribut `schemaLocation` optionnel : `<import namespace="someNS" schemaLocation="someURL">`. L'option spécifie s'il faut charger un document de schéma ou juste mettre un espace de noms sous licence, et, si un document de schéma doit être chargé, quelle information doit être utilisée pour le trouver. Défaut : `load-preferring-schemalocation`.
    Le comportement est le suivant :

    - `load-by-schemalocation`: La valeur de l'attribut `schemaLocation` est utilisée pour situer le schéma, en prenant en compte les mappages de catalogue. Si l'attribut de l'espace de nom est présent, l'espace de noms est importé (licencé).
    - `load-preferring-schemalocation`: Si l'attribut `schemaLocation` est présent, il est utilisé en prenant en compte les mappages de catalogue. Si l'attribut `schemaLocation` est présent, la valeur de l'attribut `namespace` est utilisée via un mappage de catalogue. C'est la **valeur par défaut**.
    - `load-by-namespace`: La valeur de l'attribut `namespace` est utilisée pour situer le schéma via un mappage de catalogue.
    - `load-combining-both`: Si soit l'Attribut `namespace` ou l'attribut `schemaLocation` a un mappage de catalogue, le mappage est utilisé. Si les deux ont des mappages de catalogue, alors la valeur de l'option `--schema-mapping` (option XBRL et option XML/XSD) décide de quel mappage utiliser. Si aucun mappage de catalogue n'est présent, l'attribut `schemaLocation` est utilisé.
    - `license-namespace-only`: Le nom d'espace est importé. Aucun document de schéma n'est importé.

  - schemalocation-hints

    --schemalocation-hints = `load-by-schemalocation | load-by-namespace | load-combining-both | ignore`
    Spécifie le comportement des attributs `xsi:schemaLocation` et `xsi:noNamespaceSchemaLocation` : S'il faut charger un document de schéma et, si oui, quelle information doit être utilisée pour la trouver. Défaut : `load-by-schemalocation`.

    - La valeur `load-by-schemalocation` utilise l'URL de l'emplacement de schéma

dans les attributs `xsi:schemaLocation` et `xsi:noNamespaceSchemaLocation` dans les documents d'instance XML ou XBRL. Il s'agit de la **valeur par défaut**.

- La valeur `load-by-namespace` prend la part d'espace de nom `xsi:schemaLocation` et un string vide dans le cas de `xsi:noNamespaceSchemaLocation` et localise le schéma par le biais d'un mappage de catalogue.
- Si `load-combining-both` est utilisé et si soit la partie espace de noms ou la partie URL a un mappage de catalogue, alors le mappage de catalogue est utilisé. Si tous deux ont des mappages de catalogue, alors la valeur de l'option de `--schema-mapping` (option XBRL et option XML/XSD) de quel mappage utiliser. Si ni l'espace de noms ni l'URL n'a un mappage de catalogue, l'URL est utilisée.
- Si la valeur de l'option est `ignore`, les attributs `xsi:schemaLocation` et `xsi:noNamespaceSchemaLocation` seront ignorés tous les deux.

schema-mapping

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

Si l'emplacement de schéma et l'espace de noms sont tous les deux utilisés pour trouver un document de schéma, spécifie lequel des deux doit être utilisé pendant la consultation du catalogue. (Si soit l'option `--schemalocation-hints` ou l'option `--schema-imports` a une valeur de `load-combining-both`, et si les parties d'espace de noms et d'URL impliquées ont toutes les deux des mappages de catalogue, alors la valeur de cette option spécifie lequel des deux mappages utiliser (mappage d'espace de noms ou mappage URL ; la valeur `prefer-schemalocation` réfère au mappage URL).) Défaut : `prefer-schemalocation`.

xinclude

`--xinclude =` `true|false`

Active la prise en charge de XML Inclusions (XInclude). La valeur par défaut est `false`. Si `false`, les élément `include` d'XInclude sont ignorés.
Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

xml-mode

`--xml-mode =` `wf|id|valid`

Spécifie le mode de traitement XML à utiliser : `wf`=wellformed check (vérification de la bonne forme); `id`=bien formé avec des contrôles ID/IDREF ; `valid`=validation. La valeur par défaut est `wf`. Veuillez noter qu'une valeur de `valid` exige que chaque document de schéma/instance chargé depuis les traitements référence un DTD. Si aucun DTD n'existe, une erreur est rapportée.

xml-mode-for-schemas

`--xml-mode-for-schemas =` `wf|id|valid`

Spécifie le mode de traitement XML à utiliser pour les documents de schéma XML : `wf`=vérification de la bonne formation ; `id`=contrôles de la bonne formation avec ID/IDREF ; `valid`=validation. La valeur par défaut est `wf`. Veuillez noter qu'une valeur de `valid` exige que chaque document de schéma chargé pendant le traitement référence un DTD. Si aucun DTD n'existe, une erreur est rapportée.

xsd-version

--xsd-version = 1.0|1.1|detect
Spécifie la version du langage de définition de Schéma W3C (XSD) à utiliser. Le défaut est 1.0. Cette option peut être utile pour découvrir comment un schéma compatible à 1.0 n'est pas compatible à 1.1. L'option detect est une fonction spécifique à Altova. Elle permet la détection de la version du document de Schéma XML (1.0 ou 1.1) en lisant la valeur de l'attribut vc:minVersion de l'élément du document <xs:schema>. Si la valeur de l'attribut @vc:minVersion est 1.1, le schéma est détecté comme étant de version 1.1. Pour toute autre valeur, ou si l'attribut @vc:minVersion est absent, le schéma est détecté comme étant de version 1.0.

- Catalogues et ressources globales
  - catalog

    --catalog = *FILE*
    Spécifie le chemin absolu vers un fichier de catalogue root qui n'est pas le fichier catalogue root installé. La valeur par défaut est le chemin absolu vers le fichier de catalogue root installé (<installation-folder>\Altova\RaptorXMLXBRLServer2018\etc\RootCatalog.xml). Voir la section, Catalogues XML, pour plus d'informations à propos du travail avec les catalogues.

  - user-catalog

    --user-catalog = *FILE*
    Spécifie le chemin absolu vers un catalogue XML à utiliser en plus du catalogue root. Voir la section, Catalogues XML, pour plus d'informations concernant le travail avec des catalogues.

  - enable-globalresources

    --enable-globalresources = true|false
    Active les ressources globales. La valeur par défaut est false.
    Note : les valeurs d'option booléennes sont configurées sur true si l'option est spécifiée sans une valeur.

  - globalresourceconfig [gc]

    --gc | --globalresourceconfig = *VALUE*
    Spécifie la configuration active de la ressource globale (et active les ressources globales).

  - globalresourcefile [gr]

    --gr | --globalresourcefile = *FILE*
    Spécifie le fichier de ressource globale (et active les ressources globales).

- Extensions

  Ces options définissent la gestion des fonctions d'extension spéciales disponibles dans un certain nombre de produits Altova de niveau d'entreprise (comme XMLSpy Enterprise Edition). Leur utilisation est décrite dans les manuels d'utilisateur de ces produits.
  - chartext-disable

--chartext-disable = `true|false`
Désactive les extensions de graphique. La valeur par défaut est `false`.
Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- dotnetext-disable

  --dotnetext-disable = `true|false`
  Désactive les extensions .NET. La valeur par défaut est `false`.
  Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- jvm-location

  --jvm-location = *FILE*
  FILE spécifie l'emplacement de la Java Virtual Machine (DLL sur Windows, objet partagé sur Linux). La JVM est nécessaire si vous utilisez les fonctions d'extension Java dans votre code XSLT/XQuery. La valeur par défaut est `false`.

- javaext-barcode-location

  --javaext-barcode-location = *FILE*
  Spécifie le chemin vers le dossier qui contient le fichier d'extension du code-barres `AltovaBarcodeExtension.jar`. Le chemin doit être donné dans une des formes suivantes :

  - Un URI de fichier, par exemple : `--javaext-barcode-location="file:///C:/Program Files/Altova/RaptorXMLXBRLServer2018/etc/jar/"`
  - Un chemin Windows avec un échappement par barres obliques inversées, par exemple : `--javaext-barcode-location="C:\\Program Files\\Altova\\RaptorXMLXBRLServer2018\\etc\\jar\\"`

- javaext-disable

  --javaext-disable = `true|false`
  Désactive les extensions Java. La valeur par défaut est `false`.
  Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- Messages, erreurs, aide, timeout, version

  - error-format

    --error-format = `text|shortxml|longxml`
    Spécifie le format de la sortie d'erreur. La valeur par défaut est `text`. Les autres options génèrent des formats XML, avec `longxml` générant plus de détails.

  - error-limit

    --error-limit = *N*
    Spécifie la limite d'erreur avec une plage de valeur de `1` à `999`. La valeur par défaut est `100`. Le traitement s'arrête lorsque la limite d'erreur est atteinte. Utile pour limiter l'utilisation du processeur pendant la validation/transformation.

- help

  `--help`
  Affiche le texte d'aide pour la commande. Par exemple, `valany --h`. (en alternative, la commande `help` peut être utilisée avec un argument. Par exemple : `help valany`.)

- network-timeout

  `--network-timeout =` *`VALUE`*
  Spécifie le timeout en secondes pour les opérations I/O à distance. La valeur par défaut est : `40`.

- verbose

  `--verbose =` `true|false`
  Une valeur de `true` permet la sortie d'informations supplémentaires pendant la validation. La valeur par défaut est `false`.
  Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- verbose-output

  `--verbose-output =` *`FILE`*
  Écrit la sortie verbeuse sur *`FILE`*.

- version

  `--version`
  Affiche la version de RaptorXML+XBRL Server. Si utilisé avec une commande, placer `--version` avant la commande.

 © 2018 Altova GmbH

valxqueryupdate

La commande **valxqueryupdate** prend un fichier XQuery en tant que son argument unique et le valide.

Windows	**RaptorXMLXBRL valxqueryupdate [options] *XQuery-File***
Linux	**raptorxmlxbrl valxqueryupdate [options] *XQuery-File***
Mac	**raptorxmlxbrl valxqueryupdate [options] *XQuery-File***

L'argument ***XQuery-File*** est le chemin et le nom du fichier XQuery à valider.

Exemples

- **raptorxmlxbrl** valxqueryupdae c:\Test.xqu
- **raptorxmlxbrl** valxqueryupdate --xquery-version=1 c:\Test.xqu

Casse et lignes obliques sur la ligne de commande

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

* Veuillez noter que les caractères minuscules (raptorxmlxbrl) fonctionnent sur toutes les plates-formes (Windows, Linux et Mac), alors qu'une écriture en majuscule/minuscule (RaptorXMLXBRL) ne fonctionne que sur Windows et Mac.
* Utiliser les barres obliques penchées vers l'avant pour Linux et Mac, et les barres obliques penchées vers l'arrière sur Windows.

Options

Les options de la commande sont listées ci-dessous, et organisées en groupes. Les valeurs peuvent être spécifiées sans guillemets sauf dans deux cas : (i) lorsque le string de valeur contient des espaces, ou (ii) si la présence de guillemets est explicitement requise dans la description de l'option.

Traitement XQuery

omit-xml-declaration

--omit-xml-declaration = true|false
Option de sérialisation pour spécifier si la déclaration XML doit être omise de la sortie ou pas. Si true, il n'y aura pas de déclaration XML dans le document de sortie. Si false, une déclaration XML sera incluse. La valeur par défaut est false.
Note : les valeurs d'option booléennes sont configurées sur true si l'option est spécifiée sans une valeur.

- xquery-update-version

  --xquery-update-version = 1|3
  Spécifie si le processeur XQuery devrait utiliser XQuery Update Facility 1.0 ou XQuery Update Facility 3.0. La valeur par défaut est 3.

- Schéma XML et instance XML
  - load-xml-with-psvi

    --load-xml-with-psvi = true|false
    Active la validation des fichiers XML d'entrée et génère des informations de post-schema-validation pour les fichiers. La valeur par défaut est : true.

  - schema-imports

    --schema-imports = load-by-schemalocation | load-preferring-schemalocation | load-by-namespace | load-combining-both | license-namespace-only
    Spécifie le comportement des éléments xs:import, chacun d'entre eux ayant un attribut namespace optionnel et un attribut schemaLocation optionnel : <import namespace="someNS" schemaLocation="someURL">. L'option spécifie s'il faut charger un document de schéma ou juste mettre un espace de noms sous licence, et, si un document de schéma doit être chargé, quelle information doit être utilisée pour le trouver. Défaut : load-preferring-schemalocation.
    Le comportement est le suivant :

    - load-by-schemalocation: La valeur de l'attribut schemaLocation est utilisée pour situer le schéma, en prenant en compte les mappages de catalogue. Si l'attribut de l'espace de nom est présent, l'espace de noms est importé (licencé).
    - load-preferring-schemalocation: Si l'attribut schemaLocation est présent, il est utilisé en prenant en compte les mappages de catalogue. Si l'attribut schemaLocation est présent, la valeur de l'attribut namespace est utilisée via un mappage de catalogue. C'est la **valeur par défaut**.
    - load-by-namespace: La valeur de l'attribut namespace est utilisée pour situer le schéma via un mappage de catalogue.
    - load-combining-both: Si soit l'Attribut namespace ou l'attribut schemaLocation a un mappage de catalogue, le mappage est utilisé. Si les deux ont des mappages de catalogue, alors la valeur de l'option --schema-mapping (option XBRL et option XML/XSD) décide de quel mappage utiliser. Si aucun mappage de catalogue n'est présent, l'attribut schemaLocation est utilisé.
    - license-namespace-only: Le nom d'espace est importé. Aucun document de schéma n'est importé.

  - schemalocation-hints

    --schemalocation-hints = load-by-schemalocation | load-by-namespace | load-combining-both | ignore
    Spécifie le comportement des attributs xsi:schemaLocation et xsi:noNamespaceSchemaLocation : S'il faut charger un document de schéma et, si oui, quelle information doit être utilisée pour la trouver. Défaut : load-by-schemalocation.

 © 2018 Altova GmbH

- La valeur `load-by-schemalocation` utilise l'URL de l'emplacement de schéma dans les attributs `xsi:schemaLocation` et `xsi:noNamespaceSchemaLocation` dans les documents d'instance XML ou XBRL. Il s'agit de la **valeur par défaut**.
- La valeur `load-by-namespace` prend la part d'espace de nom `xsi:schemaLocation` et un string vide dans le cas de `xsi:noNamespaceSchemaLocation` et localise le schéma par le biais d'un mappage de catalogue.
- Si `load-combining-both` est utilisé et si soit la partie espace de noms ou la partie URL a un mappage de catalogue, alors le mappage de catalogue est utilisé. Si tous deux ont des mappages de catalogue, alors la valeur de l'option de `--schema-mapping` (option XBRL et option XML/XSD) de quel mappage utiliser. Si ni l'espace de noms ni l'URL n'a un mappage de catalogue, l'URL est utilisée.
- Si la valeur de l'option est `ignore`, les attributs `xsi:schemaLocation` et `xsi:noNamespaceSchemaLocation` seront ignorés tous les deux.

schema-mapping

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

Si l'emplacement de schéma et l'espace de noms sont tous les deux utilisés pour trouver un document de schéma, spécifie lequel des deux doit être utilisé pendant la consultation du catalogue. (Si soit l'option `--schemalocation-hints` ou l'option `--schema-imports` a une valeur de `load-combining-both`, et si les parties d'espace de noms et d'URL impliquées ont toutes les deux des mappages de catalogue, alors la valeur de cette option spécifie lequel des deux mappages utiliser (mappage d'espace de noms ou mappage URL ; la valeur `prefer-schemalocation` réfère au mappage URL).) Défaut : `prefer-schemalocation`.

xinclude

`--xinclude =` `true|false`

Active la prise en charge de XML Inclusions (XInclude). La valeur par défaut est `false`. Si `false`, les élément `include` d'XInclude sont ignorés.
Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

xml-mode

`--xml-mode =` `wf|id|valid`

Spécifie le mode de traitement XML à utiliser : `wf`=wellformed check (vérification de la bonne forme); `id`=bien formé avec des contrôles ID/IDREF ; `valid`=validation. La valeur par défaut est `wf`. Veuillez noter qu'une valeur de `valid` exige que chaque document de schéma/instance chargé depuis les traitements référence un DTD. Si aucun DTD n'existe, une erreur est rapportée.

xml-mode-for-schemas

`--xml-mode-for-schemas =` `wf|id|valid`

Spécifie le mode de traitement XML à utiliser pour les documents de schéma XML : `wf`=vérification de la bonne formation ; `id`=contrôles de la bonne formation avec ID/IDREF ; `valid`=validation. La valeur par défaut est `wf`. Veuillez noter qu'une valeur de `valid` exige que chaque document de schéma chargé pendant le traitement référence un DTD. Si aucun DTD n'existe, une erreur est rapportée.

- xsd-version

  --xsd-version = `1.0|1.1|detect`
  Spécifie la version du langage de définition de Schéma W3C (XSD) à utiliser. Le défaut est `1.0`. Cette option peut être utile pour découvrir comment un schéma compatible à 1.0 n'est pas compatible à 1.1. L'option `detect` est une fonction spécifique à Altova. Elle permet la détection de la version du document de Schéma XML (`1.0` ou `1.1`) en lisant la valeur de l'attribut `vc:minVersion` de l'élément du document `<xs:schema>`. Si la valeur de l'attribut `@vc:minVersion` est `1.1`, le schéma est détecté comme étant de version `1.1`. Pour toute autre valeur, ou si l'attribut `@vc:minVersion` est absent, le schéma est détecté comme étant de version `1.0`.

- Catalogues et ressources globales
  - catalog

    --catalog = *FILE*
    Spécifie le chemin absolu vers un fichier de catalogue root qui n'est pas le fichier catalogue root installé. La valeur par défaut est le chemin absolu vers le fichier de catalogue root installé (`<installation-folder>\Altova\RaptorXMLXBRLServer2018\etc\RootCatalog.xml`). Voir la section, Catalogues XML, pour plus d'informations à propos du travail avec les catalogues.

  - user-catalog

    --user-catalog = *FILE*
    Spécifie le chemin absolu vers un catalogue XML à utiliser en plus du catalogue root. Voir la section, Catalogues XML, pour plus d'informations concernant le travail avec des catalogues.

  - enable-globalresources

    --enable-globalresources = `true|false`
    Active les ressources globales. La valeur par défaut est `false`.
    Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

  - globalresourceconfig [gc]

    --gc | --globalresourceconfig = *VALUE*
    Spécifie la configuration active de la ressource globale (et active les ressources globales).

  - globalresourcefile [gr]

    --gr | --globalresourcefile = *FILE*
    Spécifie le fichier de ressource globale (et active les ressources globales).

- Extensions

  Ces options définissent la gestion des fonctions d'extension spéciales disponibles dans un certain nombre de produits Altova de niveau d'entreprise (comme XMLSpy Enterprise Edition). Leur utilisation est décrite dans les manuels d'utilisateur de ces produits.

 © 2018 Altova GmbH

- chartext-disable

  --chartext-disable = `true|false`
  Désactive les extensions de graphique. La valeur par défaut est `false`.
  Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- dotnetext-disable

  --dotnetext-disable = `true|false`
  Désactive les extensions .NET. La valeur par défaut est `false`.
  Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- jvm-location

  --jvm-location = *FILE*
  FILE spécifie l'emplacement de la Java Virtual Machine (DLL sur Windows, objet partagé sur Linux). La JVM est nécessaire si vous utilisez les fonctions d'extension Java dans votre code XSLT/XQuery. La valeur par défaut est `false`.

- javaext-barcode-location

  --javaext-barcode-location = *FILE*
  Spécifie le chemin vers le dossier qui contient le fichier d'extension du code-barres `AltovaBarcodeExtension.jar`. Le chemin doit être donné dans une des formes suivantes :

  - Un URI de fichier, par exemple : `--javaext-barcode-location="file:///C:/Program Files/Altova/RaptorXMLXBRLServer2018/etc/jar/"`
  - Un chemin Windows avec un échappement par barres obliques inversées, par exemple : `--javaext-barcode-location="C:\\Program Files\\Altova\\RaptorXMLXBRLServer2018\\etc\\jar\\"`

- javaext-disable

  --javaext-disable = `true|false`
  Désactive les extensions Java. La valeur par défaut est `false`.
  Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

Messages, erreurs, aide, timeout, version

- error-format

  --error-format = `text|shortxml|longxml`
  Spécifie le format de la sortie d'erreur. La valeur par défaut est `text`. Les autres options génèrent des formats XML, avec `longxml` générant plus de détails.

- error-limit

  --error-limit = *N*
  Spécifie la limite d'erreur avec une plage de valeur de `1` à `999`. La valeur par défaut est `100`. Le traitement s'arrête lorsque la limite d'erreur est atteinte. Utile pour limiter

l'utilisation du processeur pendant la validation/transformation.

- help

  `--help`
  Affiche le texte d'aide pour la commande. Par exemple, `valany --h`. (en alternative, la commande `help` peut être utilisée avec un argument. Par exemple : `help valany`.)

- network-timeout

  `--network-timeout =` *`VALUE`*
  Spécifie le timeout en secondes pour les opérations I/O à distance. La valeur par défaut est : `40`.

- verbose

  `--verbose =` `true|false`
  Une valeur de `true` permet la sortie d'informations supplémentaires pendant la validation. La valeur par défaut est `false`.
  Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- verbose-output

  `--verbose-output =` *`FILE`*
  Écrit la sortie verbeuse sur *`FILE`*.

- version

  `--version`
  Affiche la version de RaptorXML+XBRL Server. Si utilisé avec une commande, placer `--version` avant la commande.

 © 2018 Altova GmbH

3.6 Commandes JSON/Avro

Les commandes JSON peuvent être utilisées pour vérifier la validité et la bonne formation de schéma JSON et de documents d'instance. Ces commandes sont listées ci-dessous et sont décrites en détails dans les sous-sections de cette section :

avroextractschema	Extrait le schéma Avro d'un fichier binaire Avro
valavro	Valide les données dans un ou plusieurs binaires Avro par rapport au schéma Avro respectif de chaque binaire
valavrojson	Valide un ou plusieurs fichiers de données JSON par rapport à un schéma Avro
valavroschema	Valide un schéma Avro par rapport à la spécification de schéma Avro
valjsonschema	Contrôle la validité des documents de schéma JSON.
valjson	Contrôle la validité des documents JSON.
wfjson	Contrôle la bonne formation des documents JSON.

avroextractschema

Un fichier binaire Avro contient un bloc de données Avro précédé du schéma Avro qui définit la structure du bloc de données. La commande **avroextractschema** extrait le schéma Avro du binaire Avro et sérialise le schéma Avro en tant que JSON.

Windows **RaptorXMLXBRL avroextractschema [options] --avrooutput=*AvroSchemaFile AvroBinaryFile***

Linux **raptorxmlxbrl avroextractschema [options] --avrooutput=*AvroSchemaFile AvroBinaryFile***

Mac **raptorxmlxbrl avroextractschema [options] --avrooutput=*AvroSchemaFile AvroBinaryFile***

L'argument *AvroBinaryFile* spécifie le fichier binaire Avro à partir duquel le schéma Avro sera extrait. L'option `--avrooutput` spécifie l'emplacement du schéma Avro extrait.

Exemples

- **raptorxmlxbrl** `avroextractschema --avrooutput=c:\MyAvroSchema.avsc c:\MyAvroBinary.avro`

Casse et lignes obliques sur la ligne de commande

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

* Veuillez noter que les caractères minuscules (`raptorxmlxbrl`) fonctionnent sur toutes les plates-formes (Windows, Linux et Mac), alors qu'une écriture en majuscule/minuscule (`RaptorXMLXBRL`) ne fonctionne que sur Windows et Mac.
* Utiliser les barres obliques penchées vers l'avant pour Linux et Mac, et les barres obliques penchées vers l'arrière sur Windows.

Options

Les options de la commande sont listées ci-dessous, et organisées en groupes. Les valeurs peuvent être spécifiées sans guillemets sauf dans deux cas : (i) lorsque le string de valeur contient des espaces, ou (ii) si la présence de guillemets est explicitement requise dans la description de l'option.

Traitement

output, **avrooutput**

--output = *FILE*, **--avrooutput =** *FILE*

Définit l'emplacement du fichier de sortie Avro.

- recurse

  --recurse = `true|false`
  Utilisé pour sélectionner des fichiers dans le cadre de sous-répertoires, y compris des archives ZIP. Si `true`, l'argument *`InputFile`* de la commande sélectionnera aussi le fichier spécifié dans les sous-répertoires. Par exemple : `test.zip|zip\test.xml` sélectionnera des fichiers nommés `test.xml` à tous les niveaux de dossier du dossier zip. Les caractères génériques `*` et `?` peuvent être utilisés. Donc, `*.xml` sélectionnera tous les fichiers `.xml` dans le dossier (zip). La valeur par défaut de l'option est `false`.
  Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- Catalogues et ressources globales
  - catalog

    --catalog = *`FILE`*
    Spécifie le chemin absolu vers un fichier de catalogue root qui n'est pas le fichier catalogue root installé. La valeur par défaut est le chemin absolu vers le fichier de catalogue root installé (`<installation-folder>\Altova\RaptorXMLXBRLServer2018\etc\RootCatalog.xml`). Voir la section, Catalogues XML, pour plus d'informations à propos du travail avec les catalogues.

  - user-catalog

    --user-catalog = *`FILE`*
    Spécifie le chemin absolu vers un catalogue XML à utiliser en plus du catalogue root. Voir la section, Catalogues XML, pour plus d'informations concernant le travail avec des catalogues.

  - enable-globalresources

    --enable-globalresources = `true|false`
    Active les ressources globales. La valeur par défaut est `false`.
    Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

  - globalresourceconfig [gc]

    --gc | --globalresourceconfig = *`VALUE`*
    Spécifie la configuration active de la ressource globale (et active les ressources globales).

  - globalresourcefile [gr]

    --gr | --globalresourcefile = *`FILE`*
    Spécifie le fichier de ressource globale (et active les ressources globales).

- Messages, erreurs, aide, timeout, version
  - error-format

--error-format = `text|shortxml|longxml`
Spécifie le format de la sortie d'erreur. La valeur par défaut est `text`. Les autres options génèrent des formats XML, avec `longxml` générant plus de détails.

- error-limit

--error-limit = *N*
Spécifie la limite d'erreur avec une plage de valeur de `1` à `999`. La valeur par défaut est `100`. Le traitement s'arrête lorsque la limite d'erreur est atteinte. Utile pour limiter l'utilisation du processeur pendant la validation/transformation.

- help

--help
Affiche le texte d'aide pour la commande. Par exemple, `valany --h`. (en alternative, la commande `help` peut être utilisée avec un argument. Par exemple : `help valany`.)

- log-output

--log-output = *FILE*
Écrit la sortie de log dans l'URL de fichier spécifié. Veuillez vous assurer que la CLI a une permission d'écriture dans l'emplacement de la sortie.

- network-timeout

--network-timeout = *VALUE*
Spécifie le timeout en secondes pour les opérations I/O à distance. La valeur par défaut est : `40`.

- verbose

--verbose = `true|false`
Une valeur de `true` permet la sortie d'informations supplémentaires pendant la validation. La valeur par défaut est `false`.
Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- verbose-output

--verbose-output = *FILE*
Écrit la sortie verbeuse sur *FILE*.

- version

--version
Affiche la version de RaptorXML+XBRL Server. Si utilisé avec une commande, placer `--version` avant la commande.

- warning-limit

--warning-limit = *VALUE*
Spécifie la limite d'avertissement dans la plage `1-65535`. Le traitement se poursuit si cette limite a été atteinte, mais d'autres avertissements ne sont pas rapportés. La valeur par défaut est `100`.

 © 2018 Altova GmbH

valavro (avro)

La commande **valavro | avro** valide le bloc de données dans un ou plusieurs fichiers binaires Avro par rapport aux schéma Avro respectifs dans chaque fichier binaire.

Windows	**RaptorXMLXBRL valavro | avro [options] *AvroBinaryFile***
Linux	**raptorxmlxbrl valavro | avro [options] *AvroBinaryFile***
Mac	**raptorxmlxbrl valavro | avro [options] *AvroBinaryFile***

L'argument *AvroBinaryFile* spécifie un ou plusieurs fichiers binaires Avro à valider. En particulier, le bloc de données dans chaque fichier binaire Avro est validé par rapport au Schéma Avro dans ce fichier binaire. Pour valider plusieurs binaires Avro, il faut soit (i) lister les fichiers à valider sur la CLI, chaque fichier étant séparé du suivant par un espace ; ou (ii) lister les fichiers à valider dans un fichier texte (fichier `.txt`), avec un nom de fichier par ligne, et fournir ce fichier de texte en tant que l'argument *AvroBinaryFile* avec l'option `--listfile` définie sur `true` (*voir la liste des Options ci-dessous*).

Exemples

- **raptorxmlxbrl** `valavro c:\MyAvroBinary.avro`
- **raptorxmlxbrl** `valavro c:\MyAvroBinary01.avro c:\MyAvroBinary02.avro`
- **raptorxmlxbrl** `avro--listfile=true c:\MyFileList.txt`

Casse et lignes obliques sur la ligne de commande

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

* Veuillez noter que les caractères minuscules (`raptorxmlxbrl`) fonctionnent sur toutes les plates-formes (Windows, Linux et Mac), alors qu'une écriture en majuscule/minuscule (`RaptorXMLXBRL`) ne fonctionne que sur Windows et Mac.
* Utiliser les barres obliques penchées vers l'avant pour Linux et Mac, et les barres obliques penchées vers l'arrière sur Windows.

Options

Les options de la commande sont listées ci-dessous, et organisées en groupes. Les valeurs peuvent être spécifiées sans guillemets sauf dans deux cas : (i) lorsque le string de valeur contient des espaces, ou (ii) si la présence de guillemets est explicitement requise dans la description de l'option.

Traitement

listfile

--listfile = `true|false`

Si `true`, traiter l'argument *`InputFile`* de la commande en tant que fichier de texte contenant un nom de fichier par ligne. La valeur par défaut est `false`. (Une alternative est de lister les fichiers sur la CLI avec un espace en tant que séparateur. Veuillez noter, néanmoins, que les CLI ont une limitation de caractères maximum.) Veuillez noter que l'option `--listfile` s'applique uniquement aux arguments, et non pas aux options.
Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- recurse

  `--recurse =` `true|false`
  Utilisé pour sélectionner des fichiers dans le cadre de sous-répertoires, y compris des archives ZIP. Si `true`, l'argument *`InputFile`* de la commande sélectionnera aussi le fichier spécifié dans les sous-répertoires. Par exemple : `test.zip|zip\test.xml` sélectionnera des fichiers nommés `test.xml` à tous les niveaux de dossier du dossier zip. Les caractères génériques `*` et `?` peuvent être utilisés. Donc, `*.xml` sélectionnera tous les fichiers `.xml` dans le dossier (zip). La valeur par défaut de l'option est `false`.
  Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- Catalogues et ressources globales
  - catalog

    `--catalog =` *`FILE`*
    Spécifie le chemin absolu vers un fichier de catalogue root qui n'est pas le fichier catalogue root installé. La valeur par défaut est le chemin absolu vers le fichier de catalogue root installé (`<installation-folder>\Altova\RaptorXMLXBRLServer2018\etc\RootCatalog.xml`). Voir la section, Catalogues XML, pour plus d'informations à propos du travail avec les catalogues.

  - user-catalog

    `--user-catalog =` *`FILE`*
    Spécifie le chemin absolu vers un catalogue XML à utiliser en plus du catalogue root. Voir la section, Catalogues XML, pour plus d'informations concernant le travail avec des catalogues.

  - enable-globalresources

    `--enable-globalresources =` `true|false`
    Active les ressources globales. La valeur par défaut est `false`.
    Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

  - globalresourceconfig [gc]

    `--gc | --globalresourceconfig =` *`VALUE`*
    Spécifie la configuration active de la ressource globale (et active les ressources globales).

  - globalresourcefile [gr]

--gr | --globalresourcefile = *FILE*
Spécifie le fichier de ressource globale (et active les ressources globales).

- Messages, erreurs, aide, timeout, version
  - error-format

    --error-format = `text|shortxml|longxml`
    Spécifie le format de la sortie d'erreur. La valeur par défaut est `text`. Les autres options génèrent des formats XML, avec `longxml` générant plus de détails.

  - error-limit

    --error-limit = *N*
    Spécifie la limite d'erreur avec une plage de valeur de `1` à `999`. La valeur par défaut est `100`. Le traitement s'arrête lorsque la limite d'erreur est atteinte. Utile pour limiter l'utilisation du processeur pendant la validation/transformation.

  - help

    --help
    Affiche le texte d'aide pour la commande. Par exemple, `valany --h`. (en alternative, la commande `help` peut être utilisée avec un argument. Par exemple : `help valany`.)

  - log-output

    --log-output = *FILE*
    Écrit la sortie de log dans l'URL de fichier spécifié. Veuillez vous assurer que la CLI a une permission d'écriture dans l'emplacement de la sortie.

  - network-timeout

    --network-timeout = *VALUE*
    Spécifie le timeout en secondes pour les opérations I/O à distance. La valeur par défaut est : `40`.

  - verbose

    --verbose = `true|false`
    Une valeur de `true` permet la sortie d'informations supplémentaires pendant la validation. La valeur par défaut est `false`.
    Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

  - verbose-output

    --verbose-output = *FILE*
    Écrit la sortie verbeuse sur *FILE*.

  - version

    --version
    Affiche la version de RaptorXML+XBRL Server. Si utilisé avec une commande, placer `--version` avant la commande.

 © 2018 Altova GmbH

- warning-limit

  `--warning-limit =` *`VALUE`*

  Spécifie la limite d'avertissement dans la plage `1-65535`. Le traitement se poursuit si cette limite a été atteinte, mais d'autres avertissements ne sont pas rapportés. La valeur par défaut est `100`.

valavrojson (avrojson)

La commande **valavrojson | avrojson** valide un document JSON par rapport à un schéma Avro.

Windows **RaptorXMLXBRL valavrojson | avrojson [options] --avroschema=*AvroSchema JSONFile***

Linux **raptorxmlxbrl valavrojson | avrojson [options] --avroschema=*AvroSchema JSONFile***

Mac **raptorxmlxbrl valavrojson | avrojson [options] --avroschema=*AvroSchema JSONFile***

L'argument *JSONFile* spécifie le document JSON à valider. L'option **--avroschema** spécifie le schéma Avro par rapport auquel le document JSON doit être validé. Pour valider plusieurs fichiers JSON, il faut soit (i) lister les fichiers à valider sur la CLI, chaque fichier étant séparé du suivant par un espace ; ou (ii) lister les fichiers à valider dans un fichier texte (fichier `.txt`), avec un nom de fichier par ligne, et fournir ce fichier de texte en tant que l'argument *JSONFile* avec l'option `--listfile` définie sur `true` (*voir la liste des Options ci-dessous*).

Exemples

- **raptorxmlxbrl** `valavrojson --avroschema=c:\MyAvroSchema.avsc c:\MyJSONDataFile.json`
- **raptorxmlxbrl** `avrojson --avroschema=c:\MyAvroSchema.avsc c:\MyJSONDataFile.json`

Casse et lignes obliques sur la ligne de commande

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

* Veuillez noter que les caractères minuscules (`raptorxmlxbrl`) fonctionnent sur toutes les plates-formes (Windows, Linux et Mac), alors qu'une écriture en majuscule/minuscule (`RaptorXMLXBRL`) ne fonctionne que sur Windows et Mac.
* Utiliser les barres obliques penchées vers l'avant pour Linux et Mac, et les barres obliques penchées vers l'arrière sur Windows.

Options

Les options de la commande sont listées ci-dessous, et organisées en groupes. Les valeurs peuvent être spécifiées sans guillemets sauf dans deux cas : (i) lorsque le string de valeur contient des espaces, ou (ii) si la présence de guillemets est explicitement requise dans la description de l'option.

Traitement

- listfile

  --listfile = `true|false`
  Si `true`, traiter l'argument *`InputFile`* de la commande en tant que fichier de texte contenant un nom de fichier par ligne. La valeur par défaut est `false`. (Une alternative est de lister les fichiers sur la CLI avec un espace en tant que séparateur. Veuillez noter, néanmoins, que les CLI ont une limitation de caractères maximum.) Veuillez noter que l'option `--listfile` s'applique uniquement aux arguments, et non pas aux options.
  Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- recurse

  --recurse = `true|false`
  Utilisé pour sélectionner des fichiers dans le cadre de sous-répertoires, y compris des archives ZIP. Si `true`, l'argument *`InputFile`* de la commande sélectionnera aussi le fichier spécifié dans les sous-répertoires. Par exemple : `test.zip|zip\test.xml` sélectionnera des fichiers nommés `test.xml` à tous les niveaux de dossier du dossier zip. Les caractères génériques `*` et `?` peuvent être utilisés. Donc, `*.xml` sélectionnera tous les fichiers `.xml` dans le dossier (zip). La valeur par défaut de l'option est `false`.
  Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- Catalogues et ressources globales

  - catalog

    --catalog = *`FILE`*
    Spécifie le chemin absolu vers un fichier de catalogue root qui n'est pas le fichier catalogue root installé. La valeur par défaut est le chemin absolu vers le fichier de catalogue root installé (`<installation-folder>\Altova\RaptorXMLXBRLServer2018\etc\RootCatalog.xml`). Voir la section, Catalogues XML, pour plus d'informations à propos du travail avec les catalogues.

  - user-catalog

    --user-catalog = *`FILE`*
    Spécifie le chemin absolu vers un catalogue XML à utiliser en plus du catalogue root. Voir la section, Catalogues XML, pour plus d'informations concernant le travail avec des catalogues.

  - enable-globalresources

    --enable-globalresources = `true|false`
    Active les ressources globales. La valeur par défaut est `false`.
    Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

  - globalresourceconfig [gc]

    --gc | --globalresourceconfig = *`VALUE`*
    Spécifie la configuration active de la ressource globale (et active les ressources globales).

- globalresourcefile [gr]

  `--gr | --globalresourcefile =` *`FILE`*
  Spécifie le fichier de ressource globale (et active les ressources globales).

- Messages, erreurs, aide, timeout, version
  - error-format

    `--error-format =` `text|shortxml|longxml`
    Spécifie le format de la sortie d'erreur. La valeur par défaut est `text`. Les autres options génèrent des formats XML, avec `longxml` générant plus de détails.

  - error-limit

    `--error-limit =` *`N`*
    Spécifie la limite d'erreur avec une plage de valeur de `1` à `999`. La valeur par défaut est `100`. Le traitement s'arrête lorsque la limite d'erreur est atteinte. Utile pour limiter l'utilisation du processeur pendant la validation/transformation.

  - help

    `--help`
    Affiche le texte d'aide pour la commande. Par exemple, `valany --h`. (en alternative, la commande `help` peut être utilisée avec un argument. Par exemple : `help valany`.)

  - log-output

    `--log-output =` *`FILE`*
    Écrit la sortie de log dans l'URL de fichier spécifié. Veuillez vous assurer que la CLI a une permission d'écriture dans l'emplacement de la sortie.

  - network-timeout

    `--network-timeout =` *`VALUE`*
    Spécifie le timeout en secondes pour les opérations I/O à distance. La valeur par défaut est : `40`.

  - verbose

    `--verbose =` `true|false`
    Une valeur de `true` permet la sortie d'informations supplémentaires pendant la validation. La valeur par défaut est `false`.
    Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

  - verbose-output

    `--verbose-output =` *`FILE`*
    Écrit la sortie verbeuse sur *`FILE`*.

  - version

    `--version`

 © 2018 Altova GmbH

Affiche la version de RaptorXML+XBRL Server. Si utilisé avec une commande, placer `--version` avant la commande.

- warning-limit

  `--warning-limit =` *`VALUE`*

  Spécifie la limite d'avertissement dans la plage `1-65535`. Le traitement se poursuit si cette limite a été atteinte, mais d'autres avertissements ne sont pas rapportés. La valeur par défaut est `100`.

valavroschema (avroschema)

La commande **valavroschema | avroschema** valide un ou plusieurs documents de schéma Avro par rapport à la spécification de schéma Avro.

Windows	**RaptorXMLXBRL valavroschema | avroschema [options] *AvroSchema***
Linux	**raptorxmlxbrl valavroschema | avroschema [options] *AvroSchema***
Mac	**raptorxmlxbrl valavroschema | avroschema [options] *AvroSchema***

L'argument *AvroSchema* spécifie le document de schéma Avro à valider. Pour valider plusieurs schémas Avro, il faut soit (i) lister les fichiers à valider sur la CLI, chaque fichier étant séparé du suivant par un espace ; ou (ii) lister les fichiers à valider dans un fichier texte (fichier `.txt`), avec un nom de fichier par ligne, et fournir ce fichier de texte en tant que l'argument *AvroSchema* avec l'option `--listfile` définie sur `true` (*voir la liste des Options ci-dessous*).

Exemples

- **raptorxmlxbrl** `valavroschema c:\MyAvroSchema.avsc`
- **raptorxmlxbrl** `valavroschema c:\MyAvroSchema01.avsc c:\MyAvroSchema02.avsc`
- **raptorxmlxbrl** `avroschema--listfile=true c:\MyFileList.txt`

Casse et lignes obliques sur la ligne de commande

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

* Veuillez noter que les caractères minuscules (`raptorxmlxbrl`) fonctionnent sur toutes les plates-formes (Windows, Linux et Mac), alors qu'une écriture en majuscule/minuscule (`RaptorXMLXBRL`) ne fonctionne que sur Windows et Mac.
* Utiliser les barres obliques penchées vers l'avant pour Linux et Mac, et les barres obliques penchées vers l'arrière sur Windows.

Options

Les options de la commande sont listées ci-dessous, et organisées en groupes. Les valeurs peuvent être spécifiées sans guillemets sauf dans deux cas : (i) lorsque le string de valeur contient des espaces, ou (ii) si la présence de guillemets est explicitement requise dans la description de l'option.

Traitement

listfile

--listfile = `true|false`
Si `true`, traiter l'argument *InputFile* de la commande en tant que fichier de texte

 © 2018 Altova GmbH

contenant un nom de fichier par ligne. La valeur par défaut est `false`. (Une alternative est de lister les fichiers sur la CLI avec un espace en tant que séparateur. Veuillez noter, néanmoins, que les CLI ont une limitation de caractères maximum.) Veuillez noter que l'option `--listfile` s'applique uniquement aux arguments, et non pas aux options.
Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- recurse

  `--recurse =` `true|false`
  Utilisé pour sélectionner des fichiers dans le cadre de sous-répertoires, y compris des archives ZIP. Si `true`, l'argument *`InputFile`* de la commande sélectionnera aussi le fichier spécifié dans les sous-répertoires. Par exemple : `test.zip|zip\test.xml` sélectionnera des fichiers nommés `test.xml` à tous les niveaux de dossier du dossier zip. Les caractères génériques `*` et `?` peuvent être utilisés. Donc, `*.xml` sélectionnera tous les fichiers `.xml` dans le dossier (zip). La valeur par défaut de l'option est `false`.
  Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- Catalogues et ressources globales
  - catalog

    `--catalog =` *`FILE`*
    Spécifie le chemin absolu vers un fichier de catalogue root qui n'est pas le fichier catalogue root installé. La valeur par défaut est le chemin absolu vers le fichier de catalogue root installé (`<installation-folder>\Altova\RaptorXMLXBRLServer2018\etc\RootCatalog.xml`). Voir la section, Catalogues XML, pour plus d'informations à propos du travail avec les catalogues.

  - user-catalog

    `--user-catalog =` *`FILE`*
    Spécifie le chemin absolu vers un catalogue XML à utiliser en plus du catalogue root. Voir la section, Catalogues XML, pour plus d'informations concernant le travail avec des catalogues.

  - enable-globalresources

    `--enable-globalresources =` `true|false`
    Active les ressources globales. La valeur par défaut est `false`.
    Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

  - globalresourceconfig [gc]

    `--gc | --globalresourceconfig =` *`VALUE`*
    Spécifie la configuration active de la ressource globale (et active les ressources globales).

  - globalresourcefile [gr]

--gr | --globalresourcefile = *FILE*
Spécifie le fichier de ressource globale (et active les ressources globales).

- Messages, erreurs, aide, timeout, version
  - error-format

    --error-format = `text|shortxml|longxml`
    Spécifie le format de la sortie d'erreur. La valeur par défaut est `text`. Les autres options génèrent des formats XML, avec `longxml` générant plus de détails.

  - error-limit

    --error-limit = *N*
    Spécifie la limite d'erreur avec une plage de valeur de `1` à `999`. La valeur par défaut est `100`. Le traitement s'arrête lorsque la limite d'erreur est atteinte. Utile pour limiter l'utilisation du processeur pendant la validation/transformation.

  - help

    --help
    Affiche le texte d'aide pour la commande. Par exemple, `valany --h`. (en alternative, la commande `help` peut être utilisée avec un argument. Par exemple : `help valany`.)

  - log-output

    --log-output = *FILE*
    Écrit la sortie de log dans l'URL de fichier spécifié. Veuillez vous assurer que la CLI a une permission d'écriture dans l'emplacement de la sortie.

  - network-timeout

    --network-timeout = *VALUE*
    Spécifie le timeout en secondes pour les opérations I/O à distance. La valeur par défaut est : `40`.

  - verbose

    --verbose = `true|false`
    Une valeur de `true` permet la sortie d'informations supplémentaires pendant la validation. La valeur par défaut est `false`.
    Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

  - verbose-output

    --verbose-output = *FILE*
    Écrit la sortie verbeuse sur *FILE*.

  - version

    --version
    Affiche la version de RaptorXML+XBRL Server. Si utilisé avec une commande, placer `--version` avant la commande.

 © 2018 Altova GmbH

- warning-limit

  `--warning-limit =` *`VALUE`*

  Spécifie la limite d'avertissement dans la plage `1-65535`. Le traitement se poursuit si cette limite a été atteinte, mais d'autres avertissements ne sont pas rapportés. La valeur par défaut est `100`.

valjsonschema (jsonschema)

La commande **valjsonschema | jsonschema** valide un ou plusieurs documents de schéma JSON conformément aux spécifications JSON Schema Draft 4.

Windows	**RaptorXMLXBRL valjsonschema | jsonschema [options] *InputFile***
Linux	**raptorxmlxbrl valjsonschema | jsonschema [options] *InputFile***
Mac	**raptorxmlxbrl valjsonschema | jsonschema [options] *InputFile***

L'argument *InputFile* spécifie le document de schéma JSON à valider. Pour valider plusieurs documents, il faut soit (i) lister les fichiers à valider sur la CLI, chaque fichier étant séparé du suivant par un espace ; ou (ii) lister les fichiers à valider dans un fichier texte (fichier `.txt`), avec un nom de fichier par ligne, et fournir ce fichier de texte en tant que l'argument *InputFile* avec l'option `--listfile` définie sur `true` (*voir la liste des Options ci-dessous*).

Exemples

- **raptorxmlxbrl** `valjsonschema c:\MyJSONSchema.json`
- **raptorxmlxbrl** `jsonschema c:\MyJSONSchema-01.json c:\MyJSONSchema-02.json`
- **raptorxmlxbrl** `jsonschema --listfile=true c:\FileList.txt`

▼ Casse et lignes obliques sur la ligne de commande

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

* Veuillez noter que les caractères minuscules (`raptorxmlxbrl`) fonctionnent sur toutes les plates-formes (Windows, Linux et Mac), alors qu'une écriture en majuscule/minuscule (`RaptorXMLXBRL`) ne fonctionne que sur Windows et Mac.
* Utiliser les barres obliques penchées vers l'avant pour Linux et Mac, et les barres obliques penchées vers l'arrière sur Windows.

Options

Les options de la commande sont listées ci-dessous, et organisées en groupes. Les valeurs peuvent être spécifiées sans guillemets sauf dans deux cas : (i) lorsque le string de valeur contient des espaces, ou (ii) si la présence de guillemets est explicitement requise dans la description de l'option.

▼ Validation et traitement

▼ listfile

--listfile = `true|false`
Si `true`, traiter l'argument *InputFile* de la commande en tant que fichier de texte

 © 2018 Altova GmbH

contenant un nom de fichier par ligne. La valeur par défaut est `false`. (Une alternative est de lister les fichiers sur la CLI avec un espace en tant que séparateur. Veuillez noter, néanmoins, que les CLI ont une limitation de caractères maximum.) Veuillez noter que l'option `--listfile` s'applique uniquement aux arguments, et non pas aux options.
Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- recurse

  `--recurse =` `true|false`
  Utilisé pour sélectionner des fichiers dans le cadre de sous-répertoires, y compris des archives ZIP. Si `true`, l'argument *`InputFile`* de la commande sélectionnera aussi le fichier spécifié dans les sous-répertoires. Par exemple : `test.zip|zip\test.xml` sélectionnera des fichiers nommés `test.xml` à tous les niveaux de dossier du dossier zip. Les caractères génériques `*` et `?` peuvent être utilisés. Donc, `*.xml` sélectionnera tous les fichiers `.xml` dans le dossier (zip). La valeur par défaut de l'option est `false`.
  Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- Catalogues et ressources globales
  - catalog

    `--catalog =` *`FILE`*
    Spécifie le chemin absolu vers un fichier de catalogue root qui n'est pas le fichier catalogue root installé. La valeur par défaut est le chemin absolu vers le fichier de catalogue root installé (`<installation-folder>\Altova\RaptorXMLXBRLServer2018\etc\RootCatalog.xml`). Voir la section, Catalogues XML, pour plus d'informations à propos du travail avec les catalogues.

  - user-catalog

    `--user-catalog =` *`FILE`*
    Spécifie le chemin absolu vers un catalogue XML à utiliser en plus du catalogue root. Voir la section, Catalogues XML, pour plus d'informations concernant le travail avec des catalogues.

  - enable-globalresources

    `--enable-globalresources =` `true|false`
    Active les ressources globales. La valeur par défaut est `false`.
    Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

  - globalresourceconfig [gc]

    `--gc | --globalresourceconfig =` *`VALUE`*
    Spécifie la configuration active de la ressource globale (et active les ressources globales).

  - globalresourcefile [gr]

--gr | --globalresourcefile = *FILE*
Spécifie le fichier de ressource globale (et active les ressources globales).

- Messages, erreurs, aide, timeout, version
  - error-format

    --error-format = `text|shortxml|longxml`
    Spécifie le format de la sortie d'erreur. La valeur par défaut est `text`. Les autres options génèrent des formats XML, avec `longxml` générant plus de détails.

  - error-limit

    --error-limit = *N*
    Spécifie la limite d'erreur avec une plage de valeur de `1` à `999`. La valeur par défaut est `100`. Le traitement s'arrête lorsque la limite d'erreur est atteinte. Utile pour limiter l'utilisation du processeur pendant la validation/transformation.

  - help

    --help
    Affiche le texte d'aide pour la commande. Par exemple, `valany --h`. (en alternative, la commande `help` peut être utilisée avec un argument. Par exemple : `help valany`.)

  - log-output

    --log-output = *FILE*
    Écrit la sortie de log dans l'URL de fichier spécifié. Veuillez vous assurer que la CLI a une permission d'écriture dans l'emplacement de la sortie.

  - network-timeout

    --network-timeout = *VALUE*
    Spécifie le timeout en secondes pour les opérations I/O à distance. La valeur par défaut est : `40`.

  - verbose

    --verbose = `true|false`
    Une valeur de `true` permet la sortie d'informations supplémentaires pendant la validation. La valeur par défaut est `false`.
    Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

  - verbose-output

    --verbose-output = *FILE*
    Écrit la sortie verbeuse sur *FILE*.

  - version

    --version
    Affiche la version de RaptorXML+XBRL Server. Si utilisé avec une commande, placer `--version` avant la commande.

 © 2018 Altova GmbH

- warning-limit

  `--warning-limit =` *`VALUE`*

  Spécifie la limite d'avertissement dans la plage `1-65535`. Le traitement se poursuit si cette limite a été atteinte, mais d'autres avertissements ne sont pas rapportés. La valeur par défaut est `100`.

valjson (json)

La commande **`valjson | json`** valide un ou plusieurs documents d'instance JSON conformément au schéma JSON fournit avec l'option `--schema`.

Windows **`RaptorXMLXBRL valjson | json [options] --jsonschema=`*`File InputFile`***

Linux **`raptorxmlxbrl valjson | json [options] --jsonschema=`*`File InputFile`***

Mac **`raptorxmlxbrl valjson | json [options] --jsonschema=`*`File InputFile`***

L'argument *`InputFile`* spécifie le document d'instance JSON à valider. Pour valider plusieurs documents, il faut soit (i) lister les fichiers à valider sur la CLI, chaque fichier étant séparé du suivant par un espace ; ou (ii) lister les fichiers à valider dans un fichier texte (fichier `.txt`), avec un nom de fichier par ligne, et fournir ce fichier de texte en tant que l'argument *`InputFile`* avec l'option `--listfile` définie sur `true` (*voir la liste des Options ci-dessous*).

Exemples

- **`raptorxmlxbrl`** `valjson --jsonschema=c:\MyJSONSchema.json c:\MyJSONInstance.json`
- **`raptorxmlxbrl`** `json --josnschema=c:\MyJSONSchema.json c:\MyJSONInstance-01.json c:\MyJSONInstance-02.json`
- **`raptorxmlxbrl`** `json --josnschema=c:\MyJSONSchema.json --listfile=true c:\FileList.txt`

Casse et lignes obliques sur la ligne de commande

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

* Veuillez noter que les caractères minuscules (`raptorxmlxbrl`) fonctionnent sur toutes les plates-formes (Windows, Linux et Mac), alors qu'une écriture en majuscule/minuscule (`RaptorXMLXBRL`) ne fonctionne que sur Windows et Mac.
* Utiliser les barres obliques penchées vers l'avant pour Linux et Mac, et les barres obliques penchées vers l'arrière sur Windows.

Options

Les options de la commande sont listées ci-dessous, et organisées en groupes. Les valeurs peuvent être spécifiées sans guillemets sauf dans deux cas : (i) lorsque le string de valeur contient des espaces, ou (ii) si la présence de guillemets est explicitement requise dans la description de l'option.

Validation et traitement

 © 2018 Altova GmbH

- schema, **jsonschema**

  --schema = *FILE*, **--jsonschema =** *FILE*
  Spécifie le chemin vers le document de Schéma JSON à utiliser pour la validation des documents d'instance JSON.

- listfile

  --listfile = `true|false`
  Si `true`, traiter l'argument *InputFile* de la commande en tant que fichier de texte contenant un nom de fichier par ligne. La valeur par défaut est `false`. (Une alternative est de lister les fichiers sur la CLI avec un espace en tant que séparateur. Veuillez noter, néanmoins, que les CLI ont une limitation de caractères maximum.) Veuillez noter que l'option `--listfile` s'applique uniquement aux arguments, et non pas aux options.
  Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- recurse

  --recurse = `true|false`
  Utilisé pour sélectionner des fichiers dans le cadre de sous-répertoires, y compris des archives ZIP. Si `true`, l'argument *InputFile* de la commande sélectionnera aussi le fichier spécifié dans les sous-répertoires. Par exemple : `test.zip|zip\test.xml` sélectionnera des fichiers nommés `test.xml` à tous les niveaux de dossier du dossier zip. Les caractères génériques `*` et `?` peuvent être utilisés. Donc, `*.xml` sélectionnera tous les fichiers `.xml` dans le dossier (zip). La valeur par défaut de l'option est `false`.
  Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- json5

  --json5 = `true|false`
  Active la prise en charge JSON5. La valeur par défaut est `false`.
  Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- Catalogues et ressources globales
  - catalog

    --catalog = *FILE*
    Spécifie le chemin absolu vers un fichier de catalogue root qui n'est pas le fichier catalogue root installé. La valeur par défaut est le chemin absolu vers le fichier de catalogue root installé (`<installation-folder>\Altova\RaptorXMLXBRLServer2018\etc\RootCatalog.xml`). Voir la section, Catalogues XML, pour plus d'informations à propos du travail avec les catalogues.

  - user-catalog

    --user-catalog = *FILE*
    Spécifie le chemin absolu vers un catalogue XML à utiliser en plus du catalogue root. Voir la section, Catalogues XML, pour plus d'informations concernant le travail avec des catalogues.

- enable-globalresources

  `--enable-globalresources =` `true|false`
  Active les ressources globales. La valeur par défaut est `false`.
  Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- globalresourceconfig [gc]

  `--gc | --globalresourceconfig =` *`VALUE`*
  Spécifie la configuration active de la ressource globale (et active les ressources globales).

- globalresourcefile [gr]

  `--gr | --globalresourcefile =` *`FILE`*
  Spécifie le fichier de ressource globale (et active les ressources globales).

- Messages, erreurs, aide, timeout, version
  - error-format

    `--error-format =` `text|shortxml|longxml`
    Spécifie le format de la sortie d'erreur. La valeur par défaut est `text`. Les autres options génèrent des formats XML, avec `longxml` générant plus de détails.

  - error-limit

    `--error-limit =` *`N`*
    Spécifie la limite d'erreur avec une plage de valeur de `1` à `999`. La valeur par défaut est `100`. Le traitement s'arrête lorsque la limite d'erreur est atteinte. Utile pour limiter l'utilisation du processeur pendant la validation/transformation.

  - help

    `--help`
    Affiche le texte d'aide pour la commande. Par exemple, `valany --h`. (en alternative, la commande `help` peut être utilisée avec un argument. Par exemple : `help valany`.)

  - log-output

    `--log-output =` *`FILE`*
    Écrit la sortie de log dans l'URL de fichier spécifié. Veuillez vous assurer que la CLI a une permission d'écriture dans l'emplacement de la sortie.

  - network-timeout

    `--network-timeout =` *`VALUE`*
    Spécifie le timeout en secondes pour les opérations I/O à distance. La valeur par défaut est : `40`.

  - verbose

    `--verbose =` `true|false`

 © 2018 Altova GmbH

Une valeur de `true` permet la sortie d'informations supplémentaires pendant la validation. La valeur par défaut est `false`.
Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- verbose-output

  `--verbose-output =` *`FILE`*
  Écrit la sortie verbeuse sur *`FILE`*.

- version

  `--version`
  Affiche la version de RaptorXML+XBRL Server. Si utilisé avec une commande, placer `--version` avant la commande.

- warning-limit

  `--warning-limit =` *`VALUE`*
  Spécifie la limite d'avertissement dans la plage `1-65535`. Le traitement se poursuit si cette limite a été atteinte, mais d'autres avertissements ne sont pas rapportés. La valeur par défaut est `100`.

wfjson

La commande **wfjson** contrôle un ou plusieurs documents JSON conformément à la spécification ECMA-404 concernant la bonne formation.

Windows	**RaptorXMLXBRL wfjson [options]** ***InputFile***
Linux	**raptorxmlxbrl wfjson [options]** ***InputFile***
Mac	**raptorxmlxbrl wfjson [options]** ***InputFile***

L'argument *InputFile* spécifie le document JSON (schéma ou instance) dont vous souhaitez contrôler la bonne formation. Pour contrôler plusieurs documents, il faut soit (i) lister les fichiers à contrôler sur la CLI, chaque fichier étant séparé du suivant par un espace ; ou (ii) lister les fichiers à contrôler dans un fichier texte (fichier `.txt`), avec un nom de fichier par ligne, et fournir ce fichier de texte en tant que l'argument *InputFile* avec l'option `--listfile` définie sur `true` (*voir la liste des Options ci-dessous*).

Exemples

- **raptorxmlxbrl** `wfjson c:\MyJSONFile.json`
- **raptorxmlxbrl** `wfjson c:\MyJSONFile-01.json c:\MyJSONFile-02.json`
- **raptorxmlxbrl** `wfjson --listfile=true c:\FileList.txt`

Casse et lignes obliques sur la ligne de commande

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

* Veuillez noter que les caractères minuscules (`raptorxmlxbrl`) fonctionnent sur toutes les plates-formes (Windows, Linux et Mac), alors qu'une écriture en majuscule/minuscule (`RaptorXMLXBRL`) ne fonctionne que sur Windows et Mac.
* Utiliser les barres obliques penchées vers l'avant pour Linux et Mac, et les barres obliques penchées vers l'arrière sur Windows.

Options

Les options de la commande sont listées ci-dessous, et organisées en groupes. Les valeurs peuvent être spécifiées sans guillemets sauf dans deux cas : (i) lorsque le string de valeur contient des espaces, ou (ii) si la présence de guillemets est explicitement requise dans la description de l'option.

Validation et traitement

json5

--json5 = `true|false`

Active la prise en charge JSON5. La valeur par défaut est `false`.
Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- listfile

  `--listfile =` `true|false`
  Si `true`, traiter l'argument *`InputFile`* de la commande en tant que fichier de texte contenant un nom de fichier par ligne. La valeur par défaut est `false`. (Une alternative est de lister les fichiers sur la CLI avec un espace en tant que séparateur. Veuillez noter, néanmoins, que les CLI ont une limitation de caractères maximum.) Veuillez noter que l'option `--listfile` s'applique uniquement aux arguments, et non pas aux options.
  Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- recurse

  `--recurse =` `true|false`
  Utilisé pour sélectionner des fichiers dans le cadre de sous-répertoires, y compris des archives ZIP. Si `true`, l'argument *`InputFile`* de la commande sélectionnera aussi le fichier spécifié dans les sous-répertoires. Par exemple : `test.zip|zip\test.xml` sélectionnera des fichiers nommés `test.xml` à tous les niveaux de dossier du dossier zip. Les caractères génériques `*` et `?` peuvent être utilisés. Donc, `*.xml` sélectionnera tous les fichiers `.xml` dans le dossier (zip). La valeur par défaut de l'option est `false`.
  Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- Catalogues et ressources globales

  - catalog

    `--catalog =` *`FILE`*
    Spécifie le chemin absolu vers un fichier de catalogue root qui n'est pas le fichier catalogue root installé. La valeur par défaut est le chemin absolu vers le fichier de catalogue root installé (`<installation-folder>\Altova\RaptorXMLXBRLServer2018\etc\RootCatalog.xml`). Voir la section, Catalogues XML, pour plus d'informations à propos du travail avec les catalogues.

  - user-catalog

    `--user-catalog =` *`FILE`*
    Spécifie le chemin absolu vers un catalogue XML à utiliser en plus du catalogue root. Voir la section, Catalogues XML, pour plus d'informations concernant le travail avec des catalogues.

  - enable-globalresources

    `--enable-globalresources =` `true|false`
    Active les ressources globales. La valeur par défaut est `false`.
    Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- globalresourceconfig [gc]

  --gc | --globalresourceconfig = *VALUE*
  Spécifie la configuration active de la ressource globale (et active les ressources globales).

- globalresourcefile [gr]

  --gr | --globalresourcefile = *FILE*
  Spécifie le fichier de ressource globale (et active les ressources globales).

- Messages, erreurs, aide, timeout, version
  - error-format

    --error-format = `text|shortxml|longxml`
    Spécifie le format de la sortie d'erreur. La valeur par défaut est `text`. Les autres options génèrent des formats XML, avec `longxml` générant plus de détails.

  - error-limit

    --error-limit = *N*
    Spécifie la limite d'erreur avec une plage de valeur de `1` à `999`. La valeur par défaut est `100`. Le traitement s'arrête lorsque la limite d'erreur est atteinte. Utile pour limiter l'utilisation du processeur pendant la validation/transformation.

  - help

    --help
    Affiche le texte d'aide pour la commande. Par exemple, `valany --h`. (en alternative, la commande `help` peut être utilisée avec un argument. Par exemple : `help valany`.)

  - log-output

    --log-output = *FILE*
    Écrit la sortie de log dans l'URL de fichier spécifié. Veuillez vous assurer que la CLI a une permission d'écriture dans l'emplacement de la sortie.

  - network-timeout

    --network-timeout = *VALUE*
    Spécifie le timeout en secondes pour les opérations I/O à distance. La valeur par défaut est : `40`.

  - verbose

    --verbose = `true|false`
    Une valeur de `true` permet la sortie d'informations supplémentaires pendant la validation. La valeur par défaut est `false`.
    Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

  - verbose-output

    --verbose-output = *FILE*

Écrit la sortie verbeuse sur *FILE*.

- version

  --version
  Affiche la version de RaptorXML+XBRL Server. Si utilisé avec une commande, placer `--version` avant la commande.

- warning-limit

  --warning-limit = *VALUE*
  Spécifie la limite d'avertissement dans la plage `1-65535`. Le traitement se poursuit si cette limite a été atteinte, mais d'autres avertissements ne sont pas rapportés. La valeur par défaut est `100`.

3.7 Commandes Signature XML

Les commandes Signature XML peuvent être utilisées pour signer un document XML et pour vérifier un document signé. Ces commandes sont listées ci-dessous et sont décrites en détails dans les sous-sections de cette section :

xmlsignature-sign	Crée un document de sortie de signature XML à partir d'un document d'entrée
xmlsignature-verify	Vérifie un document de signature XML
xmlsignature-update	Met à jour la signature d'un document XML (modifié)
xmlsignature-remove	Supprime la signature d'un document XML

 © 2018 Altova GmbH

xmlsignature-sign

La commande **xmlsignature-sign | xsign** prend un document XML en tant qu'entrée et crée un document de sortie à signature XML utilisant les options de signature spécifiées.

Windows
```
RaptorXMLXBRL xmlsignature-sign [options] --output=File --
signature-type=Value
--signature-canonicalization-method=Value InputFile
```

Linux
```
raptorxmlxbrl xmlsignature-sign [options] --output=File --
signature-type=Value
--signature-canonicalization-method=Value InputFile
```

Mac
```
raptorxmlxbrl xmlsignature-sign [options] --output=File --
signature-type=Value
--signature-canonicalization-method=Value InputFile
```

L'argument *InputFile* est le document XML à signer. L'option **--output** spécifie l'emplacement du document qui contient la signature XML.

Exemples

- **raptorxmlxbrl** xsign --output=c:\SignedFile.xml --signature-type=enveloped --signature-canonicalization-method=xml-c14n11 c:\SomeUnsigned.xml

Casse et lignes obliques sur la ligne de commande

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

* Veuillez noter que les caractères minuscules (raptorxmlxbrl) fonctionnent sur toutes les plates-formes (Windows, Linux et Mac), alors qu'une écriture en majuscule/minuscule (RaptorXMLXBRL) ne fonctionne que sur Windows et Mac.
* Utiliser les barres obliques penchées vers l'avant pour Linux et Mac, et les barres obliques penchées vers l'arrière sur Windows.

Options

Les options de la commande sont listées ci-dessous, et organisées en groupes. Les valeurs peuvent être spécifiées sans guillemets sauf dans deux cas : (i) lorsque le string de valeur contient des espaces, ou (ii) si la présence de guillemets est explicitement requise dans la description de l'option.

Options communes

output

output = *FILE*

L'URL du document de sortie qui est créé avec la nouvelle signature XML.

- verbose

  --verbose = `true|false`
  Une valeur de `true` permet la sortie d'informations supplémentaires pendant la validation. La valeur par défaut est `false`.
  Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- Options de Signature XML
  - absolute-reference-uri

    --absolute-reference-uri = `true|false`
    Spécifie si l'URI du document signé doit être lu en tant qu'absolu (`true`) ou relatif (`false`). La valeur par défaut est `false`.
    Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

  - certname, certificate-name

    --certname, --certificate-name = *VALUE*
    Le nom du certificat utilisé pour signer.

    Windows
    Il s'agit du nom du **Sujet** d'un certificat depuis le **--certificate-store** sélectionné.

    Exemple de liste des certificats (sous PowerShell)

```
% ls cert://CurrentUser/My
PSParentPath: Microsoft.PowerShell.Security\Certificate::CurrentUser\My
Thumbprint Subject
---------- -------
C9DF64BB0AAF5FA73474D78B7CCFFC37C95BFC6C CN=certificate1
... CN=...
```

    Exemple : **--certificate-name==certificate1**

    Linux/MacOS
    --certname spécifie le nom de fichier d'un certificat X.509v3 à encodage PEM avec la clé privée. Ces fichiers portent généralement l'extension **.pem**.

    Exemple : **--certificate-name==/path/to/certificate1.pem**

  - certstore, certificate-store

    --certstore, --certificate-store = *VALUE*

L'emplacement où le certificat spécifié par **--certificate-name** est stocké.

Windows
Le nom d'un magasin de certificat sous **cert://CurrentUser**. Les magasins de certificat disponibles peuvent être listés (sous PowerShell) à l'aide de **% ls cert://CurrentUser/**. Les certificats seraient donc listés comme suit :

```
Name : TrustedPublisher
Name : ClientAuthIssuer
Name : Root
Name : UserDS
Name : CA
Name : ACRS
Name : REQUEST
Name : AuthRoot
Name : MSIEHistoryJournal
Name : TrustedPeople
Name : MyCertStore
Name : Local NonRemovable Certificates
Name : SmartCardRoot
Name : Trust
Name : Disallowed
```

Exemple : **--certificate-store==MyCertStore**

Linux/MacOS
L'option **--certstore** n'est actuellement pas prise en charge.

- digest, digest-method

  --digest, --digest-method = sha1|sha256|sha384|sha512|base64
  L'algorithme utilisé pour calculer la valeur de digest par rapport au fichier XML d'entrée.

- hmackey, hmac-secret-key

  --hmackey, --hmac-secret-key = *VALUE*
  La clé secrète partagée HMAC ; doit avoir une longueur minimum de six caractères.

  Exemple : **--hmackey=secretpassword**

- hmaclen, hmac-output-length

  --hmaclen, --hmac-output-length = *LENGTH*
  Tronque la sortie de l'algorithme HMAC à des bits LENGTH. Si spécifiée, cette valeur doit être
  - un multiple de 8
  - supérieur à 80
  - supérieur à la moitié de la longueur de sortie de l'algorithme hash sous-jacent

- keyinfo, append-keyinfo

  --keyinfo, --append-keyinfo = `true|false`
  Spécifie si vous souhaitez inclure l'élément **KeyInfo** dans la signature ou pas. La valeur par défaut est **false**.

- sigc14nmeth, signature-canonicalization-method

  --sigc14nmeth, --signature-canonicalization-method = *VALUE*
  Spécifie l'algorithme de canonicalisation à appliquer à l'élément **SignedInfo**. La valeur doit être une des suivantes :

  - `REC-xml-c14n-20010315`
  - `xml-c14n11`
  - `xml-exc-c14n#`

- sigmeth, signature-method

  --sigmeth, --signature-method = *VALUE*
  Spécifie l'algorithme à utiliser pour générer la signature.

  Lorsqu'un certificat est utilisé
  Si un certificat est spécifié, alors `--signature-method` est optionnel et la valeur pour ce paramètre est dérivé du certificat. Lorsqu'il est spécifié, il doit correspondre à l'algorithme utilisé par le certificat.

  Exemple : **--signature-method=rsa-sha256**

  Lorsque --hmac-secret-key est utilisé
  Lorsque `--hmac-secret-key` est utilisé, cette option est obligatoire. La valeur doit être un des algorithmes HMAC pris en charge :
  - `hmac-sha256`
  - `hmac-sha386`
  - `hmac-sha512`
  - hmac-sha1 (découragé par la spécification)

  Exemple : **--signature-method=hmac-sha256**

- sigtype, signature-type

  --sigtype, --signature-type = `detached | enveloping | enveloped`
  Spécifie le type de signature à générer.

- transforms

  --transforms = *VALUE*
  Spécifie les transformations de Signature XML appliquées au document d'entrée. Cette option peut être spécifiée plusieurs fois. Dans ce cas, l'ordre de la spécification est important. La première transformation spécifiée reçoit le document d'entrée. La dernière transformation spécifiée est utilisée immédiatement avant le calcul de la valeur digest.

 © 2018 Altova GmbH

Les valeurs prises en charge sont :

- `REC-xml-c14n-20010315` pour Canonical XML 1.0 (sans commentaires)
- `xml-c14n11` pour Canonical XML 1.1 (sans comments)
- `xml-exc-c14n#` pour Exclusive XML Canonicalization 1.0 (sans commentaires)
- `REC-xml-c14n-20010315#WithComments` pour Canonical XML 1.0 (avec commentaires)
- `xml-c14n11#WithComments` pour Canonical XML 1.1 (with comments)
- `xml-exc-c14n#WithComments` pour Exclusive XML Canonicalization 1.0 (avec commentaires)
- `base64`
- `strip-whitespaces` extension Altova

Exemple : **`--transforms=xml-c14n11`**

- write-default-attributes

  `--write-default-attributes =` `true|false`
  Spécifie s'il faut inclure les valeur d'attribut par défaut depuis le DTD dans le document signé.

- Options d'aide et de version
  - help

    `--help`
    Affiche le texte d'aide pour la commande. Par exemple, `valany --h`. (en alternative, la commande `help` peut être utilisée avec un argument. Par exemple : `help valany`.)

  - version

    `--version`
    Affiche la version de RaptorXML+XBRL Server. Si utilisé avec une commande, placer `--version` avant la commande.

xmlsignature-verify

La commande **xmlsignature-verify | xverify** vérifie la signature XML du fichier d'entrée. Si la vérification est réussie un message de résultat **result="OK"** est affiché ; sinon, le message **result="Failed"** s'affiche.

Windows	**RaptorXMLXBRL xmlsignature-verify [options] *InputFile***
Linux	**raptorxmlxbrl xmlsignature-verify [options] *InputFile***
Mac	**raptorxmlxbrl xmlsignature-verify [options] *InputFile***

L'argument *InputFile* est le document XML signé à vérifier.

Exemples

- **raptorxmlxbrl** xverify c:\SignedFile.xml

Casse et lignes obliques sur la ligne de commande

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

* Veuillez noter que les caractères minuscules (raptorxmlxbrl) fonctionnent sur toutes les plates-formes (Windows, Linux et Mac), alors qu'une écriture en majuscule/minuscule (RaptorXMLXBRL) ne fonctionne que sur Windows et Mac.
* Utiliser les barres obliques penchées vers l'avant pour Linux et Mac, et les barres obliques penchées vers l'arrière sur Windows.

Options

Les options de la commande sont listées ci-dessous, et organisées en groupes. Les valeurs peuvent être spécifiées sans guillemets sauf dans deux cas : (i) lorsque le string de valeur contient des espaces, ou (ii) si la présence de guillemets est explicitement requise dans la description de l'option.

Options communes

verbose

--verbose = true|false
Une valeur de true permet la sortie d'informations supplémentaires pendant la validation. La valeur par défaut est false.
Note : les valeurs d'option booléennes sont configurées sur true si l'option est spécifiée sans une valeur.

Options de Signature XML

- certname, certificate-name

--certname, --certificate-name = *VALUE*
Le nom du certificat utilisé pour signer.

Windows
Il s'agit du nom du **Sujet** d'un certificat depuis le **--certificate-store** sélectionné.

Exemple de liste des certificats (sous PowerShell)

```
% ls cert://CurrentUser/My
PSParentPath: Microsoft.PowerShell.Security\Certificate::CurrentUser\My
Thumbprint Subject
---------- -------
C9DF64BB0AAF5FA73474D78B7CCFFC37C95BFC6C CN=certificate1
... CN=...
```

Exemple : **--certificate-name==certificate1**

Linux/MacOS
--certname spécifie le nom de fichier d'un certificat X.509v3 à encodage PEM avec la clé privée. Ces fichiers portent généralement l'extension **.pem**.

Exemple : **--certificate-name==/path/to/certificate1.pem**

- certstore, certificate-store

--certstore, --certificate-store = *VALUE*
L'emplacement où le certificat spécifié par **--certificate-name** est stocké.

Windows
Le nom d'un magasin de certificat sous **cert://CurrentUser**. Les magasins de certificat disponibles peuvent être listés (sous PowerShell) à l'aide de **% ls cert://CurrentUser/**. Les certificats seraient donc listés comme suit :

```
Name : TrustedPublisher
Name : ClientAuthIssuer
Name : Root
Name : UserDS
Name : CA
Name : ACRS
Name : REQUEST
Name : AuthRoot
Name : MSIEHistoryJournal
Name : TrustedPeople
Name : MyCertStore
Name : Local NonRemovable Certificates
Name : SmartCardRoot
```

```
Name : Trust
Name : Disallowed
```

Exemple : **`--certificate-store==MyCertStore`**

Linux/MacOS
L'option **`--certstore`** n'est actuellement pas prise en charge.

- hmackey, hmac-secret-key

  `--hmackey, --hmac-secret-key =` *`VALUE`*
  La clé secrète partagée HMAC ; doit avoir une longueur minimum de six caractères.

  Exemple : **`--hmackey=secretpassword`**

- Options d'aide et de version
  - help

    `--help`
    Affiche le texte d'aide pour la commande. Par exemple, `valany --h`. (en alternative, la commande `help` peut être utilisée avec un argument. Par exemple : `help valany`.)

  - version

    `--version`
    Affiche la version de RaptorXML+XBRL Server. Si utilisé avec une commande, placer `--version` avant la commande.

 © 2018 Altova GmbH

xmlsignature-update

La commande **xmlsignature-update | xupdate** met à jour la signature XML dans le fichier d'entrée signé. Si le document a été modifié, la signature XML sera différente ; sinon, la signature mise à jour sera la même que celle de la signature précédente.

Windows **RaptorXMLXBRL xmlsignature-update [options] --output=File *SignedFile***

Linux **raptorxmlxbrl xmlsignature-update [options] --output=File *SignedFile***

Mac **raptorxmlxbrl xmlsignature-update [options] --output=File *SignedFile***

L'argument *SignedFile* est le document XML signé à mettre à jour. Il faut spécifier soit (i) l'option hmac-secret-key soit (ii) les options certificate-name et certificate-store. Si les options certificate-name et certificate-store sont spécifiées, elles doivent correspondre à celles qui ont été utilisées pour signer le document XML précédemment. (Veuillez noter que l'option certificate-store n'est actuellement pas pris en charge sur Linux et macOS.)

Exemples

- **raptorxmlxbrl** xupdate --output=c:\UpdatedSignedFile.xml --certname=certificate1 --certstore=MyCertStore c:\SomeSignedFile.xml
- **raptorxmlxbrl** xupdate --output=c:\UpdatedSignedFile.xml --hmackey=SecretPassword c:\SomeSignedFile.xml

Casse et lignes obliques sur la ligne de commande

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

* Veuillez noter que les caractères minuscules (raptorxmlxbrl) fonctionnent sur toutes les plates-formes (Windows, Linux et Mac), alors qu'une écriture en majuscule/minuscule (RaptorXMLXBRL) ne fonctionne que sur Windows et Mac.
* Utiliser les barres obliques penchées vers l'avant pour Linux et Mac, et les barres obliques penchées vers l'arrière sur Windows.

Options

Les options de la commande sont listées ci-dessous, et organisées en groupes. Les valeurs peuvent être spécifiées sans guillemets sauf dans deux cas : (i) lorsque le string de valeur contient des espaces, ou (ii) si la présence de guillemets est explicitement requise dans la description de l'option.

Options communes

- output

  output = *FILE*
  L'URL du document de sortie qui est créé avec la nouvelle signature XML.

- verbose

  --verbose = `true|false`
  Une valeur de `true` permet la sortie d'informations supplémentaires pendant la validation. La valeur par défaut est `false`.
  Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- Options de Signature XML
  - certname, certificate-name

    --certname, --certificate-name = *VALUE*
    Le nom du certificat utilisé pour signer.

    Windows
    Il s'agit du nom du **Sujet** d'un certificat depuis le **--certificate-store** sélectionné.

    Exemple de liste des certificats (sous PowerShell)

    ```
    % ls cert://CurrentUser/My
    PSParentPath: Microsoft.PowerShell.Security\Certificate::CurrentUser\My
    Thumbprint Subject
    ---------- -------
    C9DF64BB0AAF5FA73474D78B7CCFFC37C95BFC6C CN=certificate1
    ... CN=...
    ```

    Exemple : **--certificate-name==certificate1**

    Linux/MacOS
    --certname spécifie le nom de fichier d'un certificat X.509v3 à encodage PEM avec la clé privée. Ces fichiers portent généralement l'extension **.pem**.

    Exemple : **--certificate-name==/path/to/certificate1.pem**

  - certstore, certificate-store

    --certstore, --certificate-store = *VALUE*
    L'emplacement où le certificat spécifié par **--certificate-name** est stocké.

    Windows
    Le nom d'un magasin de certificat sous **cert://CurrentUser**. Les magasins de

certificat disponibles peuvent être listés (sous PowerShell) à l'aide de **`% ls cert:// CurrentUser/`**. Les certificats seraient donc listés comme suit :

```
Name : TrustedPublisher
Name : ClientAuthIssuer
Name : Root
Name : UserDS
Name : CA
Name : ACRS
Name : REQUEST
Name : AuthRoot
Name : MSIEHistoryJournal
Name : TrustedPeople
Name : MyCertStore
Name : Local NonRemovable Certificates
Name : SmartCardRoot
Name : Trust
Name : Disallowed
```

Exemple : **`--certificate-store==MyCertStore`**

Linux/MacOS
L'option **`--certstore`** n'est actuellement pas prise en charge.

- hmackey, hmac-secret-key

  `--hmackey, --hmac-secret-key =` *VALUE*
  La clé secrète partagée HMAC ; doit avoir une longueur minimum de six caractères.

  Exemple : **`--hmackey=secretpassword`**

- Options d'aide et de version
  - help

    `--help`
    Affiche le texte d'aide pour la commande. Par exemple, `valany --h`. (en alternative, la commande `help` peut être utilisée avec un argument. Par exemple : `help valany`.)

  - version

    `--version`
    Affiche la version de RaptorXML+XBRL Server. Si utilisé avec une commande, placer `--version` avant la commande.

xmlsignature-remove

La commande **xmlsignature-remove | xremove** supprime la signature XML dans le fichier d'entrée signé et enregistre le document résultant non signé dans un emplacement de sortie que vous spécifiez.

Windows **RaptorXMLXBRL xmlsignature-remove [options] --output=File *SignedFile***

Linux **raptorxmlxbrl xmlsignature-remove [options] --output=File *SignedFile***

Mac **raptorxmlxbrl xmlsignature-remove [options] --output=File *SignedFile***

L'argument *SignedFile* est le document XML signé à partir duquel vous souhaitez supprimer la signature XML. L'option `--output` spécifie l'emplacement du document XML non-signé qui est généré.

Exemples

- **raptorxmlxbrl** `xremove --output=c:\UnsignedFile.xml c:\SignedFile.xml`

Casse et lignes obliques sur la ligne de commande

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

* Veuillez noter que les caractères minuscules (`raptorxmlxbrl`) fonctionnent sur toutes les plates-formes (Windows, Linux et Mac), alors qu'une écriture en majuscule/minuscule (`RaptorXMLXBRL`) ne fonctionne que sur Windows et Mac.
* Utiliser les barres obliques penchées vers l'avant pour Linux et Mac, et les barres obliques penchées vers l'arrière sur Windows.

Options

Les options de la commande sont listées ci-dessous, et organisées en groupes. Les valeurs peuvent être spécifiées sans guillemets sauf dans deux cas : (i) lorsque le string de valeur contient des espaces, ou (ii) si la présence de guillemets est explicitement requise dans la description de l'option.

Options communes

output

output = *FILE*
L'URL du document de sortie qui est créé avec la signature XML supprimée.

 © 2018 Altova GmbH

- verbose

  `--verbose =` `true|false`
  Une valeur de `true` permet la sortie d'informations supplémentaires pendant la validation. La valeur par défaut est `false`.
  Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- Options d'aide et de version
  - help

    `--help`
    Affiche le texte d'aide pour la commande. Par exemple, `valany --h`. (en alternative, la commande `help` peut être utilisée avec un argument. Par exemple : `help valany`.)

  - version

    `--version`
    Affiche la version de RaptorXML+XBRL Server. Si utilisé avec une commande, placer `--version` avant la commande.

3.8 Commande Valany

La commande **valany** est une commande générale qui valide un document sur la base du type de document dont il s'agit. Le type du document d'entrée est détecté automatiquement et la validation correspondante est effectuée conformément à la spécification respective. L'argument *InputFile* spécifie le document à valider. Veuillez noter qu'un seul document peut être soumis en tant que l'argument de la commande.

Windows	**RaptorXMLXBRL valany [options] *InputFile***
Linux	**raptorxmlxbrl valany [options] *InputFile***
Mac	**raptorxmlxbrl valany [options] *InputFile***

La commande **valany** couvre les types de validation suivants. Ces options sont celles qui sont disponibles pour la commande de validation individuelle correspondante. Voir la description des commandes de validation respectives pour une liste de leurs options respectives.

- valdtd (dtd)
- valxsd (xsd)
- valxml-withdtd (xml)
- valxml-withxsd (xsi)
- valxslt
- valxquery
- valxbrl (xbrl)
- valinlinexbrl (ixbrl)
- valxbrltaxonomy (dts)
- valavrojson (avrojson)

Exemples

- **raptorxmlxbrl** valany c:\Test.xsd
- **raptorxmlxbrl** valany --error-format=text c:\Test.xbrl

Casse et lignes obliques sur la ligne de commande

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

* Veuillez noter que les caractères minuscules (raptorxmlxbrl) fonctionnent sur toutes les plates-formes (Windows, Linux et Mac), alors qu'une écriture en majuscule/minuscule (RaptorXMLXBRL) ne fonctionne que sur Windows et Mac.
* Utiliser les barres obliques penchées vers l'avant pour Linux et Mac, et les barres obliques penchées vers l'arrière sur Windows.

 © 2018 Altova GmbH

Options

Voir la description des commandes de validation respective pour une liste de leurs options respectives. Veuillez noter, néanmoins qu'alors que la plupart des commandes de validation individuelles acceptent des documents d'entrée multiples, la commande **`valany`** n'accepte qu'un seul document d'entrée. C'est pourquoi les options de type `--listfile` ne s'appliqueront pas à **`valany`**.

3.9 Commande Script

La commande **`script`** exécute un script Python 3 qui utilise l'API Python RaptorXML.

Windows	**`RaptorXMLXBRL script [options] File`**
Linux	**`raptorxmlxbrl script [options] File`**
Mac	**`raptorxmlxbrl script [options] File`**

L'argument *`File`* est le chemin vers le script Python que vous souhaitez exécuter. Des options supplémentaires sont disponibles. Pour obtenir une liste de ces options, exécutez la commande suivante :

Windows	**`RaptorXMLXBRL script [-h | --help]`**
Linux	**`raptorxmlxbrl script [-h | --help]`**
Mac	**`raptorxmlxbrl script [-h | --help]`**

Exemples

- **`raptorxmlxbrl`** `script c:\MyPythonScript.py`
- **`raptorxmlxbrl`** `script -h`
- **`raptorxmlxbrl`** `script` # *Sans fichier script, un shell Python interactif sera démarré*
- **`raptorxmlxbrl`** `script -m pip` # *Charge et exécute le module* `pip` *; voir la section Options ci-dessous*

Casse et lignes obliques sur la ligne de commande

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

* Veuillez noter que les caractères minuscules (`raptorxmlxbrl`) fonctionnent sur toutes les plates-formes (Windows, Linux et Mac), alors qu'une écriture en majuscule/minuscule (`RaptorXMLXBRL`) ne fonctionne que sur Windows et Mac.
* Utiliser les barres obliques penchées vers l'avant pour Linux et Mac, et les barres obliques penchées vers l'arrière sur Windows.

Options

Toutes les options et arguments se trouvant après la commande `script` seront envoyés directement à l'interprète Python. Veuillez consulter la page de documentation Python https://docs.python.org/3/using/cmdline.html pour une liste complète des options disponibles.

 © 2018 Altova GmbH

3.10 Commandes Aide et Licence

Cette section décrit deux fonctions importantes de RaptorXML+XBRL Server:

- Commande Aide : Décrit comment afficher les informations concernant les commandes disponibles ou concernant les arguments et options d'une commande
- Licence : Décrit comment mettre RaptorXML sous licence

help

La commande **`help`** prend un seul argument : le nom de la commande pour laquelle l'aide est requise. Elle affiche la syntaxe de la commande et d'autres informations pertinentes à l'exécution correcte de la commande.

Windows **`RaptorXMLXBRL help`** ***`Command`***

Linux **`raptorxmlxbrl help`** ***`Command`***

Mac **`raptorxmlxbrl help`** ***`Command`***

Note : Si aucun argument n'est soumis, l'exécution de la commande `help` entraîne l'affichage de toutes les commandes disponibles, chacune avec une brève description.

Exemple

Exemple de la commande `help` :

```
raptorxmlxbrl help valany
```

La commande ci-dessus contient un argument : la commande `valany`, pour laquelle l'aide es requise. Lorsque cette commande est exécutée, elle affichera l'information d'aide concernant la commande `valany`.

Casse et lignes obliques sur la ligne de commande

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

* Veuillez noter que les caractères minuscules (`raptorxmlxbrl`) fonctionnent sur toutes les plates-formes (Windows, Linux et Mac), alors qu'une écriture en majuscule/minuscule (`RaptorXMLXBRL`) ne fonctionne que sur Windows et Mac.
* Utiliser les barres obliques penchées vers l'avant pour Linux et Mac, et les barres obliques penchées vers l'arrière sur Windows.

L'option `--help`

L'information Aide concernant une commande est aussi disponible en utilisant l'option `--help` avec cette commande. Par exemple, en utilisant l'option `--help` avec la commande `valany`, comme suit :

```
raptorxmlxbrl valany --help
```

obtient le même résultat qu'en utilisant la commande `help` avec un argument de `valany` :

```
raptorxmlxbrl help valany
```

Dans les deux cas, l'information d'aide concernant la commande `valany` est affichée.

 © 2018 Altova GmbH

Casse et lignes obliques sur la ligne de commande

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

* Veuillez noter que les caractères minuscules (`raptorxmlxbrl`) fonctionnent sur toutes les plates-formes (Windows, Linux et Mac), alors qu'une écriture en majuscule/minuscule (`RaptorXMLXBRL`) ne fonctionne que sur Windows et Mac.
* Utiliser les barres obliques penchées vers l'avant pour Linux et Mac, et les barres obliques penchées vers l'arrière sur Windows.

licenseserver

La commande `licenseserver` enregistre RaptorXML+XBRL Server auprès de l'Altova LicenseServer. Elle prend en tant que son argument le nom ou l'adresse IP du serveur exécutant le LicenseServer.

Windows **RaptorXMLXBRL licenseserver [options]** ***Server-Or-IP-Address***

Linux **raptorxmlxbrl licenseserver [options]** ***Server-Or-IP-Address***

Mac **raptorxmlxbrl licenseserver [options]** ***Server-Or-IP-Address***

En cas d'enregistrement réussi de RaptorXML+XBRL Server avec LicenseServer, l'URL de l'interface web de LicenseServer sera retournée. Saisir l'URL dans une fenêtre de navigateur pour accéder à l'interface web, puis passer par les étapes de licence telles que décrites dans la documentation LicenseServer.

Exemple

Voici un exemple de la commande `licenseserver` :

```
raptorxmlxbrl licenseserver DOC.altova.com
```

La commande spécifie que la machine nommée `DOC.altova.com` est la machine exécutant l'Altova LicenseServer.

▾ Casse et lignes obliques sur la ligne de commande

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

* Veuillez noter que les caractères minuscules (`raptorxmlxbrl`) fonctionnent sur toutes les plates-formes (Windows, Linux et Mac), alors qu'une écriture en majuscule/minuscule (`RaptorXMLXBRL`) ne fonctionne que sur Windows et Mac.
* Utiliser les barres obliques penchées vers l'avant pour Linux et Mac, et les barres obliques penchées vers l'arrière sur Windows.

Options

Les options suivantes sont disponibles :

--j|json=true|false
Imprime le résultat des tentatives d'enregistrement en tant qu'objet JSON parsable par machine.

--h|help
Affiche le texte d'aide de la commande.

 © 2018 Altova GmbH

--version

Affiche le numéro de version de RaptorXML+XBRL Server. L'option doit être placée avant la commande. Donc : `raptorxmlxbrl --version licenseserver`.

assignlicense

La commande `assignlicense` est uniquement disponible sur Windows. Elle charge un fichier de licence sur l'Altova LicenseServer sur lequel RaptorXML+XBRL Server est enregistré (voir la commande `licenseserver`), mise sous licence et attribue la licence à RaptorXML+XBRL Server. Elle prend l'URL d'un fichier de licence en tant que son argument. La commande vous permet également de tester la validité d'une licence.

Windows **RaptorXMLXBRL assignlicense [options]** ***LICENSE-FILE***

Linux ***not applicable***

Mac ***not applicable***

Exemple

- **raptorxmlxbrl** assignlicense C:\licensepool\mylicensekey.lic
- **raptorxmlxbrl** assignlicense --test-only=true C:\licensepool \mylicensekey.lic

Casse et lignes obliques sur la ligne de commande

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

* Veuillez noter que les caractères minuscules (`raptorxmlxbrl`) fonctionnent sur toutes les plates-formes (Windows, Linux et Mac), alors qu'une écriture en majuscule/minuscule (`RaptorXMLXBRL`) ne fonctionne que sur Windows et Mac.
* Utiliser les barres obliques penchées vers l'avant pour Linux et Mac, et les barres obliques penchées vers l'arrière sur Windows.

Options

Les options suivantes sont disponibles :

--t|test-only=true|false
Si la valeur est `true`, alors la licence est chargée sur LicenseServer et validée, mais pas attribuée.

Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

--h|help
Affiche le texte d'aide de la commande.

--version
Affiche le numéro de version de RaptorXML+XBRL Server. L'option doit être placée avant la

commande. Donc : `raptorxmlxbrl --version licenseserver`.

verifylicense

La commande `verifylicense` est disponible uniquement sur Windows. Elle contrôle si RaptorXML+XBRL Server a une licence valide et en option, vérifie si une clé de licence donnée a déjà été attribuée à RaptorXML+XBRL Server. Elle prend la clé de licence en tant qu'option. La commande retourne une instruction concernant le statut de la licence ou la validité de la clé de licence soumise.

Windows **RaptorXMLXBRL verifylicense [options]**

Linux ***not applicable***

Mac ***not applicable***

Exemple

- **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)

Les instructions de retour ressemblent à :

- La licence du produit est valide
- La licence du produit n'est pas valide
- La clé de licence AAAAAAA-BBBBBBB-CCCCCCC-DDDDDDD-EEEEEEE est attribuée au produit
- La clé de licence AAAAAAA-BBBBBBB-CCCCCCC-DDDDDDD-EEEEEEE n'est pas attribuée au produit

Casse et lignes obliques sur la ligne de commande

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

* Veuillez noter que les caractères minuscules (`raptorxmlxbrl`) fonctionnent sur toutes les plates-formes (Windows, Linux et Mac), alors qu'une écriture en majuscule/minuscule (`RaptorXMLXBRL`) ne fonctionne que sur Windows et Mac.
* Utiliser les barres obliques penchées vers l'avant pour Linux et Mac, et les barres obliques penchées vers l'arrière sur Windows.

Options

Les options suivantes sont disponibles :

--l|license-key=*VALUE*
`VALUE` est la clé de licence de 39 caractères de long sans aucun délimiteur. La clé consiste en

cinq blocs de sept caractères séparés par des virgules.

--h|help
Affiche le texte d'aide de la commande.

--version
Affiche le numéro de version de RaptorXML+XBRL Server. L'option doit être placée avant la commande. Donc: `raptorxmlxbrl --version licenseserver`.

3.11 Commandes de localisation

Vous pouvez créer une version localisée de l'application RaptorXML pour la langue de votre choix. Cinq versions localisées (anglais, allemand, espagnol, français et japonais) sont disponibles dans le dossier `<ProgramFilesFolder>\Altova\RaptorXMLXBRLServer2018\bin\` . Ces cinq versions ne devront donc pas être créées.

Pour créer une version localisée dans une autre langue :

1. Générer un fichier XML contenant les strings de ressource. Pour ce faire, utiliser la commande `exportresourcestrings`. Les strings de ressource dans le fichier XML généré seront dans une des cinq langues : l'anglais (`en`), l'allemand (`de`), l'espagnol (`es`), le français (`fr`) ou le japonais (`ja`), conformément à l'argument utilisé avec la commande.
2. Traduire les strings de ressource depuis la langue du fichier XML généré dans la langue cible. Les strings de ressource sont les contenus des éléments `<string>` dans le fichier XML. Ne pas traduire les variables dans les accolades, comme `{option}` ou `{product}`.
3. Contacter l'Assistance Altova pour générer un fichier DLL RaptorXML localisé depuis votre fichier XML traduit.
4. Une fois avoir reçu votre fichier DLL localisé de la part de l'Assistance Altova, enregistrer la DLL dans le dossier `<ProgramFilesFolder>\Altova\RaptorXMLXBRLServer2018\bin\` . Votre fichier DLL aura un nom de la forme `RaptorXMLXBRLServer_lc.dll`. La partie `_lc` du nom contient le code de langage. Par exemple, dans `RaptorXMLXBRLServer_de.dll`, la partie `de` est la langue de code pour l'allemand (Deutsch).
5. Exécuter la commande `setdeflang` pour définir votre fichier DLL localisé en tant que l'application RaptorXML à utiliser. En ce qui concerne l'argument de la commande `setdeflang`, utiliser le code de langage qui fait partie du nom de DLL.

Note : Altova RaptorXML+XBRL Server est fourni avec une prise en charge de cinq langues : anglais, allemand, espagnol, français et japonais. Donc vous n'aurez pas besoin de créer une version localisée pour ces langues-là. Pour définir une des ces cinq langues en tant que la langue de défaut, utiliser la commande de CLI `setdeflang`.

 © 2018 Altova GmbH

exportresourcestrings

La commande **exportresourcestrings** sort un fichier XML contenant les strings de ressource RaptorXML. La commande prend deux arguments : (i) la langue des strings de ressource dans le fichier XML de sortie, et (ii) le chemin et le nom du fichier XML de sortie. Les langues d'exportation autorisées (avec leurs codes de langue en parenthèses) sont : l'anglais (en), l'allemand (de), l'espagnol (es), le français (fr) ou le japonais (ja).

Windows **RaptorXMLXBRL exportresourcestrings *LanguageCode XMLOutputFile***

Linux **raptorxmlxbrl exportresourcestrings *LanguageCode XMLOutputFile***

Mac **raptorxmlxbrl exportresourcestrings *LanguageCode XMLOutputFile***

Arguments

La commande exportresourcestrings prend les arguments suivants :

LanguageCode	Spécifie la langue cible de l'exportation, c'est à dire, la langue des strings de ressource dans le fichier XML exporté. Les langues prises en charge sont : en, de, es, fr,ja
XMLOutputFile	Spécifie l'emplacement et le nom du fichier XML exporté.

Exemple

Cette commande crée un fichier appelé Strings.xml sous c:\ qui contient tous les strings de ressource de l'application RaptorXML traduite en allemand.

```
raptorxmlxbrl exportresourcestrings de c:\Strings.xml
```

Casse et lignes obliques sur la ligne de commande

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

* Veuillez noter que les caractères minuscules (raptorxmlxbrl) fonctionnent sur toutes les plates-formes (Windows, Linux et Mac), alors qu'une écriture en majuscule/minuscule (RaptorXMLXBRL) ne fonctionne que sur Windows et Mac.
* Utiliser les barres obliques penchées vers l'avant pour Linux et Mac, et les barres obliques penchées vers l'arrière sur Windows.

setdeflang

La commande **setdeflang** (la forme abrégée est `sdl`) définit la langue par défaut de RaptorXML. Il nécessaire un argument obligatoire `LanguageCode`.

Windows **RaptorXMLXBRL setdeflang | sdl *LangaugeCode***

Linux **raptorxmlxbrl setdeflang | sdl *LangaugeCode***

Mac **raptorxmlxbrl setdeflang | sdl *LangaugeCode***

Exemple

Cette commande définit la langue par défaut des messages d'application en allemand.

```
raptorxmlxbrl setdeflang de
```

Casse et lignes obliques sur la ligne de commande

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

* Veuillez noter que les caractères minuscules (`raptorxmlxbrl`) fonctionnent sur toutes les plates-formes (Windows, Linux et Mac), alors qu'une écriture en majuscule/minuscule (`RaptorXMLXBRL`) ne fonctionne que sur Windows et Mac.
* Utiliser les barres obliques penchées vers l'avant pour Linux et Mac, et les barres obliques penchées vers l'arrière sur Windows.

Langues prises en charge

La table ci-dessous liste les langues actuellement prises en charge avec leurs codes de langue.

en	Anglais
de	Allemand
es	Espagnol
fr	Français
ja	Japonais

 © 2018 Altova GmbH

3.12 Options

Cette section contient une description de toutes les options de CLI, organisées par les fonctions. Pour découvrir quelles options peuvent être utilisées avec chaque commande, voir la description des commandes respectives.

- Catalogues, Ressources globales, Fichiers ZIP
- Messages, Erreurs, Aide
- Traitement
- XBRL
- XML
- XSD
- XQuery
- XSLT

Catalogues, Ressources globales, fichiers ZIP

- catalog

  `--catalog =` *`FILE`*
  Spécifie le chemin absolu vers un fichier de catalogue root qui n'est pas le fichier catalogue root installé. La valeur par défaut est le chemin absolu vers le fichier de catalogue root installé (`<installation-folder>\Altova\RaptorXMLXBRLServer2018\etc\RootCatalog.xml`). Voir la section, Catalogues XML, pour plus d'informations à propos du travail avec les catalogues.

- user-catalog

  `--user-catalog =` *`FILE`*
  Spécifie le chemin absolu vers un catalogue XML à utiliser en plus du catalogue root. Voir la section, Catalogues XML, pour plus d'informations concernant le travail avec des catalogues.

- enable-globalresources

  `--enable-globalresources =` `true|false`
  Active les ressources globales. La valeur par défaut est `false`.
  Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- globalresourceconfig [gc]

  `--gc | --globalresourceconfig =` *`VALUE`*
  Spécifie la configuration active de la ressource globale (et active les ressources globales).

- globalresourcefile [gr]

  `--gr | --globalresourcefile =` *`FILE`*
  Spécifie le fichier de ressource globale (et active les ressources globales).

- recurse

  `--recurse =` `true|false`
  Utilisé pour sélectionner des fichiers dans le cadre de sous-répertoires, y compris des archives ZIP. Si `true`, l'argument *`InputFile`* de la commande sélectionnera aussi le fichier spécifié dans les sous-répertoires. Par exemple : `test.zip|zip\test.xml` sélectionnera des fichiers nommés `test.xml` à tous les niveaux de dossier du dossier zip. Les caractères génériques `*` et `?` peuvent être utilisés. Donc, `*.xml` sélectionnera tous les fichiers `.xml` dans le dossier (zip). La valeur par défaut de l'option est `false`.
  Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

 © 2018 Altova GmbH

Messages, Erreurs, Aide, Timeout, Version

- error-format

  `--error-format =` `text|shortxml|longxml`
  Spécifie le format de la sortie d'erreur. La valeur par défaut est `text`. Les autres options génèrent des formats XML, avec `longxml` générant plus de détails.

- error-limit

  `--error-limit =` *`N`*
  Spécifie la limite d'erreur avec une plage de valeur de `1` à `999`. La valeur par défaut est `100`. Le traitement s'arrête lorsque la limite d'erreur est atteinte. Utile pour limiter l'utilisation du processeur pendant la validation/transformation.

- help

  `--help`
  Affiche le texte d'aide pour la commande. Par exemple, `valany --h`. (en alternative, la commande `help` peut être utilisée avec un argument. Par exemple : `help valany`.)

- log-output

  `--log-output =` *`FILE`*
  Écrit la sortie de log dans l'URL de fichier spécifié. Veuillez vous assurer que la CLI a une permission d'écriture dans l'emplacement de la sortie.

- network-timeout

  `--network-timeout =` *`VALUE`*
  Spécifie le timeout en secondes pour les opérations I/O à distance. La valeur par défaut est : `40`.

- verbose

  `--verbose =` `true|false`
  Une valeur de `true` permet la sortie d'informations supplémentaires pendant la validation. La valeur par défaut est `false`.
  Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- verbose-output

  `--verbose-output =` *`FILE`*
  Écrit la sortie verbeuse sur *`FILE`*.

- version

  `--version`
  Affiche la version de RaptorXML+XBRL Server. Si utilisé avec une commande, placer `--version` avant la commande.

- warning-limit

  `--warning-limit =` *`VALUE`*
  Spécifie la limite d'avertissement dans la plage `1-65535`. Le traitement se poursuit si cette

limite a été atteinte, mais d'autres avertissements ne sont pas rapportés. La valeur par défaut est `100`.

© 2018 Altova GmbH

Traitement

- listfile

  --listfile = true|false

  Si true, traiter l'argument *InputFile* de la commande en tant que fichier de texte contenant un nom de fichier par ligne. La valeur par défaut est false. (Une alternative est de lister les fichiers sur la CLI avec un espace en tant que séparateur. Veuillez noter, néanmoins, que les CLI ont une limitation de caractères maximum.) Veuillez noter que l'option --listfile s'applique uniquement aux arguments, et non pas aux options.
  Note : les valeurs d'option booléennes sont configurées sur true si l'option est spécifiée sans une valeur.

- parallel-assessment [pa]

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

  Si configuré sur true, l'évaluation de validité de schéma est effectuée en parallèle. Cela signifie qu'il y a plus de 128 éléments à tout niveau, ces éléments sont traités en parallèle, en utilisant plusieurs threads. Des fichiers XML très volumineux peuvent donc être traités plus rapidement si cette option est activée. L'évaluation parallèle a lieu un niveau hiérarchique à la fois, mais peut se produire à plusieurs niveaux différents dans une seule infobulle. Veuillez noter que l'évaluation parallèle ne fonctionne pas en mode streaming. C'est pour cette raison que l'option --streaming est ignorée si l'évaluation --parallel- est définie sur true. De même, les besoins en mémoire sont plus élevés lorsque l'option --parallel-assessment est utilisée. Le paramètre par défaut est false. La forme abrégée pour l'option est --pa.
  Note : les valeurs d'option booléennes sont configurées sur true si l'option est spécifiée sans une valeur.

- script

  --script = *FILE*

  Exécute le script Python dans le fichier soumis une fois que la validation a été terminée. Ajouter l'option plusieurs fois pour spécifier plus d'un seul script.

- script-api-version

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

  Spécifie la version API Python à utiliser pour le script. La valeur par défaut est la dernière version, actuellement **2.4**. Au lieu des valeurs 1 et 2, vous pouvez aussi utiliser les valeurs 1.0 et 2.0, respectivement.

- script-param

  --script-param = *KEY:VALUE*

  Des paramètres spécifiés par l'utilisateur supplémentaires qui peuvent être accédés au cours de l'exécution des scripts Python. Ajouter l'option plusieurs fois pour spécifier plus d'un paramètre de script.

- streaming

  --streaming = true|false

  Permet la validation de streaming. La valeur par défaut est true. Dans le mode streaming, les données stockées dans la mémoire sont minimisées et le traitement est plus rapide. L'inconvénient est que l'information que peut être requise par conséquent, par exemple, un

modèle de données du document d'instance XML —ne sera pas disponible. Dans des situations où cela est significatif, le mode de streaming devra être éteint (en donnant à `--streaming` une valeur de `false`). En utilisant l'option `--script` avec la commande `valxml-withxsd`, désactiver le streaming. Veuillez noter que l'option `--streaming` est ignorée si `--parallel-assessment` est défini sur `true`.
Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- xml-validation-error-as-warning

`--xml-validation-error-as-warning =` `true|false`
Si `true`, traite les erreurs de validation en tant qu'avertissements. Si les erreurs sont traitées en tant qu'avertissements, les traitements supplémentaires, comme des transformations XSLT, continueront quelles que soient les erreurs. La valeur par défaut est `false`.

 © 2018 Altova GmbH

XBRL

- Options de validation et traitement XBRL
  - dimensions

    `--dimensions =` `true|false`
    Active les extensions XBRL Dimension 1.0. La valeur par défaut est `true`.
    Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

  - evaluate-referenced-parameters-only

    `--evaluate-referenced-parameters-only =` `true|false`
    Si `false`, force l'évaluation de tous les paramètres, même s'ils ne sont pas référencés par d'autres formules/assertions/tables. La valeur par défaut est : `true`.

  - extensible-enumerations

    `--extensible-enumerations =` `true|false`
    Si `true`, active l'extension XBRL Extensible Enumerations 1.0. La valeur par défaut est : `true`.

  - inconsistencies-limit

    `--inconsistencies-limit =` *`VALUE`*
    Spécifie la limite d'inconsistances XBRL, avec des valeurs allant entre `1-65535`. Le traitement continue lorsque la limite est atteinte mais les autres inconsistances ne sont pas rapportées. La valeur par défaut est `100`.

  - listfile

    `--listfile =` `true|false`
    Si `true`, traiter l'argument *`InputFile`* de la commande en tant que fichier de texte contenant un nom de fichier par ligne. La valeur par défaut est `false`. (Une alternative est de lister les fichiers sur la CLI avec un espace en tant que séparateur. Veuillez noter, néanmoins, que les CLI ont une limitation de caractères maximum.) Veuillez noter que l'option `--listfile` s'applique uniquement aux arguments, et non pas aux options.
    Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

  - parallel-assessment [pa]

    `--pa | --parallel-assessment =` `true|false`
    Si configuré sur `true`, l'évaluation de validité de schéma est effectuée en parallèle. Cela signifie qu'il y a plus de 128 éléments à tout niveau, ces éléments sont traités en parallèle, en utilisant plusieurs threads. Des fichiers XML très volumineux peuvent donc être traités plus rapidement si cette option est activée. L'évaluation parallèle a lieu un niveau hiérarchique à la fois, mais peut se produire à plusieurs niveaux différents dans une seule infobulle. Veuillez noter que l'évaluation parallèle ne fonctionne pas en mode streaming. C'est pour cette raison que l'option `--streaming` est ignorée si l'évaluation `--parallel-` est définie sur `true`. De même, les besoins en mémoire sont plus élevés lorsque l'option `--parallel-assessment` est utilisée. Le paramètre par défaut est `false`. La forme abrégée pour l'option est `--pa`.
    Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est

spécifiée sans une valeur.

- preload-xbrl-schemas

  --preload-xbrl-schemas = true|false
  Précharge les schémas de la spécification XBRL 2.1. La valeur par défaut est true.
  Note : les valeurs d'option booléennes sont configurées sur true si l'option est spécifiée sans une valeur.

- recurse

  --recurse = true|false
  Utilisé pour sélectionner des fichiers dans le cadre de sous-répertoires, y compris des archives ZIP. Si true, l'argument *InputFile* de la commande sélectionnera aussi le fichier spécifié dans les sous-répertoires. Par exemple : test.zip|zip\test.xml sélectionnera des fichiers nommés test.xml à tous les niveaux de dossier du dossier zip. Les caractères génériques * et ? peuvent être utilisés. Donc, *.xml sélectionnera tous les fichiers .xml dans le dossier (zip). La valeur par défaut de l'option est false.
  Note : les valeurs d'option booléennes sont configurées sur true si l'option est spécifiée sans une valeur.

- schema-imports

  --schema-imports = load-by-schemalocation | load-preferring-schemalocation | load-by-namespace | load-combining-both | license-namespace-only
  Spécifie le comportement des éléments xs:import, chacun d'entre eux ayant un attribut namespace optionnel et un attribut schemaLocation optionnel : <import namespace="someNS" schemaLocation="someURL">. L'option spécifie s'il faut charger un document de schéma ou juste mettre un espace de noms sous licence, et, si un document de schéma doit être chargé, quelle information doit être utilisée pour le trouver. Défaut : load-preferring-schemalocation.
  Le comportement est le suivant :

  - load-by-schemalocation: La valeur de l'attribut schemaLocation est utilisée pour situer le schéma, en prenant en compte les mappages de catalogue. Si l'attribut de l'espace de nom est présent, l'espace de noms est importé (licencé).
  - load-preferring-schemalocation: Si l'attribut schemaLocation est présent, il est utilisé en prenant en compte les mappages de catalogue. Si l'attribut schemaLocation est présent, la valeur de l'attribut namespace est utilisée via un mappage de catalogue. C'est la **valeur par défaut**.
  - load-by-namespace: La valeur de l'attribut namespace est utilisée pour situer le schéma via un mappage de catalogue.
  - load-combining-both: Si soit l'Attribut namespace ou l'attribut schemaLocation a un mappage de catalogue, le mappage est utilisé. Si les deux ont des mappages de catalogue, alors la valeur de l'option --schema-mapping (option XBRL et option XML/XSD) décide de quel mappage utiliser. Si aucun mappage de catalogue n'est présent, l'attribut schemaLocation est utilisé.
  - license-namespace-only: Le nom d'espace est importé. Aucun document de schéma n'est importé.

- schema-mapping

--schema-mapping = `prefer-schemalocation | prefer-namespace`
Si l'emplacement de schéma et l'espace de noms sont tous les deux utilisés pour trouver un document de schéma, spécifie lequel des deux doit être utilisé pendant la consultation du catalogue. (Si soit l'option `--schemalocation-hints` ou l'option `--schema-imports` a une valeur de `load-combining-both`, et si les parties d'espace de noms et d'URL impliquées ont toutes les deux des mappages de catalogue, alors la valeur de cette option spécifie lequel des deux mappages utiliser (mappage d'espace de noms ou mappage URL ; la valeur `prefer-schemalocation` réfère au mappage URL).) Défaut : `prefer-schemalocation`.

- schemalocation-hints

--schemalocation-hints = `load-by-schemalocation | load-by-namespace | load-combining-both | ignore`
Spécifie le comportement des attributs `xsi:schemaLocation` et `xsi:noNamespaceSchemaLocation` : S'il faut charger un document de schéma et, si oui, quelle information doit être utilisée pour la trouver. Défaut : `load-by-schemalocation`.

- La valeur `load-by-schemalocation` utilise l'URL de l'emplacement de schéma dans les attributs `xsi:schemaLocation` et `xsi:noNamespaceSchemaLocation` dans les documents d'instance XML ou XBRL. Il s'agit de la **valeur par défaut**.
- La valeur `load-by-namespace` prend la part d'espace de nom `xsi:schemaLocation` et un string vide dans le cas de `xsi:noNamespaceSchemaLocation` et localise le schéma par le biais d'un mappage de catalogue.
- Si `load-combining-both` est utilisé et si soit la partie espace de noms ou la partie URL a un mappage de catalogue, alors le mappage de catalogue est utilisé. Si tous deux ont des mappages de catalogue, alors la valeur de l'option de `--schema-mapping` (option XBRL et option XML/XSD) de quel mappage utiliser. Si ni l'espace de noms ni l'URL n'a un mappage de catalogue, l'URL est utilisée.
- Si la valeur de l'option est `ignore`, les attributs `xsi:schemaLocation` et `xsi:noNamespaceSchemaLocation` seront ignorés tous les deux.

- script

--script = *FILE*
Exécute le script Python dans le fichier soumis une fois que la validation a été terminée. Ajouter l'option plusieurs fois pour spécifier plus d'un seul script.

- script-api-version

--api, --script-api-version = `1|2|2.1|2.2|2.3|2.4`
Spécifie la version API Python à utiliser pour le script. La valeur par défaut est la dernière version, actuellement **`2.4`**. Au lieu des valeurs `1` et `2`, vous pouvez aussi utiliser les valeurs `1.0` et `2.0`, respectivement.

- script-param

--script-param = *KEY:VALUE*
Des paramètres spécifiés par l'utilisateur supplémentaires qui peuvent être accédés au cours de l'exécution des scripts Python. Ajouter l'option plusieurs fois pour spécifier plus d'un paramètre de script.

- treat-inconsistencies-as-errors

  --treat-inconsistencies-as-errors = `true|false`
  Entraîne l'échec de la validation XBRL si le fichier contient des inconsistances telles que définies par la spécification XBRL 2.1. La valeur par défaut est `false`.
  Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- utr

  --utr = `true|false`
  Si `true`, active l'extension XBRL Unit Registry 1.0. La valeur par défaut est : `false`.

- validate-dts-only

  --validate-dts-only = `true|false`
  Le DTS est découvert en démarrant depuis le document d'instance XBRL. Tous les schémas de taxonomie référencés et bases de lien sont découverts et validés. Le reste du document d'instance XBRL est ignoré. La valeur par défaut est `false`.
  Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- xinclude

  --xinclude = `true|false`
  Active la prise en charge de XML Inclusions (XInclude). La valeur par défaut est `false`. Si `false`, les élément `include` d'XInclude sont ignorés.
  Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- XBRL formula and assertion options
  - assertion-severity

    --assertion-severity = `true|false`
    Active l'extension Assertion Severity 1.0. La valeur par défaut est `true`.
    Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

  - assertions-output

    --assertions-output = *`FILE`*
    Écrit la sortie de l'évaluation d'assertion sur le *`FILE`* spécifié. Si défini, spécifie automatiquement `--formula-execution=true`.

  - assertions-output-format

    --assertions-output-format = `json|xml`
    Spécifie le format de sortie de l'évaluation d'assertion. La valeur par défaut est `json`.

  - formula

    --formula = `true|false`

 © 2018 Altova GmbH

Active l'extension XBRL Formula 1.0. La valeur par défaut est `true`.
Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- formula-assertion-set [[DEPRECATED]]

  `--formula-assertion-set =` *`VALUE`*
  Limite l'exécution de la formule au seul ensemble d'assertion donné. Ajouter l'option plusieurs fois pour spécifier plus d'un ensemble d'assertion. La forme abrégée est `--as`. La ***`VALUE`*** est soit la valeur de l'attribut `@id`, soit une URI avec un fragment XPointer qui identifie la ressource. Les valeurs spéciales `##none` et `##all` peuvent aussi être utilisées.

- formula-execution

  `--formula-execution =` `true|false`
  Active l'extension des formules XBRL. La valeur par défaut est `true`. Si `true`, spécifie automatiquement `--formula=true`.
  Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- formula-output

  `--formula-output =` *`FILE`*
  Écrit la sortie de l'évaluation de la formule pour le *`FILE`* spécifié. Si défini, spécifie automatiquement `--formula-execution=true`.

- formula-parameters

  `--formula-parameters =` *`JSON-ARRAY`*
  Spécifie les paramètres pour l'évaluation de la formule XBRL en tant qu'un array de mappages JSON directement sur la CLI. Voir la section Paramètres de formule pour plus de détails.

- formula-parameters-file

  `--formula-parameters-file =` *`FILE`*
  Spécifie un *`FILE`* contenant les paramètres pour l'évaluation de formule XBRL. Le fichier peut être soit un fichier XML soit un fichier JSON. Voir la section Paramètres de formule.

- message-lang

  `--message-lang =` *`VALUE`*
  Spécifie la langue à utiliser lors de l'affichage des messages de validation. La valeur par défaut est : `en`. Les autres valeurs acceptées sont `de`, `es`, `fr`, `ja`, pour, respectivement, l'allemand, l'espagnol, le français et le japonais.

- message-role

  `--message-role =` *`VALUE`*
  Spécifie le rôle de message préféré à utiliser lors de l'affichage des messages de validation. La valeur par défaut est : `http://www.xbrl.org/2010/role/message`.

- preload-formula-schemas

--preload-formula-schemas = `true|false`
Précharge les schémas de la spécification XBRL Formula 1.0. La valeur par défaut est `false`.
Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- process-assertion [a]

--a | --process-assertion = *VALUE*
Limite l'exécution de formule à la seule assertion donnée. Ajouter l'option plusieurs fois pour spécifier plus d'une assertion. Forme abrégée est `--a`. La ***VALUE*** est soit la valeur de l'attribut `@id`, soit un URI avec un fragment XPointer qui identifie la ressource. Les valeurs spéciales `##none` et `##all` peuvent aussi être utilisées.

- process-assertion-set [as]

--as | --process-assertion-set = *VALUE*
Limite l'exécution de formule au seul ensemble d'assertion donné. Ajouter l'option plusieurs fois pour spécifier plus d'un ensemble d'assertion. Forme abrégée est `--as`. La ***VALUE*** est soit la valeur de l'attribut `@id`, soit un URI avec un fragment XPointer qui identifie la ressource. Les valeurs spéciales `##none` et `##all` peuvent aussi être utilisées.

- process-formula [f]

--f | --process-formula = *VALUE*
Limite l'exécution de formule à la seule formule donnée. Ajouter l'option plusieurs fois pour spécifier plus d'une formule. Forme abrégée est `--f`. La ***VALUE*** est soit la valeur de l'attribut `@id`, soit un URI avec un fragment XPointer qui identifie la ressource. Les valeurs spéciales `##none` et `##all` peuvent aussi être utilisées.

- report-unsatisfied-assertion-evaluations

--report-unsatisfied-assertion-evaluations = `true|false`
Rapporte l'évaluation de l'assertion non satisfaite en tant qu'erreurs ou avertissements selon le degré de sévérité de l'assertion attribuée. La valeur par défaut est `false`.
Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- variableset-execution-timeout

--variableset-execution-timeout = *VALUE*
Appliqué lors de l'exécution de formules (**--formula-execution=true**). Spécifie le temps maximum autorisé pour l'exécution d'un ensemble de variables unique (une formule ou une valeur, ou une existence ou une assertion de consistance). Le temps est spécifié en minutes et doit être un nombre positif. La valeur par défaut est 30min. Si un ensemble de variables particulier ne termine pas l'exécution avant que le timeout n'ait été atteint, il sera abandonné. Un message d'erreur est affiché (et saisi dans le journal verbeux). Veuillez noter, néanmoins, que le contrôle de timeout est uniquement effectué après l'évaluation de chaque ensemble de variable ; et non pas pendant l'exécution des expressions XPath individuelles. Donc, si une seule expression XPath prend beaucoup de temps à être exécutée, la limite de timeout peut avoir été dépassée. L'exécution d'un ensemble de variable est abandonnée uniquement une fois que

 © 2018 Altova GmbH

l'évaluation du set de variable complet ait été exécuté.

- XBRL table options
  - concept-label-linkrole

    `--concept-label-linkrole =` *`VALUE`*
    Spécifie le rôle de lien étendu préféré à utiliser lors du rendu de libellés de concept.

  - concept-label-role

    `--cconcept-label-role =` *`VALUE`*
    Spécifie le rôle de lien préféré à utiliser lors du rendu de libellés de concept. Par défaut : `http://www.xbrl.org/2003/role/label`.

  - evaluate-referenced-parameters-only

    `--evaluate-referenced-parameters-only =` `true|false`
    Si `false`, force l'évaluation de tous les paramètres, même s'ils ne sont pas référencés par d'autres formules/assertions/tables. La valeur par défaut est : `true`.

  - generic-label-linkrole

    `--generic-label-linkrole =` *`VALUE`*
    Spécifie le rôle de lien étendu préféré à utiliser lors du rendu des libellés génériques.

  - generic-label-role

    `--generic-label-role =` *`VALUE`*
    Spécifie le rôle de libellé préféré à utiliser lors du rendu des libellés génériques. La valeur par défaut est : http://www.xbrl.org/2003/role/label.

  - label-lang

    `--label-lang =` *`VALUE`*
    Spécifie la langue préférée à utiliser lors du rendu des libellés. La valeur par défaut est : `en`.

  - preload-table-schemas

    `--preload-table-schemas =` `true|false`
    Précharge les schémas de la spécification XBRL Table 1.0. La valeur par défaut est `false`.
    Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

  - process-table [t]

    `--t | --process-table =` *`VALUE`*
    Limite l'exécution de formule à la seule table donnée. Ajouter l'option plusieurs fois pour spécifier plus d'une table. Forme abrégée est `--t`. La ***`VALUE`*** est soit la valeur de l'attribut `@id`, soit un URI avec un fragment XPointer qui identifie la ressource. Les valeurs spéciales `##none` et `##all` peuvent aussi être utilisées.

- table

  --table = `true|false`
  Active l'extension XBRL Table 1.0. La valeur par défaut est `true`. Si `true`, spécifie automatiquement `--formula=true` et `--dimensions=true`.
  Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- table-elimination

  --table-elimination = `true|false`
  Active l'élimination de lignes/colonnes vides dans la sortie HTML. La valeur par défaut est `true`.
  Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- table-execution

  --table-execution = `true|false`
  Active l'évaluation des tables XBRL. La valeur par défaut est `false`. Sera configuré sur `true` si `--table-output` est spécifié. Si `true`, spécifie automatiquement `--table=true`.
  Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- 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`

  Active le chargement des bases de lien de table rédigées par le biais d'une spécification d'ébauche précédente. Néanmoins, la validation, la résolution et la mise en page de la base de lien de table sont toujours effectués conformément à la Recommandation Table Linkbase 1.0 du 18 mars 2014. Utiliser `##detect` pour activer la détection automatique.

- table-output

  --table-output = *FILE*
  Écrit la sortie de table dans le *FILE* spécifié. Si défini, spécifie automatiquement `--table-execution=true`.

- table-output-format

  --table-output-format = `xml|html`
  Spécifie le format de la sortie de table. La valeur par défaut est `xml`.

 © 2018 Altova GmbH

XML

- assessment-mode

  --assessment-mode = `lax|strict`
  Spécifie le mode d'évaluation de validité de schéma tel que défini dans les spécifications XSD. La valeur par défaut est `strict`. Le document d'instance XML sera validé conformément au mode spécifié avec cette option.

- dtd

  --dtd = *FILE*
  Spécifie le document DTD externe à utiliser pour la validation. Si une référence à un DTD externe est présente dans le document XM, alors l'option CLI contourne la référence externe.

- load-xml-with-psvi

  --load-xml-with-psvi = `true|false`
  Active la validation des fichiers XML d'entrée et génère des informations de post-schema-validation pour les fichiers. La valeur par défaut est : `true`.

- namespaces

  --namespaces = `true|false`
  Permet le traitement sensible aux espaces de noms. Cela est utile pour contrôler l'instance XML pour des erreurs dues à des espaces de noms incorrectes. La valeur par défaut est `false`.
  Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- xinclude

  --xinclude = `true|false`
  Active la prise en charge de XML Inclusions (XInclude). La valeur par défaut est `false`. Si `false`, les élément `include` d'XInclude sont ignorés.
  Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- xml-mode

  --xml-mode = `wf|id|valid`
  Spécifie le mode de traitement XML à utiliser : `wf`=wellformed check (vérification de la bonne forme); `id`=bien formé avec des contrôles ID/IDREF ; `valid`=validation. La valeur par défaut est `wf`. Veuillez noter qu'une valeur de `valid` exige que chaque document de schéma/instance chargé depuis les traitements référence un DTD. Si aucun DTD n'existe, une erreur est rapportée.

- xml-validation-error-as-warning

  --xml-validation-error-as-warning = `true|false`
  Si `true`, traite les erreurs de validation en tant qu'avertissements. Si les erreurs sont traitées en tant qu'avertissements, les traitements supplémentaires, comme des transformations XSLT, continueront quelles que soient les erreurs. La valeur par défaut est `false`.

- xsd

--xsd = *FILE*

Spécifie un ou plusieurs documents de Schéma XML à utiliser pour la validation des documents d'instance XML. Ajouter l'option plusieurs fois pour spécifier plus d'un document de schéma.

 © 2018 Altova GmbH

XSD

assessment-mode

--assessment-mode = lax|strict
Spécifie le mode d'évaluation de validité de schéma tel que défini dans les spécifications XSD. La valeur par défaut est strict. Le document d'instance XML sera validé conformément au mode spécifié avec cette option.

namespaces

--namespaces = true|false
Permet le traitement sensible aux espaces de noms. Cela est utile pour contrôler l'instance XML pour des erreurs dues à des espaces de noms incorrectes. La valeur par défaut est false.
Note : les valeurs d'option booléennes sont configurées sur true si l'option est spécifiée sans une valeur.

schema-imports

--schema-imports = load-by-schemalocation | load-preferring-schemalocation | load-by-namespace | load-combining-both | license-namespace-only
Spécifie le comportement des éléments xs:import, chacun d'entre eux ayant un attribut namespace optionnel et un attribut schemaLocation optionnel : `<import namespace="someNS" schemaLocation="someURL">`. L'option spécifie s'il faut charger un document de schéma ou juste mettre un espace de noms sous licence, et, si un document de schéma doit être chargé, quelle information doit être utilisée pour le trouver. Défaut : load-preferring-schemalocation.
Le comportement est le suivant :

- load-by-schemalocation: La valeur de l'attribut schemaLocation est utilisée pour situer le schéma, en prenant en compte les mappages de catalogue. Si l'attribut de l'espace de nom est présent, l'espace de noms est importé (licencé).
- load-preferring-schemalocation: Si l'attribut schemaLocation est présent, il est utilisé en prenant en compte les mappages de catalogue. Si l'attribut schemaLocation est présent, la valeur de l'attribut namespace est utilisée via un mappage de catalogue. C'est la **valeur par défaut**.
- load-by-namespace: La valeur de l'attribut namespace est utilisée pour situer le schéma via un mappage de catalogue.
- load-combining-both: Si soit l'Attribut namespace ou l'attribut schemaLocation a un mappage de catalogue, le mappage est utilisé. Si les deux ont des mappages de catalogue, alors la valeur de l'option --schema-mapping (option XBRL et option XML/XSD) décide de quel mappage utiliser. Si aucun mappage de catalogue n'est présent, l'attribut schemaLocation est utilisé.
- license-namespace-only: Le nom d'espace est importé. Aucun document de schéma n'est importé.

schemalocation-hints

--schemalocation-hints = load-by-schemalocation | load-by-namespace | load-combining-both | ignore
Spécifie le comportement des attributs xsi:schemaLocation et xsi:noNamespaceSchemaLocation : S'il faut charger un document de schéma et, si oui, quelle information doit être utilisée pour la trouver. Défaut : load-by-schemalocation.

- La valeur `load-by-schemalocation` utilise l'URL de l'emplacement de schéma dans les attributs `xsi:schemaLocation` et `xsi:noNamespaceSchemaLocation` dans les documents d'instance XML ou XBRL. Il s'agit de la **valeur par défaut**.
- La valeur `load-by-namespace` prend la part d'espace de nom `xsi:schemaLocation` et un string vide dans le cas de `xsi:noNamespaceSchemaLocation` et localise le schéma par le biais d'un mappage de catalogue.
- Si `load-combining-both` est utilisé et si soit la partie espace de noms ou la partie URL a un mappage de catalogue, alors le mappage de catalogue est utilisé. Si tous deux ont des mappages de catalogue, alors la valeur de l'option de `--schema-mapping` (option XBRL et option XML/XSD) de quel mappage utiliser. Si ni l'espace de noms ni l'URL n'a un mappage de catalogue, l'URL est utilisée.
- Si la valeur de l'option est `ignore`, les attributs `xsi:schemaLocation` et `xsi:noNamespaceSchemaLocation` seront ignorés tous les deux.

schema-mapping

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

Si l'emplacement de schéma et l'espace de noms sont tous les deux utilisés pour trouver un document de schéma, spécifie lequel des deux doit être utilisé pendant la consultation du catalogue. (Si soit l'option `--schemalocation-hints` ou l'option `--schema-imports` a une valeur de `load-combining-both`, et si les parties d'espace de noms et d'URL impliquées ont toutes les deux des mappages de catalogue, alors la valeur de cette option spécifie lequel des deux mappages utiliser (mappage d'espace de noms ou mappage URL ; la valeur `prefer-schemalocation` réfère au mappage URL).) Défaut : `prefer-schemalocation`.

xsd-version

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

Spécifie la version du langage de définition de Schéma W3C (XSD) à utiliser. Le défaut est `1.0`. Cette option peut être utile pour découvrir comment un schéma compatible à 1.0 n'est pas compatible à 1.1. L'option `detect` est une fonction spécifique à Altova. Elle permet la détection de la version du document de Schéma XML (`1.0` ou `1.1`) en lisant la valeur de l'attribut `vc:minVersion` de l'élément du document `<xs:schema>`. Si la valeur de l'attribut `@vc:minVersion` est `1.1`, le schéma est détecté comme étant de version `1.1`. Pour toute autre valeur, ou si l'attribut `@vc:minVersion` est absent, le schéma est détecté comme étant de version `1.0`.

 © 2018 Altova GmbH

XQuery

- indent-characters

  --indent-characters = *VALUE*
  Spécifie le string de caractère à utiliser en tant que retrait.

- input

  --input = *FILE*
  L'URL du fichier XML à transformer.

- keep-formatting

  --keep-formatting = `true|false`
  Garde le formatage du document cible autant que possible. La valeur par défaut est : `true`.

- omit-xml-declaration

  --omit-xml-declaration = `true|false`
  Option de sérialisation pour spécifier si la déclaration XML doit être omise de la sortie ou pas. Si `true`, il n'y aura pas de déclaration XML dans le document de sortie. Si `false`, une déclaration XML sera incluse. La valeur par défaut est `false`.
  Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- output, xsltoutput

  output = *FILE*, **xsltoutput =** *FILE*
  L'URL du fichier sortie-primaire. Par exemple, dans le cas d'une sortie HTML de fichier multiple, le fichier de sortie primaire sera l'emplacement du fichier HTML du point d'entrée. Les fichiers de sortie supplémentaires, comme des fichiers d'image générée, sont rapportés en tant que `xslt-additional-output-files`. Si aucune option `--output` ou `--xsltoutput` n'est spécifiée, la sortie est écrite sur la sortie standard.

- output-encoding

  --output-encoding = *VALUE*
  La valeur de l'attribut d'encodage dans le document de sortie. Les valeurs valides sont des noms dans le registre d'ensemble de caractères IANA. La valeur par défaut est `UTF-8`.

- output-indent

  --output-indent = `true|false`
  Si `true`, la sortie sera en retrait conformément à sa structure hiérarchique. Si `false`, il n'y aura pas de retrait hiérarchique. La valeur par défaut est `false`.
  Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- output-method

  --output-method = `xml|html|xhtml|text`
  Spécifie le format de sortie. La valeur par défaut est `xml`.

- param [p]

  --p | --param = *KEY:VALUE*

XQuery

Spécifie la valeur d'un paramètre externe. Un paramètre externe est déclaré dans le document XQuery avec la déclaration `declare variable` suivi par un nom de variable puis le mot-clé `external` suivi par le point-virgule de fin de ligne. Par exemple :

```
declare variable $foo as xs:string external;
```

En raison du mot-clé `external`, `$foo` devient un paramètre externe, dont la valeur est passée pendant le runtime depuis une source externe. Le paramètre externe reçoit une value avec la commande CLI. Par exemple :

```
--param=foo:'MyName'
```

Dans l'instruction de description ci-dessus, *KEY* est le nom de paramètre externe, *VALUE* est la valeur du paramètre externe, donné en tant qu'une expression XPath. Les noms de paramètres utilisés sur la CLI doivent être déclarés dans le document XQuery. Si plusieurs paramètres externes sont des valeurs passées sur la CLI, chacun doit être recevoir une option séparée `--param`. Les doubles guillemets doivent être utilisés si l'expression XPath contient des espaces.

XSLT

Spécifie paramètre de feuille de style global. *KEY* est le nom de paramètre, *VALUE* est une expression XPath qui fournit la valeur de paramètre. Les noms de paramètre utilisés sur la CLI doivent être déclarés dans la feuille de style. Si plusieurs paramètres multiples sont utilisés, le changement `--param` doit être utilisé avant chaque paramètre. Les double guillemets doivent être utilisés autour de l'expression XPath si elle contient un espace ; que l'espace se trouve dans l'expression XPath elle-même ou dans un littéral de string dans l'expression. Par exemple :

```
raptorxmlxbrl xslt --input=c:\Test.xml --output=c:\Output.xml --
param=date://node[1]/@att1 --p=title:'stringwithoutspace' --
param=title:"'string with spaces'" --p=amount:456 c:\Test.xslt
```

updated-xml

`--updated-xml =` `discard|writeback|asmainresult`

Spécifie comment gérer le fichier XML mis à jour. Les mises à jour peuvent être soit :

- éliminées et pas écrites sur le fichier (`discard`)
- réécrites sur le fichier d'entrée XML qui est spécifié avec l'option `--input` (`writeback`)
- enregistrées soit sur une sortie standard soit sut l'emplacement spécifié dans l'option `--output` (si cela a été défini)

La valeur par défaut est : `discard`.

xquery-update-version

`--xquery-update-version =` `1|3`

Spécifie si le processeur XQuery devrait utiliser XQuery Update Facility 1.0 ou XQuery Update Facility 3.0. La valeur par défaut est `3`.

xquery-version

`--xquery-version =` `1|1.0|3|3.0|3.1`

Spécifie si le processeur XQuery devrait utiliser XSLT XQuery 1.0 ou XQuery 3.0. La valeur

 © 2018 Altova GmbH

par défaut est 3.1.

XSLT

- chartext-disable

  --chartext-disable = `true|false`
  Désactive les extensions de graphique. La valeur par défaut est `false`.
  Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- dotnetext-disable

  --dotnetext-disable = `true|false`
  Désactive les extensions .NET. La valeur par défaut est `false`.
  Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- indent-characters

  --indent-characters = *VALUE*
  Spécifie le string de caractère à utiliser en tant que retrait.

- input

  --input = *FILE*
  L'URL du fichier XML à transformer.

- javaext-barcode-location

  --javaext-barcode-location = *FILE*
  Spécifie le chemin vers le dossier qui contient le fichier d'extension du code-barres `AltovaBarcodeExtension.jar`. Le chemin doit être donné dans une des formes suivantes :

  - Un URI de fichier, par exemple : `--javaext-barcode-location="file:///C:/Program Files/Altova/RaptorXMLXBRLServer2018/etc/jar/"`
  - Un chemin Windows avec un échappement par barres obliques inversées, par exemple : `--javaext-barcode-location="C:\\Program Files\\Altova\\RaptorXMLXBRLServer2018\\etc\\jar\\"`

- javaext-disable

  --javaext-disable = `true|false`
  Désactive les extensions Java. La valeur par défaut est `false`.
  Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- output, xsltoutput

  output = *FILE*, **xsltoutput =** *FILE*
  L'URL du fichier sortie-primaire. Par exemple, dans le cas d'une sortie HTML de fichier multiple, le fichier de sortie primaire sera l'emplacement du fichier HTML du point d'entrée. Les fichiers de sortie supplémentaires, comme des fichiers d'image générée, sont rapportés en tant que `xslt-additional-output-files`. Si aucune option `--output` ou `--xsltoutput` n'est spécifiée, la sortie est écrite sur la sortie standard.

- param [p]

 © 2018 Altova GmbH

--p | --param = *KEY:VALUE*

XQuery

Spécifie la valeur d'un paramètre externe. Un paramètre externe est déclaré dans le document XQuery avec la déclaration `declare variable` suivi par un nom de variable puis le mot-clé `external` suivi par le point-virgule de fin de ligne. Par exemple :

```
declare variable $foo as xs:string external;
```

En raison du mot-clé `external`, `$foo` devient un paramètre externe, dont la valeur est passée pendant le runtime depuis une source externe. Le paramètre externe reçoit une value avec la commande CLI. Par exemple :

```
--param=foo:'MyName'
```

Dans l'instruction de description ci-dessus, *KEY* est le nom de paramètre externe, *VALUE* est la valeur du paramètre externe, donné en tant qu'une expression XPath. Les noms de paramètres utilisés sur la CLI doivent être déclarés dans le document XQuery. Si plusieurs paramètres externes sont des valeurs passées sur la CLI, chacun doit être recevoir une option séparée `--param`. Les doubles guillemets doivent être utilisés si l'expression XPath contient des espaces.

XSLT

Spécifie paramètre de feuille de style global. *KEY* est le nom de paramètre, *VALUE* est une expression XPath qui fournit la valeur de paramètre. Les noms de paramètre utilisés sur la CLI doivent être déclarés dans la feuille de style. Si plusieurs paramètres multiples sont utilisés, le changement `--param` doit être utilisé avant chaque paramètre. Les double guillemets doivent être utilisés autour de l'expression XPath si elle contient un espace ; que l'espace se trouve dans l'expression XPath elle-même ou dans un littéral de string dans l'expression. Par exemple :

```
raptorxmlxbrl xslt --input=c:\Test.xml --output=c:\Output.xml --
param=date://node[1]/@att1 --p=title:'stringwithoutspace' --
param=title:"'string with spaces'" --p=amount:456 c:\Test.xslt
```

- streaming

--streaming = `true|false`

Permet la validation de streaming. La valeur par défaut est `true`. Dans le mode streaming, les données stockées dans la mémoire sont minimisées et le traitement est plus rapide. L'inconvénient est que l'information que peut être requise par conséquent, par exemple, un modèle de données du document d'instance XML —ne sera pas disponible. Dans des situations où cela est significatif, le mode de streaming devra être éteint (en donnant à `--streaming` une valeur de `false`). En utilisant l'option `--script` avec la commande `valxml-withxsd`, désactiver le streaming. Veuillez noter que l'option `--streaming` est ignorée si`--parallel-assessment` est défini sur `true`.

Note : les valeurs d'option booléennes sont configurées sur `true` si l'option est spécifiée sans une valeur.

- initial-template, template-entry-point

--template-entry-point = *VALUE*

Donne le nom d'un modèle nommé dans la feuille de style XSLT qui est le point d'entrée de la transformation.

- initial-mode, template-mode

--template-mode = *VALUE*
Spécifie le mode de modèle à utiliser pour la transformation.

- xslt-version

--xslt-version = 1|1.0|2|2.0|3|3.0|3.1
Spécifie si le processeur XSLT devrait utiliser XSLT 1.0, XSLT 2.0 ou XSLT 3.0. La valeur par défaut est 3.

 © 2018 Altova GmbH

JSON/Avro

- schema, **jsonschema**

  --schema = *FILE*, **--jsonschema =** *FILE*
  Spécifie le chemin vers le document de Schéma JSON à utiliser pour la validation des documents d'instance JSON.

- codec

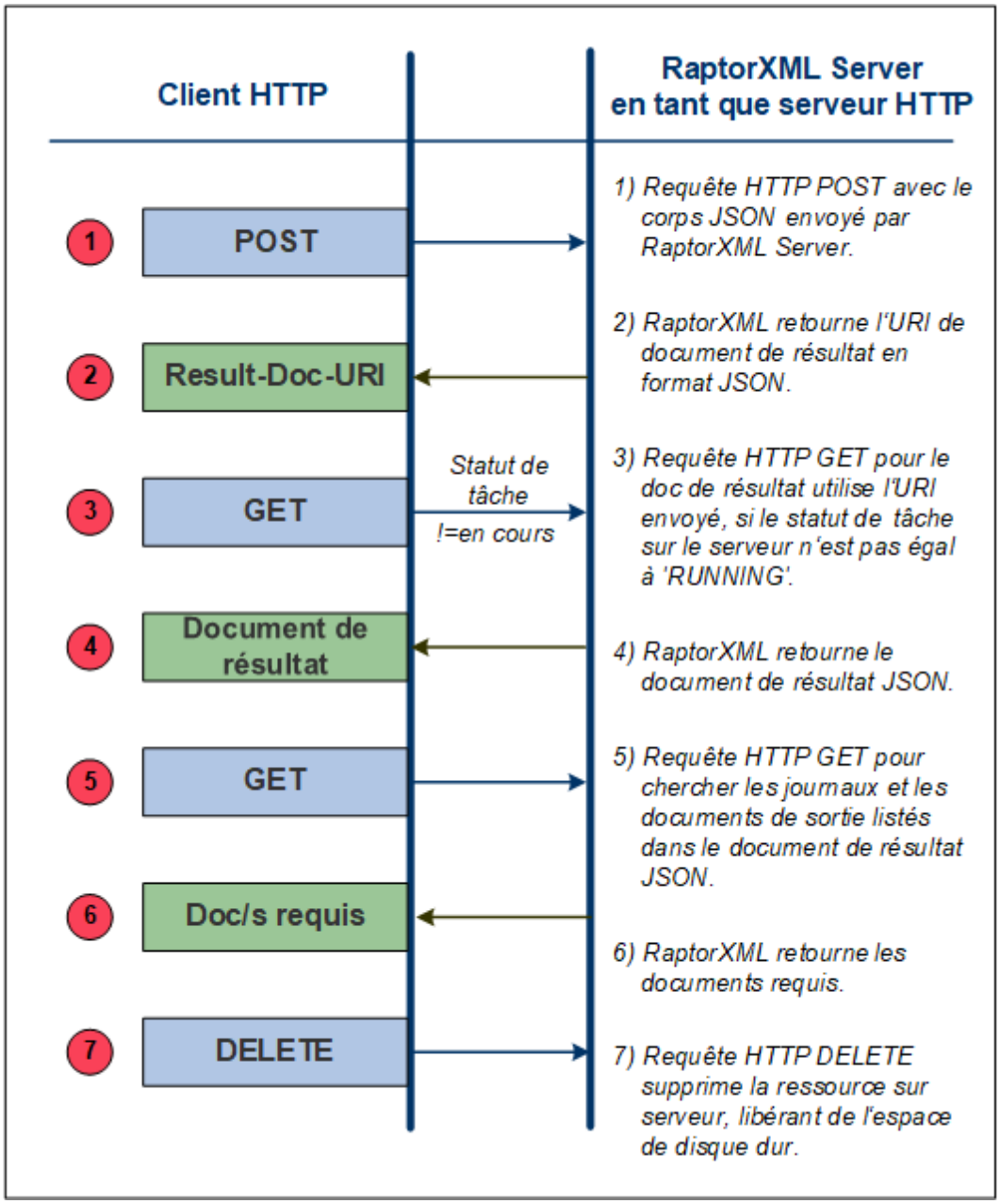
  --codec = `null|deflate`
  Spécifie le codec de compression à utiliser. La valeur par défaut est `null`.

Chapitre 4

Interface HTTP

4 Interface HTTP

RaptorXML+XBRL Server accepte les tâches de validation soumises par HTTP (ou HTTPS). La description de tâche ainsi que les résultats sont échangés en format JSON. Le flux de travail de base est tel que montré dans le diagramme ci-dessous.



 © 2018 Altova GmbH

Problèmes de sécurité liés à l'interface HTTP
L'interface HTTP, par défaut, permet aux documents de résultat d'être rédigés sur tout emplacement spécifié par le client (qui est accessible avec le protocole HTTP). C'est pourquoi il est important de considérer ces aspects de sécurité lors de la configuration de RaptorXML+XBRL Server.

Si vous estimez que la sécurité ait pu être compromise ou que l'interface ait pu être utilisée à mauvais escient, le serveur peut être configuré pour écrire les documents de résultat dans un répertoire de sortie réservé sur le serveur lui-même. Cela est spécifié en configurant l'option `server.unrestricted-filesystem-access` du fichier de configuration du serveur sur `false`. Lorsque l'accès est limité de cette manière, le client peut télécharger les documents de résultat depuis le répertoire de sortie réservé avec des requêtes `GET`. En alternative, un administrateur peut copier/charger les fichiers de document de résultat depuis le serveur vers l'emplacement cible.

Dans cette section

Avant d'envoyer une requête client, RaptorXML+XBRL Server doit être lancé et configuré correctement. La procédure est décrite dans la section Configuration de serveur. Vous trouverez une description comment envoyer les requêtes client dans la section Requêtes de client.

4.1 Configuration de serveur

Pour configurer correctement RaptorXML+XBRL Server, procéder comme suit. Nous assumons que RaptorXML+XBRL Server a déjà été installé et mis sous licence correctement.

1. RaptorXML+XBRL Server doit soit être lancé en tant qu'un service ou une application pour pouvoir être accédé correctement par le biais de HTTP ou HTTPS. La procédure diffère selon le système d'exploitation et est décrite ici : sur Windows, sur Linux, sur macOS.
2. Utiliser la configuration de serveur initiale pour tester la connexion au serveur. (La configuration de serveur initiale est la configuration par défaut que vous obtenez lors de l'installation.) Vous pouvez utiliser une simple requête HTTP `GET` comme `http://localhost:8087/v1/version` pour tester la connexion. (La requête peut aussi être saisie dans la barre d'adresse d'une fenêtre de navigateur.) Si le service est en cours d'exécution vous devez obtenir une réponse à une requête de test HTTP comme la requête de version ci-dessus.
3. Consultez le fichier de configuration de serveur, `server_config.xml`. Si vous souhaitez changer des paramètres dans le fichier, éditer le fichier de configuration de serveur et enregistrer les changements. HTTPS est désactivé par défaut et devront être activés dans le fichier de configuration.
4. Si vous avez édité le fichier de configuration de serveur, redémarrer RaptorXML+XBRL Server en tant que service de manière à ce que les nouveaux paramètres de configuration sont appliqués. Tester à nouveau de la connexion pour vous assurer que le service est en cours et accessible.

Note : Les erreurs de démarrage de serveur, le fichier de configuration de serveur qui est utilisé, et les erreurs de licence sont rapportés dans le journal de système. Veuillez donc vous référer au journal de système s'il y a des problèmes avec le serveur.

Pour plus d'informations concernant HTTPS, voir la section Paramètres HTTPS.

 © 2018 Altova GmbH

Démarrer le serveur

Cette section :

- Emplacement de l'exécutable de serveur
- Démarrer RaptorXML en tant que service sur Windows
- Démarrer RaptorXML en tant que service sur Linux
- Démarrer RaptorXML en tant que service sur macOS

Emplacement du fichier exécutable du serveur

L'exécutable RaptorXML+XBRL Server est installé par défaut dans le dossier :

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

L'exécutable peut être utilisé pour lancer RaptorXML+XBRL Server en tant que service.

Démarrage en tant que service sur Windows

Le processus d'installation aura enregistré RaptorXML+XBRL Server en tant qu'un service sur Windows. Vous devez, néanmoins, **lancer** RaptorXML+XBRL Server en tant qu'un service. Vous pouvez procéder des manières suivantes :

- Par le biais d'Altova ServiceController, que vous trouverez sous la forme d'une icône dans la zone de notification. Si l'icône n'est pas disponible, vous pouvez lancer Altova ServiceController et ajouter son icône à la zone de notification en vous rendant dans le menu **Start**, en sélectionnant **All Programs | Altova | Altova LicenseServer | Altova ServiceController**.
- Par le biais du Windows Services Management Console: **Control Panel | All Control Panel Items | Administrative Tools | Services**.
- Par le biais de l'invite de commandes que vous pouvez lancer si vous disposez des droites d'administrateur. Utilisez la commande suivante sous n'importe quel répertoire : **`net start "Altova RaptorXML+XBRL Server"`**
- Par le biais de l'exécutable RaptorXML+XBRL Server dans une fenêtre d"invite de commandes : **`RaptorXMLXBRLServer.exe debug`**. Cela démarre le serveur, les informations d'activité du serveur allant directement dans la fenêtre d'invite de commandes. L'affichage des informations d'activité du serveur peut être allumé et éteint avec le paramètre `http.log-screen` du fichier de configuration du serveur. Pour arrêter le serveur, appuyer sur **Ctrl+Break** (ou **Ctrl+Pause**). Lorsque le serveur est lancé de cette manière, au lieu de le lancer en tant qu'un service tel que décrit dans les trois premières étapes précédentes ; le serveur s'arrêtera lorsque la console de ligne de commande est fermée ou lorsque l'utilisateur se déconnecte.

Démarrage en tant que service sur Linux

Démarrer RaptorXML+XBRL Server en tant qu'un service avec la commande suivante :

[< 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, à tout moment, vous devez arrêter RaptorXML+XBRL Server, utiliser :

[< 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

Démarrage en tant que service sur macOS

Démarrer RaptorXML+XBRL Server en tant qu'un service avec la commande suivante :

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

Si, à tout moment, vous devez arrêter RaptorXML+XBRL Server, utiliser :

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

 © 2018 Altova GmbH

Tester la connexion

Cette section :

- Requête GET pour tester la connexion
- Réponse de serveur et liste de structure de données JSON

Requête GET pour tester la connexion

Une fois que RaptorXML+XBRL Server a été lancé, tester la connexion en utilisant une requête `GET`. (Vous pouvez aussi saisir cette requête dans la barre d'adresse d'une fenêtre de navigateur.)

```
http://localhost:8087/v1/version
```

Note : L'interface et le numéro port de RaptorXML+XBRL Server est spécifié dans le fichier de configuration de serveur, `server_config.xml`, qui est décrit dans la section suivante, Configuration de serveur.

Réponse de serveur et liste de structure de données JSON

Lorsque le service est exécuté et le serveur est correctement configuré, la requête ne devrait jamais échouer. RaptorXML+XBRL Server retournera son information de version en tant qu'une structure de donnée JSON (*liste ci-dessous*).

```
{
  "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"
}
```

Note : Si vous modifiez la configuration de serveur, lorsque vous éditez le fichier de configuration de serveur, vous devriez tester à nouveau la connexion.

Configurer le serveur

Cette section :

- Fichier de configuration de serveur : paramètres de base
- Fichier de configuration de serveur : modifier les paramètres de base, retourner aux paramètres de base
- Fichier de configuration de serveur : liste et paramètres
- Fichier de configuration de serveur : description des paramètres
- Configurer l'adresse de serveur

Fichier de configuration de serveur : paramètres de base

RaptorXML+XBRL Server est configuré au moyen d'un fichier de configuration appelé `server_config.xml`, qui est situé par défaut sous :

```
C:\Program Files (x86)\Altova\RaptorXMLXBRLServer2018\etc\server_config.xml
```

La configuration initiale pour RaptorXML+XBRL Server définit les éléments suivants :

- Un numéro de port de `8087` en tant que le port de serveur.
- Que le serveur écoute uniquement les connexions locales (`localhost`).
- Que le serveur écrit la sortie sous `C:\ProgramData\Altova\RaptorXMLXBRLServer2018\Output\`.

Les autres paramètres par défaut sont affichés dans la liste de `server_config.xml` ci-dessous.

Fichier de configuration de serveur : modifier les paramètres de base, retourner aux paramètres de base

Si vous souhaitez changer les paramètres de base, vous devez éditer le fichier de configuration de serveur, `server_config.xml` (voir la liste ci-dessous), l'enregistrer, puis redémarrer RaptorXML+XBRL Server en tant que service.

Si vous souhaitez recréer le fichier de configuration de serveur original (de manière à ce que le serveur soit à nouveau configuré avec les paramètres de base), exécuter la commande `createconfig`:

```
RaptorXMLXBRL.exe createconfig
```

Lors de l'exécution de cette commande, le fichier de paramètres de base sera recréé et écrasera le fichier `server_config.xml`. La commande `createconfig` est utile si vous souhaitez réinitialiser la configuration de serveur pour les paramètres de base.

Fichier de configuration de serveur : liste et paramètres

Le fichier de configuration de serveur, **`server_config.xml`**, est listé ci-dessous avec les paramètres de base. Les paramètres disponibles s'y trouvant sont expliqués en-dessous de la liste.

 © 2018 Altova GmbH

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>
```

Paramètres

language

Configure la langue des messages de serveur, dans un élément `language` optionnel. La valeur par défaut est `en` (anglais). Les valeurs autorisées sont `en|de|es|fr|ja` (anglais, allemand, espagnol, français et japonais, respectivement). Voir les Commandes de localisation pour une description de la localisation de RaptorXML.

server.unrestricted-filesystem-access

- Une fois définis sur `true` (la valeur par défaut), les fichiers de sortie seront écrits directement dans l'emplacement spécifié par l'utilisateur et dans les scripts Python (écrasant éventuellement les fichiers existants du même nom). Veuillez noter, néanmoins, que les chemins de fichiers locaux ne peuvent pas être utilisés pour accéder aux fichiers depuis une machine à distance par HTTP. Donc, si RaptorXML+XBRL Server est exécuté sur une machine à distance, régler la valeur de cette option sur **`false`**. Le réglage de la valeur sur **`true`** est uniquement viable si le client et le serveur se trouvent sur la même machine et que vous souhaitez écrire les fichiers de sortie sur un répertoire dans cette machine.
- Si définis sur `false`, les fichiers seront écrits dans le répertoire de la tâche dans le répertoire de sortie, et les URI de ces fichiers seront inclus dans le document de résultat. La configuration de la valeur sur `false` fournit un niveau de sécurité, puisque les fichiers peuvent uniquement être écrits sur le disque dans un répertoire de tâche réservé et connu sur le serveur. Les fichiers de sortie de tâche peuvent ensuite être copiés par des moyens autorisés dans d'autres emplacements.

`server.output-root-dir`
Le répertoire dans lequel la sortie de toutes les tâches soumises est enregistrée.

`server.script-root-dir`
Le répertoire dans lequel scripts Python de confiance doivent être enregistrés. L'option `script`, lorsqu'utilisée par le biais de l'interface HTTP, fonctionnera uniquement si des scripts provenant de ce répertoire de confiance sont utilisés. La spécification d'un script Python provenant d'un autre répertoire entraînera un erreur. *Voir Rendre les Scripts Python sûrs'*.

`server.default-script-api-version`
La version API Python par défaut utilisés pour exécuter les scripts Python. Par défaut, la version la plus récente de l'API Python est utilisée. Les valeurs prises en charge actuellement sont 1 et 2.

`server.catalog-file`
L'URL du fichier catalogue XML à utiliser. Par défaut, le fichier catalogue `RootCatalog.xml`, qui est situé dans le dossier `<ProgramFilesFolder>\Altova\RaptorXMLXBRLServer2018\etc`, sera utilisé. Utiliser le paramètre `server.catalog-file` uniquement si vous souhaitez modifier le fichier de catalogue par défaut.

`server.log-file`
Le nom et l'emplacement du fichier de log de serveur. Les événements sur le serveur, comme *Server started/stopped*, sont connectez en continu dans le journal de l'événement du système et affiché dans un visionneur de l'événement de système comme par exemple Windows Event Viewer. Outre l'affichage du visionneur, les messages de journal peuvent aussi être écrits sur le fichier spécifié avec l'option `server.log-file`. Le fichier de journal de serveur contiendra des informations concernant des activités sur le serveur, y compris les erreurs de lancement de serveur, le fichier de configuration utilisé, et les erreurs de licence.

`http.enable`

Une valeur booléenne pour activer ou désactiver HTTP: `true | false`. HTTP peut être activé/désactivé indépendamment ou HTTPS, et les deux peuvent être actifs en parallèle.

`http.environment`
Les environnements interne de raptorxml: `production | development`. L'environnement de Développement sera plus adapté aux besoin des développeurs, permettant un débogage plus simple que lorsque l'environnement de Production est utilisé.

`http.socket-host`
L'interface par le biais duquel RaptorXML+XBRL Server est accédé. Si vous souhaitez que RaptorXML+XBRL Server accepte les connexions depuis les machines à distance, décommenter l'élément et définir son contenu sur : `0.0.0.0`, comme cela : `<http.socket-host>0.0.0.0</http.socket-host>`. Cela héberge le service sur toute interface adressable sur l'appareil de serveur. Dans ce cas, s'assurer que les paramètres du pare-feu sont configurés correctement. Les exceptions de pare-feu entrantes pour les produits Altova doivent être enregistrées comme suit : Altova LicenseServer : port `8088`; Altova RaptorXML+XBRL Server: port `8087`; Altova FlowForce Server : port `8082`.

`http.socket-port`
Le port sur lequel le service est accédé. Le port doit être fixe et connu de manière à ce que les requêtes HTTP puissent être adressées correctement au service.

`http.log-screen`
Si RaptorXML+XBRL Server est lancé avec la commande `RaptorXMLXBRLServer.exe debug`, (voir Lancer le serveur) et si `http.log-screen` est configuré sur `true`, alors l'activité du serveur est affichée dans la console de la ligne de commande. Sinon, l'activité du serveur n'est pas affichée. L'écran journal est affiché en plus de l'écriture dans les fichiers journaux.

`http.access-file`
Le nom et l'emplacement du fichier d'accès HTTP. Le fichier d'accès contient des informations concernant l'activité liée aux accès. Il contient des informations utiles pour résoudre les problèmes de connexion.

`http.error-file`
Le nom et l'emplacement du fichier d'erreur HTTP. Le fichier d'erreur contient des erreurs liées au trafic provenant de et allant vers le serveur. En cas de problèmes de connexion, ce fichier peut fournir des informations utiles concernant la résolution d'erreur.

`http.max_request_body_size`
Cette option spécifie la taille maximum du corps de requête que RaptorXML+XBRL Server accepte. La valeur par défaut est de 100MO. Si la taille d'un corps de requête est plus grand que la valeur spécifiée pour cette option, alors le serveur répond avec `HTTP Error 413: Request entity too large`. La valeur de l'option doit être supérieure à ou égale à zéro. La limite peut être désactivée avec `http.max_request_body_size=0`.

https.enable
Une valeur booléenne pour activer ou désactiver HTTPS: `true | false`. HTTPS peut être activé/désactivé indépendamment de HTTP, et tous les deux peuvent être actives en parallèle. La prise en charge HTTPS est désactivée par défaut et doit être activée en changeant la valeur de ce paramètre sur **true**.

https.socket-host
Prend une valeur string qui est l'adresse hôte sur laquelle les connexions HTTP sont acceptées. Pour accepter les connexions depuis l'hôte local uniquement, définir **localhost** ou **127.0.0.1**. Si vous souhaitez que RaptorXML+XBRL Server accepte les connexions provenant de toutes les machines à distance, définir la valeur sur : `0.0.0.0`, comme ceci : `<http.socket-host>0.0.0.0</http.socket-host>`. Cela héberge le service sur chaque interface adressable de la machine du serveur. Dans ce cas, veuillez vous assurer que les paramètres de pare-feu sont correctement configurés. Les exceptions pare-feu entrantes pour les produits Altova doivent être enregistrés tels que suit : Altova LicenseServer : port `8088`; Altova RaptorXML+XBRL Server : port `8087`; Altova FlowForce Server : port `8082`. Vous pouvez aussi utiliser des adresses IPv6 comme : `'::'`.

https.socket-port
Une valeur de nombre entier qui est le port sur lequel HTTPS est accepté. Le port doit être fixé et connu de manière à ce que les requêtes HTTP peuvent être adressées correctement au service.

https.private-key, https.certificate
Les URI qui sont les chemins, respectivement, à la clé privée du serveur et les fichiers de certificat. Les deux sont nécessaires. Voir Paramètres HTTPS et Configurer l'encodage SSL pour plus d'informations. Sur les appareils Windows, vous pouvez aussi utiliser des chemins Windows.

https.certificate-chain
Un paramètre optionnel. il s'agit d'un URI qui situe le fichier de certificat intermédiaire. Si vous avec deux certificats intermédiaires (primaire et secondaire), puis de les combiner en un fichier tel que décrit dans l'étape 7 de Configurer l'encodage SSL. Voir Paramètres HTTPS et Configurer l'encodage SSL pour plus d'informations.

L'adresse RaptorXML+XBRL Server
L'adresse HTTP du serveur consiste en : socket-host et socket-port :

```
http://{socket-host}:{socket-port}/
```

L'adresse telle que configurée avec la configuration de base sera :

```
http://localhost:8087/
```

Pour changer l'adresse, modifier les paramètres `http.socket-host` et `http.socket-port` dans le fichier de configuration de serveur, `server_config.xml`. Si par exemple, le serveur

 © 2018 Altova GmbH

a une adresse IP de `123.12.123.1`, et que les paramètres de configuration de serveur suivants ont été fait :

```
<http.socket-host>0.0.0.0</http.socket-host>
<http.socket-port>8087</http.socket-port>
```

RaptorXML+XBRL Server peut ensuite être adressé avec :

```
http://123.12.123.1:8087/
```

Note : Une fois que `server_config.xml` aura été modifié, RaptorXML+XBRL Server doit être redémarré pour que les nouvelles valeurs soient appliquées.

Note : Si vous être confronté à des problèmes de connexion à RaptorXML+XBRL Server, l'information dans les fichiers nommé dans `http.access-file` et `http.error-file` peut aider à résoudre des problèmes.

Note : Les messages soumis à RaptorXML+XBRL Server doivent contenir les noms de chemin qui sont valides sur le serveur. Les documents sur le serveur peuvent être accédés soit localement soit à distance (dans le deuxième cas avec des URI HTTP, par exemple).

Paramètres HTTPS

RaptorXML+XBRL Server prend en charge le démarrage non seulement en tant qu'un serveur HTTP, mais aussi en tant qu'un serveur HTTPS. Les deux types de connexion peuvent être active parallèlement.

Activer HTTPS

La prise en charge HTTPS est désactivée par défaut. Pour activer HTTPS, dans le fichier de configuration de serveur, `server_config.xml`, mettre le paramètre **`https.enable`** sur **`true`**. Modifier les différents paramètres HTTPS du fichier de configuration conformément à vos exigences de serveur.

Clé privée et certificat

Vous pouvez obtenir une clé privée et des fichiers de certificat d'une des manières suivantes :

- Depuis une autorité de certificat : suivez les étapes décrites dans la section Configurer le chiffrage Up SSL.
- Créer un certificat auto-signé en utilisant la commande OpenSSL suivante (modifié en conséquence pour votre environnement) :

```
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"
```

Tester la connexion

Un bon moyen de tester votre connexion s'effectue par le biais de l'outil de ligne de commande curl pour transmettre les données avec des URL. Vous pouvez utiliser la commande suivante :

```
curl.exe https://localhost:443/v1/version
```

Si le certificat n'est pas approuvé, utiliser l'option **`-k`**, comme ceci :

```
curl.exe -k https://localhost:443/v1/version
```

La commande suivante exécute l'exemple Python HTTP qui est distribué avec RaptorXML+XBRL Server:

```
python3.exe examples\ServerAPI\python\RunRaptorXML.py --host localhost -p 443
-s
```

 © 2018 Altova GmbH

Configurer le cryptage SSL

Si vous souhaitez encoder vos transferts de données RaptorXML+XBRL Server à l'aide du protocole SSL, vous devez :

- Générer une clé privée SSL et créer un fichier de certificat de clé publique SSL
- Configurer RaptorXML+XBRL Server pour la communication SSL.

Vous trouverez ci-dessous les étapes à suivre.

Cette méthode utilise le kit outil OpenSSL open-source pour gérer l'encodage SSL. Les étapes listées ci-dessous, doivent donc être effectuées sur un ordinateur sur lequel OpenSSL est disponible. Généralement, OpenSSL est pré-installé sur la plupart des distributions Linux et sur des appareils macOS. Il peut donc être installé sur les ordinateurs Windows. Pour télécharger des liens pour les binaires d'installation, voir le Wiki OpenSSL.

Pour générer une clé privée et obtenir un certificat auprès d'une autorité de certification, procéder comme suit :

1. Générer une clé privée

   SSL nécessite l'installation d'une **clé privée** sur RaptorXML+XBRL Server. Cette clé privée sera utilisée pour encoder toutes les données RaptorXML+XBRL Server. Pour créer la clé privée, utiliser la commande OpenSSL suivante :

   ```
   openssl genrsa -out private.key 2048
   ```

   Un fichier nommé `private.key` est créé, qui contient votre clé privée. Veuillez noter où vous avez enregistré le fichier. Il vous faudra la clé privée pour (i) générer la Certificate Signing Request (CSR), et (ii) l'installer sur RaptorXML+XBRL Server.

2. Requête de signature de certificat (CSRs)

   Une Requête de signature de certificat (Certificate Signing Request) (CSR) est envoyée à l'Autorité de certification (CA), comme VeriSign ou Thawte, pour demander un certificat de clé public. La CSR est basée sur votre clé privée et contient des informations concernant votre organisation. Créer une CSR avec la commande OpenSSL suivante (qui fournit le fichier privé-clé, `private.key`, qui a été créé dans l'étape 1, en tant que l'un de ces paramètres) :

   ```
   openssl req -new -nodes -key private.key -out my.csr
   ```

   Au cours de la génération de la CSR vous devrez donner des informations concernant votre organisation, comme celle que vous trouverez ci-dessous. Cette information sera utilisée par l'autorité de certification pour vérifier l'identité de votre entreprise.

   - *Pays*
   - *Lieu* (la ville où votre commerce se situe)
   - *Organisation* (le nom de votre entreprise). Ne pas utiliser des caractères spéciaux ; ils invalideront votre certificat
   - *Nom commun* (le nom DNS de votre serveur). Cela doit correspondre

exactement au nom officiel de votre serveur, c'est à dire, le nom DNS que les applis client utiliseront pour se connecter au serveur
- *Un mot de passe.* Garder cette entrée vierge !

3. Acheter un certificat SSL

   Acheter un certificat SSL auprès d'une autorité de certificat (CA) reconnu, comme VeriSign ou Thawte. Pour le reste de ces instructions, nous suivons la procédure VeriSign. La procédure avec d'autres CA est semblable.

   - Se rendre sur le site web VeriSign.
   - Cliquer sur **Buy SSL Certificates**.
   - Plusieurs types de certificats SSL sont disponibles. En ce qui concerne RaptorXML+XBRL Server, les certificats Secure Site ou Secure Site Pro sont suffisants. Une EV (vérification étendue) n'est pas nécessaire, puisque les utilisateurs ne verront pas de "barre d'adresse verte".
   - Suivez les étapes d'inscription et saisir les informations nécessaires pour passer votre commande.
   - Lorsque vous êtes invité à indiquer la CSR (*créée dans l'étape 2*), copier et coller le contenu du fichier `my.csr` dans le formulaire de commande.
   - Payer le certificat avec votre carte de crédit.

   Permettre du temps pour obtenir un certificat
   L'obtention des certificats de clé publiques auprès de l'autorité de certificat (CA) SSL prend généralement **deux ou trois jours ouvrés**. Veuillez y penser lorsque vous configurez votre RaptorXML+XBRL Server.

4. Recevoir clé publique de la CA

   Votre autorité de certificat achèvera le processus d'inscription en deux ou trois jours ouvrés. Pendant ce temps, vous pourrez recevoir des e-mails ou des appels téléphoniques pour vérifier si vous êtes autorisé à demander un certificat SSL pour votre domaine DNS. Veuillez travailler avec l'autorité pour achever le processus.

   Une fois que le processus d'autorisation et d'inscription est achevé, vous obtiendrez un e-mail contenant la **clé publique** de votre certificat SSL. La clé publique sera en texte clair ou attaché en tant que fichier `.cer`.

5. Enregistrer la clé publique dans le fichier

   Pour pouvoir être utilisé avec RaptorXML+XBRL Server, la clé publique doit être enregistrée dans un fichier `.cer`. Si la clé publique était fournie en tant que texte, copier-coller toutes les lignes depuis

```
--BEGIN CERTIFICATE--
  ...
--END CERTIFICATE--
```

dans un fichier de texte que nous appellerons **`mycertificate.cer`**.

6. Enregistrer les certificats intermédiaires de la CA sous le fichier

   Pour terminer votre certificat SSL, il vous faut deux certificats supplémentaires : le **certificat intermédiaire primaire** et **secondaire**. Votre autorité de certificat (CA) établira la liste du contenu des certificats intermédiaires sur son site web.

   - Les certificats intermédiaires de Verisign : https://knowledge.verisign.com/support/ssl-certificates-support/index?page=content&id=AR657&actp=LIST&viewlocale=en_US
   - Les certificats intermédiaires de Verisign pour son produit Secure Site : https://knowledge.verisign.com/support/ssl-certificates-support/index?page=content&id=AR1735

   Copier-coller les deux certificats intermédiaires (primaire et secondaire) dans des fichiers de texte séparés et les enregistrer sur votre ordinateur.

7. En option, combiner des certificats dans un fichier de certificat de clé publique

   Vous disposez maintenant de trois fichiers de certificat :

   - Clé publique (`mycertificate.cer`)
   - Certificat intermédiaire secondaire
   - Certificat intermédiaire primaire

   Vous pouvez intégrer vos certificats intermédiaires dans votre certificat de clé publique si vous le souhaitez. Vous trouverez ci-dessous la description de la procédure à suivre. (En alternative, vous pouvez utiliser les paramètres du fichier de configuration **`https.certificate-chain`** pour spécifier l'emplacement des certificats intermédiaires.)

   Chacun contient des blocs de texte contenus dans des tirets :

   ```
   --BEGIN CERTIFICATE--
     ...
   --END CERTIFICATE--
   ```

   À présent, copier-coller les trois certificats dans un fichier de manière à ce qu'ils se trouvent dans le bon ordre : l'ordre de la séquence est important : (i) clé publique, (ii) certificat intermédiaire secondaire, (iii) certificat intermédiaire primaire. Veuillez vous assurer qu'il n'y a pas de lignes entre les certificats.

   --BEGIN CERTIFICATE--
   clé publique *provenant de* `mycertificate.cer` *(voir étape 5)*
   --END CERTIFICATE--
   --BEGIN CERTIFICATE--
   certificat intermédiaire secondaire *(voir étape 6)*
   --END CERTIFICATE--
   --BEGIN CERTIFICATE--
   certificat intermédiaire primaire *(voir étape 6)*
   --END CERTIFICATE--

Enregistrer le texte de certificat combiné en résultat sous un fichier nommé **`publickey.cer`** . Il s'agit du *fichier de certificat de clé publique* de votre certificat SSL. Il inclut votre certificat de clé publique et la chaîne complète de confiance sous la forme des certificats intermédiaires qui ont été utilisés par la CA pour signer votre certificat.

 © 2018 Altova GmbH

4.2 Requêtes Client

Une fois que RaptorXML+XBRL Server a été lancé en tant qu'un service, sa fonction peut être accédée par tout client HTTP capable de :

- utiliser les méthodes HTTP `GET`, `PUT`, `POST` et `DELETE`
- définir le champ d'en-tête `Content-Type`

Un client HTTP facile à utiliser
Il existe plusieurs clients web disponibles au téléchargement sur Internet. Le client web facile à utiliser et fiable que nous avons trouvé est RESTClient de Mozilla, qui peut être ajouté en tant qu'un plugin Firefox. Il est facile à utiliser, prend en charge les méthodes HTTP requises par RaptorXML et fournit une coloration syntaxique JSON suffisamment bonne. Si vous n'avez pas d'expérience préalable avec des clients HTTP, vous pouvez essayer RESTClient. Veuillez noter, néanmoins, que l'installation et l'utilisation de RESTClient est entièrement à vos risques.

Une requête client typique consiste en une série d'étapes telles qu'affichées dans le diagramme ci-dessous.

Les points importants caractérisant chaque étape sont notés ci-dessous. Les termes-clés sont gras.

1. Une méthode `POST` HTTP est utilisée pour faire une requête, le corps de la requête est en **format JSON**. La requête peut s'adresser à toute fonction de RaptorXML+XBRL Server. Par exemple, la requête peut être pour une validation, ou pour une transformation XSLT. Les commandes, les arguments et les options utilisées dans la requête sont les mêmes que ceux utilisés sur la ligne de commande. La requête est postée sur : **`http://localhost:8087/v1/queue`**, en assumant que **`localhost:8087`** est l'adresse de RaptorXML+XBRL Server (l'adresse initiale du serveur). Comme une requête est Ce type de requête est appelé une **tâche RaptorXML+XBRL Server**.

2. Si la requête est reçue et acceptée pour le traitement par RaptorXML+XBRL Server, un **document de résultat** contenant les résultats de l'action de serveur sera créé une fois que le job a été traité. L'**URI de ce document de résultat** (le Result-Doc-URI dans le diagramme ci-dessus), est retourné sur le client. Veuillez noter que l'URI sera retournée immédiatement après que la tâche ait été acceptée (queued) pour le traitement et même si le traitement n'a pas été achevé.

3. Le client envoie une requête pour le document de résultat (en utilisant l'URI de document de résultat) dans une méthode `GET` sur le serveur. Si le traitement de la tâche n'a pas encore démarré ou n'a pas encore été achevé au moment de la réception de la requête, le serveur retourne un statut de *Running*. La requête `GET` doit être répétée jusqu'à ce que le traitement de la tâche ait été achevé et que le document de résultat ait été créé.

4. RaptorXML+XBRL Server retourne le document de résultat dans le format JSON. Le document de résultat peut contenir les **URI des documents d'erreurs ou de sortie** produits par le traitement de la requête d'origine par RaptorXML+XBRL Server. Les journaux d'erreur sont retournés, par exemple, si une validation a retourné des erreurs. Des documents de sortie primaire, comme le résultat d'une transformation XSLT sont retournés si une tâche de production de sortie est achevée avec succès.

5. Le client envoie les URI des documents de sortie reçu dans l'étape 4 par le biais d'une méthode `GET` HTTP vers le serveur. Chaque requête est envoyée dans une méthode `GET` séparée.

6. RaptorXML+XBRL Server retourne les documents requis en réponse aux requêtes `GET` effectués dans l'Étape 5.

7. Le client peut supprimer des documents non souhaités sur le serveur qui ont été générés en tant qu'un résultat d'une requête de tâche. Cela s'effectue en soumettant, dans une méthode `DELETE` HTTP, l'URI du document de résultat en question. Tous les fichiers sur disque liés à cette tâche sont supprimés. Cela comprend le fichier de document de résultat, tout fichier temporaire et les fichiers de document d'erreur et de sortie. Cette étape est utile pour libérer de l'espace sur le disque dur du serveur.

Les détails de chaque étape sont décrits dans les sous-sections de cette section.

 © 2018 Altova GmbH

Initier les tâches avec POST

Cette section :

- Envoyer la requête
- Syntaxe JSON pour les requêtes POST
- Charger les fichiers avec la requête POST
- Charger des archives ZIP

Envoyer la requête

Une tâche RaptorXML+XBRL Server est initiée avec la méthode `POST` HTTP

Méthode HTTP	URI	Type de contenu	Corps
`POST`	`http://localhost:8087/v1/queue/`	`application/json`	JSON

Veuillez noter les points suivants :

- L'URI ci-dessus a une adresse de serveur qui utilise les paramètres de la configuration initiale.
- L'URI a un chemin `/v1/queue/`, qui doit être présent dans l'URI. Il peut être considéré comme un dossier abstrait dans la mémoire dans laquelle la tâche est placée.
- Le numéro de version correct `/vN` est celui que le serveur retourne (et pas nécessairement celui dans cette documentation). Le nombre que le serveur retourne est le numéro de version de l'interface HTTP actuelle. Les numéros de version précédents indiquent des versions plus anciennes de l'interface HTTP, qui sont toujours pris en charge pour la rétrocompatibilité.
- L'en-tête doit contenir le champ : `Content-Type: application/json`. Néanmoins, si vous souhaitez charger des fichiers dans le corps de la requête `POST`, alors le type de contenu de l'en-tête du message doit être configuré sur `multipart/form-data` (par ex. `Content-Type: multipart/form-data`). Voir la section Charger les fichiers avec la requête POST pour plus de détails.
- Le corps de la requête doit être en format JSON.
- Les fichiers à traiter doivent se trouver sur le serveur. Donc les fichiers doivent soit être copiés sur le serveur avant d'effectuer une requête, ou être chargés avec la requête POST. Dans ce cas, le type de contenu de l'en-tête de message doit être configuré sur `multipart/form-data`. Voir la section Charger les fichiers avec la requête POST ci-dessous pour plus de détails.

Pour vérifier la bonne formation d'un fichier XML, la requête dans le format JSON ressemblerait à l'exemple suivant :

```
{
   "command": "wfxml", "args": [ "file:///c:/Test/Report.xml" ]
}
```

Les commandes valides, et leurs arguments et options, sont telles que documenté dans la section de Ligne de commande.

Syntaxe JSON pour les requêtes POST

```
{
  "command": "Command-Name",
  "options": {"opt1": "opt1-value", "opt2": "opt2-value"},
  "args"   : ["file:///c:/filename1", "file:///c:/filename2"]
}
```

- Tout le texte noir est fixe et doit être inclus. Cela contient toutes les accolades, double guillemets, virgules, double-point et crochets. L'espace blanc peut être normalisé.
- Le texte en italique bleu sont des espaces réservés et représente les noms de commande, les options et valeurs d'option et les valeurs d'argument. Veuillez vous référer à la section de Ligne de commande pour une description des commandes.
- Les clés `command` et `args` sont obligatoires. La clé `options` est optionnelle. Certaines clés `options` ont des valeurs par défaut ; donc parmi ces options, seules ceux dont les valeurs par défaut doivent être changées doivent être spécifiées.
- Tous les strings doivent être inclus dans des doubles guillemets. Les valeurs booléennes et les nombres n'ont pas de guillemets. Donc : `{"error-limit": "unlimited"}` et `{"error-limit": 1}` est l'usage correct.
- Veuillez noter que les URI (et non pas les chemins de fichier) sont recommandés et qu'ils utilisent des barres obliques vers l'avant. Les chemins de fichier Windows, si utilisés, prennent des barres obliques vers l'arrière. De plus, les barres obliques vers l'arrière de chemin de fichier de Windows doivent être échappées dans JSON (avec des échappements de barres obliques vers l'arrière : `"c:\\dir\\filename"`). Veuillez noter que les URI de fichier et les chemins de fichier sont des strings et doivent donc se trouver entre guillemets.

Voici un exemple avec des options. Veuillez noter que certaines options (comme `input` ou `xslt-version`) prennent une valeur d'option directe, alors que d'autres (comme `param`) prennent une paire de valeur-clé en tant que leur valeur et nécessitent donc une syntaxe différente.

```
{
   "command": "xslt",
   "args": [
           "file:///C:/Work/Test.xslt"
           ],
   "options": {
              "input": "file:///C:/Work/Test.xml",
              "xslt-version": 1,
              "param": {
                       "key": "myTestParam",
```

 © 2018 Altova GmbH

```
                        "value": "SomeParamValue"
                        },
                "output": "file:///C:/temp/out2.xml"
                }
}
```

L'exemple ci-dessous montre un troisième type d'option : celle d'un table de valeurs (comme pour l'option `xsd` ci-dessous). Dans ce cas, la syntaxe à utiliser est celle d'un tableau JSON.

```
{
   "command": "xsi",
   "args": [
           "file:///C:/Work/Test.xml"
           ],
   "options": {
               "xsd" : ["file:///C:/Work/File1.xsd", "file:///C:/Work/File2.xsd"]
               }
}
```

Charger les fichiers avec la requête POST

Les fichiers à traiter peuvent être téléchargés dans le corps de la requête `POST`. Dans ce cas, la requête `POST` doit être effectuée comme suit.

En-tête de requête

Dans l'en-tête de requête, définir `Content-Type: multipart/form-data` et spécifier tout string arbitraire en tant que la limite. Voici une en-tête d'exemple :

```
Content-Type: multipart/form-data; boundary=---PartBoundary
```

L'objectif de la limite est de définir des limites des différentes parties données de forme dans le corps de la requête (*voir ci-dessous*).

Corps de la requête : partie message

Le corps de la requête détient les parties données de forme suivantes, séparées par le string de limite spécifié dans l'en-tête de requête (*voir ci-dessus*) :

- *Parties données de forme obligatoires :* **`msg`**, qui spécifie l'action de traitement requise, et **`args`**, qui contient les fichiers à charger en tant que l'argument de la commande spécifiée dans la partie de donnée de forme `msg`. *Voir la liste ci-dessous.*
- *Partie données de forme optionnelle :* un nom **`additional-files`** de données de forme **`additional-files`**, qui contient les fichiers référencés depuis les fichiers dans les parties de données de forme `msg` ou `args`. Des parties de données de forme supplémentaires nommées selon une option de la commande peut aussi contenir des

fichiers à charger.

Note : tous les fichiers chargés sont créés dans un seul répertoire virtuel.

Voir Exemple-1 (avec des légendes) : Valider XML pour une explication détaillée du code, et Exemple-2 : Utiliser un catalogue pour trouver le schéma.

Tester avec CURL

Vous pouvez utiliser une application de transfert de données tiers comme CURL (http://curl.haxx.se/) pour tester la requête POST. CURL propose une option de trace précieuse qui génère et liste les limites des parties des requêtes. Cela vous épargnera la tâche de créer manuellement les limites de partie. La section Tester avec CURL décrit comment utiliser CURL .

Charger des archives ZIP

Des archives ZIP peuvent aussi être chargés et des fichiers se trouvant dans un ZIP peuvent être référencés en utilisant le scheme `additional-files`. Par exemple :

```
additional-files:///mybigarchive.zip%7Czip/biginstance.xml
```

Note : La partie `|zip/` doit être échappée avec un URI comme `%7Czip/` pour pouvoir se conformer à l'URI RFC puisque le symbole `|` n'est pas autorisé directement. L'utilisation des motifs glob (`*` et `?`) est aussi autorisée. Donc vous pouvez utiliser quelque chose du genre suivant pour valider tous les fichiers XML dans l'archive ZIP :

```
{"command": "xsi", "args": ["additional-files:///mybigarchive.zip%7Czip/
*.xml"], "options": {...}}
```

Voir Exemple-3: Utiliser les archives ZIP pour une liste du code d'exemple.

 © 2018 Altova GmbH

Exemple-1 (avec légendes): valider XML

Ci-dessous, vous trouverez une liste du corps d'une requête `POST`. Elle contient des légendes numérotées qui sont expliquées ci-dessous. La commande soumise dans la requête de liste aurait l'équivalent CLI suivant :

```
raptorxmlxbrl xsi First.xml Second.xml --xsd=Demo.xsd
```

La requête est destinée à la validation de deux fichiers XML conformément à un schéma. Le corps de la requête ressemble à l'exemple suivant, en partant du principe que `---PartBoundary` a été spécifié dans l'en-tête en tant que le string de limite (*voir [En-tête de requête](#) ci-dessus*).

```
-----PartBoundary                                                    1
Content-Disposition: form-data; name="msg"
Content-Type: application/json

{"command": "xsi", "options": {} , "args": []}                       2

-----PartBoundary                                                    3
Content-Disposition: attachment; filename="First.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="Second.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. Le nom des limites de partie de données de forme principales sont déclarés dans l'en-tête de requête. Le séparateur de limite de partie doit être un string unique qui ne se produira nulle part ailleurs dans les documents imbriqués. Il est préfixé avec deux tirets et est utilisé pour séparer les différentes parties. La première partie de données de forme dans cet exemple est **`msg`**. Veuillez noter que le type de contenu est `application/json`.
2. Voici la syntaxe standard pour les requêtes POST HTTP. Si `args` contient une référence à un fichier et si des fichiers supplémentaires sont chargés, les deux sets de files seront passés au serveur.
3. Le premier membre du tableau `args` est un attache de fichier appelé `First.xml`.
4. Le texte du fichier `First.xml`. Il contient une référence au schéma appelé `Demo.xsd`, qui sera aussi chargé—dans la partie de donnée de forme `additional-files`.
5. Le second membre du tableau `args` est un attache appelé `Second.xml`.
6. Le texte du fichier `Second.xml`. Lui aussi contient une référence au schéma `Demo.xsd`. *Voir légende 7.*
7. Les premiers fichiers supplémentaires contiennent les métadonnées d'attache `Demo.xsd`.
8. Le texte du fichier `Demo.xsd`.
9. La fin de la partie des fichiers supplémentaires `Demo.xsd`, et la partie de données de forme `additional-files`. Veuillez noter que le dernier séparateur de limite de partie est aussi bien préfixé que postfixé avec deux tirets.

 © 2018 Altova GmbH

Exemple-2: utiliser un catalogue pour trouver le schéma

Dans cet exemple, un fichier de catalogue est utilisé pour trouver le schéma XML qui est référencé par les fichiers XML à valider.

```
-----PartBoundary
Content-Disposition: form-data; name="msg"
Content-Type: application/json

{"command": "xsi", "args": ["additional-files:///First.xml", "additional-
files:///Second.xml"], "options": {"user-catalog": "additional-files:///
catalog.xml"}}

-----PartBoundary
Content-Disposition: attachment; filename="First.xml"; name="additional-files"
Content-Type: application/octet-stream

<?xml version="1.0" encoding="UTF-8"?>
<test xsi:noNamespaceSchemaLocation="http://example.com/Demo.xsd"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">42</test>

-----PartBoundary
Content-Disposition: attachment; filename="Second.xml"; name="additional-files"
Content-Type: application/octet-stream

<?xml version="1.0" encoding="UTF-8"?>
<test xsi:noNamespaceSchemaLocation="http://example.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://example.com/Demo.xsd" uri="additional-
files:///Demo.xsd"/>
</catalog>

-----PartBoundary--
```

Exemple-3 : utiliser des archives ZIP

Des archives ZIP peuvent aussi être chargées et des fichiers dans un ZIP peuvent être référencés en utilisant le scheme **additional-files**. Par exemple :

```
additional-files:///mybigarchive.zip%7Czip/biginstance.xml
```

Note : La partie **|zip/** doit être échappée par URI en tant que **%7Czip/** pour pouvoir se conformer à l'URI RFC puisque le symbole **|** n'est pas directement autorisé. L'utilisation de motifs glob (***** et **?**) est aussi autorisée. Vous pouvez donc utiliser quelque chose comme cela pour valider tous les fichiers XML dans l'archive ZIP : `{"command": "xsi", "args": ["additional-files:///mybigarchive.zip%7Czip/*.xml"], "options": {…}}`

Note : `'Content-Disposition: form-data'` est aussi valide, outre `'Content-Disposition: attachment'`. Puisque plusieurs outils génèrent `form-data` en tant que disposition de contenu, la valeur `form-data` est acceptée comme étant valide.

Exemple : Valider tous les fichiers XML dans un archive ZIP

Dans cet exemple, on part du principe que toutes les références de schéma sont des chemins relatifs et que tous les schémas sont contenus dans le 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

Binary content of Demo.zip archive

-----PartBoundary--
```

Exemple : Valider les fichiers XML dans un archive ZIP contenant des références à des schémas externes

Dans cet exemple, les fichiers XML contenus dans un archive ZIP sont validés en utilisant des références à un schéma externe, qui est fourni dans un second archive ZIP.

```
-----PartBoundary
Content-Disposition: form-data; name="msg"
Content-Type: application/json
```

 © 2018 Altova GmbH

```
{"command": "xsi", "args": ["additional-files:///Instances.zip%7Czip/*.xml"],
"options": {"user-catalog": "additional-files:///Schemas.zip%7Czip/
catalog.xml"}}

-----PartBoundary
Content-Disposition: attachment; filename="Instances.zip"; name="additional-
files"
Content-Type: application/octet-stream

Binary content of Instances.zip archive

-----PartBoundary
Content-Disposition: attachment; filename="Schemas.zip"; name="additional-
files"
Content-Type: application/octet-stream

Binary content of Schemas.zip archive

-----PartBoundary--
```

Tester avec CURL

L'application tierce CURL (http://curl.haxx.se/) est un outil de ligne de commande que vous pouvez utiliser pour tester la requête `POST`. CURL propose une option de trace très utile qui génère et liste les limites de partie des requêtes qui peut être utilisé directement dans vos requêtes ou en tant qu'une référence.

Ci-dessous, vous trouverez un scénario test dans lequel un fichier XML est validé par rapport à un schéma XML. Considérant que :

- les commandes ci-dessous sont exécutées depuis le dossier dans lequel se trouvent les fichiers à soumettre à la validation ; (cela nous permet d'écrire des chemins relatifs simples vers ces fichiers). Si vous avez installé l'application XMLSpy d'Altova, les fichiers utilisés dans cet exemple peuvent être trouvés dans le dossier `Examples` de l'application, qui se trouve par défaut sous : `C:\Users\<username>\Documents\Altova\XMLSpy2018\Examples`
- RaptorXML+XBRL Server est exécuté localement sur le port 8087

Pour plus d'informations concernant les options de ligne de commande CURL, voir l'aide CURL.

Appeler CURL avec la commande de validation sur

```
[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
```

Note : dans powershell, si des guillemets apparaissent dans des guillemets, il faudra utiliser un type différent de guillemets (simple/double).

```
[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"}
```

Utiliser l'URL de "result" pour obtenir le résultat

```
[input]
  \path\to\curl.exe http://localhost:8087/v1/results/058F9E97-CB95-43EF-AC0A-
  496CD3AC43A3

[output]
```

 © 2018 Altova GmbH

```
{"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":{}}]}
```


L'option trace de CURL

CURL dispose d'une option trace (**`--trace-ascii`**), qui trace le trafic HTTP envoyé de et vers le serveur. L'option est très utile puisqu'elle liste les limites de partie qui sont requises pour lancer des tâches avec POST. Vous pouvez utiliser l'information contenue dans trace, soit directement soit en tant que référence, pour créer des limites de partie. La liste ci-dessous montre la trace obtenue en exécutant la commande donnée ci-dessus.

Liste de trace

```
== 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
```

Note : Veuillez noter, dans la liste supérieure `'Content-Disposition: form-data'` est aussi valide, outre `'Content-Disposition: attachment'`.

Appeler CURL avec la commande de bonne formation sur 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
```

 © 2018 Altova GmbH

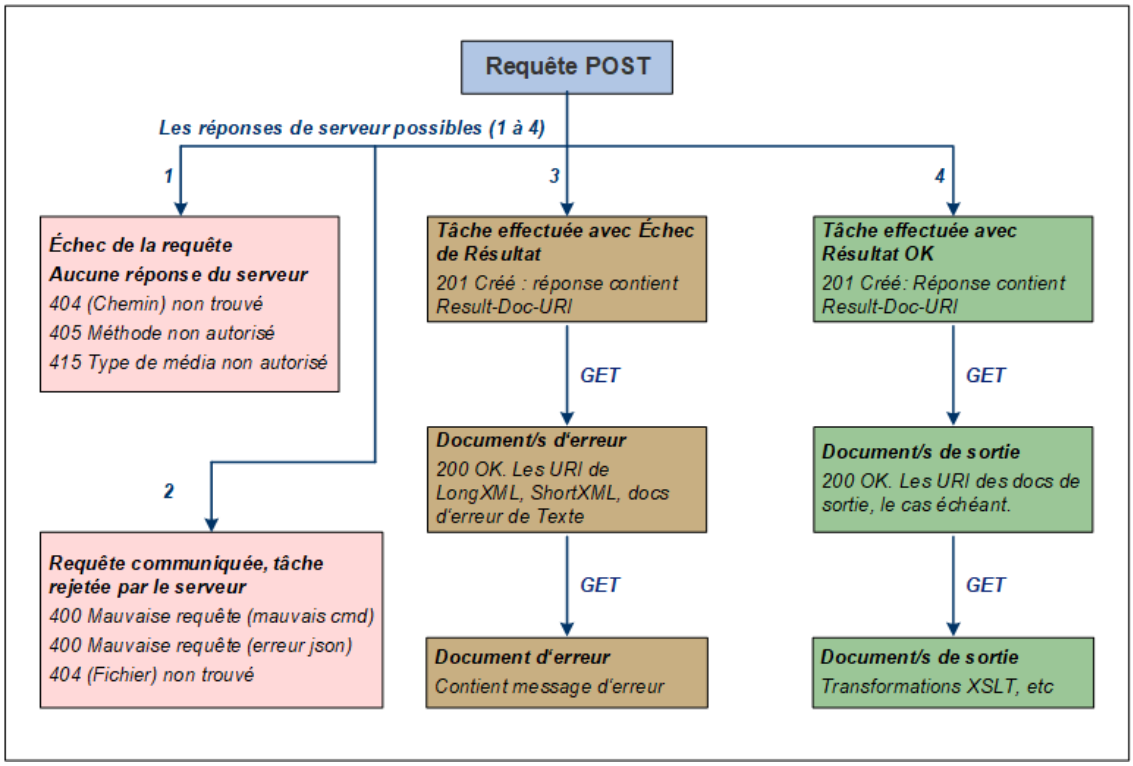
Réponse de serveur à une requête POST

Cette section :

- Aperçu des réponses de serveur possibles
- Réponse : échec de la requête, aucune réponse de la serveur
- Réponse : requête communiquée, mais la tâche est rejetée par le serveur
- Réponse : tâche exécutée (avec un résultat positif ou négatif)

Lorsqu'une requête `POST` est effectuée avec succès auprès du serveur, la tâche est placée dans la file d'attente du serveur. Un message `201 Created` et un URI de document de résultat sont retournés. La tâche sera traitée au plus tôt. Entre-temps, si le document de résultat est requis, un message `"status": "Running"` est retourné si la tâche n'a pas été terminée ; le client devra essayer à nouveau plus tard. Un état `Dispatched` indique que la tâche se trouve dans la file d'attente du serveur mais n'a pas encore été démarrée.

Le résultat de la tâche (par exemple, une requête de validation) peut être négative (échec de la validation) ou positive (validation réussie). Dans les deux cas, un message `201 Created` est retourné et un document de résultat est généré. Il est également possible que la requête `POST` n'ait pas été communiquée au serveur (*Échec de la requête*), ou bien, la requête a été communiquée mais la tâche a été rejetée par le serveur (*Requête communiquée, mais tâche rejetée*). Les résultats possibles variés sont affichés dans le diagramme ci-dessous.



Les résultats possibles de la requête `POST` du client sont les suivantes :

Échec de la requête, aucune réponse du serveur

En cas d'échec de requête sur le serveur, les erreurs les plus communes sont les suivantes :

Message	Explication
404 Not Found	Le chemin correct est : `http://localhost:8087/v1/queue/`
405 Method Not Allowed	La méthode spécifiée est invalide pour cette ressource. Utiliser la méthode `POST`.
415 Unsupported Media Type	L'en-tête de message doit être `Content-Type:application/json`.

Requête communiquée, mais la tâche est rejetée par le serveur

Lorsque les requêtes sont effectuées avec succès sur le serveur, celui-ci peut les rejeter pour les raisons suivantes :

Message	Explication
400 Bad Request (*bad cmd*)	La commande RaptorXML est incorrecte.
400 Bad Request (*json error*)	Le corps de requête contient une erreur de syntaxe JSON.
404 File Not Found	Contrôler syntaxe d'URI de fichier (ou chemin de fichier) de tous les fichiers nommés dans la commande.

Tâche exécutée (avec un résultat positif ou négatif)

Lorsqu'une tâche (par exemple, une tâche de validation) est exécutée, son résultat peut être positif (*OK*) ou négatif (*Échec*). Par exemple, le résultat d'une tâche de validation est positif (*OK*) lorsque le document à valider est valide, négatif (*Échec*) si le document est invalide.

Dans les deux cas, la tâche est exécutée, mais avec des résultats différents. Un message `201 Created` est retourné dans les deux cas dès que la tâche a été placée avec succès dans la file d'attente. De même, dans les deux cas, un URI de document de résultat est retourné vers le client HTTP qui a effectué la requête. (Le document de résultat lui-même peut ne pas avoir été créé si le traitement de la tâche n'a pas encore été lancé ou terminé.) Une fois que le document de résultat a été créé, il peut être amené avec une requête `GET` HTTP. Outre le document de résultat, d'autres documents peuvent être aussi générés, comme dans les exemples ci-dessous :

- *Tâche exécutée avec le résultat 'Échec':* un journal d'erreurs est créé en trois formats : texte, XML long et XML court. Les URI de ces trois documents sont envoyés dans le document de résultat (qui est en format JSON). Les URI peuvent être utilisés dans une requête `GET` HTTP pour aller chercher les documents d'erreur.
- *Tâche exécutée avec le résultat 'OK' :* la tâche est traitée avec succès et les documents de sortie, comme la sortie produite par une transformation XSLT, sont créés. Si des fichiers de sortie ont été générés, leurs URI sont envoyés dans le document de résultat de format JSON. Les URI peuvent ensuite être utilisés dans une requête `GET` HTTP pour

 © 2018 Altova GmbH

aller chercher les documents de sortie. Veuillez noter que toutes les tâches n'auront pas forcément de fichiers de sortie ; par exemple, une tâche de validation. Ainsi, une tâche peut se terminer avec un état '*OK*', mais il peut s'être produit des avertissements et/ou des messages qui ont été rédigés des fichiers d'erreur. Dans ce cas, les URI de fichier d'erreur sont aussi envoyés dans le document de résultat (c'est à dire, en plus des documents de sortie).

Voir Obtenir le document de résultat et Obtenir les documents d'erreur/sortie pour une description de ces documents et de leur accès.

Obtenir le document de résultat

Cette section :

- L'URI de document de résultat
- Récupérer le document de résultat
  - Document de résultat contenant les URI des documents d'erreur
  - Document de résultat contenant les URI des documents de sortie
  - Document de résultat ne contenant aucun URI
- Accéder aux documents d'erreur et de sortie listés dans le document de résultat

L'URI de document de résultat

Un document de résultat sera créé à chaque fois que qu'une tâche est créée, quel que soit le résultat d'une tâche (par exemple, une validation) est positive (document valide) ou négative (document invalide). Dans les deux cas, un message `201 Created` est retourné. Ce message sera en format JSON et contiendra un URI relatif du document de résultat. Le fragment JSON ressemblera à l'exemple ci-dessous :

```
{
  "result": "/v1/results/E6C4262D-8ADB-49CB-8693-990DF79EABEB",
  "jobid": "E6C4262D-8ADB-49CB-8693-990DF79EABEB"
}
```

L'objet `result` contient l'URI relatif du document de résultat. L'URI est relatif à l'adresse de serveur. Par exemple, si l'adresse de serveur est `http://localhost:8087/` (l'adresse de configuration initiale), alors l'URI agrandi du document de résultat spécifié dans la liste ci-dessus sera :

```
http://localhost:8087/v1/results/E6C4262D-8ADB-49CB-8693-990DF79EABEB
```

Note : le numéro de version correct `/vN` est celui que le serveur retourne (et n'est pas nécessairement celui dans cette documentation). Le numéro que le serveur retourne est le numéro de version de l'interface HTTP actuel. Les numéros de version précédents indiquent des versions plus anciens de l'interface HTTP, qui, néanmoins, sont toujours pris en charge pour la rétrocompatibilité.

Récupérer le document de résultat

Pour obtenir le document de résultat, soumettre l'URI agrandi du document (*voir ci-dessus*), dans une requête `GET` HTTP. Le document de résultat est retourné et pourrait être un des types génériques décrits ci-dessous.

Note : Lorsqu'une tâche est placée avec succès dans la file d'attente du serveur, le serveur retourne l'URI du document de résultat. Si le client requête le résultat avant que la tâche ait été lancée (elle se trouve toujours dans la file d'attente), un message `"status": "Dispatched"` sera retourné. Si la tâche a été lancée mais n'a pas été terminée (par exemple parce qu'il s'agit d'une tâche volumineuse), un message `"status": "Running"` sera retourné. Dans ces deux situations, le client devrait attendre un certain temps avant de faire une requête fraîche pour le document de résultat.

 © 2018 Altova GmbH

Note : Tous les documents d'exemple ci-dessous ont un accès de client limité. Donc les documents d'erreur, les documents de message et les documents de sortie sont tous présumés être enregistrés dans le répertoire de tâche pertinent sur le serveur. Les URI les concernant dans le document de résultat sont donc tous des URI relatifs. Aucun n'est un URI de fichier (qui serait le type d'URI généré dans les cas d'accès client non limités). En ce qui concerne les détails de ces URI, voir la section Obtenir des documents Erreur/Message/Sortie.

Document de résultat contenant les URI des documents d'erreur

Si la tâche requise s'est terminée avec un état *Échec*, alors la tâche a retourné un résultat négatif. Par exemple, une tâche de validation a retourné un résultat de document invalide. Les erreurs rencontrées lors de l'exécution de la tâche sont stockées dans le journal d'erreur, créé en trois formats de fichiers : (i) texte, (ii) XML long (journal d'erreur détaillé), et (iii) XML court (journal d'erreur moins détaillé). Voir la liste JSON ci-dessous.


```
{
  "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"
      }
    }
  ]
}
```


Veuillez noter :

- Les tâches ont des sous-tâches.
- Les erreurs au niveau de sous-tâche se propagent jusqu'à la tâche de niveau supérieur. L'état de la tâche de niveau supérieur sera seulement *OK* si toutes ses sous-tâches ont un état *OK*.
- Chaque tâche sous sous-tâche a son propre journal d'erreur.

- Les journaux d'erreur comprennent des journaux d'avertissement. Donc, même si une tâche se termine avec un état *OK*, il pourrait avoir des URI de fichiers d'erreur.
- Les URI des fichiers d'erreur sont relatifs à l'adresse de serveur (*voir ci-dessus*).

Document de résultat contenant les URI des documents de sortie

Si la tâche requise s'est terminée avec un état *OK*, la tâche a retourné un résultat positif. Par exemple, une tâche de validation a retourné un résultat de document valide. Si la tâche a produit un document de sortie, par exemple, le résultat d'une transformation XSLT, alors l'URI du document de sortie est retourné. Voir la liste JSON ci-dessous.

```
{
  "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":
      {
      }
    }
  ]
}
```

Veuillez noter :

- Le fichier de sortie dans le dossier `output` de la tâche. Vous pouvez utiliser son URI relatif pour accéder au fichier.
- Les URI des fichiers de sortie sont relatifs à l'adresse de serveur (*voir ci-dessus*).
- L'item `output-mapping` mappe le document de sortie dans le répertoire de tâche sur le serveur dans l'emplacement de fichier spécifié par le client dans la requête de tâche. Veuillez noter que seuls les documents de sortie spécifiés par le client dans la requête de tâche ont un mappage ; les fichiers liés aux tâches générés par le serveur (comme des fichiers d'erreur) n'ont pas de mappage.
- En alternative, il est possible de récupérer tous les documents de résultat générés pour une tâche spécifique en tant qu'un archive zip utilisant l'URL "`/v1/results/`*`JOBID`*`/output/zip`". Cette fonction n'est pas disponible dans le mode de filesystem non limité. Veuillez noter que l'archive zip contiendra des noms de fichier endommagé, qui nécessite d'être mappé à nouveau sur le nom réel en utilisant l'objet `output-mapping`.

Document de résultat ne contenant aucun URI
Si la tâche requise s'est terminée avec un état *OK*, la tâche a retourné un résultat positif. Par exemple, une tâche de validation a retourné un résultat de document valide. Certaines tâches, comme une validation ou un test bien formé, ne produit aucun document de sortie. Si une tâche de ce type se termine avec un état des *OK*, alors le document de résultat n'aura ni l'URI d'un document de sortie ni l'URI d'un journal d'erreur. Voir la liste de JSON ci-dessous.

```
{
  "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":
      {
      }
    }
  ]
}
```

Veuillez noter :

- Aussi bien les composants de sortie et d'erreur de la sous-tâche dans la liste ci-dessus sont vides.
- Une tâche pourrait se terminer avec un état *OK* mais contient tout de même encore des avertissements ou d'autres messages, qui sont inscrits dans les fichiers d'erreur. Dans ces cas, le document de résultat contiendra les URI de fichiers d'erreur même si la tâche s'est terminée avec l'état *OK*.

Accéder aux documents d'erreur et de sortie listés dans le document de résultat

Les documents d'erreur et de sortie peuvent être accédés avec des requêtes `GET` HTTP. Ils sont décrits dans la section suivante, Obtenir des documents Erreur/Sortie.

Obtenir les documents Erreur/Message/Sortie

Un document de résultat peut contenir des URI de fichier ou des URI relatifs des documents d'erreur, des documents de message (comme des journaux), et/ou des documents de sortie. (Il existe certaines situations dans laquelle un document de résultat peut ne pas contenir d'URI.) Les différents types d'URI sont décrits ci-dessous.

Pour accéder ces documents via HTTP, procéder comme suit :

1. Agrandir l'URI relatif du fichier dans le document de résultat à son URI absolu
2. Utiliser l'URI agrandi dans une requête GET HTTP pour accéder au fichier

Les URI (dans le document de résultat) de documents d'erreur/de message/de sortie

Le document de résultat contient des URI d'erreur, de message et/ou des documents de sortie. Les documents d'erreur et de message sont des documents liés aux tâches qui sont générés par le serveur ; ils sont toujours enregistrés dans le répertoire de la tâche sur le serveur. Les documents de sortie (comme la sortie des transformations XSLT) peuvent être enregistrés dans un des emplacements suivants :

- Dans n'importe quel emplacement de fichier accessible au serveur. Pour que les fichiers de sortie soient enregistrés dans un emplacement quelconque, le serveur doit être configuré pour permettre au client un accès non limité (le paramètre par défaut).
- Dans le répertoire de tâche sur le serveur. Le serveur est configuré pour limiter l'accès au client.

Si un client spécifie qu'un fichier de sortie soit créé, l'emplacement dans lequel le fichier de sortie est enregistré sera déterminé par l'option `server.unrestricted-filesystem-access` du fichier de configuration du serveur.

- Si l'accès n'est pas limité, le fichier sera enregistré sur l'emplacement spécifié par le client et l'URI retourné pour le document sera un URI de fichier.
- Si l'accès est limité, le fichier sera enregistré sur le répertoire de tâche et son URI sera un URI relatif. De plus, il y aura un mappage de cet URI relatif ver l'URL de fichier spécifiée par le client. (*Voir la liste du Document de résultat contenant les URI de documents de sortie.*)

En résumé, vous rencontrerez donc les types d'URI suivants :

URI de fichier des documents d'erreur/message
Ces documents sont enregistrés dans le répertoire de tâche du serveur. Les URI de fichier ont la forme suivante :

```
file:///<output-root-dir>/JOBID/message.doc
```

URI de fichier des documents de sortie
Ces documents sont enregistrés dans n'importe quel endroit. Les URI de fichier ont la forme suivante :

```
file:///<path-to-file>/output.doc
```

URI HTTP des documents d'erreur/message/sortie
Ces documents sont enregistrés dans le répertoire de tâche du serveur. Les URI sont relatifs à

l'adresse de serveur et doivent être agrandis à l'URI HTTP complet. Le chemin relatif aura cette forme :

`/vN/results/JOBID/error/error.txt`	pour les documents d'erreur
`/vN/results/JOBID/output/verbose.log`	pour les documents de message
`/vN/results/JOBID/output/1`	pour les documents de sortie

Dans le cas des documents de sortie, des mappages de sortie sont donnés (*voir liste d'exemple*). Ces mappages mappent chaque URI de document de sortie dans le document de résultat dans la requête client.

Agrandir l'URI relatif

Agrandir l'URI relatif dans le document de résultat sur un URI HTTP absolu en préfixant l'URI relatif avec l'adresse de serveur. Par exemple, si l'adresse de serveur est :

`http://localhost:8087/` (l'adresse de configuration initiale)

et que l'URI relatif d'un fichier d'erreur d'un fichier d'erreur dans le document de résultat est :

```
/v1/results/20008201-219F-4790-BB59-C091C276FED2/error/error.txt
```

alors l'adresse absolue agrandie sera

```
http://localhost:8087/v1/results/20008201-219F-4790-BB59-C091C276FED2/error/
error.txt
```

Pour plus d'informations liées à ce sujet, voir les sections : Configurer le serveur et Obtenir le document de résultat.

Utiliser une requête `GET` HTTP pour accéder au fichier

Utiliser l'URI agrandie dans une requête `GET` HTTP pour obtenir le fichier requis. RaptorXML+XBRL Server retourne le document requis.

Libérer des ressources de serveur après le traitement

RaptorXML+XBRL Server garde le fichier de document de résultat, des fichiers temporaires et des fichiers de document de sortie liés à une tâche traitée sur disque dur. Ces fichiers peuvent être supprimés d'une des deux manières :

- En fournissant le document de résultat d'URI avec la méthode `DELETE` HTTP. Cela supprime tous les fichiers liés à la tâche indiquée par l'URI de document de résultat soumis, y compris des documents d'erreur et de sortie.
- Suppression manuelle des fichiers individuels sur le serveur par un administrateur.

La structure de l'URI à utiliser avec la méthode `DELETE` HTTP tel que montré ci-dessous. Notez que l'URI complet consiste en l'adresse de serveur plus l'URI relatif du document de résultat.

Méthode HTTP	URI
DELETE	http://localhost:8087/v1/result/D405A84A-AB96-482A-96E7-4399885FAB0F

Pour situer le répertoire d'une tâche sur un disque, construire l'URI comme suit :

[<server.output-root-dir> *voir fichier de configuration de serveur*] + [jobid]

Note : Puisqu'un grand nombre de fichiers de document d'erreur et de sortie, il est recommandé de surveiller l'utilisation du disque dur et de programmer les suppressions conformément à votre environnement et vos exigences.

 © 2018 Altova GmbH

Chapitre 5

API Python et .NET

5 API Python et .NET

RaptorXML+XBRL Server est équipé des API suivantes, qui proposent un accès aux fonctions RaptorXML+XBRL Server :

- API Python
- API .NET Framework

Enregistrer une machine client avec Altova LicenseServer

Pour exécuter un pack API sur un appareil client, cette machine doit être mise sous licence en tant que client RaptorXML+XBRL Server. La mise sous licence consiste en deux étapes :

1. Enregistrer la machine en tant qu'un client de RaptorXML+XBRL Server avec Altova LicenseServer
2. Attribuer à RaptorXML+XBRL Server une licence de LicenseServer sur cet appareil.

Si vous projetez d'utiliser le pack d'API provenant d'un appareil particulier, deux situations probables apparaissent :

- Si la machine client exécute d'ores et déjà une installation sous licence de RaptorXML +XBRL Server, alors le pack API peut être exécuté sans qu'il vous soit nécessaire de prendre des mesures supplémentaires. Cela est dû au fait que la machine est déjà mise sous licence pour exécuter RaptorXML+XBRL Server. Par conséquent, l'utilisation du pack d'API sur cet appareil est couverte par la licence attribuée à RaptorXML+XBRL Server sur cet appareil.
- Si RaptorXML+XBRL Server n'est pas installé sur cet appareil client, et que vous ne souhaitez pas installer RaptorXML+XBRL Server sur cet appareil pour une raison quelconque. Dans ce cas, vous pouvez toujours enregistrer l'appareil en tant qu'un client RaptorXML+XBRL Server et l'attribuer à une licence RaptorXML+XBRL Server. Suivre les étapes suivantes :

Pour enregistrer un appareil (sur lequel RaptorXML+XBRL Server n'est pas installé) en tant qu'un client RaptorXML+XBRL Server, utiliser l'application de ligne de commande **`registerlicense.exe`**, qui se trouve dans le dossier **`bin`** de l'application :

Windows	`Program Files\Altova\RaptorXMLXBRLServer2018\bin`
Linux	`/opt/Altova/RaptorXMLXBRLServer2018/bin`
Mac	`/usr/local/Altova/RaptorXMLXBRLServer2018/bin`

Dans la ligne de commande, exécuter la commande :

```
registerlicense <LicenseServer>
```

où **`<LicenseServer>`** est l'adresse IP ou le nom d'hôte de l'appareil de LicenseServer.

Cette commande enregistrera l'appareil en tant qu'un client RaptorXML+XBRL Server auprès de Altova LicenseServer. Pour plus d'informations concernant comment attribuer une licence RaptorXML+XBRL Server à l'appareil et pour plus d'informations concernant la mise sous licence,

 © 2018 Altova GmbH

consulter la documentation LicenseServer Altova.

Déployer sur Linux

Pour déployer l'application **`registerlicense`** avec votre pack Python wheel, les bibliothèques partagées qui sont listées ci-dessous doivent être présentes dans un répertoire **`lib`** frère. Les librairies partagées peuvent être copiées depuis votre dossier d'installation Raptor :

`/opt/Altova/RaptorXMLServer2017/lib`

- `libcrypto.so.1.0.0`
- `libssl.so.1.0.0`
- `libstdc++.so.6`
- `libtbb.so.2`

5.1 API Python

L'API Python de RaptorXML+XBRL Server permet aux données contenues dans des documents XML, des documents de Schéma XML, des documents d'instance XBRL, et des documents de taxonomie XBRL d'être accédés et récupérés par le biais d'API Python pour XML, XSD et XBRL. Les données dans les documents source à traiter et le type de traitement de ces données sont spécifiées dans un script Python passé à RaptorXML+XBRL Server.

Les API Python

Les API Python (pour XML, XSD et XBRL) fournissent un accès à la méta-information, à l'information structurelle et aux données contenues dans des documents XML, XSD, et d'instance et de taxonomie XBRL. Ainsi, les scripts Python peuvent être créés permettant d'utiliser les API pour accéder et traiter l'information du document. Par exemple, un script Python peut être passé sur RaptorXML+XBRL Server qui écrit des données depuis un document XML ou un document d'instance XBRL pour une base de données ou sur un fichier CSV.

Des scripts d'exemple pour les API Python de Raptor sont disponibles sous : https://github.com/altova

Les API Python sont décrits dans leurs références API :

- Référence API Python v1
- Référence API Python v2

Note : **Python API v1** de Raptor **est obsolète**. Veuillez utiliser Python API v2.

Package RaptorXML+XBRL Server pour Python

Dans votre installation de RaptorXML+XBRL Server, vous trouverez aussi un package Python en format wheel. Vous pouvez utiliser la commande `pip` de Python pour installer ce package en tant que module de votre installation Python. Une fois que le module RaptorXMLXBRL a été installé, vous pouvez utiliser les fonctions du module dans votre code. De cette manière, les fonctions de RaptorXMLXBRL peuvent être utilisées aisément dans tout programme Python que vous écrivez, avec d'autres bibliothèques Python tierces comme des librairies de graphique.

Pour plus d'information concernant l'utilisation du package Python de RaptorXML+XBRL Server, voir la section RaptorXML+XBRL Server en tant que package Python.

Scripts Python

Un script Python utilisé par l'utilisateur est soumis avec le paramètre `--script` des commandes suivantes :

- `valxml-withxsd (xsi)`
- `valxsd (xsd)`
- `valxbrltaxonomy (dts)`
- `valxbrl (xbrl)`

Ces commandes invoquant des scripts Python peuvent être utilisées aussi bien sur l'Interface de

 © 2018 Altova GmbH

ligne de commande (CLI) qu'avec l'interface HTTP. L'utilisation des scripts Python avec les API Python de RaptorXML+XBRL Server est décrite sous : https://github.com/altova.

Rendre les scripts Python sûrs

Lorsqu'un script Python est spécifié dans une commande via HTTP pour RaptorXML+XBRL Server, le script fonctionnera uniquement s'il est situé dans le répertoire de confiance. Le script est exécuté depuis le répertoire de confiance. La spécification d'un script Python provenant d'un autre répertoire quelconque résultera en une erreur. Le répertoire de confiance est spécifié dans le paramètre `server.script-root-dir` du fichier de configuration de serveur, et un répertoire de confiance **doit** être spécifié si vous souhaitez utiliser des scripts Python. Veuillez vous assurer que tous les scripts Python à utiliser soient sauvegardés dans ce répertoire.

Bien que toutes les sorties générées par le serveur pour les requêtes de tâche HTTP soient écrites dans le répertoire de sortie de tâche (qui est un sous-répertoire de `output-root-directory`), cette limitation ne s'applique pas aux scripts Python, qui peuvent écrire dans n'importe quel emplacement. L'administrateur de serveur doit revoir les scripts Python dans le répertoire de confiance en ce qui concerne des problèmes de vulnérabilité potentiels.

Versions d'API Python

RaptorXML+XBRL Server prend en charge plusieurs versions d'API Python. Toute version d'API Python précédente est aussi prise en charge par la version actuelle de RaptorXML+XBRL Server. La version d'API Python est sélectionnée par le flag de ligne de commande **`--script-api-version=MAJOR_VERSION`**. Le défaut de l'argument `MAJOR_VERSION` est toujours la version actuelle. Un nouveau API Python `MAJOR_VERSION` de RaptorXML+XBRL Server est introduit lorsque des changements incompatibles ou des améliorations importantes sont introduites. Les utilisateurs de l'API ne doivent pas mettre à niveau leurs scripts existants lorsqu'une nouvelle version majeure est publiée.

Il est recommandé d' :

- utiliser le flag `--script-api-version=MAJOR_VERSION` pour invoquer des scripts utilitaires depuis la ligne de commande RaptorXML+XBRL Server (ou l'API Web). Cela vous garantit que les scripts fonctionnent toujours comme prévu après les mises à jour de RaptorXML+XBRL Server, même si une nouvelle `MAJOR_VERSION` d'API a été publiée.
- utiliser la dernière version de l'API pour des nouveaux projets, même si des versions précédentes seront prises en charge par des publications de RaptorXML+XBRL Server dans le futur.

Les versions d'API Python listées ci-dessous sont actuellement disponibles. La documentation des différents API est disponible en ligne dans les emplacements indiqués ci-dessous.

Fichiers d'exemple

Pour consulter des exemples de scripts qui utilisent des API Python de Raptor, se rendre sous https://github.com/altova.

API Python version 1

Introduit avec la version v2014 de RaptorXML+XBRL Server

Flag de ligne de commande :	**`--script-api-version=1`**
Documentation :	Python API Version 1 Reference

Voici l'API Python de RaptorXML+XBRL Server original. Il couvre la prise en charge pour accéder au modèle interne de RaptorXML+XBRL Server pour :

- XML 1.0 et XML 1.1 (module API **`xml`**)
- Schéma XML 1.0 et Schéma XML 1.1 (module API **`xsd`**)
- XBRL 2.1 (module API **`xbrl`**)

L'API peut être utilisé par le biais de fonctions de callback qui sont mises en place dans un fichier de script Python.

- `on_xsi_valid`
- `on_xsd_valid`

 © 2018 Altova GmbH

- `on_dts_valid`
- `on_xbrl_valid`

Un script est spécifié avec l'option `--script` sur la ligne de commande. Les fonctions de callback sont invoquées uniquement si la validation réussit. Les détails concernant les fonctions de rappel et l'API sont décrits dans la référence "Python API version 1" de RaptorXML+XBRL Server.

Note : Python API v1 de Raptor **est obsolète**. Veuillez utiliser Python API v2.

API Python version 2

Introduit avec la version v2015r3 de RaptorXML+XBRL Server. La dernière version API est `2.4.1`.

Flag de ligne de commande :	**`--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`**	*v 2015r3* *v 2015r4* *v 2016* *v 2016r2* *v 2017* *v 2018*
Documentation :	Python API Version 2 Reference	

Cette version API introduit plus de 300 nouvelles classes et réorganise les modules depuis l'API Python version 1 de RaptorXML+XBRL Server de manière à ce que des informations utilisées fréquemment (par exemple des données PSVI) peuvent être accédées plus simplement et des API liés sont regroupés logiquement (par exemple, `xbrl.taxonomy`, `xbrl.formula`, `xbrl.table`). Dans cette version, les fonctions de callback sont invoquées non seulement si la validation réussit, mais aussi si la validation échoue. Pour illustrer ce comportement, le nom des fonctions de callback a été changé en :

- `on_xsi_finished`
- `on_xsd_finished`
- `on_dts_finished`
- `on_xbrl_finished`

Pour activer la modularisation, RaptorXML+XBRL Server prend maintenant en charge plusieurs options `--script`. Les callbacks implémentés dans ces fichiers script Python sont exécutés dans l'ordre spécifié dans la ligne de commande.

RaptorXML+XBRL Server en tant que package Python

Depuis RaptorXML+XBRL Server 2017, l'API Python est disponible en tant que package wheel Python natif pour **Python 3.5**. Le package wheel Python peut être installé en tant que module d'extension dans votre distribution Python 3.5 préférée (par exemple, depuis python.org). Certaines distributions Python 3 (par exemple, provenant de jupyter.org, anaconda.org et SciPy.org) comprennent une palette étendue de modules d'extension pour les big data, les mathématiques, la science, l'ingénierie et les graphiques. Ces modules sont maintenant disponibles pour RaptorXML+XBRL Server sans qu'il soit nécessaire de générer ces modules spécifiquement pour RaptorXML+XBRL Server. Sinon, le package wheel fonctionne de la même manière que l'application `RaptorXMLXBRL-python.exe` qui est contenue dans RaptorXML+XBRL Server.

Note : le package wheel Python est un module d'extension natif Python et doit correspondre avec la version Python de votre installation.

Note: le package wheel Python ne contient pas le Python API v1.

L'information nécessaire pour installer correctement le package RaptorXML+XBRL Server est indiquée dans les sections ci-dessous :

- Nom du fichier wheel
- Emplacement du fichier wheel
- Installer un wheel avec pip
- Le fichier root catalog
- Le fichier de config JSON

Pour plus d'informations concernant l'utilisation de l'API Python de RaptorXML+XBRL Server, voir référence API Python et exemples. Voir aussi les scripts d'exemple qui utilisent l'API Python de Raptor sous https://github.com/altova.

Nom du fichier wheel

Les fichiers wheel sont nommés conformément au motif suivant :

```
raptorxmlserver-{version}(-{build tag})?-{python tag}-{abi tag}-
{platform tag}.whl
```

Exemples :

```
raptorxmlserver-2.4.0-cp35-cp35m-win_amd64.whl
raptorxmlserver-2.4.1-cp35-cp35m-win_amd64.whl
```

Emplacement du fichier wheel

Un fichier wheel est contenu avec votre installation de RaptorXML+XBRL Server. Il se trouve dans le dossier `bin` de l'application :

Windows	`Program Files\Altova\RaptorXMLXBRLServer2018\bin`
Linux	`/opt/Altova/RaptorXMLXBRLServer2018/bin`
Mac	`/usr/local/Altova/RaptorXMLXBRLServer2018/bin`

 © 2018 Altova GmbH

Installer un wheel avec pip

Pour installer le package RaptorXML+XBRL Server en tant que module Python, utiliser la commande **pip** :

```
pip install <wheel-file>.whl
python -m pip install <wheel-file>.whl
```

Si vous avez installé Python 3.5 ou plus depuis python.org, alors `pip` sera déjà installé. Sinon, vous devrez tout d'abord installer **pip**. Pour plus d'informations, voir https://docs.python.org/3/installing/.

Le fichier de catalogue root

Le module RaptorXMLXBRL pour Python doit pouvoir situer **RootCatalog.xml**, le fichier de catalogue root qui est stocké dans votre dossier d'installation RaptorXML+XBRL Server. La raison est que le module RaptorXMLXBRL peut utiliser le catalogue pour situer correctement les différentes ressources, comme des schémas et d'autres spécifications, que le module référence pour pouvoir effectuer des fonctions comme des validations et des transformations. Le module RaptorXMLXBRL situera automatiquement **RootCatalog.xml** si l'emplacement du catalogue n'a pas été modifié après l'installation de RaptorXML+XBRL Server.

Au cas où vous déplacez ou modifiez votre environnement RaptorXML+XBRL Server, ou si vous déplacez **RootCatalog.xml** depuis son emplacement d'origine, vous pouvez spécifier l'emplacement du catalogue par le moyen des variables d'environnement et le fichier de configuration JSON du module RaptorXMLXBRL. Voir la liste ci-dessous pour consulter les moyens se trouvant à votre disposition. Le module RaptorXMLXBRL détermine l'emplacement de **RootCatalog.xml** en consultant les ressources suivantes dans l'ordre donné.

1	Variable d'environnement `ALTOVA_RAPTORXML_PYTHON_CATALOGPATH`	Créer avec une valeur qui est le chemin vers **RootCatalog.xml**
2	Registre HKLM : `SOFTWARE\Altova\RaptorXMLXBRLServer\Installation_v2018_x64\Setup\CatalogPath`	La clé de registre est ajoutée par l'installateur RaptorXML+XBRL Server. La valeur est le chemin vers **RootCatalog.xml**. *Uniquement Windows*
3	Emplacement : `/opt/Altova/RaptorXMLXBRLServer2018/etc/RootCatalog.xml`	*Uniquement Linux*
4	Emplacement : `/usr/local/Altova/RaptorXMLXBRLServer2018/etc/RootCatalog.xml`	*Uniquement Mac*
5	Variable d'environnement `ALTOVA_RAPTORXML_PYTHON_CONFIG`	Créer avec une valeur qui est le chemin vers le fichier de config JSON.

6	Emplacement : `.altova/raptorxml-python.config`	Le fichier de config JSON dans le répertoire de travail actuel
7	Emplacement : `~/.config/altova/raptorxml-python.config`	Le fichier de config JSON dans le répertoire home de l'utilisateur
8	Emplacement : `/etc/altova/altova/raptorxml-python.config`	Le fichier de config JSON. *Uniquement Linux et Mac*

Le fichier de config JSON

Vous pouvez créer un fichier de config JSON pour le module RaptorXMLXBRLServer. Ce fichier sera utilisé par les options 5 à 8 dans la table ci-dessus pour situer le fichier de catalogue root. Le fichier de config JSON doit contenir un map avec une clé `"CatalogPath"` qui a une valeur qui est le chemin vers le fichier de catalogue root.

Liste du fichier config JSON

```
{
    "CatalogPath": "/path/to/RootCatalog.xml"
}
```

 © 2018 Altova GmbH

5.2 API .NET Framework

L'**API .NET Framework** de RaptorXML+XBRL Server vous permet d'intégrer RaptorXML+XBRL Server dans des applications écrites en C# et d'autres langages .NET.

Elle est mise en place en tant qu'un assembly .NET et place RaptorXML+XBRL Server directement à l'intérieur d'une application ou d'un mécanisme d'extension basé sur un .NET-framework comme VSTO (Visual Studio Tools for Office). L'API propose un accès granulaire pour valider les documents et pour requêter leur modèle de données interne depuis RaptorXML+XBRL Server. Le mécanisme utilisé est semblable à celui utilisé pour l'API Python de RaptorXML+XBRL Server.

Références et ressources

- *Documentation API :* vous trouverez la documentation API .NET Framework de RaptorXML+XBRL Server la plus récente sous http://manual.altova.com/RaptorXML/dotnetapiv2/html/index.html.
- *Code d'exemple :* le code d'exemple est hébergé sous https://github.com/altova/RaptorXML-Examples.

Chapitre 6

Interface Java

6 Interface Java

L'API de RaptorXML peut être accédé depuis le code Java. Pour accéder à RaptorXML+XBRL Server depuis le code Java, les bibliothèques listées ci-dessous doivent résider dans le classpath. Ces bibliothèques sont installées dans le dossier `bin` du dossier d'installation.

- `RaptorXMLServer.jar`: La bibliothèque qui communique avec le serveur RaptorXML en utilisant les requêtes HTTP
- `RaptorXMLServer_JavaDoc.zip`: un fichier Javadoc contenant la documentation d'aide pour l'API Java

Note : Pour pouvoir utiliser l'API Java, le fichier Jar doit se trouver sur le Java Classpath. Vous pouvez copier le fichier Jar dans un des emplacements si cela convient mieux à votre configuration de projet que de le référencer depuis l'emplacement installé.

Aperçu de l'interface

L'API Java est contenu dans le package `com.altova.raptorxml`. La classe RaptorXML fournit une méthode de point d'entrée appelée `getFactory()`, qui retourne des objets `RaptorXMLFactory`. Donc, une instance `RaptorXMLFactory` peut être créée avec l'appel : `RaptorXML.getFactory()`.

L'interface `RaptorXMLFactory` propose des méthodes pour obtenir des objets de moteur pour une validation et un traitement ultérieur (comme une transformation XSLT).

Note : La méthode `getFactory` retourne l'objet factory respectif conformément à l'édition RaptorXML installée.

L'interface publique de `RaptorXMLFactory` est décrite dans la liste suivante :

```
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);
}
```

Pour plus de détails, voir les descriptions de `RaptorXMLFactory` et les interfaces Java respective.

 © 2018 Altova GmbH

Voir aussi le Projet d'exemple Java.

6.1 Projet d'exemple Java

Le code Java listé ci-dessous montre comment accéder aux fonctions de base. Il est structuré dans les différentes parties suivantes :

- Situer le dossier d'exemple, et créer une instance d'objet COM RaptorXML
- Valider un fichier XML
- Effectuer une transformation XSLT, retourner le résultat en un string
- Traiter un document XQuery, retourner le résultat en un string
- Exécuter le projet

Cette fonction de base est incluse dans les fichiers dans le dossier `examples/API` du dossier d'application RaptorXML+XBRL Server.

```
public class RunRaptorXML
{
    // Locate samples installed with the product
    // (will be two levels higher from examples/API/Java)
    // REMARK: You might need to modify this path
       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() );
       }
```

 © 2018 Altova GmbH

```
        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 RaptorXML pour Java

Ci-dessous, vous trouverez un sommaire des interfaces Java de l'API RaptorXML. Les descriptions détaillées sont indiquées dans les sections respectives.

- `RaptorXMLFactory`
  Crée une nouvelle instance d'objet COM de RaptorXML COM par le biais d'un appel natif, et fournit un accès aux moteurs RaptorXML.

- `XMLValidator`
  Interface pour le moteur XMLValidator.

- `XSLT`
  Interface pour les moteurs XSLT.

- `XQuery`
  Interface pour les moteurs XQuery.

- `XBRL`
  Interface pour les moteurs XBRL.

- `RaptorXMLException`
  Interface pour la méthode `RaptorXMLException`.

RaptorXMLFactory

public interface RaptorXMLFactory

Utiliser `RaptorXMLFactory()` pour créer une nouvelle instance d'objet COM de RaptorXML. Cela permet l'accès aux moteurs RaptorXML. La relation entre `RaptorXMLFactory` et l'objet COM de RaptorXML est one-to-one. Cela signifie que les appels suivants à la fonction `get<ENGINENAME>()` retourneront des interfaces pour la même instance de moteur. Les méthodes de l'interface sont décrites ci-dessous, et regroupées par fonction. Les énumérations de l'interface **`RaptorXMLFactory`** sont décrites dans une section séparée.

Moteurs

Des méthodes pour appeler les moteurs respectifs.

- getXBRL

  public XBRL getXBRL
  Récupère le moteur XBRL et retourne une nouvelle instance `XBRL` de ce `RaptorXMLFactory`.

- getXMLValidator

  public XMLValidator getXMLValidator
  Récupère le moteur XMLValidator et retourne une nouvelle instance `XMLValidator` de ce `RaptorXMLFactory`.

- getXQuery

  public XQuery getXQuery
  Récupère le moteur XQuery et retourne une nouvelle instance `XQuery` de ce `RaptorXMLFactory`.

- getXSLT

  public XSLT getXSLT
  Récupère le moteur XSLT et retourne une nouvelle instance `XSLT` de ce `RaptorXMLFactory`.

Erreurs et avertissements

Des méthodes pour définir les paramètres pour les erreurs et les avertissements.

- setErrorFormat

  public void setErrorFormat(*ENUMErrorFormat* format)
  Définit le formant d'erreur RaptorXML sur un des littéraux `ENUMErrorFormat` (Text, ShortXML, LongXML).

 © 2018 Altova GmbH

- setErrorLimit

  public void setErrorLimit(*int* limit)
  Définit la limite d'erreur de validation de RaptorXML. Le paramètre `limit` est de type `int` (Java), `uint` (COM/.NET). Il spécifie le nombre d'erreurs à rapporter avant que l'exécution soit stoppée. Utiliser `-1` pour que `limit` ne soit pas limité (c'est à dire que toutes les erreurs seront rapportées). La valeur par défaut est `100`.

- setReportOptionalWarnings

  public void setReportOptionalWarnings(*boolean* report)
  Active/désactive le rapport des avertissements. Une valeur de `true` active les avertissements ; `false` les désactive.

Catalogues et ressources globales

Des méthodes qui fournissent les emplacements des fichiers de catalogue à utiliser.

- setGlobalCatalog

  public void setGlobalCatalog(*String* catalog)
  Définit l'emplacement, en tant qu'une URL, du fichier de catalogue (point-entrée) principal. Le string fourni doit être une URL absolue qui donne l'emplacement exact du fichier de catalogue principal à utiliser.

- setUserCatalog

  public void setUserCatalog(*String* catalog)
  Définit l'emplacement, en tant qu'une URL, du fichier de catalogue d'utilisateur personnalisé. Le string fourni doit être une URL absolue qui donne l'emplacement exact du fichier de catalogue d'utilisateur personnalisé.

- setGlobalResourceConfig

  public void setGlobalResourceConfig(*String* config)
  Définit la configuration active de la ressource globale. Le paramètre `config` est de type `String`, et spécifie le nom de la configuration utilisée par la ressource globale active.

- setGlobalResourceFile

  public void setGlobalResourceFile(*String* file)
  Définit l'emplacement, en tant qu'une URL, du fichier XML de Ressources globales. Le string fourni doit être une URL absolue qui donne l'emplacement exact du fichier XML de Ressources globales.

Paramètres de serveur HTTP

Des méthodes pour définir le nom de serveur HTTP et le port et pour spécifier le fichier de configuration du serveur.

- setServerFile

  public void setServerFile(*String* file)

  Définit l'emplacement du fichier de configuration du serveur HTTP relative à l'adresse de serveur HTTP. Si une erreur se produit, une **RaptorXMLException** est soulevée (*détails d'interface Java*). Le paramètre d'entrée est un string qui donne l'adresse du fichier de configuration de serveur HTTP relatif à l'adresse de serveur.

- setServerName

  public void setServerName(*String* name)

  Définit le nom du serveur HTTP. Si une erreur se produit, une **RaptorXMLException** est soulevée (*détails d'interface Java*). Le paramètre d'entrée est un string qui donne au nom du serveur HTTP.

- setServerPort

  public void setServerPort(*int* port)

  Définit le port sur le serveur HTTP par le biais duquel le service est accédé. Le port doit être fixé et connu de manière à ce que les requêtes HTTP puissent être adressées correctement au service. Si une erreur se produit, une **RaptorXMLException** est soulevée (*détails d'interface Java*). Le paramètre d'entrée est un entier qui spécifie le port d'accès sur le serveur HTTP.

Information de produit

Des méthodes pour récupérer l'information concernant le produit installé.

- getProductName

  public string getProductName()

  Retourne le nom du produit en tant que string. *Exemple :* Pour `Altova RaptorXML+XBRL Server 2018r2sp1(x64)`, retourne `Altova RaptorXML+XBRL Server`. Si une erreur se produit, une **RaptorXMLException** est soulevée (*détails d'interface Java*).

- getProductNameAndVersion

  public string getProductNameAndVersion()

  Retourne la version de pack de service du produit en tant qu'un entier. *Exemple :* pour `Altova RaptorXML+XBRL Server 2018r2sp1(x64)`, retourne `Altova RaptorXML+XBRL Server 2018r2sp1(x64)`. Si une erreur se produit, une **RaptorXMLException** est soulevée (*détails d'interface Java*).

 © 2018 Altova GmbH

- getMajorVersion

  `public int getMajorVersion()`
  Retourne la version majeure du produit en tant qu'un entier. *Exemple :* Pour `Altova RaptorXML+XBRL Server 2014r2sp1(x64)`, retourne `16` (la différence entre la version majeure (`2014`) et l'année initiale `1998`). Si une erreur se produit, a **`RaptorXMLException`** est soulevée (*Java*).

- getMinorVersion

  `public int getMinorVersion()`
  Retourne la version mineure du produit en tant qu'un entier. *Exemple :* Pour `Altova RaptorXML+XBRL Server 2018r2sp1(x64)`, retourne `2` (depuis la version mineure `r2`). Si une erreur se produit, a **`RaptorXMLException`** est soulevée (*détails d'interface Java*).

- getServicePackVersion

  `public int getServicePackVersion()`
  Retourne la version du pack de service du produit en tant qu'un entier. *Exemple :* pour `RaptorXML+XBRL Server 2018r2sp1(x64)`, retourne `1` (du pack de service du numéro de version `sp1`). Si une erreur se produit, une **`RaptorXMLException`** est soulevée (*détails d'interface Java*).

- is64Bit

  `public boolean is64Bit()`
  Vérifie si l'application est un exécutable 64-bit. Retourne une booléenne `true` si l'application est 64 bit, `false` si ce n'est pas le cas. *Exemple :* Pour `Altova RaptorXML+XBRL Server 2018r2sp1(x64)`, retourne `true`. Si une erreur se produit, une **`RaptorXMLException`** est soulevée (*détails d'interface Java*).

- getAPIMajorVersion

  `public int getAPIMajorVersion()`
  Retourne la version majeure de l'API en tant qu'un entier. La version majeure de l'API peut différer de la version majeure du produit si l'API est connectée à un autre serveur.

- getAPIMinorVersion

  `public int getAPIMinorVersion()`
  Retourne la version mineure de l'API en tant qu'un entier. La version mineure de l'API peut différer de la version mineure du produit si l'API est connectée à un autre serveur.

- getAPIServicePackVersion

  `public int getAPIServicePackVersion()`
  Retourne la version du pack de service de l'API en tant qu'un entier. La version du pack de

service de l'API peut différer de la version du pack de service du produit si l'API est connectée à un autre serveur.

© 2018 Altova GmbH

XMLValidator

public interface XMLValidator

Valide le document XML fourni, le document de schéma ou le document DTD. La validation de document XML peut être effectuée avec des DTD ou des Schémas XML internes ou externes. Contrôler également la bonne formation des documents XML, DTD et de Schéma XML. Les méthodes de l'interface sont décrites ci-dessous et sont regroupées par fonctions.

Traitement

Des méthodes pour spécifier les paramètres d'une validation et pour récupérer l'information concernant la validation.

- extractAvroSchema

  public boolean extractAvroSchema(*String* outputPath)
  Extrait un schéma Avro depuis un fichier binaire. Le paramètre `outputPath` est une URL absolue qui spécifie l'emplacement de sortie. Le résultat est `true` en cas de succès, `false` en cas d'échec. Si une erreur se produit, une **`RaptorXMLException`** est soulevée (*détails d'interface Java*). Utiliser `getLastErrorMessage` (***dans Java***), ou `LastErrorMessage` (***dans COM/.NET***) pour accéder aux informations supplémentaires.

- isValid(ENUM type)

  public boolean isValid(*ENUMValidationType* type)
  Retourne le résultat de validation du document XML, le document de schéma, ou le document DTD. Le type de document à valider est spécifié par le paramètre `type`, qui prend un littéral `ENUMValidationType` (Java, COM/.NET) en tant que cette valeur. Le résultat est `true` en cas de succès, `false` en cas d'échec. Si une erreur se produit, une **`RaptorXMLException`** est soulevée (*détails d'interface Java*). Utiliser `getLastErrorMessage` (***dans Java***), ou `LastErrorMessage` (***dans COM/.NET***) pour accéder des informations supplémentaires.

- isValid

  public boolean isValid()
  Retourne le résultat de validation du document soumis. Le résultat est `true` en cas de succès, `false` en cas d'échec.

- isWellFormed(ENUM type)

  public boolean isWellFormed(*ENUMWellformedCheckType* type)
  Retourne le résultat du contrôle du document XML ou du document DTD de bonne formation. Le type de document à contrôler est spécifié avec le paramètre `type`, qui prend un littéral `ENUMWellformedCheckType` (Java, COM/.NET) en tant que sa valeur. Le résultat est `true` en cas de succès, `false` en cas d'échec. Si une erreur se produit, une **`RaptorXMLException`** est soulevée (*détails d'interface Java*). Utiliser `getLastErrorMessage` (***dans Java***), ou `LastErrorMessage` (***dans COM/.NET***) pour accéder des informations supplémentaires.

- isWellFormed

  public boolean isWellFormed()
  Retourne le résultat du contrôle du document XML ou du document DTD de bonne formation. Le résultat est `true` en cas de succès, `false` en cas d'échec.

- getLastErrorMessage

  public String getLastErrorMessage()
  Récupère un string qui est le dernier message d'erreur provenant du moteur RaptorXML.

- setAssessmentMode

  public void setAssessmentMode(*ENUMAssessmentMode* mode)
  Définit le mode d'évaluation de la validation XML (`Strict/Lax`), qui est donnée par un littéral `ENUMAssessmentMode` (Java, COM/.NET).

- setPythonScriptFile

  public void setPythonScriptFile(*String* file)
  Définit l'emplacement du fichier de script Python. Le string fourni doit être une URL absolue qui donne l'emplacement exact du fichier Python.

- setStreaming

  public void setStreaming(*boolean* support)
  Active la validation de streaming. Dans le mode streaming, les données qui sont stockées en mémoire sont minimisées et le traitement est plus rapide. Une valeur de `true` active le streaming; `false` le désactive. Le défaut est `true`.

Fichiers d'entrée

Des méthodes pour spécifier les fichiers d'entrée d'une commande de validation (XML, Schéma XML et DTD).

- setAvroSchemaFileName

  public void setAvroSchemaFileName(*String* filePath)
  Définit l'emplacement, en tant qu'URL, du Schéma Avro externe à utiliser. Le string fournit doit être une URL absolue qui donne l'emplacement exact du fichier de schéma Avro.

- setAvroSchemaFromText

  public void setAvroSchemaFromText(*String* schemaText)
  Apporte le contenu du document de Schéma Avro à utiliser. Le string fourni est le contenu du document de Schéma Avro à utiliser.

 © 2018 Altova GmbH

- setInputFileName

```
public void setInputFileName(String filePath)
```

Définit l'emplacement, en tant qu'une URL, des données XML d'entrées à traiter. Le string fourni doit être une URL absolue qui donne l'emplacement du fichier d'entrée.

- setInputFileCollection

```
public void setInputFileCollection(Collection<?> fileCollection)
```

Fournit la collection de fichiers XML qui sera utilisée en tant que données d'entrée. Les fichiers sont identifiés par leurs URL. Une collection de strings est soumise ; chaque string est l'URL absolue d'un fichier XML d'entrée.

- setInputFromText

```
public void setInputFromText(String inputXMLText)
```

Apporte le contenu du document XML à traiter. Le string fourni est le contenu du document XML à traiter.

- setInputTextCollection

```
public void setInputTextCollection(Collection<?> stringCollection)
```

Apporte le contenu de plusieurs fichiers XML qui seront utilisés en tant que données d'entrée. Une collection de strings, chacun d'entre eux est le contenu d'un fichier XML d'entrée.

- setInputXMLFileCollection (OBSOLETE. Utiliser setInputFileCollection à la place.)

```
public void setInputXMLFileCollection(Collection<?> fileCollection)
```

Apporte la collection des fichiers XML qui seront utilisés en tant que données d'entrée. Les fichiers sont identifiés par leurs URL. Une collection de strings, chacun d'entre eux est l'URL absolue d'un fichier XML d'entrée.

- setInputXMLFileName (OBSOLÈTE. Utiliser setInputFileName à la place.)

```
public void setInputXMLFileName(String xmlFile)
```

Définit l'emplacement, en tant qu'une URL, du document XML à traiter. Le string fourni doit être une URL absolue qui donne l'emplacement exact du fichier XML.

- setInputXMLFromText OBSOLÈTE. Utiliser setInputFromText à la place.)

```
public void setInputXMLFromText(String inputXMLText)
```

Apporte le contenu du document XML à traiter. Le string fourni est le contenu du document XML à traiter.

- setInputXMLTextCollectionOBSOLÈTE. Utiliser setInputTextCollection à la place.)

```
public void setInputXMLTextCollection(Collection<?> stringCollection)
```
Apporte le contenu de plusieurs fichiers XML qui seront utilisés en tant que données d'entrée. Une collection de strings, chacun d'entre eux est le contenu d'un fichier XML d'entrée.

- setSchemaFileCollection

```
public void setSchemaFileCollection(Collection<?> fileCollection)
```
Fournit la collection des fichiers de Schéma XML qui sera utilisée en tant que Schémas XML externe. Les fichiers sont identifiés par leurs URL. L'entrée est une collection de strings, chacun d'entre eux est l'URL absolue d'un fichier de Schéma XML.

- setSchemaFileName

```
public void setSchemaFileName(String filePath)
```
Définit l'emplacement, en tant qu'une URL, du document de Schéma XML à utiliser pour la validation. Le string fourni doit être une URL absolue qui donne l'emplacement exact du fichier de Schéma XML.

- setSchemaFromText

```
public void setSchemaFromText(String schemaText)
```
Fournit les contenus du document de Schéma XML à utiliser. Le string fourni est le contenu du document de Schéma XML à utiliser.

- setSchemaTextCollection

```
public void setSchemaTextCollection(Collection<?> stringCollection)
```
Fournit le contenu de plusieurs fichiers de Schéma XML. L'entrée est une collection de strings, dont chacun est le contenu d'un document de Schéma XML.

- setDTDFileName

```
public void setDTDFileName(String filePath)
```
Définit l'emplacement en tant qu'une URL, du document DTD à utiliser pour la validation. Le string fourni doit être une URL absolue qui indique l'emplacement exact du document DTD.

- setDTDFromText

```
public void setDTDFromText(String dtdText)
```
Apporte le contenu du document DTD à utiliser pour la validation. Le string fourni est le contenu du document DTD à utiliser.

Schéma XML

Des méthodes pour définir des options pour le Schéma XML qui est utilisé pour la validation.

- setSchemaImports

  public void setSchemaImports(*ENUMSchemaImports* opt)
  Spécifie comment traiter les importations de schéma sur la base des valeurs d'attribut des éléments `xs:import`. Le type de gestion est spécifié par le littéral `ENUMSchemaImports` (Java, COM/.NET) qui est soumis.

- setSchemalocationHints

  public void setSchemalocationHints(*ENUMLoadSchemalocation* opt)
  Spécifie le mécanisme à utiliser pour situer le schéma. Le mécanisme est spécifié par le littéral `ENUMLoadSchemalocation` (Java, COM/.NET) qui a été sélectionné.

- setSchemaMapping

  public void setSchemaMapping(*ENUMSchemaMapping* opt)
  Définit quel mappage utiliser pour localiser le schéma. Le mappage est spécifié par le littéral `ENUMSchemaMapping` (Java, COM/.NET) qui est sélectionné.

- setXSDVersion

  public void setXSDVersion(*ENUMXSDVersion* version)
  Définit la version de schéma XML par rapport à laquelle le document XML sera validé. Est une énumération littérale de `ENUMXSDVersion` (Java, COM/.NET).

XML

Des méthodes pour spécifier des options liées aux données XML d'entrée.

- setEnableNamespaces

  public void setEnableNamespaces(*boolean* enable)
  Active le traitement reconnaissant l'espace de noms. Cela est utile pour contrôler la présence d'erreurs dues à des espaces de noms incorrects dans l'instance XML. Une valeur `true` permet le traitement reconnaissant les espaces de noms ; une valeur `false` la désactive. La valeur par défaut est `false`.

- setXincludeSupport

  public void setXincludeSupport(*boolean* support)
  Active ou désactive l'utilisation des éléments `XInclude`. Une valeur de `true` active la prise en charge XInclude ; `false` le désactive. La valeur par défaut est `false`.

- setXMLValidationMode

  public void setXMLValidationMode(*ENUMXMLValidationMode* mode)
  Définit le mode de validation XML, qui est une énumération littérale de

`ENUMXMLValidationMode` (Java, COM/.NET) qui détermine s'il faut contrôler la validité ou la bonne formation.

© 2018 Altova GmbH

XSLT

public interface XSLT

Transforme XML en utilisant le document XSLT 1.0, 2.0 ou 3.0 fournit. Les documents XML et XSLT peuvent être fournis en tant que fichiers (par le biais d'une URL) ou en tant que string de texte. La sortie est retournée en tant que fichier (à un emplacement nommé) ou en tant que string de texte. Les paramètres XSLT peuvent être fournis et les fonctions d'extension Altova peuvent être activées pour un traitement spécialisé, comme pour les graphiques. Le document XSLT peut aussi être validé. Lorsque des entrées de string doivent être interprétées en tant qu'URL, les chemins absolus doivent être utilisés. Les méthodes de l'interface `XSLT` sont décrites ci-dessous, et sont regroupées par fonctions.

Traitement

Des méthodes pour spécifier les paramètres d'une transformation XSLT.

isValid

public boolean isValid()

Retourne le résultat de validation du document XSLT conformément à la spécification XSLT nommée dans `ENUMXSLTVersion` (voir la méthode `setVersion`). Le résultat est `true` en cas de succès, `false` en cas d'échec. Si une erreur se produit, une `RaptorXMLException` est soulevée. Utiliser la méthode `getLastErrorMessage` pour accéder aux informations supplémentaires.

execute

public boolean execute(*String* outputFile)

Exécute la transformation XSLT conformément à la spécification XSLT nommée dans `ENUMXSLTVersion` (voir la méthode `setVersion`), et enregistre le résultat du fichier de sortie nommé dans le paramètre `outputFile`. Si une erreur se produit, une `RaptorXMLException` est soulevée. Utiliser la méthode `getLastErrorMessage` pour accéder aux informations supplémentaires.

executeAndGetResultAsString

public String executeAndGetResultAsString()

Exécute la transformation XSLT conformément à la spécification XSLT nommée dans `ENUMXSLTVersion` (voir la méthode `setVersion`), et retourne le résultat en tant que string. Cette méthode ne produit pas de fichiers de résultat supplémentaires, comme des graphiques ou des résultats secondaires. Elle ne contient pas non plus de résultats binaires comme des fichiers OOXML `.docx`. Si des fichiers de sortie supplémentaires sont nécessaires, utiliser la méthode `execute`. Si une erreur se produit, une `RaptorXMLException` est soulevée. Utiliser la méthode `getLastErrorMessage` pour accéder aux informations supplémentaires.

executeAndGetResultAsStringWithBaseOutputURI

public String executeAndGetResultAsStringWithBaseOutputURI(*String* baseURI)

Exécute la transformation XSLT conformément à la spécification XSLT nommée dans

ENUMXSLTVersion (voir la méthode setVersion), et retourne le résultat en tant que string à l'emplacement définit par l'URI de base. Le paramètre baseURI est un string qui fournit un URI. Cette méthode ne produit pas de fichiers de résultat supplémentaires, comme des graphiques ou des résultats secondaires. Elle ne contient pas non plus de résultats binaires comme des fichiers OOXML .docx. Si des fichiers de sortie supplémentaires sont nécessaires, utiliser la méthode execute. Si une erreur se produit, une RaptorXMLException est soulevée. Utiliser la méthode getLastErrorMessage pour accéder aux informations supplémentaires.

- getMainOutput

```
public String getMainOutput()
```

Retourne la sortie principale de la dernière tâche exécutée en tant que string.

- getAdditionalOutputs

```
public String getAdditionalOutputs()
```

Retourne les sorties supplémentaires de la dernière tâche exécutée.

- getLastErrorMessage

```
public String getLastErrorMessage()
```

Récupère un string qui est le dernier message d'erreur provenant du moteur RaptorXML.

- setIndentCharacters

```
public void setIndentCharacters(String chars)
```

Soumet le string de caractère qui sera utilisé en tant que retrait dans la sortie.

- setStreamingSerialization

```
public void setStreamingSerialization(boolean support)
```

Active la sérialisation de streaming. Dans le mode streaming, les données qui sont stockées en mémoire sont minimisées et le traitement est plus rapide. Une valeur de true active le streaming; false le désactive.

XSLT

Des méthodes pour spécifier les options liées à la feuille de style XSLT.

- setVersion

```
public void setVersion(ENUMXSLTVersion version)
```

Définit la version XSLT à utiliser pour le traitement (validation ou transformation XSLT).

Paramètres :

version: contient un littéral d'énumération EnumXSLTVersion eVersion10, eVersion20, ou eVersion30.

setXSLFileName

```
public void setXSLFileName(String xslFile)
```

Définit l'emplacement, en tant qu'une URL, du document XSLT à utiliser pour la transformation.

Paramètres :

`xslFile`: le string fournit doit être une URL absolue qui donne l'emplacement exact du fichier XSLT.

setXSLFromText

```
public void setXSLFromText(String xslText)
```

Fournit les contenus du document XSLT en tant que texte.

Paramètres :

`xslText`: le string fournit est le document XSLT à utiliser pour la transformation.

addExternalParameter

```
public void addExternalParameter(String name, String value)
```

Ajoute le nom et la valeur d'un nouveau paramètre externe. Chaque paramètre externe et sa valeur doivent être spécifiés dans un appel séparé à la méthode. Les paramètres doivent être déclarés dans le document XSLT. Puisque les valeurs de paramètre sont des expressions XPath, les valeurs de paramètre qui sont des strings doivent être contenues dans des guillemets simples.

Paramètres :

`name`: contient le nom du paramètre, qui est un QName, en tant que string.

`value`: contient la valeur du paramètre en tant que string.

clearExternalParameterList

```
public void clearExternalVariableList()
```

Supprime la liste des paramètres externes créés par la méthode `AddExternalParameter`.

setInitialTemplateMode

```
public void setInitialTemplateMode(String mode)
```

Définit le nom du mode de modèle initial. Le traitement commencera avec des modèles ayant cette valeur de mode. La transformation doit être lancée après avoir attribué les documents XML et XSLT.

Paramètres :

`mode`: le nom du mode de modèle initial, en tant que string.

setNamedTemplateEntryPoint

```
public void setNamedTemplateEntryPoint(boolean template)
```

Définit le nom du mode nommé avec lequel le traitement devra être lancé.

Paramètres :

`template`: le nom du mode de modèle nommé, en tant que string.

Schéma XML

Des méthodes pour définir les options pour le Schéma XML utilisé pour la validation.

- setSchemaImports

  public void setSchemaImports(*ENUMSchemaImports* opt)
  Spécifie comment traiter les importations de schéma sur la base des valeurs d'attribut des éléments `xs:import`. Le type de gestion est spécifié par le littéral `ENUMSchemaImports` (Java, COM/.NET) qui est soumis.

- setSchemalocationHints

  public void setSchemalocationHints(*ENUMLoadSchemalocation* opt)
  Spécifie le mécanisme à utiliser pour situer le schéma. Le mécanisme est spécifié par le littéral `ENUMLoadSchemalocation` (Java, COM/.NET) qui a été sélectionné.

- setSchemaMapping

  public void setSchemaMapping(*ENUMSchemaMapping* opt)
  Définit quel mappage utiliser pour localiser le schéma. Le mappage est spécifié par le littéral `ENUMSchemaMapping` (Java, COM/.NET) qui est sélectionné.

- setXSDVersion

  public void setXSDVersion(*ENUMXSDVersion* version)
  Définit la version de schéma XML par rapport à laquelle le document XML sera validé. Est une énumération littérale de `ENUMXSDVersion` (Java, COM/.NET).

XML

Des méthodes pour spécifier les paramètres liés aux données XML qui seront traitées.

- setInputXMLFileName

  public void setInputXMLFileName(*String* xmlFile)
  Définit l'emplacement, en tant qu'une URL, du document XML à traiter. Le string fourni doit être une URL absolue qui donne l'emplacement exact du fichier XML.

- setInputXMLFromText

  public void setInputXMLFromText(*String* inputXMLText)
  Apporte le contenu du document XML à traiter. Le string fourni est le contenu du document XML à traiter.

 © 2018 Altova GmbH

- setLoadXMLWithPSVI

  public void setLoadXMLWithPSVI(*boolean* psvi)
  Active la validation des fichiers XML d'entrée et génère des infos post-schema-validation pour les fichiers. Une valeur de `true` active la validation XML et génère des infos post-schema-validation pour les fichiers XML ; `false` désactive la validation. La valeur par défaut est `true`.

- setXincludeSupport

  public void setXincludeSupport(*boolean* support)
  Active ou désactive l'utilisation des éléments `XInclude`. Une valeur de `true` active la prise en charge XInclude ; `false` le désactive. La valeur par défaut est `false`.

- setXMLValidationErrorAsWarning

  public void **setXMLValidationErrorAsWarning**(boolean enable)
  Définit le mode de validation XML, qui est une énumération littérale de `ENUMXMLValidationMode` (Java, COM/.NET) qui détermine s'il faut contrôler la validité ou la bonne formation.

- setXMLValidationMode

  public void setXMLValidationMode(*ENUMXMLValidationMode* mode)
  Définit le mode de validation XML, qui est une énumération littérale de `ENUMXMLValidationMode` (Java, COM/.NET) qui détermine s'il faut contrôler la validité ou la bonne formation.

Extensions

Des méthodes pour spécifier si les fonctions d'extension Altova doivent être activées pour le traitement spécialisé, comme les graphiques.

- setChartExtensionsEnabled

  public void setChartExtensionsEnabled(*boolean* enable)
  Active ou désactive les fonctions d'extension de graphique d'Altova. Une valeur de `true` active les extensions de graphique ; `false` les désactive. La valeur par défaut est `true`.

- setDotNetExtensionsEnabled

  public void setDotNetExtensionsEnabled(*boolean* enable)
  Active ou désactive les fonctions d'extension de .NET. Une valeur de `true` active les extensions .NET ; `false` les désactive. La valeur par défaut est `true`.

- setJavaExtensionsEnabled

```
public void setJaveExtensionsEnabled(boolean enable)
```

Active ou désactive des fonctions d'extension Java. Une valeur de `true` active des extensions Java ; `false` les désactive. La valeur par défaut est `true`.

setJavaBarcodeExtensionLocation

```
public void setJavaBarcodeExtensionLocation(String path)
```

Spécifier le chemin vers le dossier contenant le fichier d'extension du code-barres `AltovaBarcodeExtension.jar`. Voir la section dans les fonctions d'extension de code-barres d'Altova pour plus d'informations. Le chemin doit être indiqué dans une des formes suivantes :

- Un ficher URI, par exemple : `--javaext-barcode-location="file:///C:/Program Files/Altova/RaptorXMLServer2015/etc/jar/"`
- Un chemin Windows échappé par des barres obliques renversées, par exemple : `--javaext-barcode-location="C:\\Program Files\\Altova\\RaptorXMLServer2015\\etc\\jar\\"`

Paramètres :

`path`: Le string fourni doit être une URI absolue qui donne l'emplacement de base du fichier à utiliser.

 © 2018 Altova GmbH

XQuery

public interface XQuery

Exécute les documents XQuery 1.0 et 3.0 à l'aide du moteur RaptorXML. Les documents XQuery et XML peuvent être fournis en tant que fichier (par le biais d'une URL) ou en tant que string de texte. La sortie est retournée en tant que fichier (à un emplacement nommé) ou en tant que string de texte. Les variables XQuery externes peuvent être fournies et un nombre d'options de sérialisation sont disponibles. Le document XQuery peut aussi être validé. Lorsque les entrées de string doivent être interprétés en tant qu'URL, il faut utiliser des chemins absolus. Les méthodes de l'interface XQuery sont décrites ci-dessous et regroupées selon les fonctionnalités.

Traitement

Des méthodes pour spécifier les paramètres d'une exécution XQuery.

isValid

public boolean isValid()

Retourne le résultat de validation du document XQuery conformément à la spécification XQuery nommée dans `ENUMXQueryVersion` (voir la méthode `setVersion`). Le résultat est `true` en cas de succès, `false` en cas d'échec. Si une erreur se produit, une `RaptorXMLException` est soulevée. Utiliser la méthode `getLastErrorMessage` pour accéder aux informations supplémentaires.

isValidUpdate

public boolean isValidUpdate()

Retourne le résultat de validation du document XQuery Update document conformément à la spécification XQuery Update nommée dans `ENUMXQueryVersion` (voir la méthode `setVersion`). Le résultat est `true` en cas de succès, `false` en cas d'échec. Si une erreur se produit, une `RaptorXMLException` soulevée. Utiliser la méthode `getLastErrorMessage` pour accéder aux informations supplémentaires.

execute

public boolean execute(*String* outputFile)

Exécute la transformation XQuery conformément à la spécification XQuery nommée dans `ENUMXQueryVersion` (voir la méthode `setVersion`), et enregistre le résultat sur le fichier de sortie nommé dans le paramètre `outputFile`. Le paramètre est un string qui fournit l'emplacement (chemin et nom de fichier) du fichier de la sortie. Le résultat est `true` en cas de succès, `false` en cas d'échec. Si une erreur se produit, une `RaptorXMLException` est soulevée. Utiliser la méthode `getLastErrorMessage` pour accéder aux informations supplémentaires.

executeAndGetResultAsString

public String executeAndGetResultAsString()

Exécute la transformation XQuery conformément à la spécification XQuery nommée dans `ENUMXQueryVersion` (voir la méthode `setVersion`), et retourne le résultat en tant que string.

Cette méthode ne produit pas de fichiers de résultat supplémentaires, comme des graphiques ou des résultats secondaires. Elle ne contient pas non plus de résultats binaires comme des fichiers OOXML `.docx`. Si des fichiers de sortie supplémentaires sont nécessaires, utiliser la méthode `execute`.

executeUpdate

```
public boolean executeUpdate(String outputFile)
```

Exécute la mise à jour XQuery conformément à la spécification XQuery Update nommée dans `ENUMXQueryVersion` (voir la méthode `setVersion`), et enregistre le résultat sur le fichier de sortie nommé dans le paramètre `outputFile`. e paramètre est un string qui fournit l'emplacement (chemin et nom de fichier) du fichier de la sortie. Le résultat est `true` en cas de succès, `false` en cas d'échec. Si une erreur se produit, une `RaptorXMLException` est soulevée. Utiliser la méthode `getLastErrorMessage` pour accéder aux informations supplémentaires.

executeUpdateAndGetResultAsString

```
public String executeUpdateAndGetResultAsString()
```

Exécute la mise à jour XQuery conformément à la spécification XQuery Update nommée dans `ENUMXQueryVersion` (voir la méthode `setVersion`), et retourne le résultat en tant que string. Cette méthode ne produit pas de fichiers de résultat supplémentaires, comme des graphiques ou des résultats secondaires. Elle ne contient pas non plus de résultats binaires comme des fichiers OOXML `.docx`. Si des fichiers de sortie supplémentaires sont nécessaires, utiliser la méthode `execute`.

getLastErrorMessage

```
public String getLastErrorMessage()
```

Récupère un string qui est le dernier message d'erreur provenant du moteur RaptorXML.

setUpdatedXMLWriteMode

```
public void setUpdatedXMLWriteMode(ENUMXQueryUpdatedXML mode)
```

Définit le mode à utiliser pour la mise à jour. Prend un littéral d'énumération `ENUMXQueryUpdatedXML` : `eUpdatedDiscard`, `eUpdatedWriteback` ou `eUpdatedAsMainResult`.

XML

Des méthodes pour spécifier les paramètres concernant les données XML qui seront traitées

setInputXMLFileName

```
public void setInputXMLFileName(String xmlFile)
```

Définit l'emplacement, en tant qu'une URL, du document XML à traiter. Le string fourni doit être une URL absolue qui donne l'emplacement exact du fichier XML.

 © 2018 Altova GmbH

- setInputXMLFromText

  public void setInputXMLFromText(*String* inputXMLText)
  Apporte le contenu du document XML à traiter. Le string fourni est le contenu du document XML à traiter.

- setLoadXMLWithPSVI

  public void setLoadXMLWithPSVI(*boolean* psvi)
  Active la validation des fichiers XML d'entrée et génère des infos post-schema-validation pour les fichiers. Une valeur de `true` active la validation XML et génère des infos post-schema-validation pour les fichiers XML ; `false` désactive la validation. La valeur par défaut est `true`.

- setXincludeSupport

  public void setXincludeSupport(*boolean* support)
  Active ou désactive l'utilisation des éléments `XInclude`. Une valeur de `true` active la prise en charge XInclude ; `false` le désactive. La valeur par défaut est `false`.

- setXMLValidationErrorAsWarning

  public void **setXMLValidationErrorAsWarning**(boolean enable)
  Définit le mode de validation XML, qui est une énumération littérale de `ENUMXMLValidationMode` (Java, COM/.NET) qui détermine s'il faut contrôler la validité ou la bonne formation.

- setXMLValidationMode

  public void setXMLValidationMode(*ENUMXMLValidationMode* mode)
  Définit le mode de validation XML, qui est une énumération littérale de `ENUMXMLValidationMode` (Java, COM/.NET) qui détermine s'il faut contrôler la validité ou la bonne formation.

- setXSDVersion

  public void setXSDVersion(*ENUMXSDVersion* version)
  Définit la version de schéma XML par rapport à laquelle le document XML sera validé. Est une énumération littérale de `ENUMXSDVersion` (Java, COM/.NET).

XQuery

Des méthodes pour spécifier les options concernant le document XQuery.

- setVersion

```
public void setVersion(ENUMXQueryVersion version)
```
Définit la version XQuery à utiliser pour le traitement (validation ou exécution XQuery). Prend un littéral d'énumération `ENUMXQueryVersion` : `eVersion10` ou `eVersion30`. Le défaut est `eVersion30ml`.

- setXQueryFileName

```
public void setXQueryFileName(String queryFile)
```
Définit l'emplacement, en tant qu'URL du fichier XQuery à exécuter. Le string fourni doit être une URL absolue qui donne l'emplacement exact du fichier XML à utiliser.

- setXQueryFromText

```
public void setXQueryFromText(String queryText)
```
Fournit les contenus du document XQuery en tant que texte. Le string fourni est le document XQuery à traiter.

- addExternalVariable

```
public void addExternalVariable(String name, String value)
```
Ajoute le nom et la valeur d'une nouvelle variable externe. Chaque variable externe et sa valeur doivent être spécifiés dans un appel séparé de la méthode. Les variables doivent être déclarées dans le document XQuery (avec une déclaration type optionnelle). Si la valeur de variable est un string, contenir la valeur dans des guillemets simples. Le paramètre `name` contient le nom de la variable, qui est un QName, en tant que string. Le paramètre `value` contient la valeur de la variable en tant que string.

- clearExternalVariableList

```
public void clearExternalVariableList()
```
Supprime la liste de variables externes créée par la méthode `AddExternalVariable`.

Options de sérialisation

Des méthodes pour spécifier les propriétés de la sortie du traitement.

- setIndentCharacters

```
public void setIndentCharacters(String chars)
```
Soumet le string de caractère qui sera utilisé en tant que retrait dans la sortie.

- setKeepFormatting

```
public void setKeepFormatting(boolean keep)
```
Active ou désactive l'option pour garder le format original des fichier qui seront mis à jour par `executeUpdate`. Le paramètre `keep` prend une booléenne `true` ou `false`.

- setOutputEncoding

  public void setOutputEncoding(*String* encoding)
  Définit l'encodage du document de résultat. Utiliser un nom d'encodage IANA officiel, comme `UTF-8`, `UTF-16`, `US-ASCII`, `ISO-8859-1`, en tant que string.

- setOutputIndent

  public void setOutputIndent(*boolean* indent)
  Active ou désactive le retrait dans le document de sortie. Une valeur de `true` active le retrait ; `false` le désactive.

- setOutputMethod

  public void setOutputMethod(*String* outputMethod)
  Spécifie la sérialisation du document de sortie. Les valeurs valides sont : `xml | xhtml | html | text`, donné en tant que string. La valeur par défaut est `xml`.

- setOutputOmitXMLDeclaration

  public void setOutputOmitXMLDeclaration(*boolean* omit)
  Active ou désactive l'inclusion de la déclaration XML dans le document de résultat. Une valeur de `true` omet la déclaration ; `false` l'inclut. La valeur par défaut est `false`.

Extensions

Des méthodes pour spécifier si les fonctions d'extension Altova doivent être activées pour un traitement spécialisé, comme des graphiques.

- setChartExtensionsEnabled

  public void setChartExtensionsEnabled(*boolean* enable)
  Active ou désactive les fonctions d'extension de graphique d'Altova. Une valeur de `true` active les extensions de graphique ; `false` les désactive. La valeur par défaut est `true`.

- setDotNetExtensionsEnabled

  public void setDotNetExtensionsEnabled(*boolean* enable)
  Active ou désactive les fonctions d'extension de .NET. Une valeur de `true` active les extensions .NET ; `false` les désactive. La valeur par défaut est `true`.

- setJavaExtensionsEnabled

  public void setJaveExtensionsEnabled(*boolean* enable)
  Active ou désactive des fonctions d'extension Java. Une valeur de `true` active des extensions Java ; `false` les désactive. La valeur par défaut est `true`.

XBRL

public interface XBRL

Valide le document d'instance XBRL fourni ou le document de taxonomie XBRL. Les méthodes de l'interface sont décrites ci-dessous. Les énumérations sont décrites dans une section séparée.

Classe d'utilité

Une classe d'utilité est définie pour `FormulaParam`. Elle contient deux membres et un constructeur.

- ParamValuePair

```
public class ParamValuePair
public class ParamValuePair
    {
        public String paramType;
        public String paramValue;
        public ParamValuePair( String type, String value )
        {
                        paramType = type;
                        paramValue = value;
        }
     };
```

Traitement

Des méthodes pour spécifier les paramètres d'une validation et pour récupérer les informations concernant la validation.

- isValid(ENUM type)

  `public boolean isValid(ENUMValidationType type)`

  Retourne le résultat de la validation du document d'instance XBRL ou du document de taxonomie XBRL. Le type de document à valider est spécifié par le paramètre `type`, qui prend un littéral `ENUMValidationType` en tant que sa valeur. Le résultat est `true` en cas de succès, `false` en cas d'échec. Si une erreur se produit, une `RaptorXMLException` est soulevée. Utiliser la méthode `getLastErrorMessage` pour accéder aux informations supplémentaires.

- isValid

  `public boolean isValid()`

  Retourne le résultat de validation du document XBRL soumis. Le résultat est `true` en cas de succès, `false` en cas d'échec.

- getLastErrorMessage

```
public String getLastErrorMessage()
```
Récupère un string qui est le dernier message d'erreur provenant du moteur RaptorXML.

- setEvaluateReferencedParametersOnly

```
public void setEvaluateReferencedParametersOnly(boolean enable)
```
Si défini sur `false`, force l'évaluation de tous les paramètres même s'ils ne sont pas référencés par une formule/assertion/table. La valeur par défaut est `true`.

- setParallelAssessment

```
public void setParallelAssessment(boolean support)
```
Active ou désactive l'utilisation des attributions parallèles. Une valeur de `true` active l'attribution parallèle ; `false` la désactive. La valeur par défaut est `false`.

- setPythonScriptFile

```
public void setPythonScriptFile(String file)
```
Définit l'emplacement, en tant qu'URL du fichier de script Python. Le string fourni doit être une URL absolue qui indique l'emplacement exact du fichier Python.

Fichiers d'entrée

Des méthodes pour spécifier l'entrée des commandes de validation.

- setInputFileName

```
public void setInputFileName(String filePath)
```
Définit l'emplacement, en tant qu'URL du document XBRL à valider. Le string fourni doit être une URL absolue qui indique l'emplacement exact du fichier XBRL.

- setInputFileCollection

```
public void setInputFileCollection(Collection<?> fileCollection)
```
Fournit la collection des fichiers XBRL qui seront utilisés entant que données d'entrée. Les fichiers sont identifiés par leurs URL. Le paramètre `fileCollection` est une collection de strings, chacun d'entre eux étant l'URL absolue d'un fichier XBRL d'entrée.

- setInputFromText

```
public void setInputFromText(String schemaText)
```
Fournit les contenus du document XBRL en tant que texte. Le string fourni est le contenu du document XBRL à valider.

- setInputTextCollection

 © 2018 Altova GmbH

```
public void setInputTextCollection(Collection<?> stringCollection)
```

Fournit le contenu de plusieurs fichiers XBRL qui seront utilisés en tant que données d'entrée. Le paramètre `stringCollection` est une collection de strings, chacun d'entre eux étant le contenu d'un document XBRL d'entrée.

Formules et Assertions

Des méthodes pour spécifier les options des Formules et Assertions XBRL.

addAssertionForProcessing

```
public void addAssertionForProcessing(String assertion)
```

Limite l'exécution d'assertion à la seule assertion donnée. Appeler plusieurs fois pour spécifier plus d'une assertion. Le string fourni contient le nom de l'assertion. Utiliser `##none` pour ne traiter aucune assertion et `##all` pour traiter toutes les assertions.

addAssertionSetForProcessing

```
public void addAssertionSetForProcessing(String assertionSet)
```

Limite l'exécution de l'ensemble d'assertion au seul ensemble d'assertion donné. Appeler plusieurs fois pour spécifier plus d'un ensemble d'assertion. Le string fourni contient le nom de l'ensemble d'assertion. Utiliser `##none` pour ne traiter aucun ensemble d'assertion et `##all` pour traiter tous les ensembles d'assertions.

addFormulaArrayParameter

```
public void addFormulaArrayParameter(String type, String name, Object[]
values)
```

Ajoute un paramètre d'array utilisé dans le processus d'évaluation de formule.

Paramètres :

`type`: Un string qui donne le type de données par défaut des valeurs non-paires dans les valeurs array. La valeur par défaut est `xs:string`.

`name`: Un string qui donne le nom du paramètre.

`values`: Un array de valeurs et des paires de valeurs de type de données.

Pour plus d'informations et des échantillons de code, voir la section, Paramètres de formule XBRL.

addFormulaForProcessing

```
public void addFormulaForProcessing(String formula)
```

Limite l'exécution de formule à la seule formule donnée. Appeler plusieurs fois pour spécifier plus d'une formule. Le string fourni contient le nom de la formule. Utiliser `##none` pour ne traiter aucune formule et `##all` pour traiter toutes les formules.

addFormulaParameter (with NS)

```
public void addFormulaParameter(String type, String name, String value,
```

`String namespace)`
Ajoute un paramètre utilisé dans le processus d'évaluation de formule.
Paramètres :
`type`: Un string qui donne le type de données du paramètre.
`name`: Un string qui donne le nom du paramètre.
`value`: Un string qui donne la valeur du paramètre.
`namespace`: Un string qui donne l'espace de noms du paramètre.
Note : Si cette méthode est utilisée, l'espace de noms est passé dans `addFormulaParameterNamespace`.

addFormulaParameter

`public void addFormulaParameter(String type, String name, String value)`
Ajoute un paramètre utilisé dans le processus d'évaluation de formule.
Paramètres :
`type`: Un string qui donne le type de données du paramètre.
`name`: Un string qui donne le nom du paramètre.
`value`: Un string qui donne la valeur du paramètre.

addFormulaParameterNamespace

`public void addFormulaParameterNamespace(String prefix, String URI)`
Définit un espace de noms utilisé dans les QNames des noms de paramètre, des types, ou des valeurs.
Paramètres :
`prefix`: Le préfixe d'espace de noms des valeurs passées sur `addFormulaArrayParameter`.
`URI`: L'URI espace de noms.

clearFormulaParameterList

`public void clearFormulaParameterList()`
Supprime la liste des paramètres de formule créé avec la méthode `addFormulaParameter`.

readFormulaAssertions

`public String readFormulaAssertions()`
Récupère les assertions de formule depuis le fichier spécifié. Retourne un string contenant les assertions de formule.

readFormulaOutput

`public String readFormulaOutput()`
Évalue les assertions de formule dans le fichier spécifié et retourne le résultat. Retourne un string qui est une évaluation des assertions de formule.

setFormulaAssertionsAsXML

`public void setFormulaAssertionsAsXML(boolean enable)`
Active le formatage XML du fichier d'assertion lorsque RaptorXML+XBRL est exécuté avec les

 © 2018 Altova GmbH

assertions activées. Une valeur de `true` active la sortie XML ; `false` génère une sortie JSON. La valeur par défaut est `false`.

- setFormulaAssertionsOutput

  `public void setFormulaAssertionsOutput(String outputFile)`
  Définit l'emplacement du fichier contenant les assertions de formule récupérées.
  Paramètres :
  `outputFile`: Le string fourni contient le chemin complet du fichier de sortie.

- setFormulaOutput

  `public void setFormulaOutput(String outputFile)`
  Définit l'emplacement du fichier contenant l'évaluation de formule de la sortie. Le string fourni contient le chemin complet du fichier de sortie.

- evaluateFormula

  `public boolean evaluateFormula()`
  Retourne le résultat d'évaluation des formules XBRL dans un fichier d'instance XBRL. Le résultat est `true` en cas de succès, `false` en cas d'échec. Si une erreur se produit, une `RaptorXMLException` est soulevée. Utiliser la méthode `getLastErrorMessage` pour accéder aux informations supplémentaires.

- setFormulaExtensionEnabled

  `public void setFormulaExtensionEnabled(boolean enable)`
  Active les extensions de formule XBRL pour la validation. Une valeur `true` active la prise en charge; `false` la désactive. La valeur par défaut est `true`.

- setFormulaPreloadSchemas

  `public void setFormulaPreloadSchemas(boolean enable)`
  Définit si les schémas de formule XBRL sera préchargé. Une valeur `true` précharge les schémas ; `false` ne le fait pas. La valeur par défaut est `false`.

Tables

Des méthodes pour spécifier les options pour les Tables XBRL.

- addTableForProcessing

  `public void addTableForProcessing(String table)`
  Limite la génération de table uniquement à la table donnée. Appeler plusieurs fois pour spécifier plus d'une table. Le string fourni contient de nom de la table. Utiliser `##none` pour ne traiter aucune formule et `##all` pour traiter toutes les formules.

generateTables

`public boolean generateTables()`

Évaluer les tables XBRL dans un fichier d'instance. Le résultat est `true` en cas de succès, `false` en cas d'échec. Si une erreur se produit, une `RaptorXMLException` est soulevée. Utiliser la méthode `getLastErrorMessage` pour accéder aux informations supplémentaires.

setTableEliminateEmptyRows

`public void setTableEliminateEmptyRows(boolean enable)`

Active l'élimination de lignes/colonnes de table vides uniquement dans une sortie HTML. Une valeur `true` active la prise en charge; `false` la désactive.

setTableExtensionEnabled

`public void setTableExtensionEnabled(boolean enable)`

Active les extensions XBRL Table 1.0 pour validation. Une valeur `true` active la prise en charge; `false` la désactive.

setTableLinkbaseNamespace

`public void setTableLinkbaseNamespace(String namespace)`

Active le chargement de bases de lien de table écrit avec une spécification de projet précédente. Le paramètre `namespace` spécifie la base de lien de table. La validation de base de lien de table, la résolution, et la mise en page est, néanmoins, toujours effectué conformément à la Recommandation Table Linkbase 1.0 du 18 mars 2014. Utiliser `##detect` pour activer la détection automatique.

Paramètres :

`namespace`: les valeurs suivantes sont reconnues.

```
##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
```

setTableOutput

`public void setTableOutput(String outputFile)`

Définit l'emplacement du fichier contenant la sortie de la génération de table. Le string fourni contient le chemin complet du fichier de sortie.

setTableOutputFormat

`public void setTableOutputAsXML(ENUMTableOutputFormat format)`

Définit le format du fichier de sortie de table. Le format sera un littéral de `ENUMTableOutputFormat`.

 © 2018 Altova GmbH

- setTablePreloadSchemas

  `public void setTablePreloadSchemas(boolean enable)`
  Active le préchargement des schémas de la spécification XBRL Table 1.0. Une valeur `true` active la prise en charge; `false` la désactive. La valeur par défaut est `false`.

XML et Schéma XML

Méthodes pour définir le niveau de prise en charge XInclude et les options pour le schéma XML.

- setXincludeSupport

  `public void setXincludeSupport(boolean support)`
  Active ou désactive l'utilisation des éléments `XInclude`. Une valeur de `true` active la prise en charge XInclude ; `false` le désactive. La valeur par défaut est `false`.

- setSchemaImports

  `public void setSchemaImports(ENUMSchemaImports opt)`
  Spécifie comment traiter les importations de schéma sur la base des valeurs d'attribut des éléments `xs:import`. Le type de gestion est spécifié par le littéral `ENUMSchemaImports` (Java, COM/.NET) qui est soumis.

- setSchemalocationHints

  `public void setSchemalocationHints(ENUMLoadSchemalocation opt)`
  Spécifie le mécanisme à utiliser pour situer le schéma. Le mécanisme est spécifié par le littéral `ENUMLoadSchemalocation` (Java, COM/.NET) qui a été sélectionné.

- setSchemaMapping

  `public void setSchemaMapping(ENUMSchemaMapping opt)`
  Définit quel mappage utiliser pour localiser le schéma. Le mappage est spécifié par le littéral `ENUMSchemaMapping` (Java, COM/.NET) qui est sélectionné.

XBRL général

Méthodes pour les options pour les libellés et des paramètres XBRL divers.

- addIXBRLTransformationRegistryLimit

  `public void addIXBRLTransformationRegistryLimit(String limit)`
  Limite les Inline XBRL Transformation Registries disponibles pour la version spécifiée. Le paramètre `limit` est la version pour laquelle les Inline XBRL Registries disponibles sont limitées.

- clearIXBRLTransformationRegistryLimit

  `public void clearIXBRLTransformationRegistryLimit()`
  Supprime toute limite spécifiée pour les Inline XBRL Transformation Registries.

- enableIXBRLValidateTarget

  `public void enableIXBRLValidateTarget(boolean enable)`
  Active la validation XBRL de la sortie XBRL générée. Prend la valeur booléenne `true` ou `false`. La valeur par défaut est `true`.

- setConceptLabelLinkrole

  `public void setConceptLabelLinkrole(String labelLinkrole)`
  Spécifie le rôle de lien étendu préféré à utiliser lors du rendu des libellés de concept. Le string fourni contient le rôle de lien préféré.

- setConceptLabelRole

  `public void setConceptLabelRole(String labelRole)`
  Spécifie le rôle de libellé préféré à utiliser lors du rendu des libellés de concept. Le string fourni contient le rôle de lien préféré. La valeur par défaut est : `http://www.xbrl.org/2008/role/label`.

- setDimensionsExtensionEnabled

  `public void setDimensionExtensionEnabled(boolean bEnable)`
  Active la validation d'extension XBRL Dimension. Prend la valeur booléenne `true` ou `false`. Une valeur `true` active la prise en charge; `false` la désactive. La valeur par défaut est `true`.

- setGenericLabelRole

  `public void setGenericLabelRole(String labelRole)`
  Spécifie le rôle de libellé préféré à utiliser lors du rendu des libellés génériques. Le string fourni contient le rôle de libellé préféré. La valeur par défaut est : `http://www.xbrl.org/2008/role/label`.

- setIXBRLOutput

  `public void setIXBRLOutput(String outputFile)`
  Si spécifié, la sortie XBRL générée sera écrite sur ce chemin. Le string fourni contient l'URL absolue de l'emplacement de sortie.

- setIXBRLTreatAsDocumentSet

  `public void setIXBRLTreatAsDocumentSet(boolean enable)`
  Traite toutes les entrées en tant qu'ensemble de document XBRL Inline unique. Prend la

 © 2018 Altova GmbH

valeur booléenne `true` ou `false`. La valeur par défaut est `false`.

setIXBRLUriTransformationStrategy

```
public void setIXBRLUriTransformationStrategy(ENUMIXBRLUriStrategy strategy)
```

Spécifie comment les URI XBRL Inline doivent être transformés dans le document XBRL généré. Le littéral d'énumération sélectionné spécifie la stratégie à suivre. L'énumération par défaut est `eStrategyKeepRelative`.

setIXBRLVersion

```
public void setIXBRLVersion(ENUMIXBRLVersion version)
```

Spécifie la version XBRL Inline à utiliser. Le littéral d'énumération sélectionné spécifie la version à utiliser. L'énumération par défaut est `eVersionDetect`.

setLabelLang

```
public void setLabelLang(String labelLang)
```

Spécifie le langage de libellé préféré à utiliser lors du rendu des libellés. Le string fourni contient le langage de libellé préféré. La valeur par défaut est : `en`.

setPreloadSchemas

```
public void setPreloadSchemas(boolean preload)
```

Définit si des schémas XBRL 2.1 sera préchargé. Prend la valeur booléenne `true` ou `false`. Une valeur de `true` indique les préchargements ; `false` la désactive. La valeur par défaut est `true`.

setTreatXBRLInconsistenciesAsErrors

```
public void setTreatXBRLInconsistenciesAsErrors(boolean treat)
```

Une valeur de `true` entraîne l'échec de la validation XBRL si le fichier contient des incohérences telles que définies par la spécification XBRL 2.1. Prend la valeur booléenne `true` ou `false`. La valeur par défaut est `false`. En cas de `false`, les incohérences XBRL, conformément à la spécification XBRL 2.1 ne sont pas traitées en tant qu'erreurs.

RaptorXMLException

```
public interface RaptorXMLException
```

Une seule méthode génère l'exception.

RaptorXMLException

```
public void RaptorXMLException(String message)
```

Génère une exception qui contient des informations concernant une erreur qui se produit pendant le traitement.

Paramètres :

`message`: un string qui fournit l'information concernant l'erreur.

 © 2018 Altova GmbH

6.3 Énumérations RaptorXML pour Java

Les énumérations suivantes sont définies. Elles sont regroupées conformément à la fonction et décrites dans les sous-sections de cette section.

RaptorXMLFactory
Validation XML
XSLT et XQuery
XBRL

RaptorXMLFactory

L'interface **RaptorXMLFactory** définit les énumérations suivantes.

▼ ENUMErrorFormat

```
public enum ENUMErrorFormat {
        eFormatText
        eFormatShortXML
        eFormatLongXML }
```

`ENUMErrorFormat` peut prendre un des littéraux d'énumération : `eFormatText`, `eFormatShortXML`, `eFormatLongXML`. Ils définissent le format des messages d'erreur, et `eLongXML` fournit les messages les plus détaillés. Le défaut est `eFormatText`.

Utilisé par (Interface::Method):

`RaptorXMLFactory` `setErrorFormat`

 © 2018 Altova GmbH

Validation XML

L'interface XMLValidator définit les énumérations suivantes.

ENUMAssessmentMode

```
public enum ENUMAssessmentMode {
        eAssessmentModeLax
        eAssessmentModeStrict    }
```

`ENUMAssessmentMode` prend un des littéraux d'énumération : `eAssessmentModeLax`, `eAssessmentModeStrict`. Ceux-ci définissent si la validation doit être laxe ou stricte.

Utiliser par (Interface::Method):

XMLValidator	setAssessmentMode

ENUMLoadSchemaLocation

```
public enum ENUMLoadSchemalocation {
        eLoadBySchemalocation
        eLoadByNamespace
        eLoadCombiningBoth
        eLoadIgnore    }
```

`ENUMLoadSchemalocation` contient le littéral d'énumération qui spécifie le mécanisme de localisation de schéma. La sélection est basée sur l'attribut d'emplacement du schéma du document d'instance XML ou XBRL. Cet attribut pourrait être `xsi:schemaLocation` ou `xsi:noNamespaceSchemaLocation`.

- `eLoadBySchemalocation` utilise l'URL de l'attribut d'emplacement du schéma dans le document d'instance XML ou XBRL. Ce littéral d'énumération est la **valeur par défaut**.
- `eLoadByNamespace` utilise la partie d'espace de noms de `xsi:schemaLocation` et un string vide dans le cas de `xsi:noNamespaceSchemaLocation` pour situer le schéma par le biais d'un mappage de catalogue.
- `eLoadCombiningBoth`: si soit l'URL d'espace de noms soit l'URL d'emplacement de schéma a un mappage de catalogue, alors le mappage de catalogue est utilisé. Si tous deux ont un mappage de catalogue, la valeur de `ENUMSchemaMapping` décide du mappage utilisé. Si ni l'espace de noms ni l'emplacement de schéma n'a un mappage de catalogue, l'URL d'emplacement de schéma est utilisé.
- `eLoadCombiningBoth`: Les attributs `xsi:schemaLocation` et `xsi:noNamespaceSchemaLocation` sont tous les deux ignorés.

Utilisé par (Interface::Method):

XMLValidator	setSchemalocationHints
XSLT	setSchemalocationHints
XBRL	setSchemalocationHints

ENUMSchemaImports

```
public enum ENUMSchemaImports {
        eSILoadBySchemalocation
```

```
        eSILoadPreferringSchemalocation
        eSILoadByNamespace
        eSILoadCombiningBoth
        eSILicenseNamespaceOnly    }
```

ENUMSchemaImports contient le littéral d'énumération qui définit le comportement des éléments xs:import du schéma, chacun d'entre eux a un attribut namespace optionnel et un attribut schemaLocation optionnel.

- eSILoadBySchemalocation utilise la valeur de l'attribut schemaLocation pour localiser le schéma, en prenant en compte les mappages de catalogue. Si l'attribut namespace est présent, l'espace de noms est importé (mis sous licence).
- eSILoadPreferringSchemalocation:Si l'attribut schemaLocation est présent, il est utilisé, en prenant en compte les mappages de catalogue. Si l'attribut schemaLocation est présent, la valeur de l'attribut namespace est utilisée par le biais d'un mappage de catalogue. Ce littéral d'énumération est la **valeur par défaut**.
- eSILoadByNamespace utilise la valeur de l'attribut namespace pour localiser le schéma par le biais d'un mappage de catalogue.
- eSILoadCombiningBoth: Si soit l'URL namespace soit l'URL schemaLocation a un mappage de catalogue, alors le mappage de catalogue sera utilisé. Si tous deux ont des mappages de catalogues, la valeur de ENUMSchemaMapping décide quel mappage utiliser. Si ni l'URL de namespace ni l'URL de schemaLocation a un mappage de catalogue, l'URL schemaLocation est utilisée.
- eSILicenseNamespaceOnly: L'espace de noms est importé. Aucun document de schéma n'est importé.

Utilisé par (Interface::Method):

XMLValidator	setSchemaImports
XSLT	setSchemaImports
XBRL	setSchemaImports

ENUMSchemaMapping

```
public enum ENUMSchemaMapping {
        eSMPreferSchemalocation
        eSMPreferNamespace    }
```

ENUMSchemaMapping contient le littéral d'énumération qui spécifie si l'espace de noms ou l'emplacement de schéma doit être sélectionné.

- eSMPreferNamespace: Sélectionne l'espace de noms.
- eSMPreferSchemalocation: Sélectionne l'emplacement de schéma. Il s'agit de la valeur par défaut.

Utilisé par (Interface::Method):

XMLValidator	setSchemaMapping
XSLT	setSchemaMapping
XBRL	setSchemaMapping

ENUMXMLValidationMode

```
public enum ENUMXMLValidationMode {
        eProcessingModeValid
        eProcessingModeWF  }
```

`ENUMXMLValidationMode` contient le littéral d'énumération qui spécifie le type de validation XML à effectuer (validation ou contrôle de la bonne formation).

- `eProcessingModeValid`: Définit le mode de traitement XML sur `validation`.
- `eProcessingModeWF`: Définit le mode de traitement XML sur `wellformed`. Il s'agit de la valeur par défaut.

Utilisé par (Interface::Method):

XMLValidator	setXMLValidationMode
XSLT	setXMLValidationMode
XQuery	setXMLValidationMode

ENUMXMLValidationType

```
public enum ENUMValidationType {
        eValidateAny
        eValidateXMLWithDTD
        eValidateXMLWithXSD
        eValidateDTD
        eValidateXSD
        eValidateJSON
        eValidateJSONSchema
        eValidateAvro
        eValidateAvroSchema
        eValidateAvroJSON }
```

`ENUMValidationType` contient le littéral d'énumération qui spécifie quelle validation effectuer et, dans le cas des documents XML, si la validation est effectuée par rapport à un DTD ou un XSD.

- `eValidateAny`: Le type du document est détecté automatiquement.
- `eValidateXMLWithDTD`: Valide un document XML par rapport à un DTD.
- `eValidateXMLWithXSD`: Valide un document XML par rapport à un XSD (Schéma XML).
- `eValidateDTD`: Valide un document DTD.
- `eValidateXSD`: Valide un document XSD.
- `eValidateJSON`: Valide un document d'instance JSON.
- `eValidateJSONSchema`: Valide un document de Schéma JSON.
- `eValidateAvro`: Valide un Avro fichier binaire. Les données Avro se trouvant dans le fichier binaire sont validées par rapport au Schéma Avro dans le fichier binaire.
- `eValidateAvroSchema`: Valide un Schéma Avro par rapport à la spécification de schéma Avro.
- `eValidateAvroJSON`: Valide un fichier de données Avro sérialisé JSON par rapport à un Schéma Avro.

Utilisé par (Interface::Method):

XMLValidator	isValid

ENUMWellFormedCheckType

```
public enum ENUMWellformedCheckType {
        eWellformedAny
        eWellformedXML
        eWellformedDTD
        eWellformedJSON    }
```

`ENUMWellformedCheckType` contient le littéral d'énumération qui spécifie le type de contrôle de bonne formation à effectuer (pour des documents XML ou DTD).

- `eWellformedAny`: Le type du document est détecté automatiquement.
- `eWellformedXML`: Contrôle la bonne formation du document XML.
- `eWellformedDTD`: Contrôle la bonne formation du document DTD.
- `eWellformedJSON`: Contrôle la bonne formation du document JSON.

Utilisé par (Interface::Method):

XMLValidator	isWellformed

ENUMXSDVersion

```
public enum ENUMXSDVersion {
        eXSDVersionAuto
        eXSDVersion10
        eXSDVersion11    }
```

`ENUMXSDVersion` contient le littéral d'énumération qui spécifie la version de Schéma XML.

- `eXSDVersionAuto`: La version de Schéma XML est détectée automatiquement depuis l'attribut `vc:minVersion` du document XSD. Si l'attribut `vc:minVersion` du document XSD a une valeur de `1.1`, le document sera considéré être un XSD 1.1. Si l'attribut a une autre valeur, ou n'existe pas, le document sera considéré être XSD 1.0.
- `eXSDVersion10`: Définit la version de Schéma XML pour une validation sur Schéma XML 1.0.
- `eXSDVersion11`: Définit la version de Schéma XML pour une validation sur Schéma XML 1.1.

Utilisé par (Interface::Method):

XMLValidator	setXSDVersion
XSLT	setXSDVersion
XQuery	setXSDVersion

 © 2018 Altova GmbH

XSLT et XQuery

L'interface XSLT définit les énumérations suivantes.

ENUMXSLTVersion

```
public enum ENUMXSLTVersion {
        eVersion10
        eVersion20
        eVersion30 }
```

`ENUMXSLTVersion` prend un des littéraux d'énumération : `eVersion10`, `eVersion20`, `eVersion30`. Ceux-ci définissent la version XSLT à utiliser pour le traitement (validation ou transformation XSLT).

Utilisé par (Interface::Method):

XSLT	setVersion

L'interface XQuery définit les énumérations suivantes.

ENUMXQueryVersion

```
public enum ENUMXQueryVersion {
        eVersion10
        eVersion30
        eVersion31 }
```

`ENUMXQueryVersion` prend un des littéraux d'énumération : `eVersion10`, `eVersion30`, `eVersion31`. Ceux-ci définissent la version XQuery à utiliser pour le traitement (exécution ou validation).

Utilisé par (Interface::Method):

XQuery	setVersion

ENUMXQueryUpdatedXML

```
public enum ENUMXQueryUpdatedXML {
        eUpdatedDiscard
        eUpdatedWriteback
        eeUpdatedAsMainResult }
```

`ENUMXQueryVersion` prend un des littéraux d'énumération :

- `eUpdatedDiscard`: les mises à jours sont éliminées et ne sont pas écrites sur le fichier.
- `eUpdatedWriteback`: les mises à jours sont écrites sur le fichier XML d'entrée spécifié avec `setInputXMLFileName`.
- `eUpdatedAsMainResult`: les mises à jours sont écrites sur l'emplacement spécifié par le paramètre `outputFile` de `ExecuteUpdate`.

Utilisé par (Interface::Method):

XQuery	setUpdatedXMLWriteMode

XBRL

L'interface XBRL définit les énumérations suivantes.

ENUMValidationType

```
public enum ENUMValidationType {
        eValidateAny
        eValidateInstance
        eValidateTaxonomy
        eValidateInline
        eValidateTaxonomyPackage }
```

`ENUMValidationType` contient le littéral d'énumération spécifiant quelle validation effectuer et, dans le cas des documents XML, si une validation est effectuée par rapport à un DTD ou un XSD.

- `eValidateAny`: Le type du document est détecté automatiquement.
- `eValidateInstance`: Valide un document d'instance XBRL (extension de fichier `.xbrl`).
- `eValidateTaxonomy`: Valide une taxonomie XBRL (extension de fichier `.xsd`).
- `eValidateInline`: Valide un document XBRL (iXBRL) Inline par rapport à la spécification Inline XBRL 1.0 or Inline XBRL 1.1.
- `eValidateTaxonomyPackage`: Valide un package de taxonomie XBRL conformément à la spécification des packages de Taxonomie 1.0.

Utilisé par (Interface::Method):

XBRL	isValid

ENUMTableOutputFormat

```
public enum ENUMTableOutputFormat {
        eFormatXML
        eFormatHTML }
```

`ENUMTableOutputFormat` contient le littéral d'énumération qui spécifie le format de sortie du document contenant les tables générées.

- `eFormatXML`: le document de sortie avec les tables générées est en format XML.
- `eFormatHTML`: le document de sortie avec les tables générées est en format HTML.

Utilisé par (Interface::Method):

XBRL	setTableOutputFormat

ENUMIXBRLVersion

```
public enum ENUMIXBRLVersion {
        eVersion10
        eVersion11
        eVersionDetect }
```

`ENUMIXBRLVersion` contient le littéral d'énumération qui spécifie la version de la spécification

Inline XBRL à utiliser : Inline XBRL 1.0 ou Inline XBRL 1.1. `eVersionDetect` active la détection automatique.

Utilisé par (Interface::Method):

XBRL	isValid

ENUMIXBRLUriStrategy

```
public enum ENUMIXBRLUriStrategy {
        eStrategyNone
        eStrategyMakeAbsolute
        eStrategyMakeRelative
        eStrategyKeepRelative }
```

`ENUMIXBRLUriStrategy` spécifie comment des URI doivent être écrits sur le document XBRL généré.

- `eStrategyNone`: Copie le verbatim URI sur le document cible.
- `eStrategyMakeAbsolute`: Change tout URI relatif en un URI absolu en résolvant par rapport à l'URI de base in-scope dans l'élément respectif dans le document d'entrée.
- `eStrategyMakeRelative`: Si possible, rend tout URI absolu et relatif en des URI relatifs dans le document de sortie (sinon, écrit l'URI absolu résolu).
- `eStrategyKeepRelative`: Si possible, rend uniquement les URI relatifs en des URI relatifs dans le document de sortie (et copie tout URI absolu).

Utilisé par (Interface::Method):

XBRL	setIXBRLUriTransformationStrategy

 © 2018 Altova GmbH

Chapitre 7

Interfaces COM et .NET

7 Interfaces COM et .NET

Deux interfaces, une API

Les interfaces COM et .NET de RaptorXML+XBRL Server utilisent une API unique : l'API COM/.NET de RaptorXML+XBRL Server. L'interface .NET est construite en tant que wrapper autour de l'interface COM.

Vous pouvez utiliser RaptorXML avec :

- des langages de script, comme JavaScript, par le biais de l'interface COM
- des langages de programmation, comme C#, par le biais de l'interface .NET

Organisation de cette section

Cette section est organisée dans les sujets suivants :

- À propos de l'interface COM, qui décrit comment l'interface COM travaille et les étapes à suivre pour travailler avec l'interface COM
- À propos de l'interface .NET, qui décrit comment configurer votre environnement pour travailler avec l'interface .NET.
- Langages de programmation, qui fournit des listes de code dans les langages de programmation les plus couramment utilisés qui montrent comment appeler la fonction RaptorXML.
- La référence d'API, qui documente le modèle d'objet, les objets et les propriétés de l'API.

 © 2018 Altova GmbH

7.1 À propos de l'interface COM

RaptorXML+XBRL Server est automatiquement enregistrée en tant qu'objet de serveur COM lorsque RaptorXML+XBRL Server est installé. Il peut donc être invoqué depuis les applications et les langages de script qui proposent une prise en charge de la programmation pour les appels COM. Si vous souhaitez changer l'emplacement du package d'installation RaptorXML+XBRL Server, il vaut mieux désinstaller RaptorXML+XBRL Server puis le réinstaller dans l'emplacement requis. De cette manière, la désinscription et l'enregistrement nécessaires sont effectués par le processus de l'installateur.

Contrôler le succès de l'inscription

Si l'enregistrement a réussi, le Registre contiendra les classes `RaptorXML.Server`. Ces classes se trouvent généralement sous `HKEY_LOCAL_MACHINE\SOFTWARE\Classes`.

Exemples de code

Un exemple de VBScript montrant comment l'API RaptorXML peut être utilisée par le biais de son interface COM est listé dans la section Langages de programmation. Un fichier d'exemple correspondant à cette liste est disponible dans le dossier `examples/API` du dossier d'application RaptorXML.

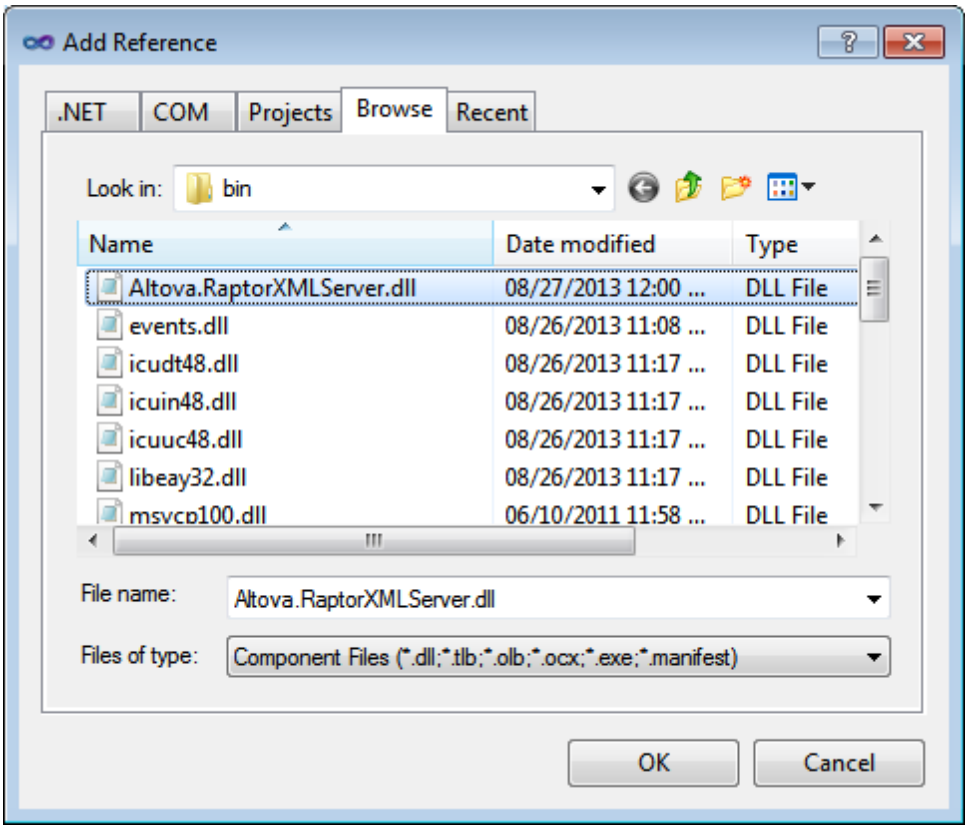
7.2 À propos de l'interface .NET

L'interface .NET est construite en tant qu'un wrapper autour de l'interface COM de RaptorXML. Elle est fournie en tant qu'assembly interop primaire signée par Altova ; elle utilise l'espace de noms `Altova.RaptorXMLServer`.

Ajouter la DLL RaptorXML en tant qu'une référence à un projet .NET Visual Studio
Pour pouvoir utiliser RaptorXML dans votre projet .NET, ajouter une référence à la DLL RaptorXML (`Altova.RaptorXMLServer.dll`) à votre projet. Votre installation RaptorXML+XBRL Server contient un fichier DLL signé, nommé `Altova.RaptorXMLServer.dll`. Ce fichier DLL sera ajouté automatiquement au cache d'assembly global (GAC) lorsque RaptorXML est installé à l'aide de l'installateur RaptorXML. Le GAC se trouve généralement dans le dossier : `C:\WINDOWS\assembly`.

Pour ajouter la DLL de RaptorXML en tant que référence dans un projet .NET, procéder comme suit :

1. Lorsque le projet .NET est ouvert, cliquer sur **Project | Add reference**. Le dialogue Ajouter Référence (*capture d'écran ci-dessous*) s'affiche.



2. Dans l'onglet Navigateur, se rendre dans le dossier : `<RaptorXML application folder>/bin`, choisir la DLL RaptorXML `Altova.RaptorXMLServer.dll`, et cliquer sur **OK**.
3. Choisir la commande View | Object Browser pour voir les objets de l'API de RaptorXML.

 © 2018 Altova GmbH

Une fois que `Altova.RaptorXMLServer.dll` est disponible sur l'interface .NET et que RaptorXML a été enregistré en tant qu'objet de serveur COM, la fonction RaptorXML sera disponible dans votre projet .NET.

Note : RaptorXML sera enregistré automatiquement en tant qu'objet de serveur COM pendant l'installation. Un enregistrement manuel n'est pas nécessaire.

Note : Si vous recevez une erreur d'accès, contrôlez que les permissions sont correctement définies. Aller au Component Services et accorder les permissions au même compte qui exécute le pool d'applications contenant RaptorXML.

Exemples de code

Un exemple C# et un exemple .NET Visual Basic montrant comment l'API RaptorXML peut être utilisée par le biais de son interface .NET est listé dans la section de programmation. Les fichiers correspondant à ces listes sont disponibles dans le dossier `examples/API` du dossier d'application RaptorXML.

7.3 Langages de programmation

Les langages de programmation diffèrent de la manière dont ils prennent en charge l'accès COM et .NET. Quelques exemples dans les langages les plus couramment utilisés vous permettront de vous lancer (*liens ci-dessous*). Les listes de code dans cette section montrent comment accéder aux fonctions de base. Cette fonction de base est contenue dans les fichiers du dossier `examples/API` du dossier d'application RaptorXML+XBRL Server.

VBScript

VBScript peut être utilisé pour accéder à l'API COM de RaptorXML+XBRL Server. La liste VBScript démontre les fonctions de base suivantes :

- Se connecter à l'API COM de RaptorXML+XBRL Server
- Valider un fichier XML
- Effectuer une Transformation XSL
- Effectuer une exécution XQuery

C#

C# peut être utilisé pour accéder à l'API .NET de RaptorXML+XBRL Server. La liste de code C# démontre comment accéder à l'API pour les fonctions de base suivantes :

- Se connecter à l'API .NET de RaptorXML+XBRL Server
- Valider un fichier XML
- Effectuer une Transformation XSL
- Effectuer une exécution XQuery

Visual Basic .NET

Visual Basic.NET est uniquement différent de C# dans sa syntaxe, l'accès API .NET fonctionne de la même manière. La liste de code Visual Basic décrit les opération de base suivantes :

- Se connecter à l'API .NET de RaptorXML+XBRL Server
- Valider un fichier XML
- Effectuer une Transformation XSL
- Effectuer une exécution XQuery

Cette section contient les exemples de code suivants :

Pour l'interface COM

- Un exemple dans VBScript

Pour l'interface .NET

- Un exemple dans C#
- Un exemple dans Visual Basic

Exemple COM : VBScript

L'exemple VBScript ci-dessous est structuré dans les parties suivantes :

- Configurer et initialiser l'objet COM RaptorXML
- Valider un fichier XML
- Effectuer une transformation XSLT, retourner le résultat en tant que string
- Traiter un document XQuery, enregistrer le résultat dans un fichier
- Configurer la séquence d'exécution du script et de son point d'entrée

```
' The RaptorXML COM object
dim objRaptor

' Initialize the RaptorXML COM object
sub Init
	objRaptor = Null
	On Error Resume Next
	' Try to load the 32-bit COM object; do not throw exceptions if object is
not found
	Set objRaptor = WScript.GetObject( "", "RaptorXML.Server" )
	On Error Goto 0
	if ( IsNull( objRaptor ) ) then
		' Try to load the 64-bit object (exception will be thrown if not
found)
		Set objRaptor = WScript.GetObject( "", "RaptorXML_x64.Server" )
	end if
	' Configure the server: error reporting, HTTP server name and port (IPv6
localhost in this example)
	objRaptor.ErrorLimit = 1
	objRaptor.ReportOptionalWarnings = true
	objRaptor.ServerName = "::1"
	objRaptor.ServerPort = 8087
end sub



' Validate one file
sub ValidateXML
	' Get a validator instance from the Server object
	dim objXMLValidator
	Set objXMLValidator = objRaptor.GetXMLValidator()

	' Configure input data
	objXMLValidator.InputFileName = "MyXMLFile.xml"

	' Validate; in case of invalid file report the problem returned by
RaptorXML
	if ( objXMLValidator.IsValid() ) then
		MsgBox( "Input string is valid" )
	else
		MsgBox( objXMLValidator.LastErrorMessage )
	end if
```

 © 2018 Altova GmbH

```
end sub



' Perform a transformation; return the result as a string
sub RunXSLT
       ' Get an XSLT engine instance from the Server object
       dim objXSLT
       set objXSLT = objRaptor.GetXSLT

       ' Configure input data
       objXSLT.InputXMLFileName = "MyXMLFile.xml"
       objXSLT.XSLFileName = "MyTransformation.xsl"

       ' Run the transformation; in case of success the result will be returned,
in case of errors the engine returns an error listing
       MsgBox( objXSLT.ExecuteAndGetResultAsString() )
end sub



' Execute an XQuery; save the result in a file
sub RunXQuery
       ' Get an XQuery engine instance from the Server object
       dim objXQ
       set objXQ = objRaptor.GetXQuery()

       ' Configure input data
       objXQ.InputXMLFileName = "MyXMLFile.xml"
       objXQ.XQueryFileName = "MyQuery.xq"

       ' Configure serialization (optional - for fine-tuning the result's
formatting)
       objXQ.OutputEncoding = "UTF8"
       objXQ.OutputIndent = true
       objXQ.OutputMethod = "xml"
       objXQ.OutputOmitXMLDeclaration = false

       ' Run the query; the result will be serialized to the given path
       call objXQ.Execute( "MyQueryResult.xml" )
end sub



' Perform all sample functions
sub main
       Init
       ValidateXML
       RunXSLT
       RunXQuery
end sub

' Script entry point; run the main function
main
```

© 2018 Altova GmbH

Exemple .NET : C#

L'exemple C# ci-dessous effectue les choses suivantes :

- Configurer et initialiser l'objet NET RaptorXML
- Valider un fichier XML
- Effectuer une transformation XSLT, retourner le résultat en tant que string
- Traiter un document XQuery, enregistrer le résultat dans un fichier
- Configurer la séquence d'exécution du script et de son point d'entrée

```
using System;
using System.Text;
using Altova.RaptorXMLServer;

namespace RaptorXMLRunner
{
    class Program
    {
    // The RaptorXML Server .NET object
       static ServerClass objRaptorXMLServer;

    // Initialize the RaptorXML Server .NET object
       static void Init()
       {
       // Allocate a RaptorXML Server object
          objRaptorXMLServer = new ServerClass();

       // Configure the server: error reporting, HTTP server name and port
       // (IPv6 localhost in this example)
          objRaptorXMLServer.ErrorLimit = 1;
          objRaptorXMLServer.ReportOptionalWarnings = true;
          objRaptorXMLServer.ServerName = "::1"
          objRaptorXMLServer.ServerPort = 8087
       }




       // Validate one file
          static void ValidateXML()
          {
          // Get a validator engine instance from the Server object
             XMLValidator objXMLValidator =
objRaptorXMLServer.GetXMLValidator();

          // Configure input data
             objXMLValidator.InputFileName = "MyXMLFile.xml";

          // Validate; in case of invalid file,
             report the problem returned by RaptorXML
```

```
      if ( objXMLValidator.IsValid() )
         Console.WriteLine( "Input string is valid" );
      else
         Console.WriteLine( objXMLValidator.LastErrorMessage );
   }




// Perform an XSLT transformation, and
// return the result as a string
   static void RunXSLT()
   {
   // Get an XSLT engine instance from the Server object
      XSLT objXSLT = objRaptorXMLServer.GetXSLT();

      // Configure input data
         objXSLT.InputXMLFileName = "MyXMLFile.xml";
         objXSLT.XSLFileName = "MyTransformation.xsl";

     // Run the transformation.
     // In case of success, the result is returned.
     // In case of errors, an error listing
        Console.WriteLine( objXSLT.ExecuteAndGetResultAsString() );
   }




// Execute an XQuery, save the result in a file
   static void RunXQuery()
   {
   // Get an XQuery engine instance from the Server object
      XQuery objXQuery = objRaptorXMLServer.GetXQuery();

     // Configure input data
        objXQuery.InputXMLFileName = exampleFolder + "simple.xml";
        objXQuery.XQueryFileName = exampleFolder + "CopyInput.xq";

    // Configure serialization (optional, for better formatting)
       objXQuery.OutputEncoding = "UTF8"
       objXQuery.OutputIndent = true
       objXQuery.OutputMethod = "xml"
       objXQuery.OutputOmitXMLDeclaration = false

    // Run the query; result serialized to given path
       objXQuery.Execute( "MyQueryResult.xml" );
   }
```

```
            static void Main(string[] args)
            {
              try
              {
              // Entry point. Perform all functions
                 Init();
                 ValidateXML();
                 RunXSLT();
                 RunXQuery();
              }
              catch (System.Exception ex)
              {
                  Console.WriteLine( ex.Message );
                  Console.WriteLine( ex.ToString() );
              }
          }
      }
  }
```

Exemple .NET : Visual Basic .NET

L'exemple Visual Basic ci-dessous effectue les choses suivantes :

- Configurer et initialiser l'objet .NET RaptorXML
- Valider un fichier XML
- Effectuer une transformation XSLT, retourner le résultat en tant que string
- Traiter un document XQuery, enregistrer le résultat dans un fichier
- Configurer la séquence d'exécution du code et de son point d'entrée

```
Option Explicit On
Imports Altova.RaptorXMLServer

Module RaptorXMLRunner

   ' The RaptorXML .NET object
     Dim objRaptor As Server

   ' Initialize the RaptorXML .NET object
     Sub Init()

      ' Allocate a RaptorXML object
        objRaptor = New Server()

      ' Configure the server: error reporting, HTTP server name and port (IPv6
localhost in this example)
        objRaptor.ErrorLimit = 1
        objRaptor.ReportOptionalWarnings = True
        objRaptor.ServerName = "::1"
        objRaptor.ServerPort = 8087
     End Sub




   ' Validate one file
     Sub ValidateXML()

      ' Get a validator instance from the RaptorXML object
        Dim objXMLValidator As XMLValidator
        objXMLValidator = objRaptor.GetXMLValidator()

      ' Configure input data
        objXMLValidator.InputFileName = "MyXMLFile.xml"

      ' Validate; in case of invalid file report the problem returned by
RaptorXML
        If (objXMLValidator.IsValid()) Then
           Console.WriteLine("Input string is valid")
        Else
           Console.WriteLine(objXMLValidator.LastErrorMessage)
        End If
     End Sub
```

 © 2018 Altova GmbH

```
  ' Perform a transformation; return the result as a string
    Sub RunXSLT()

      ' Get an XSLT engine instance from the Server object
        Dim objXSLT As XSLT
        objXSLT = objRaptor.GetXSLT()

      ' Configure input data
        objXSLT.InputXMLFileName = "MyXMLFile.xml"
        objXSLT.XSLFileName = "MyTransformation.xsl"

      ' Run the transformation; in case of success the result will be returned,
in case of errors the engine returns an error listing
        Console.WriteLine(objXSLT.ExecuteAndGetResultAsString())
    End Sub



  ' Execute an XQuery; save the result in a file
    Sub RunXQuery()

      ' Get an XQuery engine instance from the Server object
        Dim objXQ As XQuery
        objXQ = objRaptor.GetXQuery()

      ' Configure input data
        objXQ.InputXMLFileName = "MyXMLFile.xml"
        objXQ.XQueryFileName = "MyQuery.xq"

      ' Configure serialization (optional - for fine-tuning the result's
formatting)
        objXQ.OutputEncoding = "UTF8"
        objXQ.OutputIndent = true
        objXQ.OutputMethod = "xml"
        objXQ.OutputOmitXMLDeclaration = false

      ' Run the query; the result will be serialized to the given path
        objXQ.Execute( "MyQueryResult.xml" )
    End Sub



    Sub Main()
      ' Entry point; perform all sample functions
        Init()
        ValidateXML()
        RunXSLT()
        RunXQuery()
    End Sub
```

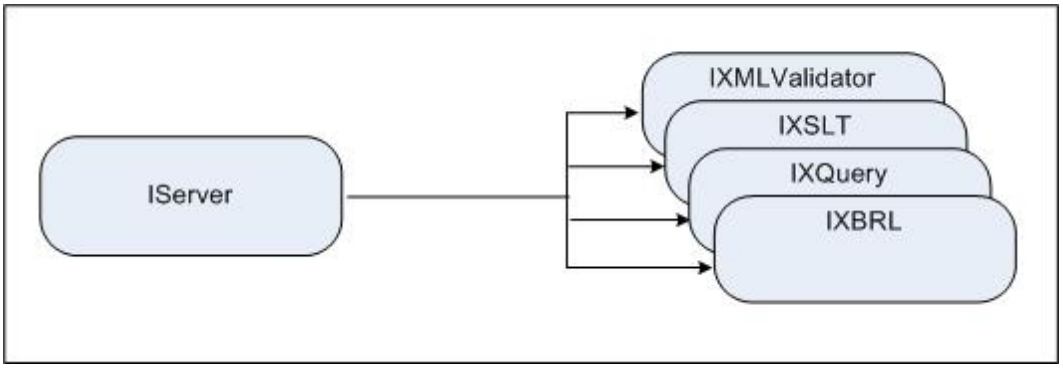
```
End Module
```


© 2018 Altova GmbH

7.4 Référence API

Cette section décrit la spécification API : son modèle d'objet et les détails de ses interfaces et énumérations.

Le point de départ pour l'utilisation de la fonction de RaptorXML est l'interface `IServer`. Cet objet contient les objets qui fournissent la fonction RaptorXML : validation XML, validation XBRL, transformations XSLT et traitement de document XQuery. Le modèle d'objet de l'API RaptorXML est illustré dans le diagramme suivant.



La hiérarchie du modèle d'objet est indiquée ci-dessous et les interfaces sont décrites en détail dans les sections correspondantes. Les méthodes et les propriétés de chaque interface sont décrites dans la section pour cette interface.

```
-- IServer
  |-- IXMLValidator
  |-- IXSLT
  |-- IXQuery
  |-- IXBRL
```

Interfaces

Les interfaces suivantes sont définies. Elles sont décrites dans les sous-sections de cette section.

IServer
IXMLValidator
IXSLT
IXQuery
IXBRL

 © 2018 Altova GmbH

IServer

L'interface **IServer** propose des méthodes pour retourner les interfaces du moteur RaptorXML respectif : XML Validator, XBRL, XSLT et XQuery. Les propriétés définissent les paramètres de l'interface.

Méthodes

Les méthodes de l'interface **IServer** retournent des interfaces du moteur RaptorXML respectif : XML Validator, XBRL, XSLT et XQuery.

- GetXMLValidator

  IXMLValidator **GetXMLValidator()**
  Retourne une instance du moteur XML Validator.

- GetXBRL

  IXBRL **GetXBRL()**
  Retourne une instance du moteur XBRL.

- GetXSLT

  IXSLT **GetXSLT()**
  Retourne une instance du moteur XSLT.

- GetXQuery

  IXQuery **GetXQuery()**
  Retourne une instance du moteur XQuery.

Propriétés

Les propriétés de l'interface **IServer** sont décrites ci-dessous dans l'ordre alphabétique. Veuillez noter que les entrées de string à interpréter en tant qu'URL doivent fournir des chemins absolus. Si un chemin relatif est utilisé, un mécanisme pour résoudre le chemin relatif doit être défini dans le module d'appel.

- APIMajorVersion

  int **APIMajorVersion**
  Retourne la version majeure de l'API en tant qu'un entier. La version majeure de l'API peut différer de la version majeure du produit si l'API est connectée à un autre serveur.

- APIMinorVersion

  int **APIMinorVersion**

Retourne la version mineure de l'API en tant qu'un entier. La version mineure de l'API peut différer de la version mineure du produit si l'API est connectée à un autre serveur.

- APIServicePackVersion

  `int` **`APIServicePackVersion`**
  Retourne la version du pack de service de l'API en tant qu'un entier. La version du pack de service de l'API peut différer de la version du pack de service du produit si l'API est connectée à un autre serveur.

- ErrorFormat

  `ENUMErrorFormat` **`ErrorFormat`**
  Définit le formant d'erreur RaptorXML sur un des littéraux `ENUMErrorFormat` (Text, ShortXML, LongXML).

- ErrorLimit

  `int` **`ErrorLimit`**
  Définit la limite d'erreur de validation de RaptorXML. Le paramètre `limit` est de type `int` (Java), `uint` (COM/.NET). Il spécifie le nombre d'erreurs à rapporter avant que l'exécution soit stoppée. Utiliser `-1` pour que `limit` ne soit pas limité (c'est à dire que toutes les erreurs seront rapportées). La valeur par défaut est `100`.

- GlobalCatalog

  `string` **`GlobalCatalog`**
  Définit l'emplacement, en tant qu'une URL, du fichier de catalogue (point-entrée) principal. Le string fourni doit être une URL absolue qui donne l'emplacement exact du fichier de catalogue principal à utiliser.

- GlobalResourceConfig

  `string` **`GlobalResourceConfig`**
  Définit la configuration active de la ressource globale. Le paramètre `config` est de type `String`, et spécifie le nom de la configuration utilisée par la ressource globale active.

- GlobalResourceFile

  `string` **`GlobalResourceFile`**
  Définit l'emplacement, en tant qu'une URL, du fichier XML de Ressources globales. Le string fourni doit être une URL absolue qui donne l'emplacement exact du fichier XML de Ressources globales.

- MajorVersion

  `int` **`MajorVersion`**
  Retourne la version majeure du produit en tant qu'un entier. *Exemple :* Pour `Altova RaptorXML+XBRL Server 2014r2sp1(x64)`, retourne `16` (la différence entre la version majeure (`2014`) et l'année initiale `1998`). Si une erreur se produit, a **`RaptorXMLException`** est soulevée (*Java*).

- MinorVersion

  `int` **`MinorVersion`**

 © 2018 Altova GmbH

Retourne la version mineure du produit en tant qu'un entier. *Exemple :* Pour `Altova RaptorXML+XBRL Server 2018r2sp1(x64)`, retourne `2` (depuis la version mineure `r2`). Si une erreur se produit, a **`RaptorXMLException`** est soulevée (*détails d'interface Java*).

- ProductName

  `string` **`ProductName`**
  Retourne le nom du produit en tant que string. *Exemple :* Pour `Altova RaptorXML+XBRL Server 2018r2sp1(x64)`, retourne `Altova RaptorXML+XBRL Server`. Si une erreur se produit, une **`RaptorXMLException`** est soulevée (*détails d'interface Java*).

- ProductNameAndVersion

  `string` **`ProductNameAndVersion`**
  Retourne la version de pack de service du produit en tant qu'un entier. *Exemple :* pour `Altova RaptorXML+XBRL Server 2018r2sp1(x64)`, retourne `Altova RaptorXML+XBRL Server 2018r2sp1(x64)`. Si une erreur se produit, une **`RaptorXMLException`** est soulevée (*détails d'interface Java*).

- ReportOptionalWarnings

  `bool` **`ReportOptionalWarnings`**
  Active/désactive le rapport des avertissements. Une valeur de `true` active les avertissements ; `false` les désactive.

- ServerName

  `string` **`ServerName`**
  Définit le nom du serveur HTTP. Si une erreur se produit, une **`RaptorXMLException`** est soulevée (*détails d'interface Java*). Le paramètre d'entrée est un string qui donne au nom du serveur HTTP.

- ServerPath

  `string` **`ServerPath`**
  Spécifie, sous la forme d'une URL, le chemin vers le serveur HTTP. Une `RaptorXMLException` est soulevée en cas d'erreur.

- ServerPort

  `int` **`ServerPort`**
  Spécifie le port de serveur HTTP. Le type est `ushort`. Une `RaptorXMLException` est soulevée en cas d'erreur.

- ServicePackVersion

  `int` **`ServicePackVersion`**
  Retourne la version du pack de service du produit en tant qu'un entier. *Exemple :* pour `RaptorXML+XBRL Server 2018r2sp1(x64)`, retourne `1` (du pack de service du numéro de version `sp1`). Si une erreur se produit, une **`RaptorXMLException`** est soulevée (*détails d'interface Java*).

- UserCatalog

  `string` **`UserCatalog`**

Définit l'emplacement, en tant qu'une URL, du fichier de catalogue d'utilisateur personnalisé. Le string fourni doit être une URL absolue qui donne l'emplacement exact du fichier de catalogue d'utilisateur personnalisé.

 © 2018 Altova GmbH

IXMLValidator

Méthodes

L'interface **IXMLValidator** propose des méthodes. Toutes ces méthodes retournent une valeur booléenne `True` ou `False` selon que l'opération a réussi ou pas.

- ExtractAvroSchema

  bool ExtractAvroSchema(string **outputPath**)

  Extrait un schéma Avro depuis un fichier binaire. Le paramètre `outputPath` est une URL absolue qui spécifie l'emplacement de sortie. Le résultat est `true` en cas de succès, `false` en cas d'échec. Si une erreur se produit, une **RaptorXMLException** est soulevée (*détails d'interface Java*). Utiliser `getLastErrorMessage` (***dans Java***), ou `LastErrorMessage` (***dans COM/.NET***) pour accéder aux informations supplémentaires.

- IsValid

  bool IsValid(ENUMValidationType **type**)

  Retourne le résultat de validation du document XML, le document de schéma, ou le document DTD. Le type de document à valider est spécifié par le paramètre `type`, qui prend un littéral `ENUMValidationType` (Java, COM/.NET) en tant que cette valeur. Le résultat est `true` en cas de succès, `false` en cas d'échec. Si une erreur se produit, une **RaptorXMLException** est soulevée (*détails d'interface Java*). Utiliser `getLastErrorMessage` (***dans Java***), ou `LastErrorMessage` (***dans COM/.NET***) pour accéder des informations supplémentaires.

- IsWellFormed

  bool IsWellFormed(ENUMWellformedCheckType **type**)

  Retourne le résultat du contrôle du document XML ou du document DTD de bonne formation. Le type de document à contrôler est spécifié avec le paramètre `type`, qui prend un littéral `ENUMWellformedCheckType` (Java, COM/.NET) en tant que sa valeur. Le résultat est `true` en cas de succès, `false` en cas d'échec. Si une erreur se produit, une **RaptorXMLException** est soulevée (*détails d'interface Java*). Utiliser `getLastErrorMessage` (***dans Java***), ou `LastErrorMessage` (***dans COM/.NET***) pour accéder des informations supplémentaires.

Propriétés

Les propriétés de l'interface **IXMLValidator** sont décrites ci-dessous dans l'ordre alphabétique. Veuillez noter que les entrées de string à interpréter en tant qu'URL doivent fournir des chemins absolus. Si un chemin relatif est utilisé, un mécanisme pour résoudre le chemin relatif doit être défini dans le module d'appel.

- AssessmentMode

  ENUMAssessmentMode **AssessmentMode**

  Définit le mode d'évaluation de la validation XML (`Strict/Lax`), qui est donnée par un littéral `ENUMAssessmentMode` (Java, COM/.NET).

AvroSchemaFileName

string AvroSchemaFileName

Définit l'emplacement, en tant qu'URL, du Schéma Avro externe à utiliser. Le string fournit doit être une URL absolue qui donne l'emplacement exact du fichier de schéma Avro.

AvroSchemaFromText

string AvroSchemaFromText

Apporte le contenu du document de Schéma Avro à utiliser. Le string fourni est le contenu du document de Schéma Avro à utiliser.

DTDFileName

string DTDFileName

Définit l'emplacement en tant qu'une URL, du document DTD à utiliser pour la validation. Le string fourni doit être une URL absolue qui indique l'emplacement exact du document DTD.

DTDFromText

string DTDFromText

Apporte le contenu du document DTD à utiliser pour la validation. Le string fourni est le contenu du document DTD à utiliser.

EnableNamespaces

bool EnableNamespaces

Active le traitement reconnaissant l'espace de noms. Cela est utile pour contrôler la présence d'erreurs dues à des espaces de noms incorrects dans l'instance XML. Une valeur `true` permet le traitement reconnaissant les espaces de noms ; une valeur `false` la désactive. La valeur par défaut est `false`.

InputFileArray

object InputFileArray

Fournit un tableau des URL des fichiers XML à utiliser en tant que données d'entrée. La propriété fournit un objet contenant, en tant que strings, les URL absolus de chaque fichiers XML.

InputFileCollection

string InputFileCollection

Fournit la collection de fichiers XML qui sera utilisée en tant que données d'entrée. Les fichiers sont identifiés par leurs URL. Une collection de strings est soumise ; chaque string est l'URL absolue d'un fichier XML d'entrée.

InputFileName

string InputFileName

 © 2018 Altova GmbH

Définit l'emplacement, en tant qu'une URL, des données XML d'entrées à traiter. Le string fourni doit être une URL absolue qui donne l'emplacement du fichier d'entrée.

- InputFromText

  `string InputFromText`

  Apporte le contenu du document XML à traiter. Le string fourni est le contenu du document XML à traiter.

- InputTextArray

  `object InputTextArray`

  Fournit un tableau des URL des fichiers texte à utiliser en tant que données d'entrée. La propriété fournit un objet contenant, en tant que strings, les URL absolus de chaque fichiers texte.

- InputTextCollection

  `string InputTextCollection`

  Apporte le contenu de plusieurs fichiers XML qui seront utilisés en tant que données d'entrée. Une collection de strings, chacun d'entre eux est le contenu d'un fichier XML d'entrée.

- InputXMLFileCollection (OBSOLÈTE. Utiliser InputFileCollection à la place.)

  `string InputXMLFileCollection`

  Apporte la collection des fichiers XML qui seront utilisés en tant que données d'entrée. Les fichiers sont identifiés par leurs URL. Une collection de strings, chacun d'entre eux est l'URL absolue d'un fichier XML d'entrée.

- InputXMLFileName (OBSOLÈTE. Utiliser InputFileName à la place.)

  `string InputXMLFileName`

  Définit l'emplacement, en tant qu'une URL, du document XML à traiter. Le string fourni doit être une URL absolue qui donne l'emplacement exact du fichier XML.

- InputXMLFromText (OBSOLÈTE. Utiliser InputFromText à la place.)

  `string InputXMLFromText`

  Apporte le contenu du document XML à traiter. Le string fourni est le contenu du document XML à traiter.

- InputXMLTextCollection(OBSOLÈTE. Utiliser InputTextCollection à la place.)

  `string InputXMLTextCollection`

  Apporte le contenu de plusieurs fichiers XML qui seront utilisés en tant que données d'entrée. Une collection de strings, chacun d'entre eux est le contenu d'un fichier XML d'entrée.

- LastErrorMessage

  string LastErrorMessage
  Récupère un string qui est le dernier message d'erreur provenant du moteur RaptorXML.

- ParallelAssessment

  bool ParallelAssessment
  Active/désactive estimation de validité de schéma parallèle.

- PythonScriptFile

  string PythonScriptFile
  Définit l'emplacement du fichier de script Python. Le string fourni doit être une URL absolue qui donne l'emplacement exact du fichier Python.

- SchemaFileArray

  object **SchemaFileArray**
  Fournit la collection des fichiers de Schéma XML qui sera utilisée en tant que Schémas XML externe. Les fichiers sont identifiés par leurs URL. L'entrée est une collection de strings, chacun d'entre eux est l'URL absolue d'un fichier de Schéma XML.

- SchemaFileName

  string SchemaFileName(*String* **filePath)**
  Définit l'emplacement, en tant qu'une URL, du document de Schéma XML à utiliser pour la validation. Le string fourni doit être une URL absolue qui donne l'emplacement exact du fichier de Schéma XML.

- SchemaFromText

  string SchemaFromText
  Fournit les contenus du document de Schéma XML à utiliser. Le string fourni est le contenu du document de Schéma XML à utiliser.

- SchemaImports

  ***ENUMSchemaImports* SchemaImports**
  Spécifie comment traiter les importations de schéma sur la base des valeurs d'attribut des éléments `xs:import`. Le type de gestion est spécifié par le littéral `ENUMSchemaImports` (Java, COM/.NET) qui est soumis.

- SchemalocationHints

  ***ENUMLoadSchemalocation* SchemalocationHints**

 © 2018 Altova GmbH

Spécifie le mécanisme à utiliser pour situer le schéma. Le mécanisme est spécifié par le littéral `ENUMLoadSchemalocation` (Java, COM/.NET) qui a été sélectionné.

- SchemaMapping

  ENUMSchemaMapping **SchemaMapping**
  Définit quel mappage utiliser pour localiser le schéma. Le mappage est spécifié par le littéral `ENUMSchemaMapping` (Java, COM/.NET) qui est sélectionné.

- SchemaTextArray

  object SchemaTextArray
  Fournit le contenu de plusieurs fichiers de Schéma XML. L'entrée est une collection de strings, dont chacun est le contenu d'un document de Schéma XML.

- Streaming

  bool Streaming
  Active la validation de streaming. Dans le mode streaming, les données qui sont stockées en mémoire sont minimisées et le traitement est plus rapide. Une valeur de `true` active le streaming; `false` le désactive. Le défaut est `true`.

- XIncludeSupport

  bool XIncludeSupport
  Active ou désactive l'utilisation des éléments `XInclude`. Une valeur de `true` active la prise en charge XInclude ; `false` le désactive. La valeur par défaut est `false`.

- XMLValidationMode

  ENUMXMLValidationMode **XMLValidationMode**
  Définit le mode de validation XML, qui est une énumération littérale de `ENUMXMLValidationMode` (Java, COM/.NET) qui détermine s'il faut contrôler la validité ou la bonne formation.

- XSDVersion

  ENUMXSDVersion **XSDVersion**
  Définit la version de schéma XML par rapport à laquelle le document XML sera validé. Est une énumération littérale de `ENUMXSDVersion` (Java, COM/.NET).

IXSLT

L'interface **IXSLT** propose des méthodes et des propriétés pour exécuter une transformation XSLT 1.0, XSLT 2.0 ou XSLT 3.0. Les résultats peuvent être enregistrés sur un fichier ou retournés en tant que string. L'interface permet aussi aux paramètres XSLT d'être passés dans la feuille de style XSLT. Les URL des fichiers XML et XSLT peuvent être fournis en tant que strings par le biais des propriétés de l'interface. En alternative, les documents XML et XSLT peuvent être établis dans le code en tant que strings de texte.

Note : Lorsque des entrées de string doivent être interprétées en tant qu'URL, des chemins absolus doivent être utilisés. Si un chemin relatif est utilisé, un mécanisme pour résoudre le chemin relatif doit être défini dans le module d'appel.

Note : Le moteur XSLT 2.0 ou 3.0 de RaptorXML peut être utilisé dans son mode de rétrocompatibilité pour traiter une feuille de style XSLT 1.0. La sortie, néanmoins, pourrait être différente de celle produite par le moteur XSLT 1.0 traitant la même feuille de style XSLT 1.0.

Méthodes

Les méthodes de l'interface **IXSLT** sont décrites ci-dessous. Veuillez noter que les entrées de string qui doivent être interprétées en tant qu'URL doivent fournir des chemins absolus. Si un chemin relatif est utilisé, un mécanisme pour résoudre le chemin relatif doit être défini dans le module d'appel.

IsValid

```
bool IsValid()
```

- Retourne le résultat de la validation de la feuille de style XSLT conformément à la spécification XSLT nommée dans `ENUMXSLTVersion` (voir la propriété `EngineVersion`). Le résultat est `true` en cas de succès, `false` en cas d'échec.
- Si une erreur se produit, une **RaptorXMLException** est soulevée. Utiliser l'opération `LastErrorMessage` pour accéder aux informations supplémentaires.

Execute

```
bool Execute(string outputPath)
```

- Exécute la transformation XSLT conformément à la spécification XSLT nommée dans `ENUMXSLTVersion` (voir la propriété `EngineVersion`), et enregistre le résultat dans un fichier de sortie.
- Le fichier de sortie est défini par `outputPath`, qui est un string qui fournit l'URL du fichier de sortie.
- Le résultat est `true` en cas de succès, `false` en cas d'échec.
- Si une erreur se produit, une **RaptorXMLException** est soulevée. Utiliser l'opération `LastErrorMessage` pour accéder aux informations supplémentaires.

ExecuteAndGetResultAsString

```
bool Execute()
```

- Exécute la transformation XSLT conformément à la spécification XSLT nommée dans `ENUMXSLTVersion` (voir la propriété `EngineVersion`), et retourne le résultat de

 © 2018 Altova GmbH

transformation en tant que string.

- Cette méthode ne produit pas de fichiers de résultat supplémentaires, comme des graphiques ou des résultats secondaires. Elle ne contient pas non plus des résultats binaires comme des fichiers `.docx` OOXML. Si des fichiers de sortie supplémentaires sont nécessaires, utiliser la méthode Exécuter.
- Si une erreur se produit, une **RaptorXMLException** est soulevée. Utiliser l'opération LastErrorMessage pour accéder aux informations supplémentaires.

▼ ExecuteAndGetResultAsStringWithBaseOutputURI

```
bool Execute(string baseURI)
```

- Exécute la transformation XSLT conformément à la spécification XSLT nommée dans ENUMXSLTVersion (voir la propriété EngineVersion), et retourne le résultat de transformation en tant que string à l'emplacement défini par le paramètre `baseURI`.
- Cette méthode ne produit pas de fichiers de résultat supplémentaires, comme des graphiques ou des résultats secondaires. Elle ne contient pas non plus des résultats binaires comme des fichiers `.docx` OOXML. Si des fichiers de sortie supplémentaires sont nécessaires, utiliser la méthode Exécuter.
- Si une erreur se produit, une **RaptorXMLException** est soulevée. Utiliser l'opération LastErrorMessage pour accéder aux informations supplémentaires.

▼ AddExternalParameter

```
void AddExternalParameter(string paramName, string paramValue)
```

- Ajoute le nom et la valeur d'un paramètre externe : `paramName` et `paramValue` sont des strings.
- Chaque paramètre externe et sa valeur doivent être spécifiés dans un appel séparé de la méthode. Les paramètres doivent être déclarés dans le document XSLT, en option avec une déclaration de type. Quelle que soit la déclaration de type dans le document XSLT, aucun délimiteur spécial n'est nécessaire lorsque la valeur de paramètre est soumise avec `AddExternalParameter`.

▼ ClearExternalParameterList

```
void ClearExternalParameterList()
```

- Supprime la liste des paramètres externes créée avec la méthode AddExternalParameter.

Propriétés

Les propriétés de l'interface **IXSLT** sont décrites ci-dessous dans l'ordre alphabétique. Veuillez noter que les entrées de string qui doivent être interprétées en tant qu'URL doivent fournir des chemins absolus. Si un chemin relatif est utilisé, un mécanisme pour résoudre le chemin relatif doit être défini dans le module d'appel.

▼ AdditionalOutputs

```
string AdditionalOutputs
```

Retourne les sorties supplémentaires de la dernière tâche exécutée.

- ChartExtensionsEnabled

  bool ChartsExtensionsEnabled
  Active ou désactive les fonctions d'extension de graphique d'Altova. Une valeur de `true` active les extensions de graphique ; `false` les désactive. La valeur par défaut est `true`.

- DotNetExtensionsEnabled

  bool DotNetExtensionsEnabled
  Active ou désactive les fonctions d'extension de .NET. Une valeur de `true` active les extensions .NET ; `false` les désactive. La valeur par défaut est `true`.

- EngineVersion

  ***ENUMXSLTVersion* EngineVersion**
  Spécifie la version XSLT à utiliser (1.0, 2.0 ou 3.0). La valeur de propriété est un littéral `ENUMXSLTVersion`.

- IndentCharacters

  string IndentCharacters
  Soumet le string de caractère qui sera utilisé en tant que retrait dans la sortie.

- InitialTemplateMode

  string InitialTemplateMode
  Définit le mode initial pour le traitement XSLT. Les modèles présentant une valeur de mode égale au string soumis seront traités.

- InputXMLFileName

  string InputXMLFileName
  Définit l'emplacement, en tant qu'une URL, du document XML à traiter. Le string fourni doit être une URL absolue qui donne l'emplacement exact du fichier XML.

- InputXMLFromText

  string InputXMLFromText
  Apporte le contenu du document XML à traiter. Le string fourni est le contenu du document XML à traiter.

- JavaBarcodeExtensionLocation

  string JavaBarcodeExtensionLocation
  Spécifie l'emplacement du fichier d'extension du code-barres. Voir la section sur les

 © 2018 Altova GmbH

fonctions d'extension du code-barres d'Altova pour plus d'informations. Le string fourni doit être une URL absolue qui donne l'emplacement de base du fichier à utiliser.

- JavaExtensionsEnabled

  bool JavaExtensionsEnabled
  Active ou désactive des fonctions d'extension Java. Une valeur de `true` active des extensions Java ; `false` les désactive. La valeur par défaut est `true`.

- LastErrorMessage

  string LastErrorMessage
  Récupère un string qui est le dernier message d'erreur provenant du moteur RaptorXML.

- LoadXMLWithPSVI

  bool LoadXMLWithPSVI
  Active la validation des fichiers XML d'entrée et génère des infos post-schema-validation pour les fichiers. Une valeur de `true` active la validation XML et génère des infos post-schema-validation pour les fichiers XML ; `false` désactive la validation. La valeur par défaut est `true`.

- MainOutput

  string MainOutput
  Retourne la sortie principale de la dernière tâche exécutée.

- NamedTemplateEntryPoint

  string NamedTemplateEntryPoint
  Spécifie le nom, en tant que string, du modèle nommé à utiliser en tant que point d'entrée pour la transformation.

- SchemaImports

  ***ENUMSchemaImports* SchemaImports**
  Spécifie comment traiter les importations de schéma sur la base des valeurs d'attribut des éléments `xs:import`. Le type de gestion est spécifié par le littéral `ENUMSchemaImports` (Java, COM/.NET) qui est soumis.

- SchemalocationHints

  ***ENUMLoadSchemalocation* SchemalocationHints**
  Spécifie le mécanisme à utiliser pour situer le schéma. Le mécanisme est spécifié par le littéral `ENUMLoadSchemalocation` (Java, COM/.NET) qui a été sélectionné.

- SchemaMapping

  ***ENUMSchemaMapping* SchemaMapping**

Définit quel mappage utiliser pour localiser le schéma. Le mappage est spécifié par le littéral `ENUMSchemaMapping` (Java, COM/.NET) qui est sélectionné.

- StreamingSerialization

`bool StreamingSerialization`
Active la sérialisation de streaming. Dans le mode streaming, les données stockées dans la mémoire sont minimisées et le traitement est plus rapide. Une valeur de `true` active la sérialisation de streaming ; `false` la désactive.

- XincludeSupport

`bool XincludeSupport`
Active ou désactive l'utilisation des éléments `XInclude`. Une valeur de `true` active la prise en charge XInclude ; `false` le désactive. La valeur par défaut est `false`.

- XMLValidationErrorAsWarning

`bool XMLValidationErrorAsWarning`
Active le traitement des erreurs de validation XML en tant qu'avertissements. Prend la booléenne `true` ou `false`.

- XMLValidationMode

`ENUMXMLValidationMode` **`XMLValidationMode`**
Définit le mode de validation XML, qui est une énumération littérale de `ENUMXMLValidationMode` (Java, COM/.NET) qui détermine s'il faut contrôler la validité ou la bonne formation.

- XSDVersion

`ENUMXSDVersion` **`XSDVersion`**
Définit la version de schéma XML par rapport à laquelle le document XML sera validé. Est une énumération littérale de `ENUMXSDVersion` (Java, COM/.NET).

- XSLFileName

`string XSLFileName`
Spécifie le fichier XSLT à utiliser pour la transformation. Le string fourni doit être une URL absolue qui donne l'emplacement du fichier XSLT à utiliser.

- XSLFromText

`string XSLFromText`
Fournit, en tant que string de texte, les contenus du document XSLT à utiliser pour la transformation.

 © 2018 Altova GmbH

IXQuery

L'interface **IXQuery** propose des méthodes et des propriétés pour exécuter un document XQuery 1.0 ou XQuery 3.0. Les résultats peuvent être enregistrés sur un fichier ou retournés en tant que string. L'interface permet aussi aux variables XQuery externes d'être passées dans le document XQuery. Les URL des fichiers XQuery et XML peuvent être fournis en tant que strings par le biais des propriétés de l'interface. En alternative, les documents XML et XQuery peuvent être établis dans le code en tant que strings de texte.

Note : Lorsque des entrées de string doivent être interprétées en tant qu'URL, des chemins absolus doivent être utilisés. Si un chemin relatif est utilisé, un mécanisme pour résoudre le chemin relatif doit être défini dans le module d'appel.

Méthodes

Les méthodes de l'interface **IXQuery** sont décrites ci-dessous. Veuillez noter que les entrées de string qui doivent être interprétées en tant qu'URL doivent fournir des chemins absolus. Si un chemin relatif est utilisé, un mécanisme pour résoudre le chemin relatif doit être défini dans le module d'appel.

IsValid

```
bool IsValid()
```

- Retourne le résultat de la validation du document XQuery conformément à la spécification XQuery nommée dans ENUMXQueryVersion (voir la propriété EngineVersion). Le résultat est `true` en cas de succès, `false` en cas d'échec.
- Si une erreur se produit, une **RaptorXMLException** est soulevée. Utiliser l'opération LastErrorMessage pour accéder aux informations supplémentaires.

IsValidUpdate

```
bool IsValidUpdate()
```

- Retourne le résultat de la validation du document XQuery Update conformément à la spécification XQuery nommée dans ENUMXQueryVersion (voir la propriété EngineVersion). Le résultat est `true` en cas de succès, `false` en cas d'échec.
- Si une erreur se produit, une **RaptorXMLException** est soulevée. Utiliser l'opération LastErrorMessage pour accéder aux informations supplémentaires.

Execute

```
bool Execute(string outputFile)
```

- Exécute le document XQuery conformément à la spécification XQuery nommée dans ENUMXQueryVersion (voir la propriété EngineVersion), et enregistre le résultat dans un fichier de sortie.
- Le fichier de sortie est défini par `outputFile`, qui est un string qui fournit l'URL du fichier de sortie.
- La valeur booléenne `true` est retournée en cas de succès, la valeur `false` en cas d'échec.
- Si une erreur se produit, une **RaptorXMLException** est soulevée. Utiliser l'opération LastErrorMessage pour accéder aux informations supplémentaires

 © 2018 Altova GmbH

ExecuteAndGetResultAsString

```
string ExecuteAndGetResultAsString()
```

- Exécute la transformation XQuery conformément à la spécification XQuery nommée dans `ENUMXQueryVersion` (voir la propriété `EngineVersion`), et retourne le résultat de transformation en tant que string.
- Cette méthode ne produit pas de fichiers de résultat supplémentaires, comme des graphiques ou des résultats secondaires. Elle ne contient pas non plus des résultats binaires comme des fichiers `.docx` OOXML. Si des fichiers de sortie supplémentaires sont nécessaires, utiliser la méthode `Exécuter`.
- Si une erreur se produit pendant la transformation, une **`RaptorXMLException`** est soulevée. Utiliser l'opération `LastErrorMessage` pour accéder aux informations supplémentaires.

ExecuteUpdate

```
bool ExecuteUpdate(string outputFile)
```

- Exécute la mise à jour XQuery conformément à la spécification XQuery nommée dans `ENUMXQueryVersion` (voir la propriété `EngineVersion`), et retourne le résultat dans un fichier de sortie.
- Le fichier de sortie est défini par `outputFile`, qui est un string qui fournit l'URL du fichier de sortie.
- La valeur booléenne `true` est retournée en cas de succès, la valeur `false` en cas d'échec.
- Si une erreur se produit pendant la transformation, une **`RaptorXMLException`** est soulevée. Utiliser l'opération `LastErrorMessage` pour accéder aux informations supplémentaires.

ExecuteUpdateAndGetResultAsString

```
string ExecuteAndGetResultAsString()
```

- Exécute la mise à jour XQuery conformément à la spécification XQuery Update nommée dans `ENUMXQueryVersion` (voir la propriété `EngineVersion`), et retourne le résultat de transformation en tant que string.
- Cette méthode ne produit pas de fichiers de résultat supplémentaires, comme des graphiques ou des résultats secondaires. Elle ne contient pas non plus des résultats binaires comme des fichiers `.docx` OOXML. Si des fichiers de sortie supplémentaires sont nécessaires, utiliser la méthode `Exécuter`.
- Si une erreur se produit pendant la transformation, une **`RaptorXMLException`** est soulevée. Utiliser l'opération `LastErrorMessage` pour accéder aux informations supplémentaires.

AddExternalVariable

```
void AddExternalVariable(string varName, string varValue)
```

- Ajoute le nom et la valeur d'une variable externe : `varName` et `varValue` sont des strings.
- Chaque variable externe et sa valeur doivent être spécifiés dans un appel séparé de la méthode. Les variables doivent être déclarées dans le document XQuery, en option avec une déclaration de type. Si la variable est un string, la variable doit être contenue

dans des guillemets simples.

- ClearExternalVariableList

  `void ClearExternalVariableList()`

  - Supprime la liste des paramètres externes créée avec la méthode AddExternalVariable.

Propriétés

Les propriétés de l'interface **IXQuery** sont décrites ci-dessous dans l'ordre alphabétique. Veuillez noter que les entrées de string qui doivent être interprétées en tant qu'URL doivent fournir des chemins absolus. Si un chemin relatif est utilisé, un mécanisme pour résoudre le chemin relatif doit être défini dans le module d'appel.

- ChartExtensionsEnabled

  `bool ChartsExtensionsEnabled`

  Active ou désactive les fonctions d'extension de graphique d'Altova. Une valeur de `true` active les extensions de graphique ; `false` les désactive. La valeur par défaut est `true`.

- DotNetExtensionsEnabled

  `bool DotNetExtensionsEnabled`

  Active ou désactive les fonctions d'extension de .NET. Une valeur de `true` active les extensions .NET ; `false` les désactive. La valeur par défaut est `true`.

- EngineVersion

  ENUMXQueryVersion **EngineVersion**

  Spécifie la version XQuery à utiliser (1.0 ou 3.0). La valeur de propriété est un littéral ENUMXQueryVersion.

- IndentCharacters

  `string IndentCharacters`

  Soumet le string de caractère qui sera utilisé en tant que retrait dans la sortie.

- InputXMLFileName

  `string InputXMLFileName`

  Définit l'emplacement, en tant qu'une URL, du document XML à traiter. Le string fourni doit être une URL absolue qui donne l'emplacement exact du fichier XML.

- InputXMLFromText

 © 2018 Altova GmbH

string InputXMLFromText
Apporte le contenu du document XML à traiter. Le string fourni est le contenu du document XML à traiter.

- JavaBarcodeExtensionLocation

string JavaBarcodeExtensionLocation
Spécifie l'emplacement du fichier d'extension du code-barres. Voir la section sur les fonctions d'extension du code-barres d'Altova pour plus d'informations. Le string fourni doit être une URL absolue qui donne l'emplacement de base du fichier à utiliser.

- JavaExtensionsEnabled

bool JavaExtensionsEnabled
Active ou désactive des fonctions d'extension Java. Une valeur de `true` active des extensions Java ; `false` les désactive. La valeur par défaut est `true`.

- KeepFormatting

bool KeepFormatting
Spécifie si le formatage du document original doit être gardé (autant que possible) ou non. Une valeur `true` garde le formatage ; une valeur `false` ne garde pas le formatage. La valeur par défaut est `true`.

- LastErrorMessage

string LastErrorMessage
Récupère un string qui est le dernier message d'erreur provenant du moteur RaptorXML.

- LoadXMLWithPSVI

bool LoadXMLWithPSVI
Active la validation des fichiers XML d'entrée et génère des infos post-schema-validation pour les fichiers. Une valeur de `true` active la validation XML et génère des infos post-schema-validation pour les fichiers XML ; `false` désactive la validation. La valeur par défaut est `true`.

- OutputEncoding

string OutputEncoding
Définit l'encodage pour le document de résultat. Utilise un nom d'encodage IANA officiel comme `UTF-8`, `UTF-16`, `US-ASCII`, `ISO-8859-1`, en tant que string.

- OutputIndent

bool OutputIndent
Active ou désactive le retrait dans le document de sortie. Une valeur de `true` active le retrait ; une valeur `false` le désactive.

- OutputMethod

  string OutputMethod
  Spécifie la sérialisation du document de sortie. Les valeurs valides sont : `xml | xhtml | html | text`. La valeur par défaut est `xml`.

- OutputOmitXMLDeclaration

  bool OutputOmitXMLDeclaration
  Active ou désactive l'inclusion de la déclaration XML dans le document de résultat. Une valeur de `true` omet la déclaration; une valeur de `false` l'inclut. La valeur par défaut est `false`.

- UpdatedXMLWriteMode

  ENUMXQueryUpdatedXML UpdatedXMLWriteMode
  Spécifie comment les mises à jour sont gérées dans le fichier XML. La valeur de propriété est un littéral `ENUMXQueryUpdatedXML`.

- XincludeSupport

  bool XincludeSupport
  Active ou désactive l'utilisation des éléments `XInclude`. Une valeur de `true` active la prise en charge XInclude ; `false` le désactive. La valeur par défaut est `false`.

- XMLValidationErrorAsWarning

  bool XMLValidationErrorAsWarning
  Active le traitement des erreurs de validation XML en tant qu'avertissements. Prend la booléenne `true` ou `false`.

- XMLValidationMode

  ***ENUMXMLValidationMode* XMLValidationMode**
  Définit le mode de validation XML, qui est une énumération littérale de `ENUMXMLValidationMode` (Java, COM/.NET) qui détermine s'il faut contrôler la validité ou la bonne formation.

- XQueryFileName

  string XQueryFileName
  Spécifie le fichier XQuery à utiliser. Le string fourni doit être une URL absolue qui donne l'emplacement du fichier XSLT à utiliser.

- XQueryFromText

  string XQueryFromText
  Fournit, en tant que string de texte, les contenus du document XQuery à utiliser.

 © 2018 Altova GmbH

- XSDVersion

  ***ENUMXSDVersion* XSDVersion**

  Définit la version de schéma XML par rapport à laquelle le document XML sera validé. Est une énumération littérale de `ENUMXSDVersion` (Java, COM/.NET).

IXBRL

L'interface **IXBRL** propose des méthodes pour valider des documents d'instance et de taxonomie XBRL, ainsi que des formules. Les résultats peuvent être des valeurs booléennes `true` ou `false`. L'interface permet aussi aux paramètres de formule d'être passés pour une évaluation de la formule. Les assertions de formules et les sorties peuvent être lues et retournées en tant que strings. Les propriétés définissent les paramètres de l'interface.

Note : Lorsque des entrées de string doivent être interprétées en tant qu'URL, des chemins absolus doivent être utilisés. Si un chemin relatif est utilisé, un mécanisme pour résoudre le chemin relatif doit être défini dans le module d'appel.

Structures

La structure suivante est définie.

```
public struct XBRLParamValuePair
{
    String ParamType;
    String ParamValue;
};
```

Méthodes

Les méthodes de l'interface **IXBRL** sont décrites ci-dessous. Veuillez noter que les entrées de string qui doivent être interprétées en tant qu'URL doivent fournir des chemins absolus. Si un chemin relatif est utilisé, un mécanisme pour résoudre le chemin relatif doit être défini dans le module d'appel.

IsValid

```
bool IsValid(ENUMXBRLValidationType type)
```

- Retourne le résultat de la validation du document d'instance XBRL ou du document de taxonomie XBRL.
- Si une erreur se produit, une **RaptorXMLException** est soulevée. Utiliser l'opération `LastErrorMessage` pour accéder aux informations supplémentaires
- `nType` est la valeur de `ENUMXBRLValidationType`. Le type de validation spécifie si le document d'instance XBRL ou la taxonomie XBRL doit être validée. La valeur par défaut est `eValidateXBRLAny`, qui indique que le type de document doit être automatiquement déterminé par RaptorXML.
- Si une erreur se produit pendant l'exécution, une `RaptorXMLException` est soulevée. Utiliser l'opération `LastErrorMessage` pour accéder aux informations supplémentaires.

EvaluateFormula

```
bool EvaluateFormula()
```

- Évalue les formules XBRL dans un document d'instance XBRL. Retourne `true` si la formule est valide, `false` si une des formule est invalide.
- Si une erreur se produit pendant l'exécution, une `RaptorXMLException` est soulevée. Utiliser l'opération `LastErrorMessage` pour accéder aux informations

 © 2018 Altova GmbH

supplémentaires.

- GenerateTables

```
bool GenerateTables()
```

  - Évalue la table XBRL dans un document d'instance XBRL. Retourne `true` en cas de réussite, `false` en cas d'échec.
  - Si une erreur se produit pendant l'exécution, une `RaptorXMLException` est soulevée. Utiliser l'opération `LastErrorMessage` pour accéder aux informations supplémentaires

- AddFormulaArrayParameter

```
void AddFormulaArrayParameter(string defaultType, string name, object[] values)
```

  - Ajoute un paramètre array utilisé dans le processus d'évaluation de la formule.
  - Tous les arguments sont des strings : `defaultType` est le type de données par défaut des valeurs non-paires dans les valeurs d'array. La valeur par défaut est `xs:string`; `name` est le nom du paramètre ; `values` est un array de valeur et des paires de valeur de type de données.
  - Pour plus d'informations et d'échantillons de code, voir la section, Paramètres de formule XBRL.

- AddFormulaParameter [DEPRECATED]

```
void AddFormulaParameter(string type, string name, string value, string namespace)
```

  - Ajoute un paramètre pour l'évaluation de formule.
  - Tous les arguments sont des strings : `type` est le type de données du paramètre ; `name` est le nom du paramètre ; `value` est la valeur du paramètre ; `namespace` est l'espace de noms du paramètre.
  - Chaque paramètre doit être spécifié dans un appel séparé à la méthode.

- AddFormulaParameter

```
void AddFormulaParameter(string type, string name, string value, string namespace="")
```

  - Ajoute un paramètre pour l'évaluation de formule.
  - Tous les arguments sont des strings : `type` est le type de données du paramètre ; `name` est le nom du paramètre ; `value` est la valeur du paramètre ; `namespace` est l'espace de noms du paramètre et est le string vide.
  - Chaque paramètre doit être spécifié dans un appel séparé à la méthode.

- AddFormulaParameterNamespace

```
void AddFormulaParameterNamespace(string prefix, string uri)
```

  - Définit un espace de noms utilisé dans les QNames des noms, types ou valeurs de paramètre.
  - Tous les arguments sont des strings : `prefix` est le préfixe d'espace de noms des valeurs

passées dans `AddFormulaArrayParameter`; `uri` est l'URI d'espace de noms.
- Chaque paramètre doit être spécifié dans un appel séparé à la méthode.

AddIXBRLTransformationRegistryLimit

```
void AddIXBRLTransformationRegistryLimit(string limit)
```

- Limite les Inline XBRL Transformation Registries disponibles à la version spécifiée.
- `limit` est la version dans laquelle les Inline XBRL Registries disponibles sont limités.

ClearFormulaParameterList

```
void ClearFormulaParameterList()
```

- Supprime la liste des paramètres de formule créés avec la méthode `AddFormulaParameter`.

ClearIXBRLTransformationRegistryLimit

```
void ClearIXBRLTransformationRegistryLimit()
```

- Supprime toutes les limitations des versions disponibles du Inline XBRL Transformation Registries.

ReadFormulaAssertions

```
void ReadFormulaAssertions()
```

- Lit les assertions de formules du fichier en cours d'évaluation.

ReadFormulaOutput

```
void ReadFormulaOutput()
```

- Lit la sortie des assertions de formules du fichier.

Propriétés

Les propriétés de l'interface **IXBRL** sont décrites ci-dessous dans l'ordre alphabétique. La table arrange les propriétés en groupes pour une référence plus aisée. Veuillez noter que les entrées de string qui doivent être interprétées en tant qu'URL doivent fournir des chemins absolus. Si un chemin relatif est utilisé, un mécanisme pour résoudre le chemin relatif doit être défini dans le module d'appel.

AddAssertionForProcessing

```
string AddAssertionForProcessing
```

Limite l'évaluation de l'assertion a la seule assertion donnée. Appeler plusieurs fois pour spécifier plus d'une assertion. Utiliser `##none` pour aucune assertion, et `##all` pour toutes les assertions.

 © 2018 Altova GmbH

- AddAssertionSetForProcessing

  string AddAssertionSetForProcessing
  Limite l'évaluation de l'ensemble de l'assertion a la seule assertion donnée. Appeler plusieurs fois pour spécifier plus d'un ensemble d'assertion. Utiliser `##none` pour aucun ensemble d'assertion, et `##all` pour tous les ensembles d'assertions.

- AddTableForProcessing

  string AddTableForProcessing
  Limite la génération de table à la seule table donnée. Appeler plusieurs fois pour spécifier plus d'une table. Utiliser `##none` pour aucune table, et `##all` pour toutes les tables.

- ConceptLabelLinkrole

  string ConceptLabelLinkrole
  Spécifie le rôle de lien étendu préféré à utiliser lors du rendu des libellés de concept.

- ConceptLabelRole

  string ConceptLabelRole
  Spécifie le rôle de libellé préféré à utiliser lors du rendu des libellés de concept. La valeur par défaut est : `http://www.xbrl.org/2008/role/label`.

- DimensionExtensionEnabled

  bool DimensionExtensionEnabled
  Active ou désactive la validation des extensions de dimension XBRL. Une valeur de `true` active la validation des extensions de dimension ; `false` la désactive. La valeur par défaut est `true`.

- EnableIXBRLValidateTarget

  bool EnableIXBRLValidateTarget
  Active la validation XBRL du document XBRL généré. La valeur par défaut est : `true`.

- EvaluateReferencedParametersOnly

  bool EvaluateReferencedParametersOnly
  Si `false`, force l'évaluation de toutes les paramètres même s'ils ne sont pas référencés par des formules/assertions/tables. La valeur par défaut est : `true`.

- FormulaAssertionsAsXML

  bool FormulaAssertionsAsXML
  Active le formatage XML du fichier d'assertions des formules lorsque RaptorXML est exécuté avec les assertions activées. Une valeur de `true` active le formatage XML ; une valeur de

`false` génère une sortie JSON. La valeur par défaut est `false`.

- FormulaAssertionsOutput

  `string FormulaAssertionsOutput`
  Spécifie l'emplacement du fichier de sortie d'assertion de formule. Le chemin complet doit être spécifié.

- FormulaExtensionEnabled

  `bool FormulaExtensionEnabled`
  Active ou désactive la validation des extensions de formule XBRL. Une valeur de `true` active la validation des extensions de formule ; `false` la désactive. La valeur par défaut est `true`.

- FormulaOutput

  `string FormulaOutput`
  Spécifie l'emplacement de la sortie du fichier d'évaluation de formule XBRL. Le chemin complet doit être spécifié.

- FormulaParameterFile

  `string FormulaParameterFile`
  Spécifie l'emplacement du fichier de paramètre de la formule. Le chemin complet doit être spécifié.

- FormulaPreloadSchemas

  `bool FormulaPreloadSchemas`
  Définit si les schémas de formule seront préchargés. Une valeur de `true` précharge les schémas. Le défaut est `false`, ce qui signifie que ces schémas ne seront pas préchargés.

- GenericLabelLinkrole

  `string GenericLabelLinkrole`
  Spécifie le rôle de lien étendu préféré à utiliser lors du rendu des libellés génériques.

- GenericLabelRole

  `string GenericLabelRole`
  Spécifie le rôle de libellé préféré à utiliser lors du rendu des libellés génériques. La valeur par défaut est : `http://www.xbrl.org/2008/role/label`.

- InputFileArray

  `object InputFileArray`
  Définit l'array des fichiers XBRL qui seront utilisés en tant que données/instances d'entrée. L'array est un objet contenant les strings des URL absolues de chaque fichier d'entrée.

 © 2018 Altova GmbH

- InputFileName

  string InputFileName
  Spécifie le nom de fichier et l'emplacement du fichier d'instance XBRL. Le string soumis doit être soit une URL absolue ; les chemins relatifs peuvent être résolus relativement à un emplacement de base, conformément à un mécanisme défini dans le module d'appel.

- InputFromText

  string InputFromText
  Fournit les contenus du document d'entrée XBRL en tant que texte.

- InputTextArray

  object InputTextArray
  Définit l'array des fichiers texte qui seront utilisés en tant que données d'entrée. L'array est un objet contenant les strings des URL absolues de chaque fichier d'entrée.

- IXBRLOutput

  string IXBRLOutput
  Définit l'emplacement de sortie du fichier de sortie XBRL généré. Le string fourni contient l'URL absolue de l'emplacement de sortie.

- IXBRLTreatAsDocumentSet

  bool IXBRLTreatAsDocumentSet
  Définit si toutes les entrées sont traitées en tant qu'un ensemble de document XBRL Inline simple. La valeur par défaut est `false`.

- IXBRLUriStrategy

  ENUMIXBRLUriStrategy IXBRLUriStrategy
  Spécifie comment les URI de XBRL Inline doivent être transformées dans le document XBRL généré. Prend le littéral `ENUMIXBRLUriStratgey` qui est sélectionné.

- IXBRLVersion

  ENUMIXBRLVersion IXBRLVersion
  Spécifie la version XBRL Inline à utiliser. Prend le littéral `ENUMIXBRLVersion` qui est sélectionné.

- LabelLang

  string LabelLang
  Spécifie le langage de libellé préféré à utiliser lors du rendu des libellés. La valeur par défaut

est : `en`.

- LastErrorMessage

  `string LastErrorMessage`
  Récupère un string qui est le dernier message d'erreur provenant du moteur RaptorXML.

- ParallelAssessment

  `bool ParallelAssessment`
  Active/désactive l'attribution de validité de schéma parallèle.

- PreloadSchemas

  `bool PreloadSchemas`
  Définit si les schémas XBRL 2.1 seront préchargés. Une valeur de `true` précharge les schémas. La valeur par défaut est `true`.

- PythonScriptFile

  `string PythonScriptFile`
  Spécifie le fichier de script Python qui fournit le traitement supplémentaire du fichier XML ou XSD soumis à la validation. Le string fourni doit être une URL absolue qui indique l'emplacement de base du script Python.

- SchemaImports

  `ENUMSchemaImports` **`SchemaImports`**
  Spécifie comment traiter les importations de schéma sur la base des valeurs d'attribut des éléments `xs:import`. Le type de gestion est spécifié par le littéral `ENUMSchemaImports` (Java, COM/.NET) qui est soumis.

- SchemalocationHints

  `ENUMLoadSchemalocation` **`SchemalocationHints`**
  Spécifie le mécanisme à utiliser pour situer le schéma. Le mécanisme est spécifié par le littéral `ENUMLoadSchemalocation` (Java, COM/.NET) qui a été sélectionné.

- SchemaMapping

  `ENUMSchemaMapping` **`SchemaMapping`**
  Définit quel mappage utiliser pour localiser le schéma. Le mappage est spécifié par le littéral `ENUMSchemaMapping` (Java, COM/.NET) qui est sélectionné.

- TableEliminateEmptyRows

  `bool TableEliminateEmptyRows`

 © 2018 Altova GmbH

Active l'élimination des lignes/colonnes vides dans la sortie HTML de la génération de la table.

- TableExtensionEnabled

  bool TableExtensionEnabled
  Active/Désactive l'extension XBRL Table 1.0.

- TableLinkbaseNamespace

  string TableLinkbaseNamespace
  Active le chargement des bases de lien de table écrits à l'aide d'un projet de spécification précédente. La valeur de string fournie spécifie la base de lien de la table. La validation de la base de lien de la table, la résolution et la mise en page sont, néanmoins, toujours effectués conformément à la Table Linkbase 1.0 Recommandation du 18 mars 2014. Utiliser `##detect` pour activer l'auto-détection. Les valeurs suivantes sont reconnues :

  ```
  ##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

  string TableOutput
  Spécifie le nom de fichier et l'emplacement de la sortie de la génération de table. Le string soumis doit être le chemin complet du fichier de sortie.

- TableOutputFormat

  ENUMTableOutputFormat TableOutputFormat
  Spécifie le format du fichier de sortie de génération de la table.

- TablePreloadSchemas

  bool TablePreloadSchemas
  Active/Désactive le préchargement des schémas de spécification de XBRL Table 1.0.

- TreatXBRLInconsistenciesAsErrors

  bool TreatXBRLInconsistenciesAsErrors
  Une valeur `true` entraîne l'échec de la validation XBRL si le fichier contient des incohérences telles que définies par la spécification XBRL 2.1. La valeur par défaut est `false`: les incohérences XBRL conformément à la spécification XBRL 2.1 ne sont pas traitées en tant qu'erreurs.

- XIncludeSupport

`bool XIncludeSupport`

Active ou désactive l'utilisation des éléments `XInclude`. Une valeur de `true` active la prise en charge XInclude ; `false` le désactive. La valeur par défaut est `false`.

© 2018 Altova GmbH

Énumérations

Les énumérations suivantes sont définies. Elles sont décrites dans les sous-sections de cette section.

ENUMAssessmentMode

ENUMErrorFormat

ENUMLoadSchemalocation

ENUMQueryVersion

ENUMSchemaImports

ENUMSchemaMapping

ENUMValidationType

ENUMWellformedCheckType

ENUMXBRLValidationType

ENUMXMLValidationMode

ENUMXQueryVersion

ENUMXSDVersion

ENUMXSLTVersion

ENUMAssessmentMode

Description
Contient les littéraux d'énumération qui définissent le mode d'attribution du Validateur XML : `Strict` ou `Lax`.

Utilisé par

Interface	*Opération*
`IXMLValidator`	`AssessmentMode`

Littéraux d'énumération

```
eAssessmentModeStrict   = 0
eAssessmentModeLax      = 1
```

`eAssessmentModeStrict`
Définit le mode d'attribution de validité de schéma sur `Strict`. Il s'agit de la valeur par défaut.

`eAssessmentModeLax`
Définit le mode d'attribution de validité de schéma sur `Lax`.

 © 2018 Altova GmbH

ENUMErrorFormat

Description
Contient les littéraux d'énumération qui spécifient le format de sortie d'erreur.

Utilisé par

Interface	*Opération*
IServer	ErrorFormat

Littéraux d'énumération

```
eFormatText         = 0
eFormatShortXML     = 1
eFormatLongXML      = 2
```

eFormatText
Définit le format de sortie d'erreur sur `Text`. Il s'agit de la valeur par défaut.

eFormatShortXML
Définit le format de sortie d'erreur sur `ShortXML`. Ce format est une forme abrégée du format `LongXML`.

eFormatLongXML
Définit le format de sortie d'erreur sur `LongXML`. Des trois formats de sortie, il s'agit du format qui fournit le plus de détails.

ENUMLoadSchemalocation

Description
Contient les littéraux d'énumération qui indiquent comment l'emplacement du schéma doit être déterminé.

Utilisé par

Interface	*Opération*
IXBRL	SchemalocationHints
IXMLValidator	SchemalocationHints
IXSLT	SchemalocationHints

Littéraux d'énumération

```
eSHLoadBySchemalocation     = 0
eSHLoadByNamespace          = 1
eSHLoadCombiningBoth        = 2
eSHLoadIgnore               = 3
```

eSHLoadBySchemalocation
Définit Load Schemalocation sur `LoadBySchemalocation`. Utilise l'URL de l'emplacement de schéma dans les attributs `xsi:schemaLocation` et `xsi:noNamespaceSchemaLocation` dans les documents d'instance XML ou XBRL. Il s'agit de la valeur par défaut.

eSHLoadByNamespace
Définit Load Schemalocation sur `LoadByNamespace`. Utilise la partie de l'espace de noms de `xsi:schemaLocation` (un string vide dans le cas de `xsi:noNamespaceSchemaLocation`), et situe le schéma par le biais d'un mappage de catalogue.

eSHLoadCombiningBoth
Définit Load Schemalocation sur `CombiningBoth`. Si l'espace de noms ou l'URL dispose d'un mappage de catalogue, le mappage de catalogue sera utilisé. Si tous deux ont un mappage de catalogue, la valeur du paramètre `ENUMSchemaMapping` décide quel mappage utiliser. Si ni l'espace de nom ni l'URL dispose d'un mappage de catalogue, l'URL sera utilisée.

eSHLoadIgnore
Définit Load Schemalocation sur `LoadIgnore`. Si la valeur du paramètre est `eSHLoadIgnore`, alors les attributs `xsi:schemaLocation` et `xsi:noNamespaceSchemaLocation` seront tous les deux ignorés.

 © 2018 Altova GmbH

ENUMQueryVersion

Description
Contient les littéraux d'énumération qui spécifient la version XQuery à utiliser : XQuery 1.0 ou 3.0.

Littéraux d'énumération

```
eXQVersion10        = 1
eXQVersion30        = 3
```

eXQVersion10
Définit la version XQuery à XQuery 1.0.

eXQVersion30

Définit la version XQuery à XQuery 3.0.

ENUMSchemaImports

Description
Contient les littéraux d'énumération qui définissent le comportement des éléments `xs:import`. L'élément `xs:import` comporte les attributs `namespace` et `schemaLocation`, tous deux optionnels.

Utilisé par

Interface	*Opération*
IXBRL	SchemaImports
IXMLValidator	SchemaImports
IXSLT	SchemaImports

Littéraux d'énumération

```
eSILoadBySchemalocation           = 0
eSILoadPreferringSchemalocation   = 1
eSILoadByNamespace                = 2
eSICombiningBoth                  = 3
eSILicenseNamespaceOnly           = 4
```

eSILoadBySchemalocation
Définit l'importation de schéma sur `LoadBySchemalocation`. La valeur de l'attribut `schemaLocation` est utilisé pour situer le schéma, en prenant en compte les mappages de catalogue. Si l'attribut `namespace` est présent, l'espace de noms sera importé (mis sous licence).

eSILoadPreferringSchemalocation
Définit l'importation de schéma sur `LoadPreferringSchemalocation`. Si l'attribut `schemaLocation` est présent, il sera utilisé, en prenant en compte les mappages de catalogue. Si aucun attribut `schemaLocation` n'est présent, la valeur de l'attribut `namespace` est utilisée par le biais d'un mappage de catalogue. Ce littéral est la **valeur par défaut** de l'énumération.

eSILoadByNamespace
Définit l'importation de schéma sur `LoadByNamespace`. La valeur de l'attribut `namespace` est utilisée pour localiser le schéma par le biais d'un mappage de catalogue.

eSICombiningBoth
Définit l'importation de schéma sur `CombiningBoth`. Si l'attribut `namespace` ou `schemaLocation` a un mappage de catalogue, celui-ci sera utilisé. Si tous deux ont des mappages de catalogue, la valeur du paramètre `ENUMSchemaMapping` décidera quel mappage est utilisé. Si aucun catalogue n'est présent, la valeur de l'attribut `schemaLocation` (qui doit être une URL) est utilisée.

 © 2018 Altova GmbH

eSILicenseNamespaceOnly
Définit l'importation de schéma sur `LicenseNamespaceOnly`. L'espace de noms est importé. Aucun document de schéma n'est importé.

ENUMSchemaMapping

Description
Contient les littéraux d'énumération qui définit lequel des deux mappages de catalogue est préféré : espace de noms ou URL de localisation de schéma. Cette énumération est utile pour différencier ENUMLoadSchemalocation et ENUMSchemaImports.

Utilisé par

Interface	*Opération*
IXBRL	SchemaMapping
IXMLValidator	SchemaMapping
IXSLT	SchemaMapping

Littéraux d'énumération

```
eSMPreferSchemalocation     = 0
eSMPreferNamespace          = 1
```

eSMPreferSchemalocation
Définit l'option de mappage de schéma pour choisir l'URL d'emplacement de schéma.

eSMPreferNamespace
Définit l'option de mappage de schéma pour choisir l'espace de noms.

 © 2018 Altova GmbH

ENUMTableOutputFormat

Description
Contient les littéraux d'énumération qui spécifie le format de sortie du document contenant les tables générées.

Utilisé par

Interface	*Opération*
IXBRL	TableOutputFormat

Littéraux d'énumération

```
eFormatXML                  = 0
eFormatHTML                 = 1
```

eSMPreferSchemalocation
Définit l'option de mappage de schéma pour choisir l'URL d'emplacement de schéma.

eSMPreferNamespace
Définit l'option de mappage de schéma pour choisir l'espace de noms.

ENUMValidationType

Description
Contient les littéraux d'énumération qui définissent le type de document à valider.

Utilisé par

Interface	*Opération*
IXMLValidator	IsValid

Littéraux d'énumération

```
eValidateAny          = 0
eValidateXMLWithDTD   = 1
eValidateXMLWithXSD   = 2
eValidateDTD          = 3
eValidateXSD          = 4
eValidateJSON         = 5
eValidateJSONSchema   = 6
eValidateAvro         = 7
eValidateAvroSchema   = 8
eValidateAvroJSON     = 9
```

eValidateAny
Définit le type de validation sur `Any`. Cela valide un document après la détection automatique de son type.

eValidateXMLWithDTD
Définit le type de validation sur `XMLWithDTD`. Cela spécifie la validation d'un document XML par rapport à un DTD.

eValidateXMLWithXSD
Définit le type de validation sur `XMLWithXSD`. Cela spécifie la validation d'un document XML par rapport à un Schéma XML.

eValidateDTD
Définit le type de validation sur `ValidateDTD`. Cela spécifie la validation d'un document DTD.

eValidateXSD
Définit le type de validation sur `ValidateXSD`. Cela spécifie la validation d'un document de

 © 2018 Altova GmbH

Schéma XML W3C.

eValidateJSON
Définit le type de validation sur `ValidateJSON`. Cela spécifie la validation d'un document d'instance JSON conformément au Schéma JSON v 4.

eValidateJSONSchema
Définit le type de validation sur `ValidateXSD`. Cela spécifie la validation d'un document d'instance JSON conformément au Schéma JSON v 4.

eValidateAvro
Définit le type de validation sur `ValidateAvro`. Cela spécifie la validation d'un fichier binaire Avro.

eValidateAvroSchema
Définit le type de validation sur `ValidateAvroSchema`. Cela spécifie la validation d'un document de schéma Avro conformément à la spécification de schéma Avro.

eValidateAvroJSON
Définit le type de validation sur `ValidateAvroJSON`. Cela spécifie la validation d'un document de données Avro dans la sérialisation JSON conformément à un schéma Avro.

ENUMWellformedCheckType

Description
Contient les littéraux d'énumération qui qui définissent le type de document à contrôler : XML ou DTD.

Utilisé par

Interface	*Opération*
IXMLValidator	IsWellFormed

Littéraux d'énumération

```
eWellFormedAny      = 0
eWellFormedXML      = 1
eWellFormedDTD      = 2
eWellFormedJSON     = 3
```

eWellformedAny
Définit le type de contrôle de bonne formation sur `Any`. Cela contrôle la bonne formation d'un document XML ou DTD pour la bonne formation après avoir détecté automatiquement duquel des deux types il s'agit.

eWellformedXML
Définit le type de contrôle de bonne formation sur `XML`. Cela contrôle la bonne formation d'un document XML conformément à la spécification XML 1.0 ou XML 1.1.

eWellformedDTD
Définit le type de contrôle de bonne formation sur `DTD`. Cela contrôle la bonne formation d'un document DTD.

eWellformedJSON
Définit le type de contrôle de bonne formation sur `JSON`. Cela contrôle la bonne formation d'un document JSON conformément à la spécification ECMA-404.

 © 2018 Altova GmbH

ENUMXBRLValidationType

Description
Contient les littéraux d'énumération qui définissent le type de document XBRL à valider : instance XBRL ou taxonomie XBRL.

Utilisé par

Interface	*Opération*
IXBRL	IsValid

Littéraux d'énumération

```
eValidateXBRLAny              = 0
eValidateXBRLInstance         = 1
eValidateXBRLTaxonomy         = 2
eValidateXBRLInline           = 3
eValidateXBRLTaxonomyPackage  = 4
```

eValidateXBRLAny
Définit le type de validation sur `Any`. Cela valide le document XBRL après avoir détecté automatiquement son type (instance ou taxonomie).

eValidateXBRLInstance
Définit le type de validation sur `Instance`. Cela spécifie la validation d'un ou de plusieurs documents d'instance XBRL.

eValidateXBRLTaxonomy
Définit le type de validation sur `Taxonomy`. Cela spécifie la validation d'un ou de plusieurs documents de taxonomie XBRL.

eValidateXBRLInline
Définit le type de validation sur `Inline`. Cela spécifie la validation d'un ou de plusieurs documents Inline XBRL.

eValidateXBRLTaxonomyPackage
Définit le type de validation sur `TaxonomyPackage`. Cela spécifie la validation d'un ou de plusieurs documents packages de taxonomie XBRL.

ENUMIXBRLVersion

Description
Contient les littéraux d'énumération qui spécifient la version de la spécification Inline XBRL à utiliser : Inline XBRL 1.0 ou Inline XBRL 1.1. **`eVersionDetect`** active l'auto-détection.

Utilisé par

Interface	*Opération*
IXBRL	IXBRLVersion

Littéraux d'énumération

```
eVersion10          = 0
eVersion11          = 1
eVersionDetect      = 2
```

`eVersion10`
Définit la spécification Inline XBRL 1.0 comme étant celle à utiliser.

`eVersion11`
Définit la spécification Inline XBRL 1.1 comme étant celle à utiliser.

`eVersionDetect`
Active l'auto-détection de la version depuis le document Inline XBRL soumis.

 © 2018 Altova GmbH

ENUMIXBRLUriStrategy

Description
Contient les littéraux d'énumération qui spécifient comment écrire les URI dans l'Inline XBRL sur le document XBRL généré.

Utilisé par

Interface	*Opération*
IXBRL	IXBRLUriStrategy

Littéraux d'énumération

```
eStrategyNone                = 0
eStrategyMakeAbsolute        = 1
eStrategyMakeRelative        = 2
eStrategyKeepRelative        = 3
```

`eStrategyNone`
Copie l'URI verbatim sur le document cible.

`eStrategyMakeAbsolute`
Rend tous les URI relatifs en URI absolus en résolvant par rapport à l'URI de base in-scope à l'endroit de l'élément respectif dans le document d'entrée.

`eStrategyMakeRelative`
Si possible, rend tous les URI absolus et relatifs en URI relatifs par rapport au document de sortie (sinon, écrit l'URI absolue résolue).

`eStrategyKeepRelative`
Si possible, rend uniquement les URI relatifs en URI relatifs par rapport au document de sortie (et copie tout URI absolu).

ENUMXMLValidationMode

Description
Contient les littéraux d'énumération qui définissent le mode de traitement XML à utiliser : Validation ou bonne formation.

Utilisé par

Interface	*Opération*
IXMLValidator	XMLValidationMode
IXQuery	XMLValidationMode
IXSLT	XMLValidationMode

Littéraux d'énumération

```
eXMLValidationModeWF      = 0
eXMLValidationModeID      = 1
eXMLValidationModeValid   = 2
```

eXMLValidationModeWF
Définit le mode de traitement XML sur `Wellformed`. Il s'agit de la valeur par défaut.

eXMLValidationModeID
Interne.

eXMLValidationModeValid
Définit le mode de traitement XML sur `Validation`.

 © 2018 Altova GmbH

ENUMXQueryVersion

Description
Contient les littéraux d'énumération qui spécifient la version XQuery à utiliser : XQuery 1.0 ou 3.0.

Utilisé par

Interface	*Opération*
IXQuery	EngineVersion

Littéraux d'énumération

```
eXQVersion10        = 1
eXQVersion30        = 3
```

eXQVersion10
Définit la version XQuery sur XQuery 1.0.

eXQVersion30
Définit la version XQuery sur XQuery 3.0. Il s'agit de la valeur par défaut.

ENUMXQueryUpdatedXML

Description
Contient les littéraux d'énumération pour spécifier comment gérer les mises à jour XQuery.

Utilisé par

Interface	*Opération*
IXQuery	UpdatedXMLWriteMode

Littéraux d'énumération

```
eUpdatedDiscard          = 1
eUpdatedWriteback        = 2
eUpdatedAsMainResult     = 3
```

eUpdatedDiscard
Les mises à jour sont éliminées et ne sont pas rédigées sur un fichier.

eUpdatedWriteback
Les mises à jour sont écrites sur le fichier d'entrée XML spécifié avec InputXMLFileName.

eUpdatedAsMainResult
Les mises à jour sont écrites sur l'emplacement spécifié par le paramètre `outputFile` de ExecuteUpdate .

 © 2018 Altova GmbH

ENUMXSDVersion

Description
Contient les littéraux d'énumération qui indiquent la version de Schéma XML à utiliser pour la validation : XSD 1.0 ou 1.1.

Utilisé par

Interface	*Opération*
IXMLValidator	XSDVersion
IXQuery	XSDVersion
IXSLT	XSDVersion

Littéraux d'énumération

```
eXSDVersionAuto     = 0
eXSDVersion10       = 1
eXSDVersion11       = 2
```

eXSDVersionAuto
Définit la version de Schéma XML pour une validation sur `Auto-detect`. La version XSD sera détectée automatiquement après avoir parsé le document XSD. Si l'attribut `vc:minVersion` du document XSD a une valeur de `1.1`, le document sera considéré être XSD 1.1. Si l'attribut a une autre valeur, ou n'existe pas, le document sera considéré être XSD 1.0.

eXSDVersion10
Définit la version de Schéma XML pour une validation au Schéma XML 1.0.

eXSDVersion11
Définit la version de Schéma XML pour une validation au Schéma XML 1.1.

ENUMXSLTVersion

Description
Contient les littéraux d'énumération qui définissent la version XSLT à utiliser : XSLT 1.0, 2.0 ou 3.0.

Utilisé par

Interface	*Opération*
IXSLT	EngineVersion

Littéraux d'énumération

```
eVersion10     = 1
eVersion20     = 2
eVersion30     = 3
```

eVersion10
Définit la version XSLT sur XSLT 1.0.

eVersion20
Définit la version XSLT sur XSLT 2.0.

eVersion30
Définit la version XSLT sur XSLT 3.0.

 © 2018 Altova GmbH

Chapitre 8

Information supplémentaire

8 Information supplémentaire

Cette section contient les informations supplémentaires suivantes :

- Codes de sortie
- Indices d'emplacement de schéma
- Paramètres de formule XBRL

© 2018 Altova GmbH

8.1 Codes de sortie

Les codes de sortie suivant sont disponibles :

0	Validation réussie
1	Validation échouée avec erreur / Processus interrompu par Ctrl+C/ Break/terminal fermé / Licence expirée pendant l'exécution
11	RaptorXML n'a pas pu démarrer ; la raison est indiquée dans le fichier log
22	Impossible de charger le catalogue root / Impossible de charger le fichier de liste
64	Commande/options invalide
77	Impossible d'acquérir une licence pendant le démarrage
128+**n**	RaptorXML a été terminée en raison du nombre de signal **n**. Tous les codes de sortie au-dessus de **128** indiquent l'arrêt en résultat d'un signal externe reçu ou un signal déclenché internement. Par exemple, si le code de sortie est **134**, le numéro de signal est **134-128=6** (le numéro de **SIGABRT**).

8.2 Indices d'emplacement de schéma

Les documents d'instance peuvent utiliser des indices pour indiquer l'emplacement de schéma. Deux attributs sont utilisés pour les indices :

- `xsi:schemaLocation` pour les documents de schéma avec des espaces de noms cible. La valeur d'attribut est une paire d'items, le premier d'entre eux est un espace de noms, le second est une URL qui situe un document de schéma. Le nom d'espace de noms doivent correspondre à l'espace de noms cible du document de schéma.

```
<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` pour les documents de schéma sans espaces de noms cible. La valeur de l'attribut est l'URL du document de schéma. Le document de schéma référencé ne doit avoir aucun espace de noms cible.

```
<document xmlns="http://www.altova.com/schemas/test03"
          xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
          xsi:noNamespaceSchemaLocation="Test.xsd">
```

L'option `--schemalocation-hints` spécifie comment ces deux attributs doivent être utilisés en tant qu'indices, en particulier comment l'information d'attribut `schemaLocation` doit être gérée (*voir la description d'option ci-dessus*). Veuillez noter que RaptorXML+XBRL Server considère que la partie espace de noms de la valeur `xsi:noNamespaceSchemaLocation` est la string vide.

Les indices d'emplacement de schéma peuvent aussi être donnés dans une instruction `import` d'un document de Schéma XML.

```
<import namespace="someNS" schemaLocation="someURL">
```

Dans l'instruction `import`, aussi, des indices peuvent être donnés par le biais d'un espace de noms qui peut être mappé vers un schéma dans un fichier de catalogue, ou directement en tant qu'une URL dans l'attribut `schemaLocation`. L'option `--schema-imports` (pour XBRL et XSD/XML) spécifie comment l'emplacement de schéma doit être sélectionnée.

 © 2018 Altova GmbH

8.3 Paramètres de formule XBRL

Cette section contient les sujets suivants :

- Formats de paramètres de formule, qui donne des exemples des formants XML et JSON de paramètres de formule XBRL.
- Utiliser les paramètres de formule contient des listes dans Java, VB.NET, C#, VBScript et JScript qui montre les paramètres de formule peuvent être utilisés en se servant d'objets provenant des bibliothèques d'API Java et COM/.NET.

Formats de paramètre de formule

Paramètres de formule peut être donnés dans un format XML ou format JSON.

Format XML

La liste ci-dessous montre les paramètres de formule en format 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>
```

Veuillez noter les points suivants :

- L'attribut `@type` est optionnelle et la valeur par défaut est `xs:string`.
- Les éléments enfants `<options:value>` multiples peuvent être spécifiés pour attribuer une séquence XPath dans un paramètre.
- `@value` et `<options:value>` ne peut pas être utilisé au même moment.

Format JSON

La liste ci-dessous montre les paramètres de formule en format 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"
    }
}
```

Veuillez noter les points suivants :

- La clé `type` est optionnelle et le défaut est `xs:string`.
- La clé `xs` est optionnelle et le défaut est `http://www.w3.org/2001/XMLSchema`.
- Le type dans le mapping de paramètre est utilisé si une valeur est spécifiée directement en tant qu'un string JSON.
- D'autres moyens d'écriture sont aussi pris en charge actuellement :

```
{
    "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"]
}
```

Utiliser les paramètres de formule

Les listes d'exemple ci-dessous montre comment les paramètres de formule XBRL peuvent être utilisés dans des langues de programmation variées. Pour Java, voir la classe XBRL d'API Java. En ce qui concerne les autres langages, référez-vous à l'interface XBRL d'API 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 =
```

 © 2018 Altova GmbH

```
"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

Puisque la bibliothèque de type Raptor ne peut pas être chargée par les langages de scripts, et parce que le type `XBRLFormulaParameters` n'existe pas, l'utilisateur VBScript, au lieu d'utiliser les objets `XBRL.FormulaParam`, doit déclarer une classe dans son programme. La classe *doit* avoir deux propriétés publiques, `ParamName` et `ParamValue` (comme c'est le cas de `XBRL.FormulaParam`). La classe devrait avoir un constructeur qui prend le type et la valeur, puisque c'est cela qui simplifie l'usage ; sinon, l'objet nécessite d'être créé et que ses membres soient définis). Voir l'interface XBRL des API 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

Puisque la bibliothèque de type Raptor ne peut pas être chargée par les langages de scripts, et parce que le type `XBRLFormulaParameters` n'existe pas, l'utilisateur JScript, au lieu d'utiliser les objets `XBRL.FormulaParam`, doit déclarer des classes-fonctions dans son programme qui contient la paire valeur-type. Les noms de membres *doivent* être ParamType et `ParamValue`. Voir 'interface XBRL des API 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()
```

Chapitre 9

Informations concernant le moteur XSLT et XQuery

9 Informations concernant le moteur XSLT et XQuery

Les moteurs XSLT et XQuery de RaptorXML+XBRL Server suivent de près les spécifications W3C et sont donc plus strictes que les moteurs Altova précédents, comme dans les versions précédentes de XMLSpy et d'AltovaXML, le prédécesseur de RaptorXML. Ainsi, de petites erreurs qui étaient ignorées par les moteurs précédents sont maintenant marquées en tant qu'erreurs par RaptorXML+XBRL Server.

Par exemple :

- Il s'agit d'une erreur de type (`err:XPTY0018`) si le résultat d'un opérateur de chemin contient aussi bien les nœuds que les non-nœuds.
- Il s'agit d'une erreur de type (`err:XPTY0019`) si `E1` dans une expression de chemin `E1/E2` n'évalue pas à une séquence de nœuds.

Si vous rencontrez ce type d'erreur, modifiez soit le document XSLT/XQuery soit le document d'instance selon vos besoins.

Cette section décrit les fonctions spécifiques à la mise en place des moteurs, organisée par spécification :

- XSLT 1.0
- XSLT 2.0
- XSLT 3.0
- XQuery 1.0
- XQuery 3.1

 © 2018 Altova GmbH

9.1 XSLT 1.0

Le moteur XSLT 1.0 de RaptorXML+XBRL Server est conforme aux Recommandations XSLT 1.0 du 16 novembre 1999 et aux Recommandations XPath 1.0 du 16 novembre 1999 du World Wide Web Consortium (W3C's). Veuillez noter les informations suivantes concernant l'implémentation.

Notes concernant l'implémentation

Lorsque l'attribut `method` de `xsl:output` est défini sur HTML, ou si la sortie HTML est sélectionnée par défaut, les caractères spéciaux dans le fichier XML ou XSLT sont insérés dans le document HTML en tant que références de caractère HTML dans la sortie. Par exemple, le caractère U+00A0 (la référence de caractère hexadécimale pour un espace insécable) est inséré dans le code HTML soit en tant que référence de caractère (` ` ou ` `) soit en tant que référence d'entité, ` ` .

9.2 XSLT 2.0

Cette section :

- Conformité du moteur
- Rétrocompatibilité
- Espaces de nom
- Compatibilité avec le schéma
- Comportement spécifique à l'implémentation

Conformité

Le moteur XSLT 2.0 de RaptorXML+XBRL Server est conforme aux Recommandations XSLT 2.0 du 23 janvier 2007 et aux Recommandations XPath 2.0 du 14 décembre 2010 du World Wide Web Consortium (W3C's).

Rétrocompatibilité

Le moteur XSLT 2.0 est rétrocompatible. Le seul moment où la compatibilité rétroactive du moteur XSLT 2.0 prend effet, est si vous utilisez le moteur XSLT 2.0 (paramètre CLI `--xslt=2`) pour traiter une feuille de style XSLT 1.0. Veuillez noter qu'il peut y avoir des différences dans les sorties produites par le moteur XSLT 1.0 et la rétrocompatibilité du moteur XSLT 2.0.

Espaces de nom

Votre feuille de style XSLT 2.0 devrait déclarer les espaces de noms suivants afin que vous puissiez utiliser les constructeurs de type et les fonctions disponibles dans XSLT 2.0. Les préfixes indiqués ci-dessous sont utilisés conventionnellement ; vous pourriez utiliser les préfixes alternatifs si vous le souhaitez.

Nom d'espace de nom	Préfixe	Espace de nom URI
Types de schéma XML	`xs:`	`http://www.w3.org/2001/XMLSchema`
Fonctions XPath 2.0	`fn:`	`http://www.w3.org/2005/xpath-functions`

Généralement ces espaces de nom seront déclarés sur l'élément `xsl:stylesheet` ou `xsl:transform`, tel que montré dans la liste suivante :

```
<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>
```

Veuillez noter les points suivants :

- Le moteur XSLT 2.0 utilise l'espace de nom XPath 2.0 et Fonctions XQuery 1.0 (recensées dans la table ci-dessus) en tant que son **espace de noms de fonctions par défaut**. Ainsi, vous pouvez utiliser les fonctions XPath 2.0 et XSLT 2.0 dans votre feuille de style sans aucun préfixe. Si vous déclarez l'espace de nom des fonctions XPath 2.0 dans votre feuille de style avec un préfixe, vous pourrez utiliser en plus le préfixe attribué dans la déclaration.
- Lors de l'utilisation des constructeurs de types et des types provenant de l'espace de nom du Schéma XML, le préfixe utilisé dans la déclaration d'espace de nom doit être utilisé lors de l'appel du constructeur de type (par exemple, `xs:date`).
- Certaines fonctions XPath 2.0 portent le même nom que les types de données du schéma XML. Par exemple, pour les fonctions XPath `fn:string` et `fn:boolean`, il existe des types de données du schéma XML portant le même nom local : `xs:string` et `xs:boolean`. Donc si vous décidez d'utiliser l'expression XPath `string('Hello')`, l'expression évalue en tant que `fn:string('Hello')` et non pas en tant que `xs:string('Hello')`.

Compatibilité avec le schéma

Le moteur XSLT 2.0 est compatible avec le schéma. Vous pouvez ainsi utiliser des types de schéma définis par l'utilisateur et l'instruction `xsl:validate`.

Comportement spécifique à l'implémentation

Ci-dessous, vous trouverez une description de la gestion du moteur XSLT 2.0 des aspects spécifiques à l'implémentation du comportement de certaines fonctions XSLT 2.0.

`xsl:result-document`

Les encodages pris en charge en supplément sont (les codes spécifiques à Altova) : `x-base16tobinary` et `x-base64tobinary`.

`function-available`

La fonction teste la disponibilité des fonctions in-scope (XSLT, XPath, et fonctions d'extension).

`unparsed-text`

L'attribut `href` accepte (i) les chemins relatifs pour les fichiers dans le dossier de base-uri, et (ii) les chemins absolus avec ou sans le protocole `file://`. De plus, les encodages pris en charge supplémentaires sont (les codes spécifiques Altova) : `x-binarytobase16` et `x-binarytobase64`.

`unparsed-text-available`

L'attribut `href` accepte (i) les chemins relatifs pour les fichiers dans le dossier de base-uri, et (ii) les chemins absolus avec ou sans le protocole `file://`. De plus, les encodages pris en charge supplémentaires sont (les codes spécifiques Altova) : `x-binarytobase16` et `x-binarytobase64`.

Note : Les valeurs d'encodage suivantes, qui ont été mises en place dans les versions plus anciennes du produit prédécesseur de RaptorXML, AltovaXML, sont maintenant obsolètes : `base16tobinary`, `base64tobinary`, `binarytobase16` et `binarytobase64`.

 © 2018 Altova GmbH

9.3 XSLT 3.0

Le moteur XSLT 3.0 de RaptorXML+XBRL Server est conforme aux Recommandation XSLT 3.0 du 8 juin 2017 et aux Recommandations XQuery 3.1 du 21 mars 2017 du World Wide Web Consortium (W3C's).

Le moteur XSLT 3.0 présente les mêmes caractéristiques spécifiques à l'implémentation que le moteur XSLT 2.0 . de plus, il contient une prise en charge pour un certain nombre de nouvelles fonctions XSLT 3.0 : les fonctions et opérateurs XPath/XQuery 3.1, et la spécification XPath 3.1.

Note : La fonction de streaming optionnelle n'est pas prise en charge actuellement. Le document complet sera chargé dans la mémoire quelle que soit la valeur de l'attribut `streamable`, et sera traité si la mémoire disponible est suffisante. Dans des applis 64-bit cela ne pose pas de problème. Si la mémoire pose un problème, une solution serait d'ajouter plus de mémoire dans le système.

9.4 XQuery 1.0

Cette section :

- Conformité du moteur
- Compatibilité du schéma (Schema awareness)
- Encodage
- Espaces de nom
- XML source et validation
- Contrôle de type statique et dynamique
- Modules bibliothèque
- Modules externes
- Collations
- Précision des données numériques
- Prise en charge des instructions XQuery

Conformité
Le moteur XQuery 1.0 de RaptorXML+XBRL Server est conforme à la Recommandation XQuery 1.0 du 14 décembre 2010 du World Wide Web Consortium (W3C's). Le standard XQuery accorde un pouvoir discrétionnaire concernant la mise en place de nombreuses fonctions. Ci-dessous, vous trouverez une liste expliquant comment le moteur XQuery 1.0 implémente ces fonctions.

Compatibilité avec le schéma
Le moteur XQuery 1.0 est **schema-aware**.

Encodage
Les encodages de caractères UTF-8 et UTF-16 sont pris en charge.

Espaces de nom
Les URI d'espace de nom suivant et leurs liaisons associées sont prédéfinies.

Nom d'espaces de nom	Préfixe	URI Espace de noms
Types de schéma XML	`xs:`	`http://www.w3.org/2001/XMLSchema`
Instance de schéma	`xsi:`	`http://www.w3.org/2001/XMLSchema-instance`

 © 2018 Altova GmbH

Fonctions intégrées	`fn:`	`http://www.w3.org/2005/xpath-functions`
Fonctions locales	`local:`	`http://www.w3.org/2005/xquery-local-functions`

Veuillez noter les points suivants :

- Le moteur XQuery 1.0 Engine reconnaît les préfixes recensés ci-dessus comme étant liés aux espaces de noms correspondants.
- Étant donné que l'espace de noms des fonctions intégrées recensé ci-dessus est l'espace de noms des fonctions par défaut dans XQuery, le préfixe `fn:` ne doit pas nécessairement être utilisé lorsque des fonctions intégrées sont invoquées (par exemple, `string("Hello")` appellera la fonction `fn:string`). Néanmoins, le préfixe `fn:` peut être utilisé pour appeler une fonction intégrée sans avoir à déclarer l'espace de noms dans le prologue query (par exemple : `fn:string("Hello")`).
- Vous pouvez changer l'espace de noms des fonctions par défaut en déclarant l'expression `default function namespace` dans le prologue de de requête.
- En cas d'utilisation des types depuis l'espace de noms du Schéma XML, le préfixe `xs:` peut être utilisé sans devoir déclarer explicitement les espace de noms et lier ces préfixes dans le prologue de requête. (Exemple : `xs:date` and `xs:yearMonthDuration`.) Si vous souhaitez utiliser d'autres préfixes pour l'espace de noms du schéma XML, cela doit être déclaré explicitement dans le prologue de requête. (Exemple: `declare namespace alt = "http://www.w3.org/2001/XMLSchema"; alt:date("2004-10-04")`.)
- Veuillez noter que les types de données `untypedAtomic`, `dayTimeDuration`, et `yearMonthDuration` ont été déplacés, avec les CR de 23 January 2007, depuis l'espace de noms des Types de données XPath vers l'espace de noms du schéma XML, donc : `xs:yearMonthDuration`.

Si des espaces de noms pour les fonctions, les constructeurs de type, les tests de nœud, etc. sont mal attribués, une erreur est rapportée. Veuillez noter, néanmoins, que certaines fonctions portent le même nom que les types de données de schéma, par ex. `fn:string` et `fn:boolean`. (Les deux `xs:string` et `xs:boolean` sont définis.) Le préfixe d'espace de noms détermine si la fonction ou le constructeur de type est utilisé.

XML document de source et validation

Les documents XML utilisés dans l'exécution d'un document XQuery avec le moteur XQuery 1.0 doit être bien formé. Néanmoins, ils ne doivent pas être valides conformément à un schéma XML. Si le fichier n'est pas valide, le fichier invalide est chargé sans information de schéma. Si le fichier XML est associé avec un schéma interne et est valide conformément à ce schéma, l'information de validation post-schéma sera générée pour les données XML et sera utilisée pour l'évaluation de requête.

Contrôle de type statique et dynamique

La phase d'analyse statique contrôle les aspects de la requête comme la syntaxe, si des références externes existent (par ex. pour les modules), si des fonctions et des variables invoquées sont définies, etc. Si une erreur est détectée dans la phase de l'analyse statique, elle

sera rapportée et l'exécution sera stoppée.

Le contrôle de type dynamique est effectué lors de l'exécution, lorsque la requête est réellement exécutée. Si un type est incompatible avec les exigences d'une opération, une erreur sera rapportée. Par exemple, l'expression `xs:string("1") + 1` retourne une erreur parce que l'opération d'édition ne peut pas être effectuée sur un opérande de type `xs:string`.

Modules de bibliothèque

Les modules de Bibliothèque stockent les fonctions et les variables de manière à ce qu'elles puissent être réutilisées. Le moteur XQuery 1.0 prend en charge des modules qui sont stockés **dans un seul fichier XQuery externe**. Un tel fichier de module doit contenir une déclaration `module` dans son prologue, qui associe un espace de noms cible. Voici un module d'exemple :

```
module namespace libns="urn:module-library";
declare variable $libns:company := "Altova";
declare function libns:webaddress() { "http://www.altova.com" };
```

Toutes les fonctions et les variables déclarées dans le module font partie de l'espace de noms associé au module. Celui-ci est utilisé en l'important dans un fichier XQuery avec l'instruction `import module` se trouvant dans le prologue de requête. L'instruction `import module` importe uniquement les fonctions et les variables déclarées directement dans le fichier de module de bibliothèque. Comme suit :

```
import module namespace modlib = "urn:module-library" at "modulefilename.xq";

if        ($modlib:company = "Altova")
then       modlib:webaddress()
else       error("No match found.")
```

Fonctions externes

Les fonctions externes ne sont pas prises en charge, c.à.d. dans les expressions utilisant le mot-clé `external`, comme dans :

```
declare function hoo($param as xs:integer) as xs:string external;
```

Collations

La collation par défaut est la collation de point de code Unicode, qui compare les chaînes sur la base de leur point de code Unicode. Les autres collations prises en charge sont les collations ICU recensées ici. Pour utiliser une collation spécifique, fournir son URI tel que donné dans la liste des collations prises en charge. Toute comparaison de chaîne, y compris pour les fonctions `fn:max` et `fn:min` seront effectuées conformément à la collation spécifiée. Si l'option de collation n'est pas spécifiée, la collation de point de code Unicode par défaut est utilisée.

 © 2018 Altova GmbH

Précision des types numériques

- Le type de données `xs:integer` est une précision arbitraire, c.à.d. il peut représenter n'importe quel chiffre.
- Le type de données `xs:decimal` a une limite de 20 chiffres après la virgule.
- Les types de données `xs:float` et `xs:double` ont une précision limitée de 15 chiffres.

Prise en charge des instructions XQuery

L'instruction `Pragma` n'est pas prise en charge. Si elle survient, elle sera ignorée et l'expression de fallback sera évaluée.

9.5 XQuery 3.1

Le moteur XQuery 3.1 de RaptorXML+XBRL Server est conforme à la Recommandation XQuery 3.1 du 21 mars 2017 du Consortium du World Wide Web (W3C's) et comprend la prise en charge des Fonctions 3.1 XPath et XQuery. La spécification XQuery 3.1 est un sur ensemble de la spécification 3.0. C'est pourquoi le moteur XQuery 3.1 prend en charge les fonctions XQuery 3.0.

Les caractéristiques spécifiques à l'implémentation sont les mêmes que pour XQuery 1.0.

 © 2018 Altova GmbH

Chapitre 10

Fonctions XSLT et XPath/XQuery

10 Fonctions XSLT et XPath/XQuery

Cette section réunit les fonctions d'extension Altova et d'autres fonctions d'extension qui peuvent être utilisées dans les expressions XPath et/ou XQuery. Les fonctions d'extension Altova peuvent être utilisées avec les moteurs XSLT et XQuery d'Altova, et elles offrent des fonctions supplémentaires à celles disponibles dans les bibliothèques de fonctions définies dans les standards W3C.

Points généraux

Les points généraux suivants devraient être notés :

Normaltrue

- Les fonctions provenant des librairies des fonctions essentielles définies dans les spécifications W3C peuvent être appelées sans un préfixe. Cela est dû au fait que les moteurs XSLT et XQuery lisent les fonctions non préfixées comme faisant partie d'un espace de nom des fonctions par défaut qui est celui spécifié dans les spécifications des fonctions XPath/XQuery`http://www.w3.org/2005/xpath-functions`. Si cet espace de nom est déclaré explicitement dans un document XSLT ou XQuery, le préfixe utilisé dans la déclaration d'espace de nom peut aussi être utilisé en option sur les noms de fonction.
- En général, si une fonction escompte une séquence d'un item en tant qu'argument, et qu'une séquence de plus d'un item est soumise, une erreur sera retournée.
- Toutes les comparaisons de strings sont réalisées en utilisant la collation de point de code Unicode.
- Les résultats qui sont des QNames sont sérialisés sous la forme `[prefix:]localname`.

Précision de la décimale xs:

La précision se réfère au nombre de chiffres dans le nombre et la spécification requiert un minimum de 18 chiffres. Pour les opérations de division qui produisent un résultat de type `xs:decimal`, la précision est de 19 chiffres après le point décimal sans arrondissement.

Fuseau horaire implicite

Lorsque deux valeurs `date`, `time`, ou `dateTime` doivent être comparées, le fuseau horaire des valeurs comparées doit être connu. Si le fuseau n'est explicitement donné dans une telle valeur, le fuseau horaire implicite est utilisé. Le fuseau horaire implicite est prélevé de l'horloge du système et sa valeur peut être contrôlée avec la fonction `implicit-timezone()`.

Collations

La collation par défaut est la collation de point de code Unicode qui compare les chaînes sur la base de leur point de code Unicode. Le processeur utilise l'Unicode Collation Algorithm. D'autres collations prises en charge sont les collations ICU regroupées ci-dessous : pour utiliser une collation spécifique, il faut fournir son URI tel que spécifié dans la table ci-dessous. Toute comparaison de chaîne, y compris en ce qui concerne les fonctions `max` et `min`, sera effectuée conformément à la collation spécifiée. Si l'option de collation n'est pas spécifiée, la collation de point de code Unicode par défaut sera utilisée.

Langue	URI

 © 2018 Altova GmbH

da: Danois	da_DK
de: Allemand	de_AT, de_BE, de_CH, de_DE, de_LI, de_LU
en: Anglais	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: Espagnol	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: Français	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: Italien	it_CH, it_IT
ja: Japonais	ja_JP
nb: Norvégien Bokmål	nb_NO
nl: Néerlandais	nl_AW, nl_BE, nl_NL
nn: Nynorsk	nn_NO
pt: Portugais	pt_AO, pt_BR, pt_GW, pt_MZ, pt_PT, pt_ST
ru: Russe	ru_MD, ru_RU, ru_UA
sv: Suédois	sv_FI, sv_SE

Axe du nom d'espace

L'axe du nom d'espace est devenu obsolète dans XPath 2.0. Néanmoins, l'utilisation de l'axe du nom d'espace est prise en charge. Pour accéder aux information de l'espace de nom, avec des mécanismes XPath 2.0, utilisez les fonctions `in-scope-prefixes()`, `namespace-uri()` et `namespace-uri-for-prefix()`.

10.1 Fonctions d'extension Altova

Les fonctions d'extension Altova peuvent être utilisées dans les expressions XPath/XQuery. Elles fournissent des fonctions supplémentaires aux fonctions d'ores et déjà disponibles dans la librairie standard des fonctions XPath, XQuery et XSLT. Les fonctions d'extension Altova se trouvent dans **l'espace de nom des fonctions d'extension Altova,** `http://www.altova.com/xslt-extensions`, et sont indiquées dans cette section par le préfixe `altova:`, qui est présupposé être lié à cet espace de nom. Veuillez noter que, en ce qui concerne les versions futures de votre produit, la prise en charge d'une fonction peut être interrompue et le comportement de certaines fonctions peut changer. Veuillez consulter la documentation lors des publications à venir pour plus d'informations concernant la prise en charge des fonctions d'extension Altova de cette version.

Les fonctions définies dans les spécifications de Fonctions XPath/XQuery de W3C peuvent être utilisées dans : (i) les expressions XPath dans un contexte XSLT, et (ii) dans les expressions XQuery dans un document XQuery. Dans cette documentation, nous indiquons les fonctions à utiliser dans le contexte précédent (XPath dans XSLT) avec un symbole **XP** et les appelons fonctions XPath ; les fonctions qui peuvent être utilisées dans le contexte à venir (XQuery) sont indiquées avec un symbole **XQ** ; elles fonctionnent en tant que fonctions XQuery. Les spécifications XSLT de W3C —pas les spécifications de Fonctions XPath/XQuery —définissent également les fonctions qui peuvent être utilisées dans des expressions XPath dans des documents XSLT. Ces fonctions sont marquées avec un symbole **XSLT** et sont appelées fonctions XSLT. Les versions XPath/XQuery et XSLT dans lesquelles une fonction peut être utilisée sont indiquées dans la description de la fonction (*voir symboles ci-dessous*). Les fonctions provenant des librairies de fonction XPath/XQuery et XSLT sont recensées dans un préfixe. Les fonctions d'extension provenant d'autres librairies, comme les fonctions d'extension Altova, sont regroupés avec un préfixe.

Fonctions XPath (utilisées dans les expressions XPath dans XSLT) :	**XP1** **XP2** **XP3.1**
Fonctions XSLT (utilisées dans les expressions XPath dans XSLT) :	**XSLT1** **XSLT2** **XSLT3**
Fonctions XQuery (utilisées dans les expressions XQuery dans XQuery) :	**XQ1** **XQ3.1**

Fonctions XSLT

Les fonctions XSLT peuvent uniquement être utilisées dans des expressions XPath dans un contexte XSLT (comme les fonctions `current-group()` ou `key()` de XSLT 2.0). Ces fonctions ne sont pas prévues pour, et ne fonctionneront pas dans un contexte non-XSLT (par exemple, dans un contexte XQuery). Veuillez noter que les fonctions XSLT pour XBRL peuvent être utilisées uniquement avec les éditions des produits Altova qui ont une prise en charge XBRL.

Fonctions XPath/XQuery

Les fonctions XPath/XQuery peuvent être utilisées tous les deux dans les expressions XPath

dans les contextes XSLT et dans les expressions XQuery :

- Date/Heure
- Géolocalisation
- Liée à l'image
- Numérique
- Séquence
- Chaîne
- Divers

Fonctions Graphiques (Éditions Enterprise et Server uniquement seulement

Fonctions d'extension Altova pour les graphiques sont prises en charge uniquement par les édition Enterprise et Server des produits Altova et permettent de générer des graphiques d'être généré à partir de données XML.

Fonctions de code-barres

Les fonctions d'extension à code-barres d'Altova permettent la génération de code-barres et leur placement dans une sortie générée par le biais des feuilles de styles XSLT.

Fonctions XSLT

Les fonctions d'extension XSLT peuvent être utilisées dans les expressions XPath dans un contexte XSLT. Elles ne fonctionneront pas dans un contexte non-XSLT (par exemple dans un contexte XQuery).

Note concernant le nommage de fonctions et de l'applicabilité de la langue

Les fonctions d'extension Altova peuvent être utilisées dans les expressions XPath/XQuery. Elles fournissent des fonctions supplémentaires aux fonctions d'ores et déjà disponibles dans la librairie standard des fonctions XPath, XQuery et XSLT. Les fonctions d'extension Altova se trouvent dans **l'espace de nom des fonctions d'extension Altova,** `http://www.altova.com/xslt-extensions`, et sont indiquées dans cette section par le préfixe `altova:`, qui est présupposé être lié à cet espace de nom. Veuillez noter que, en ce qui concerne les versions futures de votre produit, la prise en charge d'une fonction peut être interrompue et le comportement de certaines fonctions peut changer. Veuillez consulter la documentation lors des publications à venir pour plus d'informations concernant la prise en charge des fonctions d'extension Altova de cette version.

Fonctions XPath (utilisées dans les expressions XPath dans XSLT) :	XP1 XP2 XP3.1
Fonctions XSLT (utilisées dans les expressions XPath dans XSLT) :	XSLT1 XSLT2 XSLT3
Fonctions XQuery (utilisées dans les expressions XQuery dans XQuery) :	XQ1 XQ3.1

Fonctions standard

distinct-nodes [altova:]

altova:distinct-nodes(*node()****) as node()*** XSLT1 XSLT2 XSLT3

Prend un ensemble d'un ou de plusieurs nœuds en tant que son entrée et retourne le même ensemble moins les nœuds avec des valeurs dupliquées. La comparaison s'effectue en utilisant la fonction XPath/XQuery `fn:deep-equal`.

Exemples

- **altova:distinct-nodes**(`country`) retourne tous les nœuds `country` enfant moins ceux possédant des valeurs dupliquées.

evaluate [altova:]

altova:evaluate(XPathExpression *as xs:string***[, ValueOf$p1, ... ValueOf$pN])** XSLT1 XSLT2 XSLT3

Prend une expression XPath, passée en tant que chaîne, en tant que son argument obligatoire. Elle retourne la sortie de l'expression évaluée. Par exemple : **altova:evaluate('//Name[1]')** retourne les contenues du premier élément `Name` dans le document. Veuillez noter que l'expression `//Name[1]` est passée en tant que chaîne en l'enfermant dans des guillemets simples.

La fonction `altova:evaluate` peut prendre des arguments supplémentaires en option. Ces

 © 2018 Altova GmbH

arguments sont les valeurs des variables in-scope qui portent les noms `p1`, `p2`, `p3`... `pN`. Veuillez noter les points suivants concernant l'utilisation : (i) Les variables doivent être définies avec les noms de la formule `pX`, lorsque `X` est un entier ; (ii) les arguments de la fonction `altova:evaluate` (*voir signature ci-dessus*), à partir du deuxième argument, fournissent les valeurs de la variables, avec la séquence des arguments correspondant à la séquence des variables classées numériquement : `p1` à `pN`: le deuxième argument sera la valeur de la variable `p1`, le troisième argument celui de la variable `p2`, etc. ; (iii) Les valeurs de variable doivent être de type `item*`.

Exemple

```
<xsl:variable name="xpath" select="'$p3, $p2, $p1'" />
<xsl:value-of select="altova:evaluate($xpath, 10, 20, 'hi')" />
```

outputs `"hi 20 10"`

Dans les listes ci-dessus, veuillez noter les points suivants :

- Le deuxième argument de l'expression `altova:evaluate` est la valeur attribuée à la variable `$p1`, le troisième argument est celui attribué à la variable `$p2`, etc.
- Veuillez noter que le quatrième argument de la fonction est une valeur de chaîne, ce qui est indiqué par le fait qu'elle est contenue dans des guillemets.
- L'attribut `select` de l'élément `xs:variable` fournit l'expression XPath. Puisque cette expression doit être de type `xs:string`, elle est contenue dans des guillemets simples.

Exemples pour mieux illustrer l'utilisation des variables

- ```
  <xsl:variable name="xpath" select="'$p1'" />
  <xsl:value-of select="altova:evaluate($xpath, //Name[1])" />
  ```
  Sort la valeur du premier élément `Name`.

- ```
  <xsl:variable name="xpath" select="'$p1'" />
  <xsl:value-of select="altova:evaluate($xpath, '//Name[1]')" />
  ```
  Sort `"//Name[1]"`

La fonction d'extension `altova:evaluate()` est utile lorsqu'une expression XPath dans la feuille de style XSLT contient une ou plusieurs parties qui doivent être évaluées dynamiquement. Par exemple, prenez comme exemple une situation dans laquelle un utilisateur saisit sa requête pour le critère de tri et le critère est stocké dans l'attribut `UserReq/@sortkey`. Dans la feuille de style, vous pouvez ensuite avoir l'expression : `<xsl:sort select="`**`altova:evaluate(../UserReq/@sortkey)`**`" order="ascending"/>`. La fonction `altova:evaluate()` lit l'attribut `sortkey` de l'élément enfant `UserReq` du parent du nœud contextuel. Si, par exemple, la valeur de l'attribut `sortkey` est `Price`, alors `Price` est retourné par la fonction `altova:evaluate()` et devient la valeur de l'attribut `select` : `<xsl:sort select="`**`Price`**`" order="ascending"/>`. Si cette instruction `sort` apparaît dans le contexte d'un élément appelé `Order`, alors les éléments `Order` seront triés conformément aux valeurs de leurs enfants `Price`. En alternative, si la valeur de `@sortkey` était, par exemple, `Date`, alors les éléments `Order` seraient triés selon les valeurs de leurs enfants `Date`. Donc le critère de triage pour `Order` est choisi à partir de l'attribut `sortkey` lors de l'exécution. Cela n'aurait pas pu se réaliser avec une expression telle que : `<xsl:sort select="`**`../UserReq/@sortkey`**`" order="ascending"/>`. Dans le cas montré ci-dessus, le critère de tri aurait été l'attribut `sortkey` lui-même, et non pas `Price` ou `Date` (ou tout autre

contenu actuel de `sortkey`).

Note : Le contexte statique inclut des espaces de nom, des types et des fonctions, mais pas des variables, depuis l'environnement d'appel. L'URI de base et l'espace de nom par défaut sont hérités.

Plus d'exemples

- Variables statiques : `<xsl:value-of select="$i3, $i2, $i1" />`
  Sort les valeurs des trois variables.

- Expression XPath dynamique avec des variables dynamiques :
  ```
  <xsl:variable name="xpath" select="'$p3, $p2, $p1'" />
  <xsl:value-of select="altova:evaluate($xpath, 10, 20, 30)" />
  ```
  Sortie "30 20 10"

- Expression XPath dynamique sans variable dynamique :
  ```
  <xsl:variable name="xpath" select="'$p3, $p2, $p1'" />
  <xsl:value-of select="altova:evaluate($xpath)" />
  ```
  Sortie erreur : Aucune variable définie pour $p3.

encode-for-rtf [altova:]

altova:encode-for-rtf(input *as xs:string*, **preserveallwhitespace** *as xs:boolean*, **preservenewlines** *as xs:boolean*) as **xs:string** XSLT2 XSLT3

Convertit la chaîne d'entrée en tant que code pour RTF. Les espaces blancs et les nouvelles lignes seront préservés selon la valeur booléenne spécifiée pour leurs arguments respectifs.

[Haut]

Fonctions XBRL

Les fonctions XBRL Altova peuvent uniquement être utilisées avec des éditions des produits Altova qui présentent une prise en charge XBRL.

xbrl-footnotes [altova:]

altova:xbrl-footnotes(node()**)** as **node()*** XSLT2 XSLT3

Prend un nœud en tant que son argument d'entrée et retourne l'ensemble des nœuds de notes de pieds XBRL référencées par le nœud d'entrée.

xbrl-labels [altova:]

altova:xbrl-labels(xs:QName, xs:string**)** as **node()*** XSLT2 XSLT3

Prend deux arguments d'entrée : un nom de nœud et l'emplacement de fichier de taxonomie contenant le nœud. La fonction retourne les nœuds de libellés XBRL associés avec le nœud d'entrée.

 © 2018 Altova GmbH

[Haut]

Fonctions XPath/XQuery : Date et heure

Les fonctions d'extension date/heure d'Altova peuvent être utilisées dans les expressions XPath et XQuery et fournissent des fonctions supplémentaires pour le traitement des données contenues en tant que les types de données de date et d'heures variés de XML Schema. Les fonctions dans cette section peuvent être utilisées avec les moteurs **XPath 3.0** et **XQuery 3.0** d'Altova. Ils sont disponibles dans les contextes XPath/XQuery.

Note concernant le nommage de fonctions et de l'applicabilité de la langue

Les fonctions d'extension Altova peuvent être utilisées dans les expressions XPath/XQuery. Elles fournissent des fonctions supplémentaires aux fonctions d'ores et déjà disponibles dans la librairie standard des fonctions XPath, XQuery et XSLT. Les fonctions d'extension Altova se trouvent dans **l'espace de nom des fonctions d'extension Altova,** `http://www.altova.com/xslt-extensions`, et sont indiquées dans cette section par le préfixe `altova:`, qui est présupposé être lié à cet espace de nom. Veuillez noter que, en ce qui concerne les versions futures de votre produit, la prise en charge d'une fonction peut être interrompue et le comportement de certaines fonctions peut changer. Veuillez consulter la documentation lors des publications à venir pour plus d'informations concernant la prise en charge des fonctions d'extension Altova de cette version.

Fonctions XPath (utilisées dans les expressions XPath dans XSLT) :	XP1 XP2 XP3.1
Fonctions XSLT (utilisées dans les expressions XPath dans XSLT) :	XSLT1 XSLT2 XSLT3
Fonctions XQuery (utilisées dans les expressions XQuery dans XQuery) :	XQ1 XQ3.1

Regroupées selon les fonctionnalités

- Ajouter une durée à xs:dateTime et retourner xs:dateTime
- Ajouter une durée à xs:date et retourner xs:date
- Ajouter une durée à xs:time et retourner à xs:time
- Formater et récupérer des durées
- Supprimer des fuseaux horaires de fonctions qui génèrent des date/heures actuels
- Retourne un jour de la semaine en tant qu'un entier à partir de la date
- Retourne le numéro de la semaine en tant qu'un entier à partir de la date
- Construire le type de date, d'heure ou de durée à partir des composants lexicaux de chaque type
- Construire le type de date, dateHeure ou heure à partir de l'entrée de chaîne
- Fonctions liées à l'âge

Regroupées alphabétiquement

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

 © 2018 Altova GmbH

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

[Haut]

Ajouter une durée à `xs:dateTime` XP3.1 XQ3.1

Ces fonctions ajoutent une durée à **`xs:dateTime`** et retournent **`xs:dateTime`**. Le type `xs:dateTime` a un format de `CCYY-MM-DDThh:mm:ss.sss`. Il s'agit d'une concaténation des formats `xs:date` et `xs:time` séparés par la lettre `T`. Un suffixe de fuseau horaire`+01:00 (par exemple)` est optionnel.

add-years-to-dateTime [altova:]

`altova:add-years-to-dateTime(DateTime` *`as xs:dateTime`*`, Years` *`as xs:integer`*`)` as **`xs:dateTime` XP3.1 XQ3.1

Ajoute une durée en années à `xs:dateTime` (*voir exemples ci-dessous*). Le deuxième argument est le nombre d'années à être ajouté à `xs:dateTime` fourni en tant que le premier argument. Le résultat est de type `xs:dateTime`.

Exemples

- **`altova:add-years-to-dateTime`**`(xs:dateTime("2014-01-15T14:00:00"), 10)` retourne `2024-01-15T14:00:00`
- **`altova:add-years-to-dateTime`**`(xs:dateTime("2014-01-15T14:00:00"), -4)` retourne `2010-01-15T14:00:00`

add-months-to-dateTime [altova:]

`altova:add-months-to-dateTime(DateTime` *`as xs:dateTime`*`, Months` *`as xs:integer`*`)** as **`xs:dateTime` XP3.1 XQ3.1

Ajoute une durée en mois à `xs:dateTime` (*voir exemples ci-dessous*). Le deuxième argument est le nombre de mois à être ajouté à `xs:dateTime` fourni en tant que le premier argument. Le résultat est de type `xs:dateTime`.

Exemples

- `altova:add-months-to-dateTime(xs:dateTime("2014-01-15T14:00:00"), 10)` retourne `2014-11-15T14:00:00`
- `altova:add-months-to-dateTime(xs:dateTime("2014-01-15T14:00:00"), -2)` retourne `2013-11-15T14:00:00`

add-days-to-dateTime [altova:]

altova:add-days-to-dateTime(DateTime *as xs:dateTime*, **Days** *as xs:integer*) as **xs:dateTime** **XP3.1** **XQ3.1**

Ajoute une durée en jours à `xs:dateTime` (*voir exemples ci-dessous*). Le deuxième argument est le nombre de jours à être ajouté à `xs:dateTime` fourni en tant que le premier argument. Le résultat est de type `xs:dateTime`.

Exemples

- `altova:add-days-to-dateTime(xs:dateTime("2014-01-15T14:00:00"), 10)` retourne `2014-01-25T14:00:00`
- `altova:add-days-to-dateTime(xs:dateTime("2014-01-15T14:00:00"), -8)` retourne `2014-01-07T14:00:00`

add-hours-to-dateTime [altova:]

altova:add-hours-to-dateTime(DateTime *as xs:dateTime*, **Hours** *as xs:integer*) as **xs:dateTime** **XP3.1** **XQ3.1**

Ajoute une durée en heures à `xs:dateTime` (*voir exemples ci-dessous*). Le deuxième argument est le nombre d'heures à être ajouté à `xs:dateTime` fourni en tant que le premier argument. Le résultat est de type `xs:dateTime`.

Exemples

- `altova:add-hours-to-dateTime(xs:dateTime("2014-01-15T13:00:00"), 10)` retourne `2014-01-15T23:00:00`
- `altova:add-hours-to-dateTime(xs:dateTime("2014-01-15T13:00:00"), -8)` retourne `2014-01-15T05:00:00`

add-minutes-to-dateTime [altova:]

altova:add-minutes-to-dateTime(DateTime *as xs:dateTime*, **Minutes** *as xs:integer*) as **xs:dateTime** **XP3.1** **XQ3.1**

Ajoute une durée en minutes à `xs:dateTime` (*voir exemples ci-dessous*). Le deuxième argument est le nombre of minutes à être ajouté à `xs:dateTime` fourni en tant que le premier argument. Le résultat est de type `xs:dateTime`.

Exemples

- `altova:add-minutes-to-dateTime(xs:dateTime("2014-01-15T14:10:00"), 45)` retourne `2014-01-15T14:55:00`
- `altova:add-minutes-to-dateTime(xs:dateTime("2014-01-15T14:10:00"), -5)` retourne `2014-01-15T14:05:00`

add-seconds-to-dateTime [altova:]

 © 2018 Altova GmbH

altova:add-seconds-to-dateTime(DateTime *as xs:dateTime*, **Seconds** *as xs:integer*) as **xs:dateTime** XP3.1 XQ3.1

Ajoute une durée en secondes à `xs:dateTime` (*voir exemples ci-dessous*). Le deuxième argument est le nombre de secondes à être ajouté à `xs:dateTime` fourni en tant que le premier argument. Le résultat est de type `xs:dateTime`.

Exemples

- **altova:add-seconds-to-dateTime**(xs:dateTime("2014-01-15T14:00:10"), 20) retourne 2014-01-15T14:00:30
- **altova:add-seconds-to-dateTime**(xs:dateTime("2014-01-15T14:00:10"), -5) retourne 2014-01-15T14:00:05

[Haut]

Ajouter une durée à xs:date XP3.1 XQ3.1

Ces fonctions ajoutent une durée à **xs:date** et retournent **xs:date**. Le type `xs:date` a un format `CCYY-MM-DD`.

add-years-to-date [altova:]

altova:add-years-to-date(Date *as xs:date*, **Years** *as xs:integer*) as **xs:date** XP3.1 XQ3.1

Ajoute une durée en années à une date. Le deuxième argument est le nombre d'années à être ajouté à `xs:date` fourni en tant que le premier argument. Le résultat est de type `xs:date`.

Exemples

- **altova:add-years-to-date**(xs:date("2014-01-15"), 10) retourne 2024-01-15
- **altova:add-years-to-date**(xs:date("2014-01-15"), -4) retourne 2010-01-15

add-months-to-date [altova:]

altova:add-months-to-date(Date *as xs:date*, **Months** *as xs:integer*) as **xs:date** XP3.1 XQ3.1

Ajoute une durée en mois à une date. Le deuxième argument est le nombre de mois à être ajouté à `xs:date` fourni en tant que le premier argument. Le résultat est de type `xs:date`.

Exemples

- **altova:add-months-to-date**(xs:date("2014-01-15"), 10) retourne 2014-11-15
- **altova:add-months-to-date**(xs:date("2014-01-15"), -2) retourne 2013-11-15

add-days-to-date [altova:]

altova:add-days-to-date(Date *as xs:date*, **Days** *as xs:integer*) as **xs:date** XP3.1 XQ3.1

Ajoute une durée en jours à une date. Le deuxième argument est le nombre de jours à être ajouté à `xs:date` fourni en tant que le premier argument. Le résultat est de type `xs:date`.

Exemples

- **altova:add-days-to-date**(xs:date("2014-01-15"), 10) retourne 2014-01-25
- **altova:add-days-to-date**(xs:date("2014-01-15"), -8) retourne 2014-01-07

[Haut]

Formater et récupérer des durées XP3.1 XQ3.1

Ces fonctions ajoutent une durée à **xs:date** et retournent **xs:date**. Le type xs:date a un format de CCYY-MM-DD.

format-duration [altova:]

altova:format-duration(**Duration** *as xs:duration*, **Picture** *as xs:string*) as **xs:string** XP3.1 XQ3.1

Formate une durée qui est soumise en tant que le premier argument, selon une chaîne d'image soumise en tant que le second argument. La sortie est une chaîne de texte formatée conformément à la chaîne d'image.

Exemples

- **altova:format-duration**(xs:duration("P2DT2H53M11.7S"), "Days:[D01] Hours:[H01] Minutes:[m01] Seconds:[s01] Fractions:[f0]") retourne "Days:02 Hours:02 Minutes:53 Seconds:11 Fractions:7"
- **altova:format-duration**(xs:duration("P3M2DT2H53M11.7S"), "Months:[M01] Days:[D01] Hours:[H01] Minutes:[m01]") retourne "Months:03 Days:02 Hours:02 Minutes:53"

parse-duration [altova:]

altova:parse-duration(**InputString** *as xs:string*, **Picture** *as xs:string*) as **xs:duration** XP3.1 XQ3.1

Prend un patterned string en tant que le premier argument et une chaîne image en tant que le second argument. La chaîne d'entrée est parsée sur la base de la chaîne d'image et une xs:duration est retournée.

Exemples

- **altova:parse-duration**("Days:02 Hours:02 Minutes:53 Seconds:11 Fractions:7"), "Days:[D01] Hours:[H01] Minutes:[m01] Seconds:[s01] Fractions:[f0]") retourne "P2DT2H53M11.7S"
- **altova:parse-duration**("Months:03 Days:02 Hours:02 Minutes:53 Seconds:11 Fractions:7", "Months:[M01] Days:[D01] Hours:[H01] Minutes:[m01]") retourne "P3M2DT2H53M"

[Haut]

Ajouter une durée à xs:time XP3.1 XQ3.1

Ces fonctions ajoutent une durée à **xs:time** et retournent **xs:time**. Le type xs:time a une forme lexicale de hh:mm:ss.sss. Un fuseau horaire en option peut être suffixé. La lettre Z indique le Temps universel coordonné (UTC). Tous les autres fuseaux horaires sont représentés par leur différence de l'UTC dans le format +hh:mm, ou -hh:mm. Si aucune valeur de fuseau horaire n'est

présente, elle est considérée inconnue ; elle n'est pas considérée être UTC.

add-hours-to-time [altova:]

altova:add-hours-to-time(Time *as xs:time*, **Hours** *as xs:integer*) as **xs:time**
XP3.1 **XQ3.1**
Ajoute une durée en heures à une heure de temps. Le deuxième argument est le nombre d'heures à être ajouté à `xs:time` fourni en tant que le premier argument. Le résultat est de type `xs:time`.

Exemples

- **altova:add-hours-to-time**(xs:time("11:00:00"), 10) retourne 21:00:00
- **altova:add-hours-to-time**(xs:time("11:00:00"), -7) retourne 04:00:00

add-minutes-to-time [altova:]

altova:add-minutes-to-time(Time *as xs:time*, **Minutes** *as xs:integer*) as **xs:time**
XP3.1 **XQ3.1**
Ajoute une durée en minutes à une heure. Le deuxième argument est le nombre de minutes à être ajouté à `xs:time` fourni en tant que le premier argument. Le résultat est de type `xs:time`.

Exemples

- **altova:add-minutes-to-time**(xs:time("14:10:00"), 45) retourne 14:55:00
- **altova:add-minutes-to-time**(xs:time("14:10:00"), -5) retourne 14:05:00

add-seconds-to-time [altova:]

altova:add-seconds-to-time(Time *as xs:time*, **Minutes** *as xs:integer*) as **xs:time**
XP3.1 **XQ3.1**
Ajoute une durée en secondes à une heure. Le deuxième argument est le nombre de secondes à être ajouté à `xs:time` fourni en tant que le premier argument. Le résultat est de type `xs:time`. Le composant Secondes peut être contenu dans une plage de `0` à `59.999`.

Exemples

- **altova:add-seconds-to-time**(xs:time("14:00:00"), 20) retourne 14:00:20
- **altova:add-seconds-to-time**(xs:time("14:00:00"), 20.895) retourne 14:00:20.895

[Haut]

Supprimer la partie du fuseau horaire des types de données date/heures

XP3.1 **XQ3.1**

Ces fonctions permettent de supprimer le fuseau horaire des valeurs **xs:dateTime**, **xs:date** ou **xs:time** actuelles, respectivement. Veuillez noter que la différence entre `xs:dateTime` et `xs:dateTimeStamp` est que dans le cas de ce dernier, la partie fuseau horaire est requise (alors qu'elle est optionnelle dans le premier des deux cas). Donc, le format d'une valeur `xs:dateTimeStamp` est : `CCYY-MM-DDThh:mm:ss.sss±hh:mm`. ou `CCYY-MM-DDThh:mm:ss.sssZ`. Si la date et l'heure sont lues depuis l'horloge du système, en tant que `xs:dateTimeStamp`, la

fonction `current-dateTime-no-TZ()` peut être utilisée pour supprimer le fuseau horaire s'il est requis.

current-dateTime-no-TZ [altova:]

altova:current-dateTime-no-TZ() as **xs:dateTime** XP3.1 XQ3.1

Cette fonction ne prend aucun argument. Elle supprime la partie fuseau horaire de `current-dateTime()` (qui est la date-heure actuelle selon l'horloge système) et retourne une valeur`xs:dateTime`.

Exemples

Si la date-heure actuelle est **2014-01-15T14:00:00+01:00**:

- **altova:current-dateTime-no-TZ()** retourne `2014-01-15T14:00:00`

current-date-no-TZ [altova:]

altova:current-date-no-TZ() as **xs:date** XP3.1 XQ3.1

Cette fonction ne prend aucun argument. Elle supprime la partie fuseau horaire de `current-date()` (qui est la date actuelle selon l'horloge système) et retourne une valeur `xs:date`.

Exemples

Si la date actuelle est **2014-01-15+01:00**:

- **altova:current-date-no-TZ()** retourne `2014-01-15`

current-time-no-TZ [altova:]

altova:current-time-no-TZ() as **xs:time** XP3.1 XQ3.1

Cette fonction ne prend aucun argument. Elle supprime la partie fuseau horaire de `current-time()` (qui est l'heure actuelle selon l'horloge système) et retourne une valeur `xs:time`.

Exemples

Si l'heure actuelle est **14:00:00+01:00**:

- **altova:current-time-no-TZ()** retourne `14:00:00`

[Haut]

Retourne le jour de la semaine à partir de xs:dateTime ou xs:date XP3.1 XQ3.1

Ces fonctions retournent le jour de la semaine (en tant qu'entier) depuis `xs:dateTime` ou `xs:date`. Les jours de la semaine sont numérotés (format américain) de `1` à `7`, avec `Sunday=1`. Dans le format européen, la semaine commence par Lundi (`=1`). Dans le format américain elle commence par `Sunday=1`. Configurer en utilisant l'entier `0` et où un entier est accepté pour indiquer le format.

weekday-from-dateTime [altova:]

altova:weekday-from-dateTime(DateTime *as xs:dateTime***)** as **xs:integer** XP3.1

XQ3.1
Prend une date-avec-heure en tant que son seul argument et retourne le jour de la semaine de cette date sous forme d'un entier. Les jours de la semaine sont numérotés en commençant avec `Sunday=1`. Si le format européen est requis (où `Monday=1`), utiliser l'autre signature de cette fonction (*voir signature suivante ci-dessous*).

Exemples

- `altova:weekday-from-dateTime(xs:dateTime("2014-02-03T09:00:00"))` retourne `2`, ce qui indique un lundi.

altova:weekday-from-dateTime(DateTime *as xs:dateTime*, **Format** *as xs:integer*) as **xs:integer** XP3.1 XQ3.1
Prend une date-avec-heure en tant que son premier argument et retourne le jour de la semaine de cette date sous forme d'un entier. Les jours de la semaine sont numérotés en commençant avec `Monday=1`. Si le second argument (entier) est `0`, les jours de la semaine sont numérotés de `1` à `7` en commençant avec `Sunday=1`. Si le second argument est un entier différent de `0`, alors `Monday=1`. S'il n'y a pas de second argument, la fonction est lue comme possédant l'autre signature de cette fonction (*voir signature précédente*).

Exemples

- `altova:weekday-from-dateTime(xs:dateTime("2014-02-03T09:00:00"), 1)` retourne `1`, ce qui indique un lundi
- `altova:weekday-from-dateTime(xs:dateTime("2014-02-03T09:00:00"), 4)` retourne `1`, ce qui indique un lundi
- `altova:weekday-from-dateTime(xs:dateTime("2014-02-03T09:00:00"), 0)` retourne `2`, ce qui indique un lundi.

weekday-from-date [altova:]

altova:weekday-from-date(Date *as xs:date*) as **xs:integer** XP3.1 XQ3.1
Prend une date en tant que son seul argument et retourne le jour de la semaine de cette date sous forme d'un entier. Les jours de la semaine sont numérotés en commençant avec `Sunday=1`. Si le format européen est requis (où `Monday=1`), utiliser l'autre signature de cette fonction (*voir signature suivante ci-dessous*).

Exemples

- `altova:weekday-from-date(xs:date("2014-02-03+01:00"))` retourne `2`, ce qui indique un lundi.

altova:weekday-from-date(Date *as xs:date*, **Format** *as xs:integer*) as **xs:integer** XP3.1 XQ3.1
Prend une date en tant que son premier argument et retourne le jour de la semaine de cette date sous forme d'un entier. Les jours de la semaine sont numérotés en commençant avec `Monday=1`. Si le second argument (`Format`) est `0`, les jours de la semaine sont numérotés de `1` à `7` en commençant avec `Sunday=1`. Si le second argument est un entier différent de `0`, alors `Monday=1`. S'il n'y a pas de second argument, la fonction est lue comme possédant l'autre signature de cette fonction (*voir signature précédente*).

Exemples

- `altova:weekday-from-date(xs:date("2014-02-03"), 1)` retourne `1`, ce qui indique un lundi
- `altova:weekday-from-date(xs:date("2014-02-03"), 4)` retourne `1`, ce qui indique un lundi

- `altova:weekday-from-date(xs:date("2014-02-03"), 0)` retourne `2`, ce qui indique un lundi.

[Haut]

Retourne le nombre de la semaine à partir de `xs:dateTime` ou `xs:date` XP2 XQ1 XP3.1 XQ3.1

Ces fonctions retournent le numéro de la semaine (en tant qu'un entier) depuis `xs:dateTime` or `xs:date`. La numérotation des semaines est disponible dans les formats de calendrier US, ISO/European et Islamiques. La numérotation des semaines est différente dans ces formats de calendrier parce que la semaine est considérée démarrer avec un jour différent selon le format (dimanche pour le format US, lundi pour le format ISO/Européen, et samedi dans le format islamique).

weeknumber-from-date [altova:]

altova:weeknumber-from-date(Date *as xs:date*, **Calendar** *as xs:integer*) as **xs:integer** XP2 XQ1 XP3.1 XQ3.1

Retourne le numéro de la semaine de l'argument **Date** soumis en tant qu'entier. Le deuxième argument (**Calendar**) spécifie le système de calendrier à suivre.
Les valeurs de **Calendar** prises en charge sont :

- **0 = US calendar** (*semaine commence dimanche*)
- **1 = ISO standard, European calendar** (*semaine commence lundi*)
- **2 = Islamic calendar** (*semaine commence samedi*)

Le réglage par défaut est `0`.

Exemples

- `altova:weeknumber-from-date(xs:date("2014-03-23"), 0)` retourne `13`
- `altova:weeknumber-from-date(xs:date("2014-03-23"), 1)` retourne `12`
- `altova:weeknumber-from-date(xs:date("2014-03-23"), 2)` retourne `13`
- `altova:weeknumber-from-date(xs:date("2014-03-23") )` retourne `13`

Le jour de la date dans les exemples ci-dessus (`2014-03-23`) est dimanche. Les calendriers US et musulmans sont donc une semaine en avant par rapport au calendrier européen à ce jour.

weeknumber-from-dateTime [altova:]

altova:weeknumber-from-dateTime(DateTime *as xs:dateTime*, **Calendar** *as xs:integer*) as **xs:integer** XP2 XQ1 XP3.1 XQ3.1

Retourne le numéro de la semaine de l'argument **DateTime** soumis en tant qu'entier. Le deuxième argument (**Calendar**) spécifie le système de calendrier à suivre.
Les valeurs de **Calendar** prises en charge sont :

- `0 = US calendar` (*semaine commence dimanche*)
- `1 = ISO standard, European calendar` (*semaine commence lundi*)
- `2 = Islamic calendar` (*semaine commence samedi*)

Le réglage par défaut est `0`.

Exemples

- `altova:weeknumber-from-dateTime(xs:dateTime("2014-03-23T00:00:00"), 0)` retourne `13`
- `altova:weeknumber-from-dateTime(xs:dateTime("2014-03-23T00:00:00"), 1)` retourne `12`
- `altova:weeknumber-from-dateTime(xs:dateTime("2014-03-23T00:00:00"), 2)` retourne `13`
- `altova:weeknumber-from-dateTime(xs:dateTime("2014-03-23T00:00:00") )` retourne `13`

Le jour du dateTime dans les exemples ci-dessus (`2014-03-23T00:00:00`) est dimanche. Les calendriers US et musulmans sont donc une semaine en avant par rapport au calendrier européen à ce jour.

[Haut]

Construire le type de date, d'heure ou de durée à partir de leurs composants lexicaux XP3.1 XQ3.1

Les fonctions prennent les composants lexicaux du type de données `xs:date`, `xs:time` ou `xs:duration` en tant qu'arguments d'entrée et les combinent pour construire le type de données respectif.

build-date [altova:]

altova:build-date(Year *as xs:integer*, **Month** *as xs:integer*, **Date** *as xs:integer*) as **xs:date** XP3.1 XQ3.1

Les premier, second et troisième arguments sont respectivement l'année, le mois et la date. Ils sont combinés pour construire une valeur de type `xs:date`. Les valeurs de l'entier doivent se situer dans le cadre de la plage correcte de cette partie de la date. Par exemple, le deuxième argument (pour la partie du mois) ne devrait pas être supérieur à `12`.

Exemples

- `altova:build-date(2014, 2, 03)` retourne `2014-02-03`

build-time [altova:]

altova:build-time(Hours *as xs:integer*, **Minutes** *as xs:integer*, **Seconds** *as xs:integer*) as **xs:time** XP3.1 XQ3.1

Les premiers, seconds et troisièmes arguments sont, respectivement, les valeurs d'heure (`0` to `23`), de minutes (`0` to `59`) et de secondes (`0` to `59`). Ils sont combinés pour construire une valeur de type `xs:time`. Les valeurs des entiers doivent se trouver dans le cadre de la plage correcte de cette partir de temps en particulier. Par exemple, le deuxième argument

(`Minutes`) ne devrait pas être supérieur à `59`. Pour ajouter une partie fuseau horaire à la valeur, utiliser l'autre signature de cette fonction (*voir signature suivante*).

Exemples

- **`altova:build-time`**`(23, 4, 57)` retourne `23:04:57`

`altova:build-time(Hours` *`as xs:integer`*`, **Minutes`** *`as xs:integer`*`, **Seconds`** *`as xs:integer`*`, **TimeZone`** *`as xs:string`*`)` as **`xs:time`** **XP3.1** **XQ3.1**

Les premiers, seconds et troisièmes arguments sont, respectivement, les valeurs d'heure (`0` to `23`), de minutes (`0` to `59`) et de secondes (`0` to `59`). Le quatrième argument est une chaîne qui fournit la partie fuseau horaire de la valeur. Les quatre arguments sont combinés pour construire une valeur de type `xs:time`. Les valeurs des entiers doivent se trouver dans le cadre de la plage correcte de cette partie de temps en particulier. Par exemple, le deuxième argument (`Minutes`) ne doit pas être supérieur à `59`.

Exemples

- **`altova:build-time`**`(23, 4, 57, '+1')` retourne `23:04:57+01:00`

build-duration [altova:]

`altova:build-duration(Years` *`as xs:integer`*`, **Months`** *`as xs:integer`*`)` as **`xs:yearMonthDuration`** **XP3.1** **XQ3.1**

Prend deux arguments pour construire une valeur de type `xs:yearMonthDuration`. Les premiers arguments fournissent la partie `Years` de la valeur de durée, alors que le deuxième argument fournie la partie `Months`. Si le deuxième argument (`Months`) est supérieur ou égale à `12`, alors l'entier est divisé par `12`; le quotient est ajouté au premier argument pour fournir la partie `Years` de la valeur de durée alors que le reste (de la division) fournit la partie `Months`. Pour construire une durée de type `xs:dayTimeDuration`., voir la signature suivante.

Exemples

- **`altova:build-duration`**`(2, 10)` retourne `P2Y10M`
- **`altova:build-duration`**`(14, 27)` retourne `P16Y3M`
- **`altova:build-duration`**`(2, 24)` retourne `P4Y`

`altova:build-duration(Days` *`as xs:integer`*`, **Hours`** *`as xs:integer`*`, **Minutes`** *`as xs:integer`*`, **Seconds`** *`as xs:integer`*`)` as **`xs:dayTimeDuration`** **XP3.1** **XQ3.1**

Prend quatre arguments et les combine pour construire une valeur de type `xs:dayTimeDuration`. Le premier argument fournit la partie `Days` de la valeur de durée, le deuxième, troisième et quatrième argument fournit respectivement les parties `Hours`, `Minutes` et `Seconds` de la valeur de durée. Chacun des trois arguments Time est converti en une valeur équivalente en termes de l'unité suivante plus élevée et le résultat est utilisé pour le calcul d'une valeur de durée totale. Par exemple, 72 secondes est converti en `1M+12S` (1 minute et 12 secondes), et cette valeur est utilisée pour le calcul de la valeur de durée totale. Pour construire une durée de type `xs:yearMonthDuration`., voir la signature précédente.

Exemples

- **`altova:build-duration`**`(2, 10, 3, 56)` retourne `P2DT10H3M56S`
- **`altova:build-duration`**`(1, 0, 100, 0)` retourne `P1DT1H40M`
- **`altova:build-duration`**`(1, 0, 0, 3600)` retourne `P1DT1H`

[Haut]

Construire le type de date, dateHeure ou heure à partir de l'entrée de chaîne

XP2 XQ1 XP3.1 XQ3.1

Ces fonctions prennent des chaînes en tant qu'arguments et construisent des types de données `xs:date`, `xs:dateTime`, ou `xs:time`. La chaîne est analysée pour les composants du type de données basé sur un argument de modèle soumis.

parse-date [altova:]

altova:parse-date(**Date** *as xs:string*, **DatePattern** *as xs:string*) as **xs:date**

XP2 XQ1 XP3.1 XQ3.1

Retourne la chaîne d'entrée **Date** en tant qu'une valeur **xs:date**. Le deuxième argument **DatePattern** spécifie le modèle (séquence des composants) de la chaîne d'entrée. **DatePattern** est décrit avec les spécificateurs de composants regroupés ci-dessous et avec les séparateurs de composant qui peuvent être n'importe quel caractère. Voir les exemples ci-dessous.

D	Jour
M	Mois
Y	Année

Le modèle dans **DatePattern** doit correspondre au modèle dans **Date**. Puisque la sortie est de type **xs:date**, la sortie aura toujours le format lexical **YYYY-MM-DD**.

Exemples

- **altova:parse-date**(xs:string("09-12-2014"), "[D]-[M]-[Y]") retourne 2014-12-09
- **altova:parse-date**(xs:string("09-12-2014"), "[M]-[D]-[Y]") retourne 2014-09-12
- **altova:parse-date**("06/03/2014", "[M]/[D]/[Y]") retourne 2014-06-03
- **altova:parse-date**("06 03 2014", "[M] [D] [Y]") retourne 2014-06-03
- **altova:parse-date**("6 3 2014", "[M] [D] [Y]") retourne 2014-06-03

parse-dateTime [altova:]

altova:parse-dateTime(**DateTime** *as xs:string*, **DateTimePattern** *as xs:string*) as **xs:dateTime** XP2 XQ1 XP3.1 XQ3.1

Retourne la chaîne d'entrée **DateTime** en tant que valeur **xs:dateTime**. Le deuxième argument **DateTimePattern** spécife le modèle (séquence des composants) de la chaîne d'entrée. **DateTimePattern** est décrit avec les spécificateurs de composants regroupés ci-dessous et avec les séparateurs de composant qui peuvent être n'importe quel caractère. Voir les exemples ci-dessous.

D	Date
M	Mois
Y	Année
H	Heure
m	Minutes
s	Secondes

Le modèle dans **DateTimePattern** doit correspondre au modèle dans **DateTime**. Puisque la sortie est de type **xs:dateTime**, la sortie aura toujours le format lexical **YYYY-MM-DDTHH:mm:ss**.

Exemples

- **altova:parse-dateTime**(xs:string("09-12-2014 13:56:24"), "[M]-[D]-[Y] [H]:[m]:[s]") retourne 2014-09-12T13:56:24
- **altova:parse-dateTime**("time=13:56:24; date=09-12-2014", "time=[H]:[m]:[s]; date=[D]-[M]-[Y]") retourne 2014-12-09T13:56:24

parse-time [altova:]

altova:parse-time(**Time** *as xs:string*, **TimePattern** *as xs:string*) as **xs:time**
XP2 **XQ1** **XP3.1** **XQ3.1**

Retourne la chaîne d'entrée **Time** en tant qu'une valeur **xs:time**.Le deuxième argument **TimePattern** spécifie le modèle (séquence des composants) de la chaîne d'entrée. **TimePattern** est décrit avec les spécificateurs de composants regroupés ci-dessous et avec les séparateurs de composant qui peuvent être n'importe quel caractère. Voir les exemples ci-dessous.

H	Heure
m	minutes
s	secondes

Le modèle dans **TimePattern** doit correspondre au modèle dans **Time**. Puisque la sortie est de type **xs:time**, la sortie aura toujours le format lexical **HH:mm:ss**.

Exemples

- **altova:parse-time**(xs:string("13:56:24"), "[H]:[m]:[s]") retourne 13:56:24
- **altova:parse-time**("13-56-24", "[H]-[m]") retourne 13:56:00
- **altova:parse-time**("time=13h56m24s", "time=[H]h[m]m[s]s") retourne 13:56:24
- **altova:parse-time**("time=24s56m13h", "time=[s]s[m]m[H]h") retourne 13:56:24

[Top]

Fonctions liées à l'âge XP3.1 XQ3.1

Ces fonctions retournent l'âge tel que calculé (i) entre une date d'argument d'entrée et la date actuelle, ou (ii) entre deux dates d'argument d'entrée. La fonction **altova:age** retourne l'âge en termes d'années, la fonction **altova:age-details** retourne l'âge en tant qu'une séquence de trois entiers indiquant les années, mois et jours de l'âge.

age [altova:]

altova:age(StartDate *as xs:date*) as **xs:integer** XP3.1 XQ3.1
Retourne un entier représentant l'âge *en années* d'un objet, en comptant depuis une date de départ soumise en tant que l'argument et se terminant avec la date actuelle (prise depuis l'horloge système). Si l'argument d'entrée est une date supérieure ou égale à une année dans le futur, la valeur de retour sera négative.

Exemples

Si la date actuelle est **2014-01-15** :

- **altova:age**(xs:date("2013-01-15")) retourne 1
- **altova:age**(xs:date("2013-01-16")) retourne 0
- **altova:age**(xs:date("2015-01-15")) retourne -1
- **altova:age**(xs:date("2015-01-14")) retourne 0

altova:age(StartDate *as xs:date*, **EndDate** *as xs:date*) as **xs:integer** XP3.1 XQ3.1
Retourne un entier représentant l'âge *en années* d'un objet, en comptant depuis une date de départ soumise en tant que l'argument jusqu'à une date de fin qui est de deuxième argument. La valeur de retour sera négative si le premier argument est tardif d'une année ou plus que le deuxième argument.

Exemples

Si la date actuelle est **2014-01-15**:

- **altova:age**(xs:date("2000-01-15"), xs:date("2010-01-15")) retourne 10
- **altova:age**(xs:date("2000-01-15"), current-date()) retourne 14 si la date actuelle est 2014-01-15
- **altova:age**(xs:date("2014-01-15"), xs:date("2010-01-15")) retourne -4

age-details [altova:]

altova:age-details(InputDate *as xs:date*) as (**xs:integer**)* XP3.1 XQ3.1
Retourne trois entiers qui sont respectivement les années, les mois et les jours entre la date soumise en tant que l'argument et la date actuelle (prise depuis l'horloge système). Le résultat de la somme de years+months+days donne le total de la différence de temps entre les deux dates (la date d'entrée et la date actuelle). La date d'entrée peut avoir une valeur précédant ou succédant à la date actuelle mais que la date d'entrée soit précédente ou succédant n'est pas indiqué par le signe des valeurs de retour ; les valeurs de retour sont toujours positives.

Exemples

Si la date actuelle est **2014-01-15**:

- **altova:age-details**(xs:date("2014-01-16")) retourne (0 0 1)
- **altova:age-details**(xs:date("2014-01-14")) retourne (0 0 1)
- **altova:age-details**(xs:date("2013-01-16")) retourne (1 0 1)
- **altova:age-details**(current-date()) retourne (0 0 0)

altova:age-details(Date-1 *as xs:date*, **Date-2** *as xs:date*) as (**xs:integer**)* XP3.1 XQ3.1
Retourne trois entiers qui sont respectivement les années, les mois et les jours entre les deux dates d'argument. Le résultat de la somme de years+months+days donne le total de la différence de temps entre les deux dates d'entrée ; peu importe que la date soit la

précédente ou la subséquente des deux dates, elle est soumise en tant que le premier argument. Les valeurs de retour n'indiquent pas si la date d'entrée se produit avant ou après la date actuelle. Les valeurs de retour sont toujours positives.

Exemples

- `altova:age-details(xs:date("2014-01-16"), xs:date("2014-01-15"))` retourne `(0 0 1)`
- `altova:age-details(xs:date("2014-01-15"), xs:date("2014-01-16"))` retourne `(0 0 1)`

[Top]

Fonctions XPath/XQuery : Géolocalisation

Les fonctions d'extension de géolocalisation XPath/XQuery suivantes sont prises en charge dans la version actuelle de RaptorXML+XBRL Server et peuvent être utilisées dans (i) des expressions XPath dans un contexte XSLT, ou (ii) des expressions XQuery dans un document XQuery.

Note concernant le nommage de fonctions et de l'applicabilité de la langue

Les fonctions d'extension Altova peuvent être utilisées dans les expressions XPath/XQuery. Elles fournissent des fonctions supplémentaires aux fonctions d'ores et déjà disponibles dans la librairie standard des fonctions XPath, XQuery et XSLT. Les fonctions d'extension Altova se trouvent dans **l'espace de nom des fonctions d'extension Altova,** `http://www.altova.com/xslt-extensions`, et sont indiquées dans cette section par le préfixe `altova:`, qui est présupposé être lié à cet espace de nom. Veuillez noter que, en ce qui concerne les versions futures de votre produit, la prise en charge d'une fonction peut être interrompue et le comportement de certaines fonctions peut changer. Veuillez consulter la documentation lors des publications à venir pour plus d'informations concernant la prise en charge des fonctions d'extension Altova de cette version.

Fonctions XPath (utilisées dans les expressions XPath dans XSLT) :	XP1 XP2 XP3.1
Fonctions XSLT (utilisées dans les expressions XPath dans XSLT) :	XSLT1 XSLT2 XSLT3
Fonctions XQuery (utilisées dans les expressions XQuery dans XQuery) :	XQ1 XQ3.1

parse-geolocation [altova:]

`altova:parse-geolocation(GeolocationInputString as xs:string)` as `xs:decimal+` XP3.1 XQ3.1

Parse l'argument `GeolocationInputString` fourni et retourne la latitude et longitude de géolocalisation (dans cet ordre) en séquence deux items décimaux `xs:decimal`. Les formats dans lesquels la chaîne d'entrée de géolocalisation peut être fournie sont recensés ci-dessous.

Note : La fonction `image-exif-data` et l'attribut `@Geolocation` de métadonnées Exif peuvent être utilisés pour fournir la chaîne d'entrée de géolocalisation (*voir exemple ci-dessous*).

Exemples

- `altova:parse-geolocation("33.33  -22.22")` retourne la séquence de deux `xs:decimals (33.33, 22.22)`
- `altova:parse-geolocation("48°51'29.6""N  24°17'40.2""")` retourne la séquence de deux `xs:decimals (48.8582222222222, 24.2945)`
- `altova:parse-geolocation('48°51''29.6"N  24°17''40.2"')` retourne la séquence de deux `xs:decimals (48.8582222222222, 24.2945)`
- `altova:parse-geolocation( image-exif-data(//MyImages/Image20141130.01)/@Geolocation )` retourne une séquence de deux `xs:decimals`

Formats de string d'entrée de géolocalisation :

Le string d'entrée de géolocalisation doit contenir la latitude et la longitude (dans cet ordre) séparées par un espace. Les strings peuvent tous présenter les formats suivants. Les combinaisons sont permises. La latitude peut donc être dans un format et la longitude dans un autre. Les valeurs de latitude varient de `+90` à `-90` (`N` à `S`). Les valeurs de longitude varient de `+180` à **-**`180` (`E` à `W`).

Note : L'utilisation de guillemets simples ou doubles pour la délimitation des arguments de string entraînera une non-concordance avec l'utilisation de guillemets simples ou doubles pour indiquer, respectivement les valeurs de minutes et de secondes. Dans ces cas, les guillemets utilisés pour indiquer les minutes et les secondes doivent être échappés en les doublant. Dans les exemples présentés dans cette section, les guillemets utilisés pour délimiter les strings d'entrée sont marqués en jaune (`"`) alors que les indicateurs d'unité échappés sont marqués en bleu (`""`).

- Degrés, minutes, secondes décimales, avec orientation suffixée (`N/S`, `W/E`)
  `D°M'S.SS"N/S  D°M'S.SS"W/E`
  Exemple : **`33°55'11.11"N  22°44'55.25"W`**

- Degrés, minutes, secondes décimales, avec signe préfixé (`+/-`) ; le signe plus pour (**`N/W`**) est optionnel
  `+/-D°M'S.SS"  +/-D°M'S.SS"`
  Exemple : **`33°55'11.11"  -22°44'55.25"`**

- Degrés, minutes décimales, avec orientation suffixée (`N/S`, `W/E`)
  `D°M.MM'N/S  D°M.MM'W/E`
  Exemple : **`33°55.55'N  22°44.44'W`**

- Degrés, minutes décimales, avec signe préfixé (`+/-`) ; le signe plus pour (**`N/W`**) est optionnel
  `+/-D°M.MM'  +/-D°M.MM'`
  Exemple : **`+33°55.55'  -22°44.44'`**

- Degrés décimaux, avec orientation suffixée (`N/S`, `W/E`)
  `D.DDN/S  D.DDW/E`
  Exemple : **`33.33N  22.22W`**

- Degrés décimaux, avec signe préfixé (`+/-`) ; le signe plus pour (**`N/W`**) est optionnel
  `+/-D.DD  +/-D.DD`
  Exemple : **`33.33  -22.22`**

Exemples de combinaisons de format :

```
33.33N  -22°44'55.25"
33.33  22°44'55.25"W
33.33  22.45
```

Attribut Altova Exif : Géolocalisation

La machine Altova XPath/XQuery génère l'attribut **`Geolocation`** personnalisable depuis les onglets standard de métadonnées Exif. **`Geolocation`** est une concaténation de quatre onglets Exif : `GPSLatitude`, `GPSLatitudeRef`, `GPSLongitude`, `GPSLongitudeRef`, avec des unités ajoutées (voir table ci-dessous).

 © 2018 Altova GmbH

GPSLatitude	GPSLatitudeRef	GPSLongitude	GPSLongitudeRef	Geolocation
33 51 21.91	S	151 13 11.73	E	33°51'21.91"S 151° 13'11.73"E

geolocation-distance-km [altova:]

altova:geolocation-distance-km(GeolocationInputString-1 *as xs:string*, **GeolocationInputString-2** *as xs:string*) as **xs:decimal** **XP3.1** **XQ3.1**

Calcule la distance entre deux géolocalisations en kilomètres. Les formats dans lesquels une chaîne d'entrée de géolocalisation peut être fournie sont recensés ci-dessous. Les valeurs de latitude vont de `+90` à `-90` (`N` à `S`). Les valeurs de longitude vont de `+180` à `-180` (`E` à `O`).

Note : La fonction `image-exif-data` et l'attribut `@Geolocation` des métadonnées d'Exif peuvent être utilisés pour fournir les chaînes d'entrée de géolocalisation.

Exemples

- **altova:geolocation-distance-km**("33.33 -22.22", "48°51'29.6""N 24° 17'40.2""") retourne `xs:decimal` `4183.08132372392`

Formats de string d'entrée de géolocalisation :

Le string d'entrée de géolocalisation doit contenir la latitude et la longitude (dans cet ordre) séparées par un espace. Les strings peuvent tous présenter les formats suivants. Les combinaisons sont permises. La latitude peut donc être dans un format et la longitude dans un autre. Les valeurs de latitude varient de `+90` à `-90` (`N` à `S`). Les valeurs de longitude varient de `+180` à `-180` (`E` à `W`).

Note : L'utilisation de guillemets simples ou doubles pour la délimitation des arguments de string entraînera une non-concordance avec l'utilisation de guillemets simples ou doubles pour indiquer, respectivement les valeurs de minutes et de secondes. Dans ces cas, les guillemets utilisés pour indiquer les minutes et les secondes doivent être échappés en les doublant. Dans les exemples présentés dans cette section, les guillemets utilisés pour délimiter les strings d'entrée sont marqués en jaune (`"`) alors que les indicateurs d'unité échappés sont marqués en bleu (`""`).

- Degrés, minutes, secondes décimales, avec orientation suffixée (`N/S`, `W/E`)
  D°M'S.SS"N/S D°M'S.SS"W/E
  Exemple : **33°55'11.11"N 22°44'55.25"W**

- Degrés, minutes, secondes décimales, avec signe préfixé (`+/-`) ; le signe plus pour (**N/W**) est optionnel
  +/-D°M'S.SS" +/-D°M'S.SS"
  Exemple : **33°55'11.11" -22°44'55.25"**

- Degrés, minutes décimales, avec orientation suffixée (`N/S`, `W/E`)

D°M.MM'N/S D°M.MM'W/E
Exemple : **33°55.55'N 22°44.44'W**

- Degrés, minutes décimales, avec signe préfixé (+/-) ; le signe plus pour (**N/W**) est optionnel
  +/-D°M.MM' +/-D°M.MM'
  Exemple : **+33°55.55' -22°44.44'**

- Degrés décimaux, avec orientation suffixée (N/S, W/E)
  D.DDN/S D.DDW/E
  Exemple : **33.33N 22.22W**

- Degrés décimaux, avec signe préfixé (+/-) ; le signe plus pour (**N/W**) est optionnel
  +/-D.DD +/-D.DD
  Exemple : **33.33 -22.22**

Exemples de combinaisons de format :
33.33N -22°44'55.25"
33.33 22°44'55.25"W
33.33 22.45

Attribut Altova Exif : Géolocalisation

La machine Altova XPath/XQuery génère l'attribut **Geolocation** personnalisable depuis les onglets standard de métadonnées Exif. **Geolocation** est une concaténation de quatre onglets Exif : GPSLatitude, GPSLatitudeRef, GPSLongitude, GPSLongitudeRef, avec des unités ajoutées (voir table ci-dessous).

GPSLatitude	GPSLatitudeRef	GPSLongitude	GPSLongitudeRef	Geolocation
33 51 21.91	S	151 13 11.73	E	33°51'21.91"S 151° 13'11.73"E

geolocation-distance-mi [altova:]

altova:geolocation-distance-mi(GeolocationInputString-1 *as xs:string*, **GeolocationInputString-2** *as xs:string*) as **xs:decimal** **XP3.1** **XQ3.1**

Calcule la distance entre deux géolocalisations en miles. Les formats dans lesquels une chaîne d'entrée de géolocalisation peut être fournie sont recensés ci-dessous. Les valeurs de latitude vont de +90 à -90 (N à S). Les valeurs de longitude vont de +180 à -180 (E à O).

Note : La fonction image-exif-data et l'attribut @Geolocation des métadonnées d'Exif peuvent être utilisés pour fournir les chaînes d'entrée de géolocalisation.

Exemples

- **altova:geolocation-distance-mi**("33.33 -22.22", "48°51'29.6""N 24° 17'40.2""") retourne xs:decimal 2599.40652340653

Formats de string d'entrée de géolocalisation :

Le string d'entrée de géolocalisation doit contenir la latitude et la longitude (dans cet ordre) séparées par un espace. Les strings peuvent tous présenter les formats suivants. Les combinaisons sont permises. La latitude peut donc être dans un format et la longitude dans un autre. Les valeurs de latitude varient de `+90` à `-90` (`N` à `S`). Les valeurs de longitude varient de `+180` à `-180` (`E` à `W`).

Note : L'utilisation de guillemets simples ou doubles pour la délimitation des arguments de string entraînera une non-concordance avec l'utilisation de guillemets simples ou doubles pour indiquer, respectivement les valeurs de minutes et de secondes. Dans ces cas, les guillemets utilisés pour indiquer les minutes et les secondes doivent être échappés en les doublant. Dans les exemples présentés dans cette section, les guillemets utilisés pour délimiter les strings d'entrée sont marqués en jaune (`"`) alors que les indicateurs d'unité échappés sont marqués en bleu (`""`).

- Degrés, minutes, secondes décimales, avec orientation suffixée (`N/S`, `W/E`)
  `D°M'S.SS"N/S  D°M'S.SS"W/E`
  Exemple : **`33°55'11.11"N  22°44'55.25"W`**

- Degrés, minutes, secondes décimales, avec signe préfixé (`+/-`) ; le signe plus pour (**`N/W`**) est optionnel
  `+/-D°M'S.SS"  +/-D°M'S.SS"`
  Exemple : **`33°55'11.11"  -22°44'55.25"`**

- Degrés, minutes décimales, avec orientation suffixée (`N/S`, `W/E`)
  `D°M.MM'N/S  D°M.MM'W/E`
  Exemple : **`33°55.55'N  22°44.44'W`**

- Degrés, minutes décimales, avec signe préfixé (`+/-`) ; le signe plus pour (**`N/W`**) est optionnel
  `+/-D°M.MM'  +/-D°M.MM'`
  Exemple : **`+33°55.55'  -22°44.44'`**

- Degrés décimaux, avec orientation suffixée (`N/S`, `W/E`)
  `D.DDN/S  D.DDW/E`
  Exemple : **`33.33N  22.22W`**

- Degrés décimaux, avec signe préfixé (`+/-`) ; le signe plus pour (**`N/W`**) est optionnel
  `+/-D.DD  +/-D.DD`
  Exemple : **`33.33  -22.22`**

Exemples de combinaisons de format :

```
33.33N  -22°44'55.25"
33.33  22°44'55.25"W
33.33  22.45
```

Attribut Altova Exif : Géolocalisation

La machine Altova XPath/XQuery génère l'attribut **`Geolocation`** personnalisable depuis les onglets standard de métadonnées Exif. **`Geolocation`** est une concaténation de

quatre onglets Exif : `GPSLatitude`, `GPSLatitudeRef`, `GPSLongitude`, `GPSLongitudeRef`, avec des unités ajoutées (voir table ci-dessous).

GPSLatitude	GPSLatitudeRef	GPSLongitude	GPSLongitudeRef	Geolocation
33 51 21.91	S	151 13 11.73	E	33°51'21.91"S 151° 13'11.73"E

geolocation-within-polygon [altova:]

altova:geolocation-within-polygon(Geolocation *as xs:string*, **((PolygonPoint** *as xs:string*)+)) as **xs:boolean** XP3.1 XQ3.1

Détermine si **Geolocation** (le premier argument) se trouve dans l'espace polygonal décrit par les arguments **PolygonPoint**. Si les arguments `PolygonPoint` ne forment pas une figure fermée (formée lorsque le premier point et le dernier point sont identiques), alors le premier point est implicitement ajouté en tant que le dernier point afin de pouvoir clore la figure. Tous les arguments (`Geolocation` et `PolygonPoint+`) sont donnés par chaînes d'entrées de géolocalisation (*formats recensés ci-dessous*). Si l'argument `Geolocation` se trouve dans l'espace polygonal, la fonction retourne `true()`; sinon, elle retourne `false()`. Les valeurs de latitude vont de `+90` à `-90` (`N` à `S`). Les valeurs de longitude vont de `+180` à `-180` (`E` à `O`).

Note : La fonction `image-exif-data` et l'attribut `@Geolocation` de métadonnées d'Exif peut être utilisée pour fournir les chaînes d'entrée de géolocalisation.

Exemples

- **altova:geolocation-within-polygon("33 -22", ("58 -32", "-78 -55", "48 24", "58 -32"))** retourne `true()`
- **altova:geolocation-within-polygon("33 -22", ("58 -32", "-78 -55", "48 24"))** retourne `true()`
- **altova:geolocation-within-polygon("33 -22", ("58 -32", "-78 -55", "48°51'29.6""N 24°17'40.2""")))** retourne `true()`

Formats de string d'entrée de géolocalisation :

Le string d'entrée de géolocalisation doit contenir la latitude et la longitude (dans cet ordre) séparées par un espace. Les strings peuvent tous présenter les formats suivants. Les combinaisons sont permises. La latitude peut donc être dans un format et la longitude dans un autre. Les valeurs de latitude varient de `+90` à `-90` (`N` à `S`). Les valeurs de longitude varient de `+180` à `-180` (`E` à `W`).

Note : L'utilisation de guillemets simples ou doubles pour la délimitation des arguments de string entraînera une non-concordance avec l'utilisation de guillemets simples ou doubles pour indiquer, respectivement les valeurs de minutes et de secondes. Dans ces cas, les guillemets utilisés pour indiquer les minutes et les secondes doivent être échappés en les doublant. Dans les exemples présentés dans cette section, les guillemets utilisés pour délimiter les strings d'entrée sont marqués en jaune (`"`) alors que les indicateurs d'unité échappés sont marqués en bleu (`""`).

 © 2018 Altova GmbH

- Degrés, minutes, secondes décimales, avec orientation suffixée (N/S, W/E)
  D°M'S.SS"N/S D°M'S.SS"W/E
  Exemple : **33°55'11.11"N 22°44'55.25"W**

- Degrés, minutes, secondes décimales, avec signe préfixé (+/-) ; le signe plus pour (**N/W**) est optionnel
  +/-D°M'S.SS" +/-D°M'S.SS"
  Exemple : **33°55'11.11" -22°44'55.25"**

- Degrés, minutes décimales, avec orientation suffixée (N/S, W/E)
  D°M.MM'N/S D°M.MM'W/E
  Exemple : **33°55.55'N 22°44.44'W**

- Degrés, minutes décimales, avec signe préfixé (+/-) ; le signe plus pour (**N/W**) est optionnel
  +/-D°M.MM' +/-D°M.MM'
  Exemple : **+33°55.55' -22°44.44'**

- Degrés décimaux, avec orientation suffixée (N/S, W/E)
  D.DDN/S D.DDW/E
  Exemple : **33.33N 22.22W**

- Degrés décimaux, avec signe préfixé (+/-) ; le signe plus pour (**N/W**) est optionnel
  +/-D.DD +/-D.DD
  Exemple : **33.33 -22.22**

Exemples de combinaisons de format :
33.33N -22°44'55.25"
33.33 22°44'55.25"W
33.33 22.45

Attribut Altova Exif : Géolocalisation

La machine Altova XPath/XQuery génère l'attribut **Geolocation** personnalisable depuis les onglets standard de métadonnées Exif. **Geolocation** est une concaténation de quatre onglets Exif : GPSLatitude, GPSLatitudeRef, GPSLongitude, GPSLongitudeRef, avec des unités ajoutées (voir table ci-dessous).

GPSLatitude	GPSLatitudeRef	GPSLongitude	GPSLongitudeRef	Geolocation
33 51 21.91	S	151 13 11.73	E	33°51'21.91"S 151° 13'11.73"E

geolocation-within-rectangle [altova:]

altova:geolocation-within-rectangle(Geolocation *as xs:string*, **RectCorner-1** *as xs:string*, **RectCorner-2** *as xs:string*) as **xs:boolean** XP3.1 XQ3.1

Détermine si **Geolocation** (le premier argument) se trouve dans le rectangle défini par le second et le troisième argument, **RectCorner-1** et **RectCorner-2**, qui spécifient les coins opposés du rectangle. Tous les arguments (Geolocation, **RectCorner-1** et **RectCorner-2**) sont indiqués par des chaînes d'entrées de géolocalisation (*formats recensés ci-dessous*). Si l'argument Geolocation se trouve dans le rectangle, la fonction retourne true(); sinon, elle retourne false(). Les valeurs de latitude vont de +90 à -90 (N à S). Les valeurs de longitude vont de +180 à -180 (E à O).

Note : La fonction image-exif-data et l'attribut @Geolocation de métadonnées Exif peuvent être utilisés pour fournir les chaînes d'entrée de géolocalisation.

Exemples

- **altova:geolocation-within-rectangle("33 -22", "58 -32", "-48 24")** retourne true()
- **altova:geolocation-within-rectangle("33 -22", "58 -32", "48 24")** retourne false()
- **altova:geolocation-within-rectangle("33 -22", "58 -32", "48°51'29.6""S 24°17'40.2""")** retourne true()

Formats de string d'entrée de géolocalisation :

Le string d'entrée de géolocalisation doit contenir la latitude et la longitude (dans cet ordre) séparées par un espace. Les strings peuvent tous présenter les formats suivants. Les combinaisons sont permises. La latitude peut donc être dans un format et la longitude dans un autre. Les valeurs de latitude varient de +90 à -90 (N à S). Les valeurs de longitude varient de +180 à -180 (E à W).

Note : L'utilisation de guillemets simples ou doubles pour la délimitation des arguments de string entraînera une non-concordance avec l'utilisation de guillemets simples ou doubles pour indiquer, respectivement les valeurs de minutes et de secondes. Dans ces cas, les guillemets utilisés pour indiquer les minutes et les secondes doivent être échappés en les doublant. Dans les exemples présentés dans cette section, les guillemets utilisés pour délimiter les strings d'entrée sont marqués en jaune (") alors que les indicateurs d'unité échappés sont marqués en bleu ("").

- Degrés, minutes, secondes décimales, avec orientation suffixée (N/S, W/E)
  D°M'S.SS"N/S D°M'S.SS"W/E
  Exemple : **33°55'11.11"N 22°44'55.25"W**

- Degrés, minutes, secondes décimales, avec signe préfixé (+/-) ; le signe plus pour (**N/W**) est optionnel
  +/-D°M'S.SS" +/-D°M'S.SS"
  Exemple : **33°55'11.11" -22°44'55.25"**

- Degrés, minutes décimales, avec orientation suffixée (N/S, W/E)
  D°M.MM'N/S D°M.MM'W/E
  Exemple : **33°55.55'N 22°44.44'W**

- Degrés, minutes décimales, avec signe préfixé (+/-) ; le signe plus pour (**N/W**) est optionnel
  +/-D°M.MM' +/-D°M.MM'

 © 2018 Altova GmbH

Exemple : `+33°55.55'  -22°44.44'`

- Degrés décimaux, avec orientation suffixée (`N/S`, `W/E`)
  `D.DDN/S  D.DDW/E`
  Exemple : `33.33N  22.22W`

- Degrés décimaux, avec signe préfixé (`+/-`) ; le signe plus pour (`N/W`) est optionnel
  `+/-D.DD  +/-D.DD`
  Exemple : `33.33  -22.22`

Exemples de combinaisons de format :

```
33.33N  -22°44'55.25"
33.33  22°44'55.25"W
33.33  22.45
```

Attribut Altova Exif : Géolocalisation

La machine Altova XPath/XQuery génère l'attribut `Geolocation` personnalisable depuis les onglets standard de métadonnées Exif. `Geolocation` est une concaténation de quatre onglets Exif : `GPSLatitude`, `GPSLatitudeRef`, `GPSLongitude`, `GPSLongitudeRef`, avec des unités ajoutées (voir table ci-dessous).

GPSLatitude	GPSLatitudeRef	GPSLongitude	GPSLongitudeRef	Geolocation
33 51 21.91	S	151 13 11.73	E	33°51'21.91"S 151° 13'11.73"E

[Top]

Fonctions XPath/XQuery : Relatives aux images

Les fonctions d'extension XPath/XQuery relatives à l'image suivantes sont prises en charge dans la version actuelle de RaptorXML+XBRL Server et peuvent être utilisées dans (i) des expressions XPath dans un contexte XSLT, ou dans (ii) des expressions XQuery dans un document XQuery.

Note concernant le nommage de fonctions et de l'applicabilité de la langue

Les fonctions d'extension Altova peuvent être utilisées dans les expressions XPath/XQuery. Elles fournissent des fonctions supplémentaires aux fonctions d'ores et déjà disponibles dans la librairie standard des fonctions XPath, XQuery et XSLT. Les fonctions d'extension Altova se trouvent dans **l'espace de nom des fonctions d'extension Altova,** `http://www.altova.com/xslt-extensions`, et sont indiquées dans cette section par le préfixe `altova:`, qui est présupposé être lié à cet espace de nom. Veuillez noter que, en ce qui concerne les versions futures de votre produit, la prise en charge d'une fonction peut être interrompue et le comportement de certaines fonctions peut changer. Veuillez consulter la documentation lors des publications à venir pour plus d'informations concernant la prise en charge des fonctions d'extension Altova de cette version.

Fonctions XPath (utilisées dans les expressions XPath dans XSLT) :	XP1 XP2 XP3.1
Fonctions XSLT (utilisées dans les expressions XPath dans XSLT) :	XSLT1 XSLT2 XSLT3
Fonctions XQuery (utilisées dans les expressions XQuery dans XQuery) :	XQ1 XQ3.1

suggested-image-file-extension [altova:]

`altova:suggested-image-file-extension(Base64String as string) as string?` XP3.1 XQ3.1

Prend le code Base64 d'un fichier d'image en tant que son argument et retourne l'extension de fichier de l'image comme enregistré dans le codage Base64 de l'image. La valeur retournée est une suggestion basée sur l'information du type d'image disponible dans le codage. Si cette information n'est pas disponible, une chaîne vide est retournée. Cette fonction est utile si vous souhaitez enregistrer une image Base64 en tant que fichier et que vous souhaitez extraire dynamiquement une extension de fichier appropriée.

Exemples

- `altova:suggested-image-file-extension(/MyImages/MobilePhone/Image20141130.01)` retourne `'jpg'`
- `altova:suggested-image-file-extension($XML1/Staff/Person/@photo)` retourne `''`

Dans les exemples ci-dessus, les nœuds fournis en tant qu'arguments de la fonction sont assumés contenir une image codée Base64. Le premier exemple extrait `jpg` en tant que type et extension de fichier. Dans le second exemple, le codage Base64 soumis ne fournit pas une information de fichier d'extension utile.

image-exif-data [altova:]

`altova:image-exif-data(Base64BinaryString as string) as element?` XP3.1 XQ3.1

Prend une image codée Base64 en tant que son argument et retourne un élément appelé **Exif** qui contient les métadonnées Exif de l'image. Celles-ci sont créées en tant que paires attribute-value pairs de l'élément `Exif`. Les noms d'attribut sont les onglets de données Exif trouvés dans le codage Base64. La liste des onglets des spécifications Exif est indiquée ci-dessous. Si un onglet spécifique à un distributeur est présent dans les données Exif, cet onglet et sa valeur seront aussi retournés en tant que paire attribute-value. Outre les onglets de métadonnées Exif standard (*voir la liste ci-dessous*), des paires attribute-value spécifiques à Altova sont également générées. Ces attributs Exif Altova sont recensés ci-dessous.

Exemples

- Pour accéder à n'importe quel attribut, utiliser la fonction comme suit :
  ```
  image-exif-data(//MyImages/Image20141130.01)/@GPSLatitude
  image-exif-data(//MyImages/Image20141130.01)/@Geolocation
  ```
- Pour accéder à tous les attributs, utiliser la fonction comme suit:
  ```
  image-exif-data(//MyImages/Image20141130.01)/@*
  ```
- Pour accéder au nom de tous les attributs, utiliser l'expression suivante :
  ```
  for $i in image-exif-data(//MyImages/Image20141130.01)/@* return
  name($i)
  ```
  Cela est utile pour trouver les noms des attributs retournés par la fonction.

Attribut Altova Exif : Géolocalisation

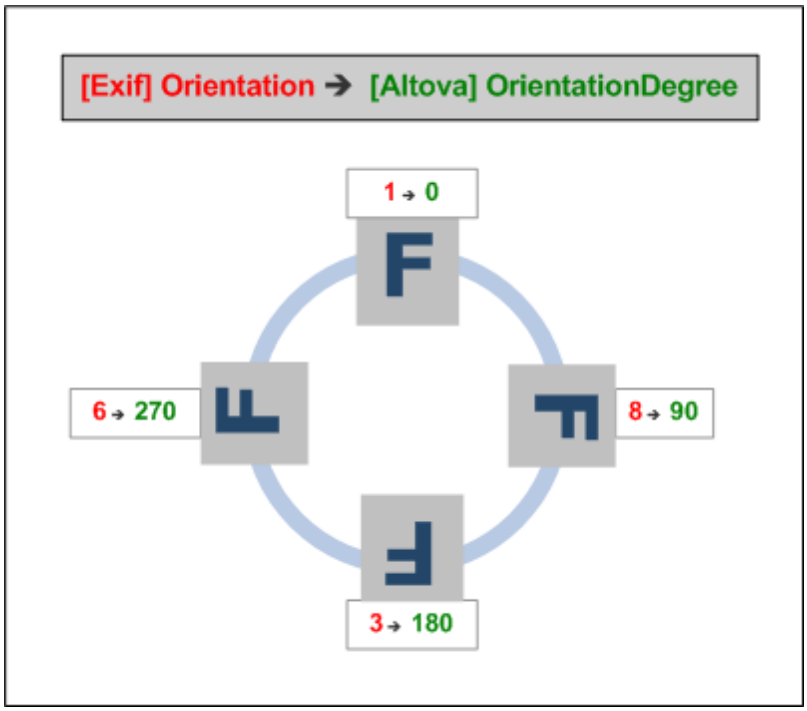
La machine Altova XPath/XQuery génère l'attribut **Geolocation** personnalisable depuis les onglets standard de métadonnées Exif. **Geolocation** est une concaténation de quatre onglets Exif : `GPSLatitude`, `GPSLatitudeRef`, `GPSLongitude`, `GPSLongitudeRef`, avec des unités ajoutées (voir table ci-dessous).

GPSLatitude	GPSLatitudeRef	GPSLongitude	GPSLongitudeRef	Geolocation
33 51 21.91	S	151 13 11.73	E	33°51'21.91"S 151° 13'11.73"E

Altova Exif Attribute: OrientationDegree

La machine Altova XPath/XQuery génère l'attribut personnalisé **OrientationDegree** à partir de l'onglet de métadonnées Exif **Orientation**.

OrientationDegree traduit l'onglet standard Exif `Orientation` à partir d'une valeur d'entier (`1`, `8`, `3`, ou `6`) aux valeurs de degrés respectives de chacun (`0`, `90`, `180`, `270`), tel que montré dans la figure ci-dessous. Veuillez noter qu'il n'y a pas de traductions de la valeur `Orientation` de `2`, `4`, `5`, `7`. (Ces orientations sont obtenus en basculant l'image `1` à travers son centre axial vertical pour obtenir l'image avec une valeur de `2`, puis en pivotant cette image par sauts de 90° dans le sens des aiguilles d'une montre pour obtenir les valeurs de `7`, `4`, et `5`, respectivement).



Listing of standard Exif meta tags

- 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

 © 2018 Altova GmbH

- Model
- Software
- Artist
- Copyright

- ExifVersion
- FlashpixVersion
- ColorSpace
- ComponentsConfiguration
- CompressedBitsPerPixel
- PixelXDimension
- PixelYDimension
- MakerNote
- UserComment
- RelatedSoundFile
- DateTimeOriginal
- DateTimeDigitized
- SubSecTime
- SubSecTimeOriginal
- SubSecTimeDigitized
- ExposureTime
- FNumber
- ExposureProgram
- SpectralSensitivity
- ISOSpeedRatings
- OECF
- ShutterSpeedValue
- ApertureValue
- BrightnessValue
- ExposureBiasValue
- MaxApertureValue
- SubjectDistance
- MeteringMode
- LightSource
- Flash
- FocalLength
- SubjectArea
- FlashEnergy
- SpatialFrequencyResponse
- FocalPlaneXResolution
- FocalPlaneYResolution
- FocalPlaneResolutionUnit
- SubjectLocation
- ExposureIndex
- SensingMethod
- FileSource
- SceneType
- CFAPattern
- CustomRendered
- ExposureMode
- WhiteBalance
- DigitalZoomRatio
- FocalLengthIn35mmFilm
- SceneCaptureType

- GainControl
- Contrast
- Saturation
- Sharpness
- DeviceSettingDescription
- SubjectDistanceRange
- ImageUniqueID

- GPSVersionID
- GPSLatitudeRef
- GPSLatitude
- GPSLongitudeRef
- GPSLongitude
- GPSAltitudeRef
- GPSAltitude
- GPSTimeStamp
- GPSSatellites
- GPSStatus
- GPSMeasureMode
- GPSDOP
- GPSSpeedRef
- GPSSpeed
- GPSTrackRef
- GPSTrack
- GPSImgDirectionRef
- GPSImgDirection
- GPSMapDatum
- GPSDestLatitudeRef
- GPSDestLatitude
- GPSDestLongitudeRef
- GPSDestLongitude
- GPSDestBearingRef
- GPSDestBearing
- GPSDestDistanceRef
- GPSDestDistance
- GPSProcessingMethod
- GPSAreaInformation
- GPSDateStamp
- GPSDifferential

[Top]

Fonctions XPath/XQuery : Numérique

Les fonctions d'extension numériques d'Altova peuvent être utilisées dans des expressions XPath et XQuery et proposent des fonctions supplémentaires pour le traitement des données. Les fonctions dans cette section peuvent être utilisée avec les moteurs **XPath 3.0** et **XQuery 3.0** d'Altova. Ils sont disponibles dans des contextes XPath/XQuery.

Note concernant le nommage de fonctions et de l'applicabilité de la langue

Les fonctions d'extension Altova peuvent être utilisées dans les expressions XPath/XQuery. Elles fournissent des fonctions supplémentaires aux fonctions d'ores et déjà disponibles dans la librairie standard des fonctions XPath, XQuery et XSLT. Les fonctions d'extension Altova se trouvent dans **l'espace de nom des fonctions d'extension Altova,** `http://www.altova.com/xslt-extensions`, et sont indiquées dans cette section par le préfixe `altova:`, qui est présupposé être lié à cet espace de nom. Veuillez noter que, en ce qui concerne les versions futures de votre produit, la prise en charge d'une fonction peut être interrompue et le comportement de certaines fonctions peut changer. Veuillez consulter la documentation lors des publications à venir pour plus d'informations concernant la prise en charge des fonctions d'extension Altova de cette version.

Fonctions XPath (utilisées dans les expressions XPath dans XSLT) :	XP1 XP2 XP3.1
Fonctions XSLT (utilisées dans les expressions XPath dans XSLT) :	XSLT1 XSLT2 XSLT3
Fonctions XQuery (utilisées dans les expressions XQuery dans XQuery) :	XQ1 XQ3.1

Fonctions de numérotation automatique

generate-auto-number [altova:]

altova:generate-auto-number(**ID** *as xs:string*, **StartsWith** *as xs:double*, **Increment** *as xs:double*, **ResetOnChange** *as xs:string*) as `xs:integer` XP1 XP2 XQ1 XP3.1 XQ3.1

Génère un numéro à chaque fois que la fonction est appelée. Le premier numéro, qui est généré la première fois que la fonction est appelée, est spécifié par l'argument `StartsWith`. Chaque appel subséquent vers la fonction génère un nouveau numéro, ce numéro est augmenté au-dessus du numéro précédemment généré par la valeur spécifiée dans l'argument `Increment`. En effet, la fonction `altova:generate-auto-number` crée un compteur comportant un nom spécifié par l'argument `ID`, et dont le compteur est augmenté à chaque fois que la fonction est appelée. Si la valeur de l'argument `ResetOnChange` change de celle de l'appel de fonction précédent, la valeur du numéro à générer est réinitialisée à la valeur `StartsWith`. La numérotation automatique peut être réinitialisée en utilisant la fonction `altova:reset-auto-number`.

Exemples

- **altova:generate-auto-number**(`"ChapterNumber"`, `1`, `1`, `"SomeString"`) retournera un nombre à chaque fois que la fonction est appelée, en commençant avec `1`, et en augmentant de `1` avec chaque appel de la fonction. Tant que le quatrième argument demeure `"SomeString"` dans chaque appel subséquent, l'augmentation se poursuivra. Lorsque la valeur du quatrième argument change, le

compteur (appelé `ChapterNumber`) sera réinitialisé à 1. La valeur de `ChapterNumber` peut aussi être réinitialisée par un appel de la fonction `altova:reset-auto-number` comme ceci : `altova:reset-auto-number("ChapterNumber")`.

reset-auto-number [altova:]

altova:reset-auto-number(ID *as xs:string*) XP1 XP2 XQ1 XP3.1 XQ3.1

Cette fonction réinitialise le numéro du compteur de numérotation automatique nommé dans l'argument `ID`. Le numéro est réinitialisé au numéro spécifié par l'argument `StartsWith` de la fonction `altova:generate-auto-number` qui a créé le compteur nommé dans l'argument `ID`.

Exemples

- **altova:reset-auto-number**("ChapterNumber") réinitialise le numéro du compteur de numérotation automatique nommé `ChapterNumber` qui a été créé par la fonction `altova:generate-auto-number`. Le numéro est réinitialisé à la valeur de l'argument `StartsWith` de la fonction `altova:generate-auto-number` qui a créé `ChapterNumber`.

[Top]

Fonctions numériques

hex-string-to-integer [altova:]

altova:hex-string-to-integer(HexString *as xs:string*) as **xs:integer** XP3.1 XQ3.1

Prend un argument de chaîne qui est l'équivalent Base-16 d'un entier dans le système décimal (Base-10), et retourne l'entier décimal.

Exemples

- **altova:hex-string-to-integer**('1') retourne 1
- **altova:hex-string-to-integer**('9') retourne 9
- **altova:hex-string-to-integer**('A') retourne 10
- **altova:hex-string-to-integer**('B') retourne 11
- **altova:hex-string-to-integer**('F') retourne 15
- **altova:hex-string-to-integer**('G') retourne une erreur
- **altova:hex-string-to-integer**('10') retourne 16
- **altova:hex-string-to-integer**('01') retourne 1
- **altova:hex-string-to-integer**('20') retourne 32
- **altova:hex-string-to-integer**('21') retourne 33
- **altova:hex-string-to-integer**('5A') retourne 90
- **altova:hex-string-to-integer**('USA') retourne une erreur

integer-to-hex-string [altova:]

altova:integer-to-hex-string(Integer *as xs:integer*) as **xs:string** XP3.1 XQ3.1

Prend un argument d'entier et retourne son équivalent de Base-16 en tant que chaîne.

Exemples

- **altova:integer-to-hex-string**(1) retourne '1'

 © 2018 Altova GmbH

- `altova:integer-to-hex-string(9)` retourne `'9'`
- `altova:integer-to-hex-string(10)` retourne `'A'`
- `altova:integer-to-hex-string(11)` retourne `'B'`
- `altova:integer-to-hex-string(15)` retourne `'F'`
- `altova:integer-to-hex-string(16)` retourne `'10'`
- `altova:integer-to-hex-string(32)` retourne `'20'`
- `altova:integer-to-hex-string(33)` retourne `'21'`
- `altova:integer-to-hex-string(90)` retourne `'5A'`

[Top]

Fonctions de formatage de numéro

generate-auto-number [altova:]

altova:generate-auto-number(ID *as xs:string*, **StartsWith** *as xs:double*, **Increment** *as xs:double*, **ResetOnChange** *as xs:string*) as `xs:integer` **XP1** **XP2** **XQ1** **XP3.1** **XQ3.1**

Génère un numéro à chaque fois que la fonction est appelée. Le premier numéro, qui est généré la première fois que la fonction est appelée, est spécifié par l'argument `StartsWith`. Chaque appel subséquent vers la fonction génère un nouveau numéro, ce numéro est augmenté au-dessus du numéro précédemment généré par la valeur spécifiée dans l'argument `Increment`. En effet, la fonction `altova:generate-auto-number` crée un compteur comportant un nom spécifié par l'argument `ID`, et dont le compteur est augmenté à chaque fois que la fonction est appelée. Si la valeur de l'argument `ResetOnChange` change de celle de l'appel de fonction précédent, la valeur du numéro à générer est réinitialisée à la valeur `StartsWith`. La numérotation automatique peut être réinitialisée en utilisant la fonction `altova:reset-auto-number`.

Exemples

- `altova:generate-auto-number("ChapterNumber", 1, 1, "SomeString")` retournera un nombre à chaque fois que la fonction est appelée, en commençant avec `1`, et en augmentant de `1` avec chaque appel de la fonction. Tant que le quatrième argument demeure `"SomeString"` dans chaque appel subséquent, l'augmentation se poursuivra. Lorsque la valeur du quatrième argument change, le compteur (appelé `ChapterNumber`) sera réinitialisé à 1. La valeur de `ChapterNumber` peut aussi être réinitialisée par un appel de la fonction `altova:reset-auto-number` comme ceci : `altova:reset-auto-number("ChapterNumber")`.

[Top]

Fonctions XPath/XQuery : Séquence

Les fonctions d'extension de la séquence d'Altova peuvent être utilisées dans les expressions XPath et XQuery et proposent des fonctions supplémentaires pour le traitement des données. Les fonctions dans cette section peuvent être utilisées avec les moteurs **XPath 3.0** et **XQuery 3.0** d'Altova. Ils sont disponibles dans des contextes XPath/XQuery.

Note concernant le nommage de fonctions et de l'applicabilité de la langue

Les fonctions d'extension Altova peuvent être utilisées dans les expressions XPath/XQuery. Elles fournissent des fonctions supplémentaires aux fonctions d'ores et déjà disponibles dans la librairie standard des fonctions XPath, XQuery et XSLT. Les fonctions d'extension Altova se trouvent dans **l'espace de nom des fonctions d'extension Altova,** `http://www.altova.com/xslt-extensions`, et sont indiquées dans cette section par le préfixe `altova:`, qui est présupposé être lié à cet espace de nom. Veuillez noter que, en ce qui concerne les versions futures de votre produit, la prise en charge d'une fonction peut être interrompue et le comportement de certaines fonctions peut changer. Veuillez consulter la documentation lors des publications à venir pour plus d'informations concernant la prise en charge des fonctions d'extension Altova de cette version.

Fonctions XPath (utilisées dans les expressions XPath dans XSLT) :	XP1 XP2 XP3.1
Fonctions XSLT (utilisées dans les expressions XPath dans XSLT) :	XSLT1 XSLT2 XSLT3
Fonctions XQuery (utilisées dans les expressions XQuery dans XQuery) :	XQ1 XQ3.1

attributes [altova:]

altova:attributes(AttributeName *as xs:string*) as **attribute()*** XP3.1 XQ3.1

Retourne tous les attributs possédant un nom local qui est le même que le nom fourni dans l'argument d'entrée, `AttributeName`. La recherche est sensible à la casse et est conduite le long de l'axe `attribute::` . Cela signifie que le nœud contextuel doit être le nœud d'élément parent.

Exemples

- **altova:attributes**("MyAttribute") retourne `MyAttribute()*`

altova:attributes(AttributeName *as xs:string*, **SearchOptions** *as xs:string*) as **attribute()*** XP3.1 XQ3.1

Retourne tous les attribut possédant un nom local qui est le même que le nom fourni dans l'argument d'entrée, `AttributeName`. La recherche est sensible à la casse et est conduite le long de l'axe `attribute::` . Le nœud contextuel doit être le nœud d'élément parent. Le deuxième argument est une chaîne contenant des flags optionnels. Les flags disponibles sont :

r = passe à une recherche d'expression régulière ; `AttributeName` doit alors être une chaîne de recherche d'expression régulière ;

f = si cette option est spécifiée, alors `AttributeName` fournit une concordance complète ; dans le cas contraire, `AttributeName` ne nécessite qu'une concordance partielle d'un nom d'attribut pour retourner cet attribut. Par exemple : si **f** n'est pas spécifié, `MyAtt` retournera `MyAttribute`;

i = passe à une recherche insensible à la casse ;
p = comprend le préfixe d'espace de nom dans la recherche ; `AttributeName` devrait ensuite contenir le préfixe d'espace de nom, par exemple : `altova:MyAttribute`.
Les flags peuvent être écrits dans n'importe quel ordre. Les flags invalides généreront des erreurs. Un ou plusieurs flags peuvent être omis. La chaîne vide est permise et produire le même effet que la fonction n'ayant qu'un seul argument (*signature précédente*). Néanmoins, une séquence vide n'est pas permise en tant que le deuxième argument.

Exemples

- **altova:attributes**("MyAttribute", "rfip") retourne MyAttribute()*
- **altova:attributes**("MyAttribute", "pri") retourne MyAttribute()*
- **altova:attributes**("MyAtt", "rip") retourne MyAttribute()*
- **altova:attributes**("MyAttributes", "rfip") ne retourne aucune correspondance.
- **altova:attributes**("MyAttribute", "") retourne MyAttribute()*
- **altova:attributes**("MyAttribute", "Rip") retourne une erreur de flag non reconnu.
- **altova:attributes**("MyAttribute",) retourne une erreur d'argument manquant.

elements [altova:]

altova:elements(ElementName *as xs:string*) as **element()*** **XP3.1** **XQ3.1**

Retourne tous les éléments qui ont un nom local identique au nom fourni dans l'argument d'entrée, `ElementName`. La recherche est sensible à la casse et est conduite le long de l'axe `child::`. Le nœud contextuel doit être le nœud parent de/s l'élément/s recherché.

Exemples

- **altova:elements**("MyElement") retourne MyElement()*

altova:elements(ElementName *as xs:string*, **SearchOptions** *as xs:string*) as **element()*** **XP3.1** **XQ3.1**

Retourne tous les éléments qui ont un nom local identique au nom fourni dans l'argument d'entrée, `ElementName`. La recherche est sensible à la casse et est conduite le long de l'axe `child::`. Le nœud contextuel doit être le nœud parent de/s l'élément/s recherché. Le second argument est une chaîne contenant des flags optionnels. Les flags disponibles sont :
r = passe à une recherche d'expression régulière ; `ElementName` doit alors être une chaîne de recherche d'expression régulière ;
f = si cette option est spécifiée, alors `ElementName` fournit une concordance complète ; dans le cas contraire, `ElementName` ne nécessite qu'une concordance partielle d'un nom d'élément pour retourner cet élément. Par exemple : si **f** n'est pas spécifié, `MyElem` retournera `MyElement` ;
i = passe à une recherche insensible à la casse ;
p = comprend le préfixe d'espace de nom dans la recherche ; `ElementName` devrait ensuite contenir le préfixe d'espace de nom, par exemple : `altova:MyElement`.
Les flags peuvent être écrits dans n'importe quel ordre. Les flags invalides généreront des erreurs. Un ou plusieurs flags peuvent être omis. La chaîne vide est autorisée et produira le même effet que la fonction n'ayant qu'un argument (*signature précédente*). Néanmoins, une séquence vide n'est pas autorisée.

Exemples

- **altova:elements**("MyElement", "rip") retourne MyElement()*

- **altova:elements**("MyElement", "pri") retourne MyElement()*
- **altova:elements**("MyElement", "") retourne MyElement()*
- **altova:attributes**("MyElem", "rip") retourne MyElement()*
- **altova:attributes**("MyElements", "rfip") retourne aucune correspondance
- **altova:elements**("MyElement", "Rip") retourne une erreur flag-non reconnu.
- **altova:elements**("MyElement",) retourne une erreur second-argument-manquant.

find-first [altova:]

altova:find-first((Sequence *as item()**)*, **(Condition(Sequence-Item** *as xs:boolean*)) as **item()?** **XP3.1** **XQ3.1**

Cette fonction prend deux arguments. Le premier argument est une séquence d'un ou de plusieurs items de tout type de données. Le second argument, Condition, est une référence à une fonction XPath qui prend un argument (possède une arité de **1**) et retourne un booléen. Chaque item de **Sequence** est soumis à son tour à la fonction référencée dans Condition. (*Rappel :* cette fonction prend un seul argument.) Le premier item **Sequence** qui cause la fonction dans **Condition** à évaluer à **true()** est retourné en tant que le résultat de **altova:find-first**, et l'itération s'arrête.

Exemples

- **altova:find-first**(5 to 10, function($a) {$a mod 2 = 0}) retourne xs:integer 6

  L'argument **Condition** référence la fonction en ligne XPath 3.0, **function()**, qui déclare une fonction en ligne nommée **$a** puis la définit. Chaque item dans l'argument **Sequence** de **altova:find-first** est passé à son tour à **$a** en tant que sa valeur d'entrée. La valeur d'entrée est testée sur la condition dans la définition de la fonction ($a mod 2 = 0). La première valeur d'entrée pour satisfaire cette condition est retournée en tant que le résultat de **altova:find-first** (dans ce cas **6**).

- **altova:find-first**((1 to 10), (function($a) {$a+3=7})) retourne xs:integer 4

Autres exemples

Si le fichier **C:\Temp\Customers.xml** existe :

- **altova:find-first**(("C:\Temp\Customers.xml", "http://www.altova.com/index.html"), (doc-available#1)) retourne xs:string C:\Temp\Customers.xml

Si le fichier **C:\Temp\Customers.xml** n'existe pas et que **http://www.altova.com/index.html** existe :

- **altova:find-first**(("C:\Temp\Customers.xml", "http://www.altova.com/index.html"), (doc-available#1)) retourne xs:string http://www.altova.com/index.html

Si le fichier **C:\Temp\Customers.xml** n'existe pas, et que **http://www.altova.com/index.html** n'existe pas non plus :

- **altova:find-first**(("C:\Temp\Customers.xml", "http://www.altova.com/index.html"), (doc-available#1)) ne retourne aucun résultat

Notes à propos des exemples indiqués ci-dessus

- La fonction XPath 3.0, doc-available, prend un seul argument de chaîne, qui est utilisé en tant qu'URI, et retourne true si un nœud de document est trouvé à l'URI soumis. (Le document à l'URI soumis doit donc être un document XML.)
- La fonction doc-available peut être utilisée pour **Condition**, le second argument de altova:find-first, parce qu'il ne prend qu'un seul argument (arité=1), parce qu'il prend un item() en tant qu'entrée (une chaîne qui est utilisée en tant qu'URI), et retourne une valeur booléenne.
- Veuillez noter que la fonction doc-available est uniquement référencée, elle n'est pas appelée. Le suffixe #1 qui y est attaché indique une fonction avec une arité de 1. Sous sa forme complète, doc-available#1 signifie simplement : *Utiliser la fonction doc-availabe() à l'arité=1, en l'y passant en tant que son seul argument, chacun à son tour, chacun des items dans la première séquence*. En résultat, chacun des deux chaînes sera passée à **doc-available()**, qui utilise la chaîne en tant qu'URI et teste si un nœud de document existe à l'URI. S'il en existe un, **doc-available()** évalue à true() et cette chaîne est retournée en tant que le résultat de la fonction **altova:find-first**. *Note à propos de la fonction doc-available() : les chemins relatifs sont résolus relativement à l'URI de base actuel, qui est par défaut l'URI du document XML à partir duquel la fonction est chargée.*

find-first-combination [altova:]

altova:find-first-combination((Seq-01 *as item()**), **(Seq-02** *as item()**), **(Condition(Seq-01-Item, Seq-02-Item** *as xs:boolean*)) as **item()*** **XP3.1** **XQ3.1**

Cette fonction prend trois arguments :

- Les deux premiers arguments, **Seq-01** et **Seq-02**, sont des séquences d'un ou de plusieurs items de tout type de données.
- Le troisième argument, **Condition**, est une référence à une fonction XPath qui prend deux arguments (a une arité de **2**) et retourne un booléen.

Les items de **Seq-01** et **Seq-02** sont passés dans des paires ordonnées (un item de chaque séquence faisant une paire) en tant que les arguments de la fonction dans **Condition**. Les paires sont classées comme suit :

```
If   Seq-01 = X1, X2, X3 ... Xn
And  Seq-02 = Y1, Y2, Y3 ... Yn
Then (X1 Y1), (X1 Y2), (X1 Y3) ... (X1 Yn), (X2 Y1), (X2 Y2) ... (Xn Yn)
```

La première paire ordonnée qui entraîne la fonction **Condition** à évaluer à **true()** est retournée en tant que le résultat de **altova:find-first-combination**. Veuillez noter que : (i) si la fonction **Condition** itère par le biais des paires d'argument soumises et n'évalue pas une fois à **true()**, alors **altova:find-first-combination** retournera *Aucun résultat* ; (ii) Le résultat de **altova:find-first-combination** sera toujours une paire d'items (de tout type de données) ou aucun item.

Exemples

- **altova:find-first-combination**(11 to 20, 21 to 30, function($a, $b) {$a +$b = 32}) retourne la séquence de xs:integers (11, 21)
- **altova:find-first-combination**(11 to 20, 21 to 30, function($a, $b) {$a +$b = 33}) retourne la séquence de f xs:integers (11, 22)
- **altova:find-first-combination**(11 to 20, 21 to 30, function($a, $b) {$a +$b = 34}) retourne la séquence de xs:integers (11, 23)

find-first-pair [altova:]

altova:find-first-pair((Seq-01 *as item()**)**, (Seq-02** *as item()**)**, (Condition(Seq-01-Item, Seq-02-Item** *as xs:boolean*)) as **item()*** XP3.1 XQ3.1

Cette fonction prend trois arguments :

- Les deux premiers arguments, **Seq-01** et **Seq-02**, sont des séquences d'un ou de plusieurs items de tout type de données.
- Le troisième argument, **Condition**, est une référence à une fonction XPath qui prend deux arguments (a une arité de **2**) et retourne un booléen.

Les items de **Seq-01** et **Seq-02** sont passés dans des paires ordonnées en tant que les arguments de la fonction dans **Condition**. Les paires sont classées comme suit :

```
If   Seq-01 = X1, X2, X3 ... Xn
And  Seq-02 = Y1, Y2, Y3 ... Yn
Then (X1 Y1), (X2 Y2), (X3 Y3) ... (Xn Yn)
```

La première paire ordonnée qui cause la fonction **Condition** à évaluer à **true()** est retournée en tant que le résultat de **altova:find-first-pair**. Veuillez noter que : (i) Si la fonction **Condition** itère par le biais des paires d'arguments soumis et n'évalue pas une seule fois à **true()**, alors **altova:find-first-pair** retournera *Aucun résultat*; (ii) Le résultat de **altova:find-first-pair** sera toujours une paire d'items (de tout type de données) ou aucun item.

Exemples

- **altova:find-first-pair**(11 to 20, 21 to 30, function($a, $b) {$a+$b = 32}) retourne la séquence de xs:integers (11, 21)
- **altova:find-first-pair**(11 to 20, 21 to 30, function($a, $b) {$a+$b = 33}) retourne *Aucun résultat*

Veuillez noter à partir des deux exemples ci-dessus que l'ordonnance des paires est : (11, 21) (12, 22) (13, 23)...(20, 30). C'est pourquoi le second exemple retourne *Aucun résultat* (parce qu'aucune paire ordonnée de donne une somme de 33).

find-first-pair-pos [altova:]

altova:find-first-pair-pos((Seq-01 *as item()**)**, (Seq-02** *as item()**)**, (Condition(Seq-01-Item, Seq-02-Item** *as xs:boolean*)) as **xs:integer** XP3.1 XQ3.1

Cette fonction prend trois arguments :

- Les deux premiers arguments, **Seq-01** and **Seq-02**, sont des séquences d'un ou de

plusieurs items de tout type de données.
- Le troisième argument, **Condition**, est une référence à une fonction XPath qui prend deux arguments (a une arité de **2**) et retourne un booléen.

Les items de **Seq-01** et **Seq-02** sont passés dans des paires ordonnées en tant que les arguments de la fonction dans **Condition**. Les paires sont classées comme suit :

```
If   Seq-01 = X1, X2, X3 ... Xn
And  Seq-02 = Y1, Y2, Y3 ... Yn
Then (X1 Y1), (X2 Y2), (X3 Y3) ... (Xn Yn)
```

La position d'index de la première paire ordonnée qui entraîne la fonction **Condition** à évaluer à **true()** est retournée en tant que le résultat de **altova:find-first-pair-pos**. Veuillez noter que si la fonction **Condition** itère par le biais des paires d'arguments soumises et n'évalue pas une seule fois à **true()**, alors **altova:find-first-pair-pos** retournera *Aucun résultat*.

Exemples

- **altova:find-first-pair-pos**`(11 to 20, 21 to 30, function($a, $b) {$a+$b = 32})` retourne `1`
- **altova:find-first-pair-pos**`(11 to 20, 21 to 30, function($a, $b) {$a+$b = 33})` retourne *Aucun résultat*

Veuillez noter à partir des deux exemples ci-dessus que l'ordonnance des paires est : `(11, 21) (12, 22) (13, 23)...(20, 30)`. dans le premier exemple, la première paire entraîne la fonction **Condition** à évaluer à **true()**, et donc sa position d'index dans la séquence, **1**, est retournée. Le second exemple retourne *Aucun résultat* parce qu'aucune paire ne totalise pas une somme de `33`.

find-first-pos [altova:]

altova:find-first-pos((Sequence *as item()**)**, (Condition(Sequence-Item** *as xs:boolean*)**)** as **xs:integer** **XP3.1** **XQ3.1**

Cette fonction prend deux arguments. Le premier argument est une séquence d'un ou de plusieurs items de tout type de données. Le second argument, `Condition`, est une référence à une fonction XPath qui prend un argument (a une arité de **1**) et retourne une booléenne. Chaque item de **Sequence** est soumis à son tour à la fonction référencée dans `Condition`. (*Rappel :* cette fonction prend un seul argument.) Le premier item **Sequence** qui cause la fonction dans **Condition** à évaluer à **true()** voit sa position index dans **Sequence** retournée en tant que résultat de **altova:find-first-pos**, et l'itération stoppe.

Exemples

- **altova:find-first-pos**`(5 to 10, function($a) {$a mod 2 = 0})` retourne `xs:integer 2`

  L'argument **Condition** référence la fonction en ligne XPath 3.0, **function()**, qui déclare une fonction en ligne nommée **$a** et puis la définit. Chaque item dans l'argument de **Sequence** de **altova:find-first-pos** est passé à son tour à **$a** en tant que sa valeur d'entrée. La valeur d'entrée est testée à la condition dans la définition de la fonction (`$a mod 2 = 0`). La position d'index dans la séquence de la première valeur d'entrée pour satisfaire à cette condition est retournée en tant que le résultat de

altova:find-first-pos (dans ce cas **2**, puisque **6**, la première valeur (dans la séquence) afin de satisfaire à la condition, est à la position d'index **2** dans la séquence).

- **altova:find-first-pos**((2 to 10), (function($a) {$a+3=7})) retourne xs:integer 3

Autres exemples

Si le fichier **C:\Temp\Customers.xml** existe :

- **altova:find-first-pos**(("C:\Temp\Customers.xml", "http://www.altova.com/index.html"), (doc-available#1)) retourne 1

Si le fichier **C:\Temp\Customers.xml** n'existe pas, et que **http://www.altova.com/index.html** existe :

- **altova:find-first-pos**(("C:\Temp\Customers.xml", "http://www.altova.com/index.html"), (doc-available#1)) retourne 2

Si le fichier **C:\Temp\Customers.xml** n'existe pas, et que **http://www.altova.com/index.html** n'existe pas non plus :

- **altova:find-first-pos**(("C:\Temp\Customers.xml", "http://www.altova.com/index.html"), (doc-available#1)) ne retourne aucun résultat

Notes à propos des exemples donnés ci-dessus

- La fonction XPath 3.0, `doc-available`, prend un seul argument de chaîne qui est utilisé en tant qu'un URI, et retourne `true` si un nœud de document est trouvé à l'URI soumis. (Le document à l'URI soumis doit donc être un document XML.)
- La fonction `doc-available` peut être utilisée pour **Condition**, le second argument de `altova:find-first-pos`, parce qu'il ne prend qu'un seul argument (arité=1), parce qu'il prend un `item()` en tant qu'entrée (une chaîne qui est utilisée en tant qu'un URI), et retourne une valeur booléenne.
- Veuillez noter que la fonction `doc-available` est uniquement référencée, elle n'est pas appelée. Le suffixe `#1` qui y est attaché indique une fonction avec une arité de 1. dans sa totalité, `doc-available#1` signifie simplement : *Utiliser la fonction doc-availabe() qui a arité=1, y passant, en tant que son argument simple, chacun à son tour, chaque item dans la première séquence*. En tant que résultat. chacune des deux chaînes sera passée à **doc-available()**, qui utilise la chaîne en tant qu'un URI et teste si un nœud de document existe à l'URI. Si c'est le cas, la fonction **doc-available()** évalue à `true()` et la position de l'index de cette chaîne dans la séquence est retournée en tant que le résultat de la fonction **altova:find-first-pos**. *Note à propos de la fonction doc-available() : les chemins relatifs sont résolus relativement à l'URI de base actuel, qui par défaut est l'URI du document XML à partir duquel la fonction est chargée.*

substitute-empty [altova:]

altova:substitute-empty(FirstSequence *as item()**, **SecondSequence** *as item()***)**

as `item()*` XP3.1 XQ3.1

Si `FirstSequence` est vide, retourne `SecondSequence`. Si `FirstSequence` n'est pas vide, retourne `FirstSequence`.

Exemples

- `altova:substitute-empty( (1,2,3), (4,5,6) )` retourne `(1,2,3)`
- `altova:substitute-empty( (), (4,5,6) )` retourne `(4,5,6)`

Fonctions XPath/XQuery : Chaîne

Les fonctions d'extension de chaîne d'Altova peuvent être utilisées dans les expressions XPath et XQuery et proposent des fonctions supplémentaires pour le traitement des données. Les fonctions dans cette section peuvent être utilisées avec les moteurs **XPath 3.0** et **XQuery 3.0** d'Altova. Ils sont disponibles dans des contextes XPath/XQuery.

Note concernant le nommage de fonctions et de l'applicabilité de la langue

Les fonctions d'extension Altova peuvent être utilisées dans les expressions XPath/XQuery. Elles fournissent des fonctions supplémentaires aux fonctions d'ores et déjà disponibles dans la librairie standard des fonctions XPath, XQuery et XSLT. Les fonctions d'extension Altova se trouvent dans **l'espace de nom des fonctions d'extension Altova,** `http://www.altova.com/xslt-extensions`, et sont indiquées dans cette section par le préfixe `altova:`, qui est présupposé être lié à cet espace de nom. Veuillez noter que, en ce qui concerne les versions futures de votre produit, la prise en charge d'une fonction peut être interrompue et le comportement de certaines fonctions peut changer. Veuillez consulter la documentation lors des publications à venir pour plus d'informations concernant la prise en charge des fonctions d'extension Altova de cette version.

Fonctions XPath (utilisées dans les expressions XPath dans XSLT) :	XP1 XP2 XP3.1
Fonctions XSLT (utilisées dans les expressions XPath dans XSLT) :	XSLT1 XSLT2 XSLT3
Fonctions XQuery (utilisées dans les expressions XQuery dans XQuery) :	XQ1 XQ3.1

camel-case [altova:]

`altova:camel-case(InputString as xs:string) as xs:string` XP3.1 XQ3.1

Retourne la chaîne d'entrée `InputString` en CamelCase. La chaîne est analysée en utilisant l'expression régulière `'\s'` (qui est un raccourci pour le caractère d'espace blanc). Le premier caractère non-espace blanc après un espace blanc ou une séquence de plusieurs espaces blancs est mis en majuscule. Le premier caractère dans la chaîne de sortie est mis en majuscule.

Exemples

- `altova:camel-case("max")` retourne `Max`
- `altova:camel-case("max max")` retourne `Max Max`
- `altova:camel-case("file01.xml")` retourne `File01.xml`
- `altova:camel-case("file01.xml file02.xml")` retourne `File01.xml File02.xml`
- `altova:camel-case("file01.xml  file02.xml")` retourne `File01.xml File02.xml`
- `altova:camel-case("file01.xml -file02.xml")` retourne `File01.xml -file02.xml`

`altova:camel-case(InputString as xs:string, SplitChars as xs:string, IsRegex as xs:boolean) as xs:string` XP3.1 XQ3.1

Convertit la chaîne d'entrée `InputString` en camel case en utilisant `SplitChars` pour déterminer le/s caractère/s qui déclenche/nt la prochaine mise en majuscule. `SplitChars`

est utilisé en tant qu'expression régulière quand **IsRegex = true()**, ou en tant que caractères normaux quand **IsRegex = false()**. Le premier caractère dans la chaîne de sortie est mis en majuscule.

Exemples

- **altova:camel-case**("setname getname", "set|get", true()) retourne setName getName
- **altova:camel-case**("altova\documents\testcases", "\", false()) retourne Altova\Documents\Testcases

char [altova:]

altova:char(Position *as xs:integer***)** as **xs:string** **XP3.1** **XQ3.1**

Retourne une chaîne contenant le caractère à la position spécifiée par l'argument Position, dans la chaîne obtenue en convertissant la valeur de l'item contextuel en xs:string. La chaîne de résultat sera vide si aucun caractère n'existe à l'index soumis par l'argument Position.

Exemples

Si l'item contextuel est **1234ABCD**:

- **altova:char**(2) retourne 2
- **altova:char**(5) retourne A
- **altova:char**(9) retourne la chaîne vide.
- **altova:char**(-2) retourne la chaîne vide.

altova:char(InputString *as xs:string*, **Position** *as xs:integer***)** as **xs:string** **XP3.1** **XQ3.1**

Retourne une chaîne contenant la caractère à la position spécifiée par l'argument Position, dans la chaîne soumise en tant que l'argument InputString. La chaîne de résultat sera vide si aucun caractère n'existe à l'index soumis par l'argument Position.

Exemples

- **altova:char**("2014-01-15", 5) retourne -
- **altova:char**("USA", 1) retourne U
- **altova:char**("USA", 10) retourne la chaîne vide.
- **altova:char**("USA", -2) retourne la chaîne vide.

first-chars [altova:]

altova:first-chars(X-Number *as xs:integer***)** as **xs:string** **XP3.1** **XQ3.1**

Retourne une chaîne contenant le premier X-Number des caractères de la chaîne obtenue en convertissant la valeur de l'item de contexte en xs:string.

Exemples

Si l'item de contexte est **1234ABCD** :

- **altova:first-chars**(2) retourne 12
- **altova:first-chars**(5) retourne 1234A
- **altova:first-chars**(9) retourne 1234ABCD

altova:first-chars(InputString *as xs:string*, **X-Number** *as xs:integer*) as **xs:string** XP3.1 XQ3.1
Retourne une chaîne contenant le premier X-Number des caractères de la chaîne soumise en tant que l'argument InputString.

Exemples

- **altova:first-chars**("2014-01-15", 5) retourne 2014-
- **altova:first-chars**("USA", 1) retourne U

last-chars [altova:]

altova:last-chars(X-Number *as xs:integer*) as **xs:string** XP3.1 XQ3.1
Retourne une chaîne contenant le dernier X-Number de caractères de la chaîne obtenue en convertissant la valeur de l'item contextuel en xs:string.

Exemples

Si l'item contextuel est **1234ABCD** :

- **altova:last-chars**(2) retourne CD
- **altova:last-chars**(5) retourne 4ABCD
- **altova:last-chars**(9) retourne 1234ABCD

altova:last-chars(InputString *as xs:string*, **X-Number** *as xs:integer*) as **xs:string** XP3.1 XQ3.1
Retourne une chaîne contenant le dernier X-Number de caractères de la chaîne soumise en tant que l'argument InputString.

Exemples

- **altova:last-chars**("2014-01-15", 5) retourne 01-15
- **altova:last-chars**("USA", 10) retourne USA

pad-string-left [altova:]

altova:pad-string-left(StringToPad *as xs:string*, **StringLength** *as xs:integer*, **PadCharacter** *as xs:string*) as **xs:string** XP3.1 XQ3.1
L'argument PadCharacter est un caractère unique. Il est bourré à la gauche de la chaîne pour augmenter le nombre de caractères dans StringToPad de manière à ce que ce nombre soit équivalent à la valeur d'entier de l'argument StringLength. L'argument StringLength peut avoir toute valeur d'entier (positive ou négative), mais le padding n'aura lieu que si la valeur de StringLength est supérieure au nombre de caractères dans StringToPad. Si StringToPad. comporte plus de caractères que la valeur de StringLength, alors StringToPad ne sera pas modifié.

Exemples

- **altova:pad-string-left**('AP', 1, 'Z') retourne 'AP'
- **altova:pad-string-left**('AP', 2, 'Z') retourne 'AP'
- **altova:pad-string-left**('AP', 3, 'Z') retourne 'ZAP'
- **altova:pad-string-left**('AP', 4, 'Z') retourne 'ZZAP'
- **altova:pad-string-left**('AP', -3, 'Z') retourne 'AP'
- **altova:pad-string-left**('AP', 3, 'YZ') retourne une erreur pad-character-too-long

 © 2018 Altova GmbH

pad-string-right [altova:]

altova:pad-string-right(StringToPad *as xs:string*, **StringLength** *as xs:integer*, **PadCharacter** *as xs:string*) as **xs:string** XP3.1 XQ3.1

L'argument `PadCharacter` est un caractère unique. Il est bourré à la droite de la chaîne pour augmenter le nombre de caractères dans `StringToPad` de manière à ce que ce nombre soit équivalent à la valeur d'entier de l'argument `StringLength`. L'argument `StringLength` peut avoir toute valeur d'entier (positive ou négative), mais le padding n'aura lieu que si la valeur de `StringLength` est supérieure au nombre de caractères dans `StringToPad`. Si `StringToPad`. comporte plus de caractères que la valeur de `StringLength`, alors `StringToPad` ne sera pas modifié.

Exemples

- **altova:pad-string-right**('AP', 1, 'Z') retourne 'AP'
- **altova:pad-string-right**('AP', 2, 'Z') retourne 'AP'
- **altova:pad-string-right**('AP', 3, 'Z') retourne 'APZ'
- **altova:pad-string-right**('AP', 4, 'Z') retourne 'APZZ'
- **altova:pad-string-right**('AP', -3, 'Z') retourne 'AP'
- **altova:pad-string-right**('AP', 3, 'YZ') retourne une erreur pad-character-too-long

repeat-string [altova:]

altova:repeat-string(InputString *as xs:string*, **Repeats** *as xs:integer*) as **xs:string** XP2 XQ1 XP3.1 XQ3.1

Génère une chaîne qui est composée du premier argument `InputString` répété `Repeats` nombre de fois.

Exemples

- **altova:repeat-string**("Altova #", 3) retourne "Altova #Altova #Altova #"

substring-after-last [altova:]

altova:substring-after-last(MainString *as xs:string*, **CheckString** *as xs:string*) as **xs:string** XP3.1 XQ3.1

Si `CheckString` est trouvé dans `MainString`, alors la sous-chaîne qui se produit après `CheckString` dans `MainString` est retournée. Si `CheckString` n'est pas trouvé dans `MainString`, la chaîne vide est retournée. Si `CheckString` est une chaîne vide, alors `MainString` est retourné dans sa totalité. S'il y a plus d'une survenance de `CheckString` dans `MainString`, alors la sous-chaîne après la dernière survenance de `CheckString` est retournée.

Exemples

- **altova:substring-after-last**('ABCDEFGH', 'B') retourne 'CDEFGH'
- **altova:substring-after-last**('ABCDEFGH', 'BC') retourne 'DEFGH'
- **altova:substring-after-last**('ABCDEFGH', 'BD') retourne ''
- **altova:substring-after-last**('ABCDEFGH', 'Z') retourne ''
- **altova:substring-after-last**('ABCDEFGH', '') retourne 'ABCDEFGH'
- **altova:substring-after-last**('ABCD-ABCD', 'B') retourne 'CD'

- `altova:substring-after-last('ABCD-ABCD-ABCD', 'BCD')` retourne `''`

substring-before-last [altova:]

`altova:substring-before-last(MainString as xs:string, CheckString as xs:string) as xs:string` XP3.1 XQ3.1

Si `CheckString` est trouvé dans `MainString`, alors la sous-chaîne qui se produit avant `CheckString` dans `MainString` est retournée. Si `CheckString` n'est pas trouvé dans `MainString`, ou si `CheckString` est une chaîne vide, la chaîne vide est retournée. S'il y a plus d'une survenance de `CheckString` dans `MainString`, alors la sous-chaîne avant la dernière survenance de `CheckString` est retournée.

Exemples

- `altova:substring-before-last('ABCDEFGH', 'B')` retourne `'A'`
- `altova:substring-before-last('ABCDEFGH', 'BC')` retourne `'A'`
- `altova:substring-before-last('ABCDEFGH', 'BD')` retourne `''`
- `altova:substring-before-last('ABCDEFGH', 'Z')` retourne `''`
- `altova:substring-before-last('ABCDEFGH', '')` retourne `''`
- `altova:substring-before-last('ABCD-ABCD', 'B')` retourne `'ABCD-A'`
- `altova:substring-before-last('ABCD-ABCD-ABCD', 'ABCD')` retourne `'ABCD-ABCD-'`

substring-pos [altova:]

`altova:substring-pos(StringToCheck as xs:string, StringToFind as xs:string) as xs:integer` XP3.1 XQ3.1

Retourne la position de caractère de la première occurrence de `StringToFind` dans la chaîne `StringToCheck`. La position du caractère est retournée en tant qu'un entier. Le premier caractère de `StringToCheck` a la position `1`. Si `StringToFind` ne se produit pas dans le cadre de `StringToCheck`, l'entier `0` est retourné. Pour contrôler la deuxième occurrence ou une occurrence ultérieure de `StringToCheck`, utiliser la signature suivante de cette fonction.

Exemples

- `altova:substring-pos('Altova', 'to')` retourne `3`
- `altova:substring-pos('Altova', 'tov')` retourne `3`
- `altova:substring-pos('Altova', 'tv')` retourne `0`
- `altova:substring-pos('AltovaAltova', 'to')` retourne `3`

`altova:substring-pos(StringToCheck as xs:string, StringToFind as xs:string, Integer as xs:integer) as xs:integer` XP3.1 XQ3.1

Retourne la position de caractère de `StringToFind` dans la chaîne, `StringToCheck`. La recherche de `StringToFind` commence à partir de la position de caractère indiquée par l'argument `Integer` ; la sous-chaîne du caractère avant cette position n'est pas recherchée. Néanmoins, l'entier retourné, est la position de la chaîne trouvée dans le cadre de la chaîne *entière*, `StringToCheck`. Cette signature est utile pour trouver la deuxième position ou une position ultérieure d'une chaîne qui se produit plusieurs fois avec `StringToCheck`. Si `StringToFind` ne se produit pas dans le cadre de `StringToCheck`, l'entier `0` est retourné.

Exemples

- `altova:substring-pos('Altova', 'to', 1)` retourne `3`

- `altova:substring-pos('Altova', 'to', 3)` retourne `3`
- `altova:substring-pos('Altova', 'to', 4)` retourne `0`
- `altova:substring-pos('Altova-Altova', 'to', 0)` retourne `3`
- `altova:substring-pos('Altova-Altova', 'to', 4)` retourne `10`

trim-string [altova:]

`altova:trim-string(InputString as xs:string)` as `xs:string` XP3.1 XQ3.1

Cette fonction prend un argument `xs:string`, supprime tout espace blanc de tête et de fin et retourne une `xs:string` « nettoyée ».

Exemples

- `altova:trim-string("  Hello World  ")` retourne `"Hello World"`
- `altova:trim-string("Hello World  ")` retourne `"Hello World"`
- `altova:trim-string("  Hello World")` retourne `"Hello World"`
- `altova:trim-string("Hello World")` retourne `"Hello World"`
- `altova:trim-string("Hello  World")` retourne `"Hello  World"`

trim-string-left [altova:]

`altova:trim-string-left(InputString as xs:string)` as `xs:string` XP3.1 XQ3.1

Cette fonction prend un argument `xs:string`, supprime tout espace blanc de tête, et retourne une `xs:string` nettoyée à gauche.

Exemples

- `altova:trim-string-left("  Hello World  ")` retourne `"Hello World  "`
- `altova:trim-string-left("Hello World  ")` retourne `"Hello World  "`
- `altova:trim-string-left("  Hello World")` retourne `"Hello World"`
- `altova:trim-string-left("Hello World")` retourne `"Hello World"`
- `altova:trim-string-left("Hello  World")` retourne `"Hello  World"`

trim-string-right [altova:]

`altova:trim-string-right(InputString as xs:string)` as `xs:string` XP3.1 XQ3.1

Cette fonction prend un argument `xs:string`, supprime tout espace blanc de fin de ligne, et retourne une `xs:string` nettoyée à droite.

Exemples

- `altova:trim-string-right("  Hello World  ")` retourne `"  Hello World"`
- `altova:trim-string-right("Hello World  ")` retourne `"Hello World"`
- `altova:trim-string-right("  Hello World")` retourne `"  Hello World"`
- `altova:trim-string-right("Hello World")` retourne `"Hello World"`
- `altova:trim-string-right("Hello  World")` retourne `"Hello  World"`

Fonctions XPath/XQuery : Divers

L'objectif général suivant des fonctions d'extension XPath/XQuery sont prises en charge dans la version actuelle de RaptorXML+XBRL Server et celles-ci peuvent être utilisées dans (i) des expressions XPath dans un contexte XSLT, ou dans (ii) des expressions XQuery dans un document XQuery.

Note concernant le nommage de fonctions et de l'applicabilité de la langue

Les fonctions d'extension Altova peuvent être utilisées dans les expressions XPath/XQuery. Elles fournissent des fonctions supplémentaires aux fonctions d'ores et déjà disponibles dans la librairie standard des fonctions XPath, XQuery et XSLT. Les fonctions d'extension Altova se trouvent dans **l'espace de nom des fonctions d'extension Altova,** `http://www.altova.com/xslt-extensions`, et sont indiquées dans cette section par le préfixe `altova:`, qui est présupposé être lié à cet espace de nom. Veuillez noter que, en ce qui concerne les versions futures de votre produit, la prise en charge d'une fonction peut être interrompue et le comportement de certaines fonctions peut changer. Veuillez consulter la documentation lors des publications à venir pour plus d'informations concernant la prise en charge des fonctions d'extension Altova de cette version.

Fonctions XPath (utilisées dans les expressions XPath dans XSLT) :	XP1 XP2 XP3.1
Fonctions XSLT (utilisées dans les expressions XPath dans XSLT) :	XSLT1 XSLT2 XSLT3
Fonctions XQuery (utilisées dans les expressions XQuery dans XQuery) :	XQ1 XQ3.1

Fonctions URI

get-temp-folder [altova:]

`altova:get-temp-folder()` as `xs:string` XP2 XQ1 XP3.1 XQ3.1

Cette fonction ne prend aucun argument. Elle retourne le chemin vers le dossier temporaire de l'utilisateur actuel.

Exemples

- `altova:get-temp-folder()` retournerait, sur une machine Windows, quelque chose du genre `C:\Users\<UserName>\AppData\Local\Temp\` en tant que `xs:string`.

[Top]

Fonctions de graphique

Les fonctions de graphique recensées ci-dessous vous permettent de créer, de générer et d'enregistrer des graphiques en tant qu'images. Elles sont supportées dans la version actuelle de votre produit Altova de la manière décrite ci-dessous. Néanmoins, veuillez noter que dans des versions futures de votre produit, le support de l'une ou de plusieurs de ces fonctions peut être interrompu, en outre, le comportement de certaines fonctions peut changer. Veuillez consulter la documentation des publications futures pour obtenir des informations concernant le support des fonctions d'extension Altova dans cette publication.

Les fonctions de graphique sont des **fonctions XPath** (pas des fonctions XSLT), et sont organisées dans deux groupes :

- Fonctions pour générer et enregistrer des graphiques
- Fonctions pour créer des graphiques

Note : Les fonctions de graphique sont supportées uniquement dans les **produits de serveur d'Altova** et les **produits d'édition Enterprise de Altova**.

Note : Les formats d'image pris en charge pour les graphiques dans les éditions de serveur sont `jpg`, `png` et `bmp`. La meilleure option est png parce que ce format est comprimé sans pertes. Dans les éditions Enterprise, les formats pris en charge sont `jpg`. `png`, `bmp` et `gif`.

Fonctions de génération et d'enregistrement des graphiques

Ces fonctions prennent l'objet de graphique (obtenu avec les fonctions de création de graphique) et soit génèrent une image soit enregistrent une image dans le fichier

`altova:generate-chart-image` `($chart, $width, $height, $encoding) en tant que atomic`

alors que

- `$chart` est l'item d'extension de graphique obtenu avec la fonction `altova:create-chart`
- `$width` et `$height` doivent être spécifiés avec une unité de longueur
- `$encoding` peut être `binarytobase64` ou `binarytobase16`

La fonction retourne l'image de graphique dans le codage spécifié.

`altova:generate-chart-image` `($chart, $width, $height, $encoding, $imagetype) en tant que atomic`

alors que

- `$chart` est l'item d'extension de graphique obtenu avec la fonction `altova:create-chart`
- `$width` et `$height` doivent être spécifiés avec une unité de longueur

- `$encoding` peut être `base64Binary` ou `hexBinary`
- `$imagetype` peut être un des formats d'image suivants : `png`, `gif`, `bmp`, `jpg`, `jpeg`. Veuillez noter que `gif` n'est pas pris en charge pour les produits de serveur. *Voir également la note en haut de la page*.

La fonction retourne l'image de graphique dans le format d'encodage et d'image spécifié.

`altova:save-chart-image` `($chart, $filename, $width, $height) en tant que empty()` ***(Windows uniquement)***

alors que

- `$chart` est l'item d'extension de graphique obtenu avec la fonction `altova:create-chart`
- `$filename` est le nom du fichier et le chemin sous lequel l'image de graphique doit être enregistré
- `$width` et `$height` doit être spécifié avec une unité de longueur

La fonction enregistre l'image de graphique sous le fichier spécifié dans `$filename`.

`altova:save-chart-image` `($chart, $filename, $width, $height, $imagetype) en tant que empty()` ***(Windows uniquement)***

alors que

- `$chart` est l'item d'extension de graphique obtenu avec la fonction `altova:create-chart`
- `$filename` est le nom du fichier et le chemin sous lequel l'image de graphique doit être enregistré
- `$width` et `$height` doivent être spécifiés avec une unité de longueur
- `$imagetype` peut être un des formats d'image suivants : `png`, `gif`, `bmp`, `jpg`, `jpeg`. Veuillez noter que `gif` n'est pas pris en charge pour les produits de serveur. *Voir également la note en haut de la page*.

La fonction enregistre l'image de graphique sous le fichier spécifié dans `$filename` dans le format d'image spécifié.

Fonctions de création de graphiques

Les fonctions suivantes sont utilisées pour créer des graphiques.

`altova:create-chart``($chart-config, $chart-data-series*) en tant que chart extension item`

alors que

- `$chart-config` est l'item d'extension chart-config obtenu avec la fonction

altova:create-chart-config ou via la fonction altova:create-chart-config-from-xml
- $chart-data-series est l'item d'extension chart-data-series obtenu avec la fonction altova:create-chart-data-series ou la fonction altova:create-chart-data-series-from-rows

La fonction retourne un item d'extension de graphique qui est créé à partir des données fournies par le biais des arguments.

altova:create-chart-config($type-name, $title) en tant qu'item d'extension chart-config

alors que

- $type-name spécifie le type de graphique à créer : Pie, Pie3d, BarChart, BarChart3d, BarChart3dGrouped, LineChart, ValueLineChart, RoundGauge, BarGauge
- $title est le nom du graphique

La fonction retourne un item d'extension chart-config contenant les informations de configuration du graphique.

altova:create-chart-config-from-xml($xml-struct) en tant qu'item d'extension chart-config

alors que

- $xml-struct est la structure XML contenant les informations de configuration du graphique

La fonction retourne un item d'extension chart-config contenant les informations de configuration du graphique. Cette information est fournie dans un fragment de données XML.

altova:create-chart-data-series($series-name?, $x-values*, $y-values*) en tant qu'item d'extension chart-data-series

alors que

- $series-name spécifie le nom de la série
- $x-values donne la liste des valeurs de l'axe X
- $y-values donne la liste des valeurs de l'axe Y

La fonction retourne un item d'extension chart-data-series contenant les données pour la construction du graphique : c'est à dire, les noms de la série et les données des Axes.

altova:create-chart-data-row(x, y1, y2, y3, ...) en tant qu'item d'extension

`chart-data-x-Ny-row`

alors que

- `x` est la valeur de la colonne de l'axe X de la rangée des données de graphique
- `yN` sont les valeurs des colonnes de l'axe Y

La fonction retourne un item d'extension chart-data-x-Ny-row qui contient les données pour la colonne de l'axe X et les colonnes de l'axe Y d'une seule série.

altova:create-chart-data-series-from-rows`($series-names as xs:string*, $row*) en tant qu'item d'extension chart-data-series`

alors que

- `$series-name` est le nom de la série à créer
- `$row` est l'item d'extension chart-data-x-Ny-row qui doit être créé en tant qu'une série

La fonction retourne un item d'extension chart-data-series qui contient les données pour l'axe X et l'axe Y de la série.

altova:create-chart-layer`($chart-config, $chart-data-series*) en tant qu'item d'extension chart-layer`

alors que

- `$chart-config` est l'item d'extension chart-config obtenu avec la fonction `altova:create-chart-config` ou via la fonction `altova:create-chart-config-from-xml`
- `$chart-data-series` est l'item d'extension chart-data-series obtenu avec la fonction `altova:create-chart-data-series` ou la fonction `altova:create-chart-data-series-from-rows`

La fonction retourne un item d'extension chart-layer qui contient des données chart-layer.

altova:create-multi-layer-chart`($chart-config, $chart-data-series*, $chart-layer*)`

alors que

- `$chart-config` est l'item d'extension chart-config obtenu avec la fonction `altova:create-chart-config` ou via la fonction `altova:create-chart-config-from-xml`
- `$chart-data-series` est l'item d'extension chart-data-series obtenu avec la fonction `altova:create-chart-data-series` ou la fonction `altova:create-chart-data-series-from-rows`
- `$chart-layer` est l'item d'extension chart-layer obtenu avec la fonction

 © 2018 Altova GmbH

`altova:create-chart-layer`

La fonction retourne un item multi-layer-chart.

`altova:create-multi-layer-chart``($chart-config, $chart-data-series*, $chart-layer*, xs:boolean $mergecategoryvalues)`

alors que

- `$chart-config` est l'item d'extension chart-config obtenu avec la fonction `altova:create-chart-config` ou via la fonction `altova:create-chart-config-from-xml`
- `$chart-data-series` est l'item d'extension chart-data-series obtenu avec la fonction `altova:create-chart-data-series` ou la fonction `altova:create-chart-data-series-from-rows`
- `$chart-layer` est l'item d'extension chart-layer obtenu avec la fonction `altova:create-chart-layer`

La fonction retourne un item multi-layer-chart.

Structure XML des données de graphique

Ci-dessous, vous trouverez la structure XML des données de graphique, telle qu'elle pourrait s'afficher pour les Fonctions d'extension Altova pour les graphiques. Cela affecte l'apparence du graphique spécifique. Tous les éléments ne sont pas utilisés pour tous les types de graphiques, par ex. l'élément `<Pie>` est ignoré pour les graphiques à barres.

Note : Les fonctions de graphique sont uniquement prises en charge dans les **Édition Enterprise et Server** des produits Altova.

```
<chart-config>
        <General
        SettingsVersion="1"   must be provided
        ChartKind="BarChart"   Pie, Pie3d, BarChart, StackedBarChart, BarChart3d,
BarChart3dGrouped, LineChart, ValueLineChart, AreaChart, StackedAreaChart, RoundGauge,
BarGauge, CandleStick
        BKColor="#ffffff"   Color
        BKColorGradientEnd="#ffffff"   Color. In case of a gradient, BKColor and
BKColorGradientEnd define the gradient's colors
        BKMode="#ffffff"   Solid, HorzGradient, VertGradient
        BKFile="Path+Filename"   String. If file exists, its content is drawn over the
background.
        BKFileMode="Stretch"   Stretch, ZoomToFit, Center, Tile
        ShowBorder="1"   Bool
        PlotBorderColor="#000000"   Color
        PlotBKColor="#ffffff"   Color
        Title=""   String
        ShowLegend="1"   Bool
        OutsideMargin="3.%"   PercentOrPixel
        TitleToPlotMargin="3.%"   PercentOrPixel
        LegendToPlotMargin="3.%"   PercentOrPixel
        Orientation="vert"   Enumeration: possible values are: vert, horz
        >
        <TitleFont
           Color="#000000"   Color
           Name="Tahoma"   String
           Bold="1"   Bool
           Italic="0"   Bool
           Underline="0"   Bool
           MinFontHeight="10.pt"   FontSize (only pt values)
           Size="8.%"   FontSize />
        <LegendFont
           Color="#000000"
           Name="Tahoma"
           Bold="0"
           Italic="0"
           Underline="0"
           MinFontHeight="10.pt"
           Size="3.5%" />
        <AxisLabelFont
           Color="#000000"
           Name="Tahoma"
           Bold="1"
```

```
   Italic="0"
   Underline="0"
   MinFontHeight="10.pt"
   Size="5.%" />
</General>

<Line
       ConnectionShapeSize="1.%"   PercentOrPixel
       DrawFilledConnectionShapes="1"   Bool
       DrawOutlineConnectionShapes="0"   Bool
       DrawSlashConnectionShapes="0"   Bool
       DrawBackslashConnectionShapes="0"   Bool
/>

<Bar
       ShowShadow="1"   Bool
       ShadowColor="#a0a0a0"   Color
       OutlineColor="#000000"   Color
       ShowOutline="1"   Bool
/>

<Area
       Transparency="0"   UINT ( 0-255 ) 255 is fully transparent, 0 is opaque
       OutlineColor="#000000"   Color
       ShowOutline="1"   Bool
/>

<CandleStick
       FillHighClose="0"   Bool. If 0, the body is left empty. If 1, FillColorHighClose
is used for the candle body
       FillColorHighClose="#ffffff"   Color. For the candle body when close >
open
       FillHighOpenWithSeriesColor="1"   Bool. If true, the series color is used to
fill the candlebody when open > close
       FillColorHighOpen="#000000"   Color. For the candle body when open > close
and FillHighOpenWithSeriesColor is false
/>

<Colors   User-defined color scheme: By default this element is empty except for the
style and has no Color attributes
       UseSubsequentColors ="1"  Boolean. If 0, then color in overlay is used. If 1,
then subsequent colors from previous chart layer is used
       Style="User"   Possible values are: "Default", "Grayscale", "Colorful", "Pastel",
"User"
       Colors="#52aca0"   Color: only added for user defined color set
       Colors1="#d3c15d"   Color: only added for user defined color set
       Colors2="#8971d8"   Color: only added for user defined color set
       ...
       ColorsN=""   Up to ten colors are allowed in a set: from Colors to Colors9
</Colors>

<Pie
       ShowLabels="1"   Bool
```

```
                OutlineColor="#404040"   Color
                ShowOutline="1"   Bool
                StartAngle="0."   Double
                Clockwise="1"   Bool
                Draw2dHighlights="1"   Bool
                Transparency="0"   Int (0 to 255: 0 is opaque, 255 is fully transparent)
                DropShadowColor="#c0c0c0"   Color
                DropShadowSize="5.%"   PercentOrPixel
                PieHeight="10.%"   PercentOrPixel. Pixel values might be different in the result
because of 3d tilting
                Tilt="40.0"   Double (10 to 90: The 3d tilt in degrees of a 3d pie)
                ShowDropShadow="1"   Bool
                ChartToLabelMargin="10.%"   PercentOrPixel
                AddValueToLabel="0"   Bool
                AddPercentToLabel="0"   Bool
                AddPercentToLabels_DecimalDigits="0"  UINT ( 0 – 2 )
                >
                <LabelFont
                       Color="#000000"
                       Name="Arial"
                       Bold="0"
                       Italic="0"
                       Underline="0"
                       MinFontHeight="10.pt"
                       Size="4.%"    />
        </Pie>

        <XY>
                <XAxis   Axis
                       AutoRange="1"   Bool
                       AutoRangeIncludesZero="1"   Bool
                       RangeFrom="0."   Double: manual range
                       RangeTill="1."   Double : manual range
                       LabelToAxisMargin="3.%"   PercentOrPixel
                       AxisLabel=""   String
                       AxisColor="#000000"   Color
                       AxisGridColor="#e6e6e6"   Color
                       ShowGrid="1"   Bool
                       UseAutoTick="1"   Bool
                       ManualTickInterval="1."   Double
                       AxisToChartMargin="0.px"   PercentOrPixel
                       TickSize="3.px"   PercentOrPixel
                       ShowTicks="1"   Bool
                       ShowValues="1"   Bool
                       AxisPosition="LeftOrBottom"   Enums: "LeftOrBottom",
"RightOrTop", "AtValue"
                       AxisPositionAtValue = "0"   Double
                       >
                       <ValueFont
                              Color="#000000"
                              Name="Tahoma"
                              Bold="0"
                              Italic="0"
                              Underline="0"
```

```
                        MinFontHeight="10.pt"
                        Size="3.%"    />
            </XAxis>
            <YAxis  Axis (same as for XAxis)
                  AutoRange="1"
                  AutoRangeIncludesZero="1"
                  RangeFrom="0."
                  RangeTill="1."
                  LabelToAxisMargin="3.%"
                  AxisLabel=""
                  AxisColor="#000000"
                  AxisGridColor="#e6e6e6"
                  ShowGrid="1"
                  UseAutoTick="1"
                  ManualTickInterval="1."
                  AxisToChartMargin="0.px"
                  TickSize="3.px"
                  ShowTicks="1"   Bool
                  ShowValues="1"   Bool
                  AxisPosition="LeftOrBottom"   Enums: "LeftOrBottom",
"RightOrTop", "AtValue"
                  AxisPositionAtValue = "0"   Double
                  >
                  <ValueFont
                        Color="#000000"
                        Name="Tahoma"
                        Bold="0"
                        Italic="0"
                        Underline="0"
                        MinFontHeight="10.pt"
                        Size="3.%"/>
            </YAxis>
      </XY>

      <XY3d
            AxisAutoSize="1"   Bool: If false, XSize and YSize define the aspect ration of
x and y axis. If true, aspect ratio is equal to chart window
            XSize="100.%"   PercentOrPixel. Pixel values might be different in the result
because of 3d tilting and zooming to fit chart
            YSize="100.%"   PercentOrPixel. Pixel values might be different in the result
because of 3d tilting and zooming to fit chart
            SeriesMargin="30.%"   PercentOrPixel. Pixel values might be different in the
result because of 3d tilting and zooming to fit chart
            Tilt="20."   Double. -90 to +90 degrees
            Rot="20."   Double. -359 to +359 degrees
            FoV="50.">   Double. Field of view: 1-120 degree
            >
            <ZAxis
                  AutoRange="1"
                  AutoRangeIncludesZero="1"
                  RangeFrom="0."
                  RangeTill="1."
                  LabelToAxisMargin="3.%"
                  AxisLabel=""
                  AxisColor="#000000"
```

```
                AxisGridColor="#e6e6e6"
                ShowGrid="1"
                UseAutoTick="1"
                ManualTickInterval="1."
                AxisToChartMargin="0.px"
                TickSize="3.px" >
                <ValueFont
                        Color="#000000"
                        Name="Tahoma"
                        Bold="0"
                        Italic="0"
                        Underline="0"
                        MinFontHeight="10.pt"
                        Size="3.%"/>
        </ZAxis>
</XY3d>

<Gauge
        MinVal="0."   Double
        MaxVal="100."   Double
        MinAngle="225"   UINT: -359-359
        SweepAngle="270"   UINT: 1-359
        BorderToTick="1.%"   PercentOrPixel
        MajorTickWidth="3.px"   PercentOrPixel
        MajorTickLength="4.%"   PercentOrPixel
        MinorTickWidth="1.px"   PercentOrPixel
        MinorTickLength="3.%"   PercentOrPixel
        BorderColor="#a0a0a0"   Color
        FillColor="#303535"   Color
        MajorTickColor="#a0c0b0"   Color
        MinorTickColor="#a0c0b0"   Color
        BorderWidth="2.%"   PercentOrPixel
        NeedleBaseWidth="1.5%"   PercentOrPixel
        NeedleBaseRadius="5.%"   PercentOrPixel
        NeedleColor="#f00000"   Color
        NeedleBaseColor="#141414"   Color
        TickToTickValueMargin="5.%"   PercentOrPixel
        MajorTickStep="10."   Double
        MinorTickStep="5."   Double
        RoundGaugeBorderToColorRange="0.%"   PercentOrPixel
        RoundGaugeColorRangeWidth ="6.%"   PercentOrPixel
        BarGaugeRadius="5.%"   PercentOrPixel
        BarGaugeMaxHeight="20.%"   PercentOrPixel
        RoundGaugeNeedleLength="45.%"   PercentOrPixel
        BarGaugeNeedleLength="3.%"   PercentOrPixel
        >
        <TicksFont
                Color="#a0c0b0"
                Name="Tahoma"
                Bold="0"
                Italic="0"
                Underline="0"
                MinFontHeight="10.pt"
                Size="4.%"
        />
```

```
                <ColorRanges>   User-defined color ranges. By default empty with no child
element entries
                        <Entry
                                From="50. "   Double
                                FillWithColor="1"   Bool
                                Color="#00ff00"   Color
                        />
                        <Entry
                                From="50.0"
                                FillWithColor="1"
                                Color="#ff0000"
                        />
                        ...
                </ColorRanges>
        </Gauge>
</chart-config>
```

Exemple : fonctions de graphique

Le document XSLT présenté ci-dessous à titre d'exemple montre comment les fonctions d'extension Altova pour les graphiques peuvent être utilisées. Ci-dessous, vous trouverez un document XML et une capture d'écran de l'image de sortie générée lorsque le document XML est traité avec le document XSLT en utilisant XSLT 2.0 ou 3.0 Engine.

Note: Les fonctions de graphique sont uniquement prises en charge dans les **Éditions Enterprise et Server** des produits Altova.

Note: Pour plus d'informations concernant la création des tables de données de graphique, voir la documentation des produits Altova XMLSpy et StyleVision.

Document XSLT

Ce document XSLT (*liste ci-dessous*) utilise fonctions d'extension de graphique Altova pour générer un camembert. Elles peuvent être utilisées pour traiter le document XML recensé ci-dessous.

```
<?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>
```

Document XML

Ce document XML peut être traité avec le document XSLT ci-dessus. Les données se trouvant dans le document XML sont utilisées pour générer le camembert affiché dans la capture d'écran ci-dessous.

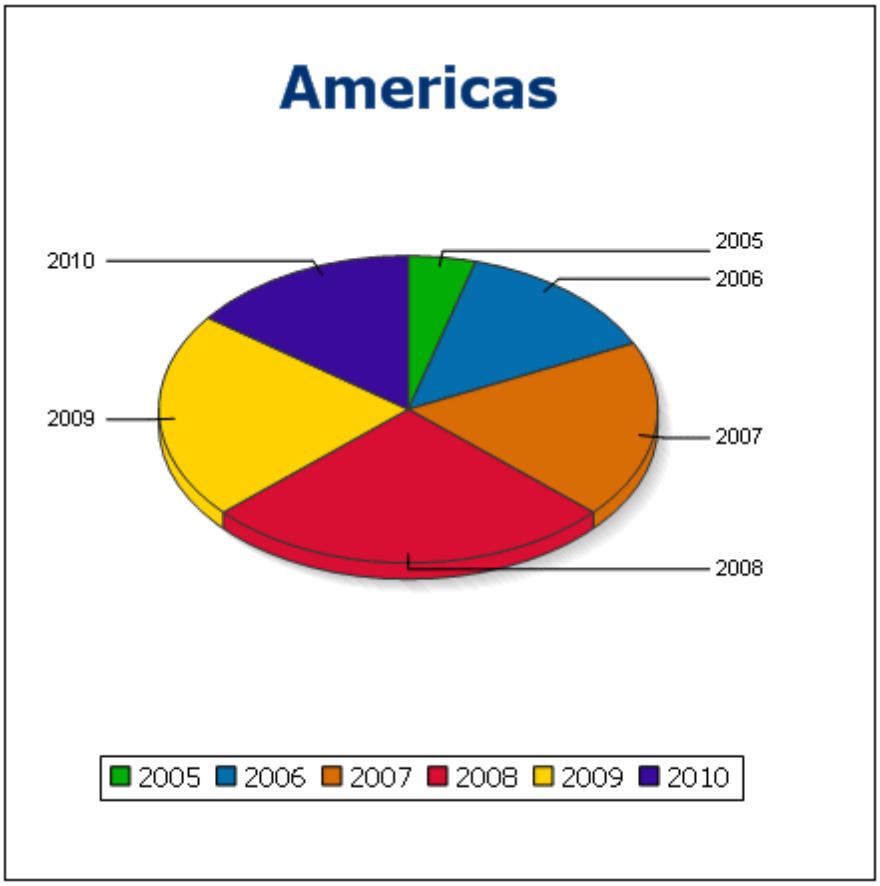
```
<?xml version="1.0" encoding="UTF-8"?>
<Data xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xsi:noNamespaceSchemaLocation="YearlySales.xsd">
   <ChartType>Pie Chart 2D</ChartType>
   <Region id="Americas">
      <Year id="2005">30000</Year>
      <Year id="2006">90000</Year>
      <Year 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>
```

Image de sortie

Le camembert affiché ci-dessous est généré lorsque le document XML recensé ci-dessus est traité avec le document XSLT.

 © 2018 Altova GmbH


Americas
2005
2006
2007
2008
2009
2010
2005 2006 2007 2008 2009 2010

Fonctions de code-barres

Le moteur XSLT utilise des librairies Java tierces pour créer des codes-barres. Ci-dessous, vous trouverez les classes et les méthodes publiques utilisées. Les classes sont emballées dans `AltovaBarcodeExtension.jar`, qui se trouve dans le dossier `<ProgramFilesFolder>\Altova\Common2018\jar`.

Les librairies Java utilisées se trouvent dans les sous-dossiers du dossier `<ProgramFilesFolder>\Altova\Common2018\jar`:

- `barcode4j\barcode4j.jar` (Site web : http://barcode4j.sourceforge.net/)
- `zxing\core.jar` (Site web : http://code.google.com/p/zxing/)

Les fichiers de licence sont également situés dans les dossiers respectifs.

Le package `com.altova.extensions.barcode`

Le package, `com.altova.extensions.barcode`, est utilisé pour générer la plupart des types de code-barres.

Les classes suivantes sont utilisées :

```
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`** *Used to store the barcode properties that will be dynamically set later*

```
    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`** *Enregistre la classe* `com.altova.extensions.barcode.proxy.zxing.QRCodeBean` *pour le* `qrcode` *bean, en plus des classes enregistrées par le* `org.krysalis.barcode4j.DefaultBarcodeClassResolver`.

Le package `com.altova.extensions.barcode.proxy.zxing`

Le package, `com.altova.extensions.barcode.proxy.zxing`, est utilisé pour générer le code-barres de type QRCode.

 © 2018 Altova GmbH

Les classes suivantes sont utilisées :

classe **QRCodeBean**

- *Élargit* `org.krysalis.barcode4j.impl.AbstractBarcodeBean`
- *Crée une* interface `AbstractBarcodeBean` pour `com.google.zxing.qrcode.encoder`

```
void generateBarcode(CanvasProvider canvasImp, String msg)
void setQRErrorCorrectionLevel(QRCodeErrorCorrectionLevel level)
BarcodeDimension calcDimensions(String msg)
double getVerticalQuietZone()
double getBarWidth()
```

classe **QRCodeErrorCorrectionLevel** *Erreur niveau de correction pour le* `QRCode`

```
static QRCodeErrorCorrectionLevel byName(String name)
"L" = ~7% correction
"M" = ~15% correction
"H" = ~25% correction
"Q" = ~30% correction
```

Exemple XSLT

Ci-dessous, vous trouverez un exemple XSLT pour démontrer comment les fonctions de code-barres sont utilisées dans une feuille de style 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 Fonctions d'extension diverses

Il existe plusieurs types de fonctions prêtes à l'utilisation dans les langages de programmation comme Java et C# qui ne sont pas disponibles en tant que fonctions XQuery/XPath ou en tant que fonctions XSLT. Un bon exemple sont les fonctions mathématiques disponibles en Java, comme `sin()` et `cos()`. Si ces fonctions étaient disponibles aux designers des feuilles de style XSLT et des requêtes XQuery, cela augmenterait le champ d'application des feuilles de style et des requêtes et faciliterait considérablement les tâches des créateurs de feuilles de style. Les moteurs XSLT et XQuery utilisés dans un grand nombre de produits Altova prennent en charge l'utilisation des fonctions d'extension dans Java et .NET, et pour les scripts MSXSL pour XSLT. Ils prennent également en charge les fonctions XBRL pour XSLT. Cette section décrit comment utiliser les fonctions d'extension et les scripts MSXSL dans vos feuilles de scripts XSLT et les documents XQuery. Les fonctions d'extension disponibles sont organisées dans les sections suivantes :

- Fonctions d'extension Java
- Fonctions d'extension .NET
- Fonctions XBRL pour XSLT
- Scripts MSXSL pour XSLT

Les deux problèmes principaux considérés pour les descriptions sont : (i) comment sont appelées les fonctions dans les librairies respectives ; et (ii) quelles sont les règles à suivre pour la conversion d'arguments dans un appel de fonction pour obtenir le format d'entrée requis de la fonction, et quelles sont les règles à suivre pour la conversion de retour (résultat de la fonction à l'objet de données XSLT/XQuery).

Exigences

Pour une prise en charge des fonctions d'extension, un Java Runtime Environment (pour l'accès aux fonctions Java) et le cadre de travail .NET Framework 2.0 (minimum, pour l'accès aux fonctions .NET) doit être installé sur la machine qui effectue la transformation XSLT ou l'exécution XQuery, ou doit être accessible pour les transformations.

Fonctions d'extension Java

Une fonction d'extension Java peut être utilisée dans le cadre d'une expression XPath ou XQuery pour invoquer un constructeur Java ou pour appeler une méthode Java (statique ou d'instance).

Un champ dans une classe Java est considéré être une méthode sans argument. Un champ peut être statique ou d'instance. L'accès aux champs est décrit dans les sous-sections respectives, statiques et d'instance.

Cette section est organisée dans les sous-sections suivantes :

- Java: Constructeurs
- Java: Méthodes statiques et champs statiques
- Java: Méthodes d'instance et champs d'instance
- Types de données : XPath/XQuery en Java
- Types de données : Java en XPath/XQuery

Forme la fonction d'extension

La fonction d'extension dans l'expression XPath/XQuery doit prendre la forme `prefix:fname()`.

- La partie `prefix:` identifie la fonction d'extension en tant qu'une fonction Java. Elle procède en associant la fonction d'extension avec une déclaration d'espace de nom in-scope, dont l'URI doit commencer avec `java:` (*voir les exemples ci-dessous*). La déclaration d'espace de nom devrait identifier une classe Java, par exemple : `xmlns:myns="java:java.lang.Math"`. Néanmoins, elle pourrait être simplement : `xmlns:myns="java"` (sans les doubles points), l'identification de la classe Java se faisant dans la partie `fname()` de la fonction d'extension.
- La partie `fname()` identifie la méthode Java appelée, et fournit les arguments pour la méthode (*voir les exemples ci-dessous*). Néanmoins, si l'URI d'espace de nom identifié par la partie `prefix:` n'identifie pas une classe Java (*voir point précédent*), alors la classe Java devrait être identifiée dans la partie `fname()`, avant la classe et séparé de la classe par un point (*voir le second exemple XSLT ci-dessous*).

Note : La classe appelée doit être sur le classpath de la machine.

Exemple XSLT

Voici deux exemples pour démontrer comment appeler une méthode statique. Dans le premier exemple, le nom de classe (`java.lang.Math`) est inclus dans l'URI d'espace de nom et ne doit donc pas être dans la même partie `fname()`. Dans le second exemple, la partie `prefix:` fournit le préfixe `java:` alors que la partie `fname()` identifie la classe et la méthode.

```
<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)" />
```

La méthode nommée dans la fonction d'extension (`cos()` dans l'exemple ci-dessus) doit correspondre au nom d'une méthode statique publique dans la classe Java nommée (`java.lang.Math` dans l'exemple ci-dessus).

Exemple XQuery

Voici un exemple XQuery semblable à l'exemple XSLT ci-dessus:

```
<cosine xmlns:jMath="java:java.lang.Math">
   {jMath:cos(3.14)}
</cosine>
```

Classes Java définies par l'utilisateur

Si vous avez créé vos propres classes Java, les méthodes dans ces classes sont appelées différemment selon que : (i) les classes sont accédées par un fichier JAR ou un fichier classe et (ii) si ces fichiers (JAR ou classe) sont situés dans le répertoire actuel (le même répertoire que le document XSLT ou XQuery) ou pas. Consulter les sections Fichiers de classe définis par l'utilisateur et Fichiers Jar définis par l'utilisateur pour apprendre comment localiser ces fichiers. Veuillez noter que les chemins vers les fichiers de classe ne se trouvant pas dans le répertoire actuel et vers tous les fichiers Jar doivent être spécifiés.

Fichiers de classe définis par l'utilisateur

Si l'accès se produit par un fichier de classe, deux possibilités s'ouvrent à vous :

- Le fichier de classe se trouve dans un package. Le fichier XSLT ou XQuery se trouve dans le même dossier que le package Java. (*Voir exemple ci-dessous*.)
- Le fichier de classe ne se trouve pas dans un package. Le fichier XSLT ou XQuery se trouve dans le même dossier que le fichier de classe. (*Voir exemple ci-dessous*.)
- Le fichier de classe se trouve dans un package. Le fichier XSLT ou XQuery se trouve dans un emplacement aléatoire. (*Voir exemple ci-dessous*.)
- Le fichier de classe ne se trouve pas dans un package. Le fichier XSLT ou XQuery se trouve dans un emplacement aléatoire. (*Voir exemple ci-dessous*.)

Prenons le cas où le fichier de classe ne se trouve pas dans un package et se trouve dans le même dossier que le document XSLT ou XQuery document. Dans ce cas, puisque toutes les classes dans le dossier sont trouvées, l'emplacement du fichier ne doit pas être spécifié. La syntaxe pour identifier une classe est :

```
java:classname
```

alors que

`java:` indique qu'une fonction Java définie par l'utilisateur est appelée ; (les classes Java dans le répertoire actuelles seront chargées par défaut)
`classname` est le nom de la classe de la méthode requise

La classe est identifiée dans un URI d'espace de nom, et l'espace de nom est utilisé pour préfixer un appel de méthode.

Fichier de classe dans un package, le fichier XSLT/XQuery se trouve dans le même dossier que le package Java

L'exemple ci-dessous appelle la méthode `getVehicleType()` de la classe `Car` du package `com.altova.extfunc`. Le package `com.altova.extfunc` se trouve dans le dossier `JavaProject`. Le fichier XSLT se trouve également dans le dossier `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>
```

 © 2018 Altova GmbH

Fichier de classe ne se trouve pas dans un package, le fichier XSLT/XQuery se trouve dans le même dossier que le fichier de classe

L'exemple ci-dessous appelle la méthode `getVehicleType()` de la classe `Car` du package `com.altova.extfunc`. Le fichier de classe `Car` se trouve dans l'emplacement de dossier suivant : `JavaProject/com/altova/extfunc`. Le fichier XSLT se trouve également dans le dossier `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>
```

Fichier de classe dans un package, le fichier XSLT/XQuery se trouve dans n'importe quel emplacement

L'exemple ci-dessous appelle la méthode `getCarColor()` de la classe `Car` du package `com.altova.extfunc`. Le package `com.altova.extfunc` se trouve dans le dossier `JavaProject`. Le fichier XSLT se trouve dans n'importe quel emplacement. Dans ce cas, l'emplacement du package doit être spécifié dans l'URI en tant que chaîne de requête. La syntaxe est :

```
java:classname[?path=uri-of-package]
```

alors que

`java`: indique qu'une fonction Java définie par l'utilisateur est appelée
`uri-of-package` est l'URI du package Java
`classname` est le nom de la classe de méthode requise

La classe est identifiée dans un URI d'espace de nom et l'espace de nom est utilisé pour préfixer un appel de méthode. L'exemple ci-dessous montre comment accéder à un fichier de classe qui se situe dans un autre répertoire que le répertoire actuel.

```
<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>
```

Fichier de classe ne se trouve pas dans un package, XSLT/XQuery se trouve dans n'importe quel emplacement

L'exemple ci-dessous appelle la méthode `getCarColor()` de la classe `Car` du package `com.altova.extfunc`. Le package `com.altova.extfunc` se trouve dans le dossier `JavaProject`. Le fichier XSLT se trouve dans n'importe quel emplacement. L'emplacement du fichier de classe est spécifié dans le cadre de l'URI d'espace de nom en tant que chaîne de requête. La syntaxe est :

```
java:classname[?path=uri-of-classfile]
```

alors que

`java:` indique qu'une fonction Java définie par l'utilisateur est appelée
`uri-of-classfile` est l'URI du dossier contenant le fichier de classe
`classname` est le nom de la classe de méthode requise

La classe est identifiée dans un URI d'espace de nom et l'espace de nom est utilisé pour préfixer un appel de méthode. L'exemple ci-dessous montre comment accéder à un fichier de classe qui se situe dans un autre répertoire que le répertoire actuel.

```
<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>
```

Note : Lorsqu'un chemin d'accès est fourni par le biais de la fonction d'extension, le chemin est ajouté au ClassLoader.

 © 2018 Altova GmbH

Fichiers Jar définis par l'utilisateur

Si l'accès s'effectue par le biais d'un fichier JAR, l'URI du fichier JAR doit être spécifié à l'aide de la syntaxe suivante :

```
xmlns:classNS="java:classname?path=jar:uri-of-jarfile!/"
```

La méthode est ensuite appelée en utilisant le préfixe de l'URI d'espace de nom qui identifie la classe : `classNS:method()`

Dans l'exemple ci-dessus :

`java:` indique qu'une fonction Java est appelée
`classname` est le nom de la classe définie par l'utilisateur
`?` est le séparateur entre le nom de classe et le chemin d'accès
`path=jar:` indique qu'un chemin d'accès vers un fichier JAR est donné
`uri-of-jarfile` est l'URI du fichier jar
`!/` est le délimiteur de fin du chemin d'accès
`classNS:method()` est l'appel de la méthode

En alternative, le nom de classe peut être donné avec l'appel de la méthode. Voici deux exemples de la syntaxe :

```
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()
```

Voici un exemple XSLT complet qui utilise un fichier JAR pour appeler une fonction d'extension 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>
```

Note : Lorsqu'un chemin d'accès est fourni par le biais de la fonction d'extension, le chemin est ajouté au ClassLoader.

Java: Constructeurs

Une fonction d'extension peut être utilisée pour appeler un constructeur Java. Tous les constructeurs sont appelés avec la pseudo-fonction `new()`.

Si le résultat d'un appel de constructeur Java peut être converti implicitement en des types de données XPath/XQuery, alors la fonction d'extension Java retournera une séquence qui est un type de données XPath/XQuery. Si le résultat d'un appel de constructeur Java ne peut pas être converti en un type de données XPath/XQuery adéquat, le constructeur crée un objet Java encapsulé avec un type qui est le nom de la classe retournant cet objet Java. Par exemple, si un constructeur pour la classe `java.util.Date` est appelé (`java.util.Date.new()`), alors un objet ayant un type `java.util.Date` est retourné. Le format lexical de l'objet retourné peut ne pas être conforme au format lexical d'un type de données XPath et la valeur devrait donc être convertie dans le format lexical du type de données XPath requis, puis au type de données XPath requis.

Deux choses peuvent être réalisées avec un objet Java créé par un constructeur :

- Il peut être attribué à une variable :
```
<xsl:variable name="currentdate" select="date:new()"
xmlns:date="java:java.util.Date" />
```
- Il peut être passé à une fonction d'extension (*voir Méthode d'instance et Champs d'instance*) :
```
<xsl:value-of select="date:toString(date:new())"
xmlns:date="java:java.util.Date" />
```

 © 2018 Altova GmbH

Java: Méthodes statiques et Champs statiques

Une méthode est appelée directement par son nom Java et en fournissant les arguments pour la méthode. Les champs statiques (méthodes qui ne prennent aucun argument), comme les champs de valeur constante `E` et `PI`, sont accédés sans devoir spécifier aucun argument.

Exemples XSLT

Voici quelques exemples montrant comment appeler des méthodes et des champs statiques :

```
<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)" />
```

Veuillez noter que les fonctions d'extension ci-dessus prennent la forme `prefix:fname()`. Le préfixe dans les trois cas indiqués est `jMath:`, qui est associé avec l'URI d'espace de nom `java:java.lang.Math`. (L'URI d'espace de nom doit commencer avec `java:`. Dans les exemples ci-dessus, il est étendu pour contenir le nom de classe (`java.lang.Math`).) La partie `fname()` des fonctions extension doit correspondre au nom d'une classe publique (par ex. `java.lang.Math`) suivi du nom d'une méthode statique publique avec son/ses argument/s (comme `cos(3.14)`) ou un champ statique public (comme `PI()`).

Dans les exemples ci-dessus, Le nom de classe a été inclus dans l'espace de nom URI. S'il n'était pas contenu dans l'URI d'espace de nom, il devrait être inclus dans la partie `fname()` de la fonction d'extension. Par exemple :

```
<xsl:value-of xmlns:java="java:"
              select="java:java.lang.Math.cos(3.14)" />
```

Exemple XQuery

Un exemple semblable dans XQuery serait :

```
<cosine xmlns:jMath="java:java.lang.Math">
   {jMath:cos(3.14)}
</cosine>
```

Java: Méthodes d'instance et Champs d'instance

Une méthode d'instance a un objet Java qui lui est passé en tant que son premier argument de l'appel de méthode. Un tel objet Java est généralement créé en utilisant une fonction d'extension (par exemple un appel de constructeur) ou un paramètre/variable de feuille de style. Un exemple XSLT de ce type serait :

```
<xsl:stylesheet version="1.0" exclude-result-prefixes="date"
  xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
  xmlns:date="java:java.util.Date"
  xmlns:jlang="java:java.lang">
  <xsl:param name="CurrentDate" select="date:new()"/>
  <xsl:template match="/">
      <enrollment institution-id="Altova School"
                date="{date:toString($CurrentDate)}"
                type="
{jlang:Object.toString(jlang:Object.getClass( date:new() ))}">
      </enrollment>
  </xsl:template>
</xsl:stylesheet>
```

Dans l'exemple ci-dessus, la valeur du nœud `enrollment/@type` est créée comme suit :

1. Un objet est créé avec un constructeur pour la classe `java.util.Date` (avec le constructeur `date:new()`).
2. Cet objet Java est passé en tant que l'argument de la méthode `jlang.Object.getClass`.
3. L'objet obtenu par la méthode `getClass` est passé en tant que l'argument de la méthode `jlang.Object.toString`.

Le résultat (la valeur de `@type`) aura une chaîne avec la valeur : `java.util.Date`.

Un champ d'instance est théoriquement différent d'une méthode d'instance dans le sens où ce n'est pas un objet Java en soit qui est passé en tant qu'argument au champ d'instance. Au lieu, un paramètre ou une variable est passée en tant que l'argument. Néanmoins, le paramètre/variable peut contenir elle-même la valeur retournée par un objet Java. Par exemple, le paramètre `CurrentDate` prend la valeur retournée par un constructeur pour la classe `java.util.Date`. Cette valeur est ensuite passée en tant qu'un argument à la méthode d'instance `date:toString` afin de fournir la valeur de `/enrollment/@date`.

 © 2018 Altova GmbH

Types de données : XPath/XQuery en Java

Lorsqu'une fonction Java est appelée depuis l'intérieur d'une expression XPath/XQuery, le type de données des arguments de la fonction est important pour déterminer laquelle des multiples classes Java possédant le même nom est appelé.

dans Java, les règles suivantes sont suivies :

- S'il y a plus d'une seule méthode Java portant le même nom, mais chacune contenant un nombre différent d'arguments que les autres, alors c'est la méthode Java qui correspond le mieux au nombre d'arguments dans l'appel de la fonction qui est sélectionnée.
- La chaîne XPath/XQuery, le nombre et les types de données booléens (*voir la liste ci-dessous*) sont convertis implicitement en un type de données Java correspondant. Si le type XPath/XQuery fourni peut être converti en plus d'un seul type Java (par exemple, `xs:integer`), alors, le type Java sélectionné est celui qui est déclaré pour la méthode sélectionnée. Par exemple, si la méthode Java appelée est `fx(decimal)` et que le type de données XPath/XQuery fourni est `xs:integer`, alors `xs:integer` sera converti dans le type de données `decimal` de Java.

La table ci-dessous recense les conversions implicites de la chaîne XPath/XQuery, le nombre et les types booléens en types de données Java.

`xs:string`	`java.lang.String`
`xs:boolean`	`boolean` (primitive), `java.lang.Boolean`
`xs:integer`	int, long, short, byte, float, double, et les classes de wrapper de ces catégories comme `java.lang.Integer`
`xs:float`	`float` (primitive), `java.lang.Float`, `double` (primitive)
`xs:double`	`double` (primitive), `java.lang.Double`
`xs:decimal`	`float` (primitive), `java.lang.Float`, `double`(primitive), `java.lang.Double`

Les sous-types des types de données du schéma XML recensés ci-dessus (et qui sont utilisés dans XPath et XQuery) seront aussi convertis dans le/s type/s Java correspondant à ce type d'ancêtre du sous-type.

Dans certains cas, il peut ne pas être possible de sélectionner la méthode Java correcte sur la base du cas suivant.

- L'argument fourni est une valeur `xs:untypedAtomic` de 10 et il est prévu pour la méthode `mymethod(float)`.
- Néanmoins, il existe une autre méthode dans la classe qui prend un argument d'un autre type de données : `mymethod(double)`.
- Puisque les noms de la méthode sont les mêmes et que le type fourni (`xs:untypedAtomic`) peut être converti correctement soit à `float` ou `double`, il est possible que `xs:untypedAtomic` soit converti en `double` au lieu de `float`.
- Par conséquence, la méthode sélectionnée ne sera pas la méthode requise et peut ne pas produire le résultat attendu. Pour contourner ce problème, vous pouvez créer une

méthode définie par l'utilisateur avec un autre nom et utiliser cette méthode.

Les types qui ne sont pas couverts dans la liste ci-dessus (par exemple, `xs:date`) ne sera pas converti et générera une erreur. Néanmoins, veuillez noter que dans certains cas, il peut être possible de créer le type Java requis en utilisant un constructeur Java.

 © 2018 Altova GmbH

Types de données : Java en XPath/XQuery

Lorsqu'une méthode Java retourne une valeur, le type de donnée de la valeur est de type chaîne, numérique ou booléen, alors il est converti dans le type XPath/XQuery correspondant. Par exemple, les types de données `java.lang.Boolean` et `boolean` de Java sont convertis en `xsd:boolean`.

Des arrays unidimensionnels retournés par des fonctions sont étendus à une séquence. Les arrays multidimensionnels ne seront pas convertis et devraient donc être encapsulés.

Lorsqu'un objet Java encapsulé ou un type de données différent de chaîne, numérique ou booléen est retourné, vous pouvez assurer la conversion dans le type XPath/XQuery requis tout d'abord en utilisant une méthode Java (par ex.`toString`) pour convertir l'objet Java en une chaîne. Dans XPath/XQuery, la chaîne peut être modifiée pour correspondre à la représentation lexicale du type requis puis convertie dans le type requis (par exemple, en utilisant l'expression `cast as`).

Fonctions d'extension .NET

Si vous travaillez sur la plate-forme .NET d'une machine Windows, vous pouvez utiliser les fonctions d'extension écrites dans un des langages .NET (par exemple, C#). Une fonction d'extension .NET peut être utilisée dans une expression XPath ou XQuery pour invoquer un constructeur, une propriété ou une méthode (statique ou instance) dans une classe .NET.

Une propriété d'une classe .NET est appelée en utilisant la syntaxe `get_PropertyName()`.

Cette section est organisée dans les sous-sections suivantes :

- .NET: Constructeurs
- .NET: Méthodes statiques et champs statiques
- .NET: Méthodes d'instance et Champs d'instance
- Types de données : XPath/XQuery en .NET
- Types de données : .NET en XPath/XQuery

Forme de la fonction d'extension

La fonction d'extension de l'expression XPath/XQuery doit avoir la forme `prefix:fname()`.

- La partie `prefix:` est associée avec un URI qui identifie la classe .NET en cours d'adressage.
- La partie `fname()` identifie le constructeur, la propriété ou la méthode (statique ou d'instance) dans la classe .NET, et fournit un/des argument/s, le cas échéant.
- L'URI doit commencer avec `clitype:` (qui identifie la fonction comme étant une fonction d'extension .NET).
- La forme `prefix:fname()` de la fonction d'extension peut être utilisée avec les classes de système et avec les classes dans un assemblage chargé. Néanmoins, si une classe doit être chargée, les paramètres supplémentaires contenant l'information requise devront être fournis.

Paramètres

Pour charger un assemblage, utiliser les paramètres suivants :

`asm`	Le nom de l'assemblage à charger.
`ver`	Le numéro de la version (quatre entiers maximum séparés par des points).
`sn`	Le jeton clé du nom solide de l'assemblage (16 chiffres hex).
`from`	Un URI qui donne l'emplacement de l'assemblage (DLL) à charger. Si l'URI est relatif, il est relatif au document XSLT ou XQuery. Si ce paramètre est présent, tout autre paramètre est ignoré.
`partialname`	Le nom partiel de l'assemblage. Il est fourni dans `Assembly.LoadWith.PartialName()`, qui tentera de charger l'assemblage. Si `partialname` est présent, tout autre paramètre est

 © 2018 Altova GmbH

	ignoré.
`loc`	La région, par exemple, `en-US`. Le défaut est `neutral`.

Si l'assemblage doit être chargé depuis un DLL, utiliser le paramètre `from` et omettre le paramètre `sn`. Si l'assemblage doit être chargé depuis le Global Assembly Cache (GAC), utiliser le paramètre `sn` et omettre le paramètre `from`.

Un signe d'interrogation doit être inséré avant le premier paramètre, et les paramètres doivent être séparés par un point-virgule. Le nom du paramètre donne sa valeur avec un signe égal (*voir exemple ci-dessous*).

Exemples de déclarations d'espace de nom

Un exemple de déclaration d'espace de nom dans XSLT qui identifie le système class `System.Environment`:

```
xmlns:myns="clitype:System.Environment"
```

Un exemple d'une déclaration d'espace de nom dans XSLT qui identifie la classe devant être chargée en tant que `Trade.Forward.Scrip`:

```
xmlns:myns="clitype:Trade.Forward.Scrip?asm=forward;version=10.6.2.1"
```

Un exemple d'une déclaration d'espace de nom dans XQuery qui identifie la classe de système `MyManagedDLL.testClass`:. Deux cas peuvent être distingués :

1. Lorsque l'assemblage est chargé depuis le GAC:
```
declare namespace cs="clitype:MyManagedDLL.testClass?asm=MyManagedDLL;
       ver=1.2.3.4;loc=neutral;sn=b9f091b72dccfba8";
```
2. Lorsque l'assemblage est chargé depuis le DLL (références partielles ou complète ci-dessous) :
```
       declare namespace cs="clitype:MyManagedDLL.testClass?from=file:///
C:/Altova
       Projects/extFunctions/MyManagedDLL.dll;

declare namespace cs="clitype:MyManagedDLL.testClass?
from=MyManagedDLL.dll;
```

Exemple XSLT

Voici un exemple complet XSLT qui appelle les fonctions dans la classe de système `System.Math`:

```
<xsl:stylesheet version="2.0"
   xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
   xmlns:xs="http://www.w3.org/2001/XMLSchema"
   xmlns:fn="http://www.w3.org/2005/xpath-functions">
   <xsl:output method="xml" omit-xml-declaration="yes" />
```

```
   <xsl:template match="/">
      <math xmlns:math="clitype:System.Math">
         <sqrt><xsl:value-of select="math:Sqrt(9)"/></sqrt>
         <pi><xsl:value-of select="math:PI()"/></pi>
         <e><xsl:value-of select="math:E()"/></e>
         <pow><xsl:value-of select="math:Pow(math:PI(), math:E())"/></pow>
      </math>
   </xsl:template>
</xsl:stylesheet>
```

La déclaration d'espace de nom de l'élément `math` associe le préfixe `math:` avec l'URI `clitype:System.Math`. Le début `clitype:` de l'URI indique que ce qui suit identifie soit une classe de système soit une classe chargée. Le préfixe `math:` dans les expressions XPath associe les fonctions d'extension avec l'URI (et, par extension, la classe) `System.Math`. Les fonctions d'extension identifient des méthodes dans la classe `System.Math` et fournissent les arguments le cas échéant.

Exemple XQuery

Voici un fragment d'exemple XQuery similaire à l'exemple XSLT ci-dessus :

```
<math xmlns:math="clitype:System.Math">
   {math:Sqrt(9)}
</math>
```

Comme pour l'exemple XSLT ci-dessus, la déclaration d'espace de nom identifie la classe .NET, dans ce cas une classe de système. L'expression XQuery identifie la méthode à appeler et fournit l'argument.

 © 2018 Altova GmbH

.NET: Constructeurs

Une fonction d'extension peut être utilisée pour appeler un constructeur .NET. Tous les constructeurs sont appelés avec la pseudo-fonction `new()`. S'il y a plus d'un constructeur pour une classe, le constructeur qui correspond le mieux au nombre d'arguments fourni est sélectionné. Si aucun constructeur n'est jugé conforme aux arguments fournis, alors une erreur `'No constructor found'` sera retournée.

Des constructeurs qui retournent des types de données XPath/XQuery

Si le résultat d'un appel de constructeur .NET peut être converti implicitement dans des types de données XPath/XQuery, alors la fonction d'extension .NET retournera une séquence qui est un type de données XPath/XQuery.

Des constructeurs qui retournent des objets .NET

Si le résultat d'un appel de constructeur .NET ne peut pas être converti en un type de données XPath/XQuery adéquat, le constructeur crée un objet .NET encapsulé avec un type qui est le nom de la classe retournant cet objet. Par exemple, si un constructeur pour la classe `System.DateTime` est appelé (avec `System.DateTime.new()`), alors un objet ayant le type `System.DateTime` est retourné.

Le format lexical de l'objet retourné peut ne pas être conforme au format lexical d'un type de données XPath. Dans ce cas, la valeur retournée devrait donc être : (i) convertie dans le format lexical du type de données XPath requis ; puis (ii) au type de données XPath requis.

Trois choses peuvent être réalisées avec un objet .NET créé par un constructeur :

- Il peut être réalisé dans une variable :
  ```
  <xsl:variable name="currentdate" select="date:new(2008, 4, 29)"
  xmlns:date="clitype:System.DateTime" />
  ```
- Il peut être passé à une fonction d'extension (*voir Méthode d'instance et Champs d'instance*):
  ```
  <xsl:value-of select="date:ToString(date:new(2008, 4, 29))"
  xmlns:date="clitype:System.DateTime" />
  ```
- Il peut être converti en une chaîne, nombre ou booléenne :
- ```
  <xsl:value-of select="xs:integer(data:get_Month(date:new(2008, 4, 29)))"
  xmlns:date="clitype:System.DateTime" />
  ```
  ```

.NET: Méthodes statiques et Champs statiques

Une méthode est appelée directement par son nom et en fournissant les arguments pour la méthode. Le nom utilisé dans l'appel doit être exactement conforme à une méthode statique publique dans la classe spécifiée. Si le nom de la méthode et le nombre des arguments qui ont été donnés dans l'appel de la fonction correspondent à plus d'une méthode dans une classe, alors les types des arguments fournis sont évalués pour obtenir la meilleure correspondance possible. Si une correspondance ne peut pas être trouvée sans ambiguïté, une erreur sera rapportée.

Note : Un champ dans une classe .NET est considéré être une méthode sans argument. Une propriété est appelée en utilisant la syntaxe `get_PropertyName()`.

Exemples

Un exemple XSLT montrant un appel vers une méthode avec un argument (`System.Math.Sin(arg)`):

```
<xsl:value-of select="math:Sin(30)" xmlns:math="clitype:System.Math"/>
```

Un exemple XSLT montrant un appel vers un champ (considéré une méthode sans argument) (`System.Double.MaxValue()`):

```
<xsl:value-of select="double:MaxValue()" xmlns:double="clitype:System.Double"/>
```

Un exemple XSLT montrant un appel vers une propriété (syntaxe est `get_PropertyName()`) (`System.String()`):

```
<xsl:value-of select="string:get_Length('my string')"
xmlns:string="clitype:System.String"/>
```

Un exemple XQuery montrant un appel vers une méthode avec un argument (`System.Math.Sin(arg)`):

```
<sin xmlns:math="clitype:System.Math">
   { math:Sin(30) }
</sin>
```

 © 2018 Altova GmbH

.NET: Méthodes d'instance et Champs d'instance

Une méthode d'instance a un objet .NET qui lui est passé en tant que son premier argument de l'appel de méthode. Un tel objet .NET est généralement créé en utilisant une fonction d'extension (par exemple un appel de constructeur) ou un paramètre/variable de feuille de style. Un exemple XSLT de ce type serait :

```
<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>
```

Dans l'exemple ci-dessus, un constructeur `System.DateTime` (`new(2008, 4, 29)`) est utilisé pour créer un objet .NET de type `System.DateTime`. Cet objet est créé deux fois, une fois en tant que la valeur de la variable `releasedate`, une deuxième fois en tant que le premier et seul argument de la méthode `System.DateTime.ToString()`. La méthode d'instance `System.DateTime.ToString()` est appelée deux fois, les deux fois avec le constructeur `System.DateTime` (`new(2008, 4, 29)`) en tant que son premier et seul argument. Dans une de ces instances, la variable `releasedate` est utilisée pour obtenir l'objet .NET.

Méthodes d'instance et champs d'instance

La différence entre une méthode d'instance et un champ d'instance est théorique. Dans une méthode d'instance, un objet .NET est passé directement en tant qu'un argument ; dans un champ d'instance, un paramètre ou une variable est passé à la place—bien que le paramètre ou la variable puisse contenir un objet .NET. Par exemple, dans l'exemple ci-dessus, la variable `releasedate` contient un objet .NET, et c'est cette variable qui est passée en tant que l'argument de `ToString()` dans le deuxième constructeur d'élément `date`. C'est pourquoi, l'instance `ToString()` dans le premier élément `date` est une méthode d'instance alors que le deuxième est considéré être un champ d'instance. Le résultat produit dans les deux instances, néanmoins, est le même.

Types de données : XPath/XQuery en .NET

Lorsqu'une fonction d'extension .NET est utilisée dans une expression XPath/XQuery, les types de données des arguments de la fonction sont importants pour déterminer laquelle des méthodes .NET multiples possédant le même nom est appelée.

Dans .NET, les règles suivantes sont suivies :

- S'il y a plus d'une méthode portant le même nom dans une classe, les méthodes disponibles pour la sélection sont réduites à celles qui ont le même nombre d'arguments que l'appel de fonction.
- Les types de données chaîne, nombre et booléen XPath/XQuery (*voir la liste ci-dessous*) sont convertis implicitement en un type de données .NET correspondant. Si le type XPath/XQuery fourni peut être converti en plus d'un type .NET (par exemple, `xs:integer`), puis, ce type .NET est sélectionné ce qui est déclaré pour la méthode sélectionnée. Par exemple, si la méthode .NET appelée est `fx(double)` et que le type de données fourni XPath/XQuery est `xs:integer`, alors `xs:integer` sera converti dans le type de données `double` de .NET.

La table ci-dessous recense les conversions implicites des types chaîne, nombre et booléen XPath/XQuery en types de données .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`

Les sous-types des types de données de XML Schema recensés ci-dessus (et qui sont utilisés dans XPath et Query) seront aussi convertis dans le/s type/s .NET correspondant au type ancêtre du sous-type.

Dans certains cas, il peut ne pas être possible de sélectionner la méthode .NET correcte sur la base des informations fournies. Veuillez considérer le cas suivant, par exemple.

- L'argument fourni est une valeur `xs:untypedAtomic` de 10 et est prévue pour la méthode `mymethod(float)`.
- Néanmoins, il y a une autre méthode dans la classe qui prend un argument d'un autre type de données : `mymethod(double)`.
- Puisque les noms de la méthode et le type fourni (`xs:untypedAtomic`) ont pu être convertis correctement en `float` ou `double`, il est possible que `xs:untypedAtomic` soit converti en `double` au lieu de `float`.
- Par conséquence, la méthode sélectionnée ne sera pas celle requise et peut ne pas produire le résultat attendu. Pour contourner ce problème, vous pouvez créer une méthode définie par l'utilisateur avec un nom différent et utiliser cette méthode.

 © 2018 Altova GmbH

Les types qui ne sont pas couverts dans la liste ci-dessus (par exemple `xs:date`) ne seront pas convertis et généreront une erreur.

Types de données : .NET en XPath/XQuery

Lorsqu'une méthode .NET retourne une valeur et que le type de données de la valeur est de type chaîne, numérique ou booléenne, il est converti dans le type XPath/XQuery correspondant. Par exemple, le type de données `decimal` de .NET est converti en `xsd:decimal`.

Lorsqu'un objet .NET ou un type de données différent de chaîne, numérique ou booléenne est retourné, vous pouvez assurer la conversion dans le type XPath/XQuery requis tout d'abord en utilisant une méthode .NET (par ex. `System.DateTime.ToString()`) pour convertir l'objet .NET en une chaîne. Dans XPath/XQuery, la chaîne peut être modifiée pour correspondre à la représentation lexicale du type requis puis convertie dans le type requis (par exemple, en utilisant l'expression `cast as`).

 © 2018 Altova GmbH

Fonctions XBRL pour XSLT

Les fonctions définies dans le répertoire de fonction XBRL peut être appelé depuis un contexte XSLT pour la transformation de documents d'instance XBRL. Ces fonctions XBRL sont définies dans un des deux espaces de nom :

`http://www.xbrl.org/2008/function/instance` (généralement utilisé avec le préfixe `xfi:`)
`http://www.xbrl.org/2010/function/formula` (généralement utilisé avec le préfixe `xff:`)

Ainsi, la fonction XBRL `xfi:context`, par exemple, s'étend à `http://www.xbrl.org/2008/function/instance:context` (à condition que cet espace de nom a été lié au préfixe `xfi:`).

Pour une liste complète des fonctions, veuillez consulter http://www.xbrl.org/functionregistry/functionregistry.xml.

Scripts MSXSL pour XSLT

L'élément `<msxsl:script>` contient des fonctions définies par l'utilisateur et des variables qui peuvent être appelées depuis des expressions XPath dans la feuille de style XSLT. Le `<msxsl:script>` et un élément de niveau supérieur, c'est à dire, qu'il doit être un élément enfant de `<xsl:stylesheet>` ou `<xsl:transform>`.

L'élément `<msxsl:script>` doit être dans l'espace de nom `urn:schemas-microsoft-com:xslt` (*voir exemple ci-dessus*).

Langage de script et espace de nom

Le langage de script utilisé dans le bloc est spécifié dans l'attribut `language` de l'élément `<msxsl:script>` et l'espace de nom à utiliser pour les appels de fonction depuis les expressions XPath est identifié avec l'attribut `implements-prefix` (*voir ci-dessous*).

```
<msxsl:script language="scripting-language" implements-prefix="user-namespace-
prefix">

   function-1 or variable-1
   ...
   function-n or variable-n

</msxsl:script>
```

L'élément `<msxsl:script>` interagit avec le Windows Scripting Runtime, donc seuls des langages installés sur votre machine peuvent être utilisés dans l'élément `<msxsl:script>`. **La plate-forme .NET Framework 2.0 ou plus récente doit être installée pour pouvoir utiliser les scripts MSXSL.** Par conséquent, les langages de script .NET peuvent être utilisés dans l'élément `<msxsl:script>`.

L'attribut de langage doit accepter les mêmes valeurs que l'attribut `language` dans l'élément HTML `<script>`. Si l'attribut de langage n'est pas spécifié, alors Microsoft JScript est assumé par défaut.

L'attribut `implements-prefix` prend une valeur qui est un préfixe d'un nom d'espace in-scope déclaré. Cet espace de nom sera généralement un espace de nom d'utilisateur qui a été réservé pour une librairie de fonction. Toutes les fonctions et les variables définies dans l'élément `<msxsl:script>` se trouveront dans l'espace de nom identifié par le préfixe spécifié dans l'attribut `implements-prefix`. Lorsqu'une fonction est appelée depuis une expression XPath, le nom de la fonction entièrement qualifié doit se trouver dans le même espace de nom que la définition de la fonction.

Exemple

Voici un exemple d'une feuille de stype XSLT complète qui utilise une fonction définie dans un élément `<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>
```

Types de données

Les valeurs des paramètres passés dans et hors du bloc de script sont limitées aux types de données XPath. Cette restriction ne s'applique pas aux données passées parmi les fonctions et les variables dans le bloc du script.

Assemblages

Un assemblage peut être importé dans le script en utilisant l'élément **`msxsl:assembly`**. L'assemblage est identifié par le biais d'un nom ou d'un URI. L'assemblage est importé lors de la compilation de la feuille de style. Voici une simple représentation de la manière d'utiliser l'élément `msxsl:assembly`.

```
<msxsl:script>
       <msxsl:assembly name="myAssembly.assemblyName" />
       <msxsl:assembly href="pathToAssembly" />
```

```
        ...

</msxsl:script>
```

Le nom d'assemblage peut être un nom complet comme :

```
"system.Math, Version=3.1.4500.1 Culture=neutral
PublicKeyToken=a46b3f648229c514"
```

ou un nom court comme `"myAssembly.Draw"`.

Espaces de nom

Les espaces de nom peuvent être déclarés avec l'élément **`msxsl:using`**. Cela permet aux classes d'assemblage d'être écrits dans le script sans leurs espaces de nom, ce qui vous épargne un gros travail de saisie. Voici comment l'élément `msxsl:using` est utilisé de manière à déclarer des espaces de nom.

```
<msxsl:script>
        <msxsl:using namespace="myAssemblyNS.NamespaceName" />

        ...

</msxsl:script>
```

La valeur de l'attribut `namespace` est le nom de l'espace de nom.

 © 2018 Altova GmbH

Chapitre 11

Altova LicenseServer

11 Altova LicenseServer

- **Altova LicenseServer** (ci-dessous désigné sous **LicenseServer**) fournit un emplacement central pour la gestion des licences pour les produits Altova. Les applications Altova fonctionnant dans un réseau peuvent se voit attribuer des licences par le biais du LicenseServer, donnant ainsi aux administrateurs la flexibilité de gérer et de surveiller des licences.

Version actuelle : 2.7*

** LicenseServer 2.7 peut être utilisé pour enregistrer (i) les produits logiciels Altova de version 2018r2 ou plus anciens, et les (ii) Altova MobileTogether Server de version 4.1, ou plus anciens. Il ne peut pas être utilisé pour mettre sous licence des versions de produit qui sont plus récentes que celles regroupées ici. Pour plus d'informations concernant la compatibilité, voir Mise à jour du LicenseServer.*

Processus d'obtention de la licence avec Altova LicenseServer

Pour attribuer une licence à un produit Altova par le biais d'Altova LicenseServer, veuillez suivre les étapes suivantes :

1. Lancer LicenseServer
2. Ouvrir la page de configuration LicenseServer qui est la UI Web de LicenseServer, sur Windows, Linux ou macOS.
3. Charger la/les licence/s du produit Altova que vous avez reçue/s de la part de Altova dans le pool de licences du LicenseServer. Pour ce faire, rendez-vous dans l'onglet License Pool de la page Configuration.
4. Enregistrer le/s produits Altova avec LicenseServer.
5. Attribuer des licences au/x produit/s Altova dans l'onglet Client Management de la page de Configuration.

Par la suite, les licences peuvent être contrôlées confortablement et gérées centralement avec LicenseServer. Voir la Référence de la page de Configuration pour les fonctions disponibles.

Note : La page de Configuration LicenseServer ne prend pas en charge SSL.

Versions LicenseServer et leur compatibilité avec les produits Altova

Les nouvelles versions des produits Altova peuvent uniquement être mises sous licence avec la version de LicenseServer la plus récente au moment de la publication du produit Altova. Néanmoins, les versions plus anciennes des produits Altova fonctionneront avec les versions plus récentes de LicenseServer.

Donc, si vous installez une nouvelle version d'un produit Altova et si votre version actuelle de LicenseServer n'est pas la plus récente, désinstallez cette version plus ancienne de LicenseServer et installez la toute dernière version disponible sur le site Altova. Toutes les informations d'inscription et concernant la licence contenues dans votre ancienne version de LicenseServer seront enregistrées au moment de la désinstallation sur une base de données sur votre machine et seront importées automatiquement dans la version de LicenseServer plus récente. Lorsque vous installez une version plus récente de LicenseServer, l'ancienne version sera désinstallée avant que la nouvelle version soit installée.

 © 2018 Altova GmbH

Le numéro de version du LicenseServer actuellement installé est indiqué au bas de la page de configuration de LicenseServer (tous les onglets).

Version actuelle : 2.7

À propos de cette documentation

Cette documentation est organisée dans les parties suivantes :

- Informations introductives sur les : exigences de réseau ; installation sur Windows, Linux et macOS et Altova ServiceController.
- Comment attribuer des Licences : description étape par étape comment attribuer des licences avec Altova LicenseServer.
- Référence de la page de Configuration : une description de l'interface de l'administrateur avec LicenseServer.

Dernière mise à jour: 28-05-2018

11.1 Informations de réseau

Altova LicenseServer doit être installé sur une machine de serveur accessible par tous les clients faisant fonctionner les produits Altova nécessitant une licence. Tout pare-feu aussi bien sur le client que sur le serveur doit permettre le trafic réseau depuis et vers le LicenseServer, trafic nécessaire au bon fonctionnement de LicenseServer.

Sur la machine LicenseServer, le **port 35355** est utilisé pour distribuer des licences, et donc, il doit être ouvert pour le trafic réseau avec des machines client.

Ci-après, vous trouverez les paramètres et exigences de réseau par défaut de LicenseServer:

- *Pour la distribution de licence LicenseServer :*
  Soir l'un ou les deux
  Connexion TCP IPv4 sur le port `35355`
  Connexion TCP IPv6 sur le port `35355`

Pour des tâches administratives, le LicenseServer est accédé par le biais d'une interface web qui utilise le port 8088. Le port utilisé peut être configuré pour se conformer à vos besoins.

Connexion au Serveur de licence principal sous altova.com
L'Altova LicenseServer doit pouvoir communiquer avec le Serveur de licence principal sous `altova.com` pour valider et authentifier les données relatives aux licences et pour assurer la conformité continue avec les accords de licence Altova. Cette communication a lieu par le biais de HTTPS avec le port 443. Si Altova LicenseServer, après avoir établi la vérification initiale avec le Serveur de licence principal `altova.com`, n'est pas capable de se reconnecter avec `altova.com` pour une durée de plus de 5 jours (= 120 heures), alors Altova LicenseServer ne pourra plus utiliser les produits de logiciels Altova connectés à cet Altova LicenseServer.

Toute perte de connexion de la sorte avec les serveurs principaux `altova.com` sera inscrite dans l'onglet Messages de la page de Configuration d'Altova LicenseServer. En outre, l'administrateur peut configurer Altova LicenseServer pour envoyer automatiquement un e-mail d'alerte lorsque la connexion à `altova.com` est perdue. Les paramètres des e-mails d'alerte sont disponibles dans l'onglet Paramètres de la page de Configuration.

Comment LicenseServer identifie les appareils client

Lorsqu'un produit Altova est mis sous licence avec LicenseServer, la licence est enregistrée par LicenseServer comme étant émise pour un appareil client spécifique. LicenseServer identifie l'appareil client par son nom d'hôte, que LicenseServer obtient par une recherche de serveur DNS de l'adresse IP du client. Si la recherche ne donne aucun nom d'hôte, alors l'adresse IP du client est utilisée en tant que son identifiant. Par conséquence, à chaque fois que le produit Altova est lancé sur l'appareil client, LicenseServer contrôle si l'adresse IP du client qu'elle reçoit correspond au nom d'hôte ou à l'adresse IP de tout client pour qui une licence a été émise. En cas de correspondance, les détails de la licence sont vérifiés.

 © 2018 Altova GmbH

Connexions VPN avec des adresses IP dynamiques

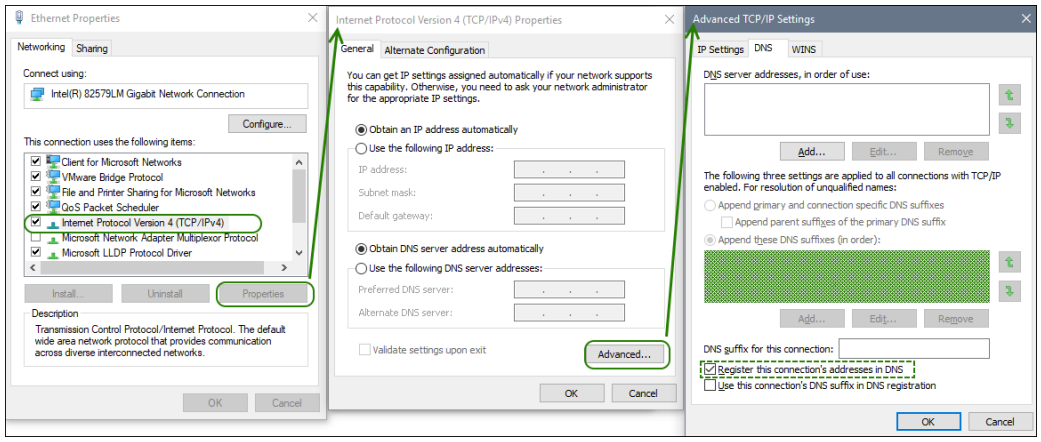
Si un appareil client se connecte à LicenseServer via un service Virtual Private Network (VPN), la connexion sera souvent établie avec une adresse IP attribuée dynamiquement. Dans ce cas, chaque nouvelle connexion établie par le client sera enregistrée par LicenseServer comme provenant d'une nouvelle adresse IP, et donc, provenant d'un client inconnu (*voir "Comment LicenseServer identifie les appareils client" ci-dessus*).

Cela entraîne les conséquences suivantes :

- Si une licence supplémentaire **est disponible** sur LicenseServer au moment de la connexion du client, une nouvelle licence sera attribuée au client (qui peut éventuellement déjà avoir une ou plusieurs licences qui lui ont été attribuées précédemment). Les licences qui ont été attribuées précédemment au client ne sont pas publiées. En conséquence, un seul client consommerait plusieurs licences, entraînant un manque de licences sur le serveur.
- Si une licence supplémentaire **n'est pas disponible** sur LicenseServer, le produit sur ce client ne pourra pas avoir de licence, même si des licences lui ont été éventuellement attribuées précédemment (pour les adresses IP que le client n'utilise plus).

Il y a deux moyens de régler ce problème :

- *Enregistrer les clients VPN avec un DNS.* La manière de procéder varie selon les systèmes d'exploitation. Sur les clients Windows 10, par exemple, vous pouvez faire la configuration avec les propriétés de la connexion de réseau du client (*voir capture d'écran ci-dessous*).



- *Attribuer une adresse IP statique à l'utilisateur VPN.* La manière de procéder varie selon les logiciels VPN. Vous obtiendrez les informations nécessaires auprès de votre fournisseur ou administrateur VPN. Vous trouverez l'exemple pour OpenVPN ici.

11.2 Mise à jour du LicenseServer

Chaque nouvelle publication des produits de serveur Altova (comme RaptorXML Server) est accompagnée par la publication d'une version plus récente de LicenseServer (la version correspondante). **Si vous souhaitez mettre à jour votre produit de serveur Altova, vous devez aussi mettre à jour votre LicenseServer** à la version de LicenseServer correspondante. Cela est dû au fait que le produit Altova de serveur mis à jour ne peut pas être enregistré avec une version plus ancienne de LicenseServer que la version correspondante. Veuillez noter, néanmoins que les version de LicenseServer sont rétrocompatibles. (Cela signifie que LicenseServer peut être utilisé pour enregistrer des produits de serveur Altova correspondants et plus anciens.)

Mettre à jour le LicenseServer comme suit :

- *Sur les système Windows* : double-cliquer sur le programme d'installation de la nouvelle version (fichier exécutable) ou appeler le programme depuis la ligne de commande. Le programme d'installation désinstallera l'ancienne version de LicenseServer et installera la nouvelle version.
- *Sur Linux et macOS*: désinstaller manuellement l'ancienne version et lancer le fichier d'installation de la nouvelle version.

Les conditions qui s'appliquent sont les mêmes que celles valides lorsque vous avez installé LicenseServer à l'origine (voir pour Windows, Linux, macOS).

 © 2018 Altova GmbH

11.3 Installation (Windows)

Altova LicenseServer peut être installé sur les systèmes Windows de deux manières :

- En tant qu'installation indépendante.
- Dans le cadre d'une installation d'un produit de serveur Altova. (Les produits de serveur Altova sont : Altova FlowForce Server, Altova MapForce Server, Altova StyleVision Server, Altova RaptorXML(+XBRL) et Altova MobileTogether Server.)

Si LicenseServer n'est pas installé sur votre système au moment de l'installation d'un produit de serveur Altova, l'option pour installer LicenseServer est sélectionnée par défaut pendant la configuration d'installation. Si LicenseServer est déjà installé, l'option pour l'installer est désélectionnée par défaut. Vous pouvez changer l'option par défaut si vous le souhaitez.

Pour plus d'informations concernant l'attribution de licences, voir la section par le biais de LicenseServer, voir la section Comment attribuer des licences.

Exigences du système

- *Windows*

  Windows 7 SP1 avec mise à jour de la plateforme, Windows 8, Windows 10

- *Windows Server*

  Windows Server 2008 R2 SP1 avec mise à jour de la plateforme ou plus récent

- Versions LicenseServer et leur compatibilité avec les produits Altova

  Les nouvelles versions des produits Altova peuvent uniquement être mises sous licence avec la version de LicenseServer la plus récente au moment de la publication du produit Altova. Néanmoins, les versions plus anciennes des produits Altova fonctionneront avec les versions plus récentes de LicenseServer.

  Donc, si vous installez une nouvelle version d'un produit Altova et si votre version actuelle de LicenseServer n'est pas la plus récente, désinstallez cette version plus ancienne de LicenseServer et installez la toute dernière version disponible sur le site Altova. Toutes les informations d'inscription et concernant la licence contenues dans votre ancienne version de LicenseServer seront enregistrées au moment de la désinstallation sur une base de données sur votre machine et seront importées automatiquement dans la version de LicenseServer plus récente. Lorsque vous installez une version plus récente de LicenseServer, l'ancienne version sera désinstallée avant que la nouvelle version soit installée.

  Le numéro de version du LicenseServer actuellement installé est indiqué au bas de la page de configuration de LicenseServer (tous les onglets).

  Version actuelle : 2.7

Le numéro de version de LicenseServer approprié pour toute version particulière d'un produit de serveur est affiché pendant le processus d'installation du produit serveur. Vous pouvez choisir

d'installer cette version du LicenseServer avec le produit de serveur ou vous pouvez installer la nouvelle version du LicenseServer séparément. Dans les deux cas, le programme d'installation désinstallera automatiquement la version précédente (si elle n'a pas déjà été désinstallée) et installera la nouvelle version.

 © 2018 Altova GmbH

11.4 Installation (Linux)

Altova LicenseServer peut être installé sur les systèmes Linux (Debian, Ubuntu, CentOS, RedHat).

Exigences du système

Linux

- CentOS 6 ou plus récent
- RedHat 6 ou plus récent
- Debian 7 ou plus récent
- Ubuntu 12.04 ou plus récent

Les bibliothèques suivantes sont exigées en tant que condition préalable pour installer et exécuter l'application. Si les paquets ci-dessous ne sont pas encore disponibles sur votre appareil Linux, vous pourrez exécuter la commande command `yum` (ou `apt-get` le cas échéant) pour les installer.

Exigé par	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

Désinstaller les anciennes versions de LicenseServer

Dans l'interface de la ligne de commande (CLI), vous pouvez vérifier si LicenseServer est installé avec la commande suivante :

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

Si LicenseServer n'est pas installé, procédez avec l'installation telle que décrite dans les étapes suivantes. Si LicenseServer est installé et que vous souhaitez installer une version plus récente, désinstallez la nouvelle version avec la commande :

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

Installer Altova LicenseServer

Sur des systèmes Linux, LicenseServer doit être installé indépendamment d'autres produits de serveur Altova. Il n'est pas compris dans les paquets d'installation des produits de serveur Altova. Télécharger Altova LicenseServer du site Internet Altova et copier le paquet dans n'importe quel

répertoire sur le système Linux.

Distribution	Installer extension
Debian	`.deb`
Ubuntu	`.deb`
CentOS	`.rpm`
RedHat	`.rpm`

Dans une fenêtre de terminal, passer au répertoire où vous avez copié le paquet Linux. Par exemple, si vous l'avez copié sur un répertoire utilisateur nommé `MyAltova` (qui est situé par exemple sur le répertoire `/home/User`), passez à ce répertoire comme suit :

```
cd /home/User/MyAltova
```

Installer LicenseServer en tant qu'utilisateur racine. Si vous ne souhaitez pas être enregistré en tant que racine, vous pouvez utiliser la commande **sudo** pour exercer temporairement des privilèges d'utilisateur racine. Installer LicenseServer avec la commande suivante :

```
[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
```

- Le package LicenseServer sera installé sous : **/opt/Altova/LicenseServer/bin**
- Un utilisateur nommé **altovalicenseserver** sera créé, qui aura les droits nécessaires pour exécuter LicenseServer. Lorsque LicenseServer est lancé en tant que daemon (ou en tant que service), il sera lancé automatiquement en tant que cet utilisateur. (Bien entendu, vous pouvez aussi (i) créer un nouvel utilisateur avec des privilèges qui permettront à cet utilisateur d'accéder à LicenseServer, ou (ii) lancer LicenseServer en tant qu'utilisateur racine (qui dispose d'un maximum de privilèges), mais aucune de ces étapes n'est requise ou recommandée parce qu'elles (i) présentent toutes certains inconvénients, et (ii) l'utilisateur **altovalicenseserver** a tous les droits nécessaires pour exécuter LicenseServer et est un utilisateur séparé et spécifié.)

  Note : Vous pouvez également exécuter LicenseServer en tant que **altovalicenseserver** dans le mode interactif. Par exemple, sur CentOS 7, vous pouvez déboguer de la manière suivante : **sudo runuser -l altovalicenseserver -c '/opt/Altova/LicenseServer/bin/licenseserver debug'**. L'inconvénient de ce mode interactif est que LicenseServer cessera de fonctionner lors de la fermeture de la session de terminal.

Informations complémentaire concernant l'exécution de LicenseServer

Veuillez noter les points suivants :

- Il est recommandé d'exécuter LicenseServer en tant que daemon (ou service). Si vous exécutez LicenseServer dans un mode interactif, il cessera de fonctionner lors

 © 2018 Altova GmbH

de la fermeture de la session de terminal.

- Lors de l'exécution de LicenseServer en tant que daemon, il est recommandé de gérer LicenseServer par le biais de la commande **`initctl`** ou **`systemctl`**. Ces deux commandes peuvent uniquement être exécutées avec des privilèges root-user. Donc, si vous l'exécutez en tant qu'utilisateur non-root, utilisez la commande **`sudo`** pour utiliser temporairement des privilèges root-user (**`sudo launchctl... et sudo systemctl...`**).
- Une fois que le LicenseServer a été lancé en tant que daemon, il sera automatiquement exécuté en tant que l'utilisateur **`altovalicenseserver`**, qui (i) a été créé par l'installateur de LicenseServer au moment de l'installation, et (ii) dispose de tous les droits nécessaires pour l'exécution de LicenseServer. Pour plus d'informations, voir la section concernant le lancement de LicenseServer

Pour plus d'informations concernant comment procéder pour l'attribution de licences, voir la section Comment attribuer des licences.

Versions LicenseServer et leur compatibilité avec les produits Altova

Les nouvelles versions des produits Altova peuvent uniquement être mises sous licence avec la version de LicenseServer la plus récente au moment de la publication du produit Altova. Néanmoins, les versions plus anciennes des produits Altova fonctionneront avec les versions plus récentes de LicenseServer.

Donc, si vous installez une nouvelle version d'un produit Altova et si votre version actuelle de LicenseServer n'est pas la plus récente, désinstallez cette version plus ancienne de LicenseServer et installez la toute dernière version disponible sur le site Altova. Toutes les informations d'inscription et concernant la licence contenues dans votre ancienne version de LicenseServer seront enregistrées au moment de la désinstallation sur une base de données sur votre machine et seront importées automatiquement dans la version de LicenseServer plus récente. Lorsque vous installez une version plus récente de LicenseServer, l'ancienne version sera désinstallée avant que la nouvelle version soit installée.

Le numéro de version du LicenseServer actuellement installé est indiqué au bas de la page de configuration de LicenseServer (tous les onglets).

Version actuelle : 2.7

11.5 Installation (macOS)

Altova LicenseServer peut être installé sur les systèmes (Mac) OS X, macOS (*voir les exigences ci-dessous*). Étant donné que vous risquez de devoir désinstaller une version précédente, le processus de désinstallation est décrit en premier.

Exigences du système

- (*Mac) OS X, macOS*

  OS X 10.10 ou plus récent

Désinstaller les anciennes versions de LicenseServer

Avant de désinstaller LicenseServer, arrêter le service avec la commande suivante :

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

Pour vérifier si le service a été arrêté, ouvrir le terminal Activity Monitor et assurez-vous que LicenseServer ne se trouve pas dans la liste.

Dans **Applications**, cliquer avec le bouton de droite sur l'icône LicenseServer et choisir **Move to Trash** (Déplacer dans la poubelle). L'application sera déplacée dans la poubelle. Néanmoins, vous devrez également supprimer l'application du dossier `usr`. Pour ce faire, utiliser la commande :

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

Installer Altova LicenseServer

Ouvrir la page de téléchargement http://www.altova.com/download.html et chercher le programme d'installation Altova LicenseServer sous "Server Software Products" pour Mac. Après avoir téléchargé le fichier d'image disque (`.dmg`), cliquer pour l'ouvrir. Cela instaure un nouveau lecteur virtuel sur votre ordinateur. Sur le lecteur virtuel, double-cliquer le fichier paquet (.pkg) et suivre les instructions à l'écran. Vous devrez accepter le contrat de licence pour pouvoir procéder à l'installation.

Veuillez noter les points suivants :

- Le paquet LicenseServer sera installé dans le dossier : **`/usr/local/Altova/LicenseServer`**
- Un utilisateur nommé **`altovalicenseserver`** sera créé, qui aura les droits nécessaires pour exécuter LicenseServer. Lorsque LicenseServer est lancé en tant que daemon (ou en tant que service), il sera lancé automatiquement en tant que cet utilisateur.

Informations complémentaire concernant l'exécution de LicenseServer

 © 2018 Altova GmbH

Veuillez noter les points suivants :

- Il est recommandé d'exécuter LicenseServer en tant que daemon (ou service). Si vous exécutez LicenseServer dans un mode interactif, il cessera de fonctionner lors de la fermeture de la session terminale.
- Lors de l'exécution de LicenseServer en tant que daemon, il est recommandé de gérer LicenseServer par le biais de la commande **`launchctl`**. Cette commande peut uniquement être exécutée avec des privilèges root-user. Donc, si vous l'exécutez en tant qu'utilisateur non-root, utilisez la commande **`sudo`** pour utiliser temporairement des privilèges root-user (**`sudo launchctl...`**).
- Une fois que le LicenseServer a été lancé en tant que daemon, il sera automatiquement exécuté en tant que l'utilisateur **`altovalicenseserver`**, qui (i) a été créé par l'installateur de LicenseServer au moment de l'installation, et (ii) dispose de tous les droits nécessaires pour l'exécution de LicenseServer. Pour plus d'informations, voir la section concernant le lancement de LicenseServer.

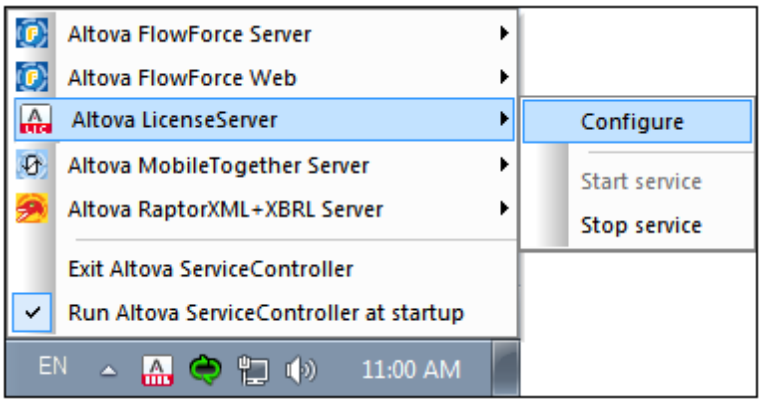
Pour éjecter le lecteur virtuel après l'installation, cliquer avec le bouton de droite et puis sélectionner **Eject**.

11.6 Altova ServiceController

Altova ServiceController (ServiceController en abrégé) est une application permettant de lancer, d'arrêter et de configurer confortablement les services Altova **sur les systèmes Windows**.

ServiceController est installé avec l'Altova LicenseServer et avec les *produits de serveur Altova installés en tant que services* (FlowForce Server, RaptorXML(+XBRL) Server, and Mobile Together Server). Il peut être démarré en cliquant sur **Start | Altova LicenseServer | Altova ServiceController**. (Cette commande est également disponible dans les dossiers de menu **Start** des *produits de serveur Altova qui sont installés en tant que services* (FlowForce Server, RaptorXML(+XBRL) Server, and Mobile Together Server).) Une fois que ServiceController a été démarré, il peut être accédé depuis la zone de notification (*capture d'écran ci-dessous*).

Pour spécifier que ServiceController démarre automatiquement lors de la connexion au système, cliquer sur l'icône **ServiceController** dans la zone de notification pour afficher le menu **ServiceController** (*capture d'écran ci-dessous*) et basculer sur la commande **Run Altova ServiceController at Startup**. (Cette commande est activée par défaut.) Pour quitter ServiceController, cliquer sur l'icône **ServiceController** dans la zone de notification et, dans le menu qui apparaît, (*voir capture d'écran ci-dessous*), cliquer sur **Exit Altova ServiceController**.



Démarrer et arrêter les services Altova

Chaque composant de service Altova installé aura une entrée dans le menu de ServiceController (*voir capture d'écran ci-dessus*). Un service Altova peut être lancé ou arrêté par le biais d'une commande sans son sous-menu ServiceController. De plus, des tâches importantes d'administration pour des services individuels peuvent être accédées par le biais du menu ServiceController. Dans la capture d'écran ci-dessus, par exemple, le service Altova LicenseServer a un sous-menu dans lequel vous pouvez choisir d'accéder à la page de Configuration de LicenseServer par le biais de la commande **Configurer**.

 © 2018 Altova GmbH

11.7 Comment attribuer des licences

Pour attribuer une licence à un produit Altova en utilisant Altova LicenseServer, procéder comme suit :

1. Lancer LicenseServer
2. Ouvrir la page de Configuration de LicenseServer qui est l'interface de l'administrateur avec LicenseServer, sur Windows, Linux ou macOS.
3. Charger la/les license/s que vous avez reçue/s de la part d'Altova dans le pool de licences de votre Altova LicenseServer. Pour ce faire, utiliser l'onglet License Pool de la page de Configuration LicenseServer.
4. Enregistrer le produit Altova avec LicenseServer. Selon le type de produit, la méthode d'enregistrement auprès de LicenseServer sera différente : respectivement soit par le biais dialogue d'activation du logiciel, l'UI Web du produit, ou par le biais de la ligne de commande. Pour plus d'informations, consulter la documentation de votre produit Altova.
5. Dans l'onglet Client Management de la page de Configuration de LicenseServer, attribuer une licence au produit Altova.

Note sur les cœurs et les licences
L'obtention de la licence des produits de serveur Altova, est basée sur le nombre de cœurs de processeurs disponibles sur la machine de produit. Par exemple, un processeur double cœur a deux cœurs, un processeur quadricœur a quatre cœurs, un processeur hexacœurs a six cœurs, etc. Le nombre de cœurs pour lesquels une licence a été délivrée pour un produit doit être supérieur ou égal au nombre de cœurs disponibles sur cette machine de serveur, que ce serveur soit une machine physique ou virtuelle. Par exemple, si un serveur a huit cœurs (un processeur octacœur), vous devrez acheter au moins une licence octacœur. Vous pouvez aussi additionner les licences pour obtenir le nombre souhaité de cœurs. Ainsi, deux licences de quadricœurs peuvent être utilisées pour un serveur octacœur au lieu d'acheter une licence octacœur.

Si vous utilisez un serveur d'ordinateur avec un grand nombre de cœurs CPU, mais ne disposez que d'un faible volume à traiter, vous pouvez aussi créer une machine virtuelle qui disposera d'un plus petit nombre de cœurs et acheter une licence pour ce nombre de cœurs. Il va de soi que la vitesse de traitement d'un tel déploiement sera moins rapide que si tous les cœurs disponibles sur le serveur étaient utilisés.

Note: Chaque licence de produit de serveur Altova peut être utilisée pour une seule machine client à la fois (l'appareil sur lequel le produit de serveur Altova est installé), même si la licence a une capacité de licence n'est pas utilisée. Par exemple, si une licence de 10-cœurs est utilisée pour une machine client qui détient 6 cœurs CPU, les 4 cœurs restants de la capacité de licence ne pourront pas être utilisés simultanément pour une autre machine client.

Note : En raison des fonctions de ses services, MobileTogether Server Advanced Edition ne sera exécuté que sur des machines comportant **deux ou plus de cœurs**.

Les licences MobileTogether Server
Les licences de MobileTogether Server sont basées sur le nombre des cœurs CPU dans l'appareil MobileTogether Server. Les licences cœur permettent à un nombre illimité d'appareils MobileTogether Client de se connecter au serveur. Néanmoins, si vous cochez la case *Limiter à une seule exécution thread*, un seul appareil mobile pourra se connecter au MobileTogether

Server à tout moment. Cela est utile pour l'évaluation et les tests à petite échelle. Veuillez noter que, dans ce cas, un second appareil se connectera à MobileTogether Server, et prendra la licence. Le premier appareil ne pourra plus se connecter et recevra un message d'erreur.

 © 2018 Altova GmbH

Lancer le LicenseServer

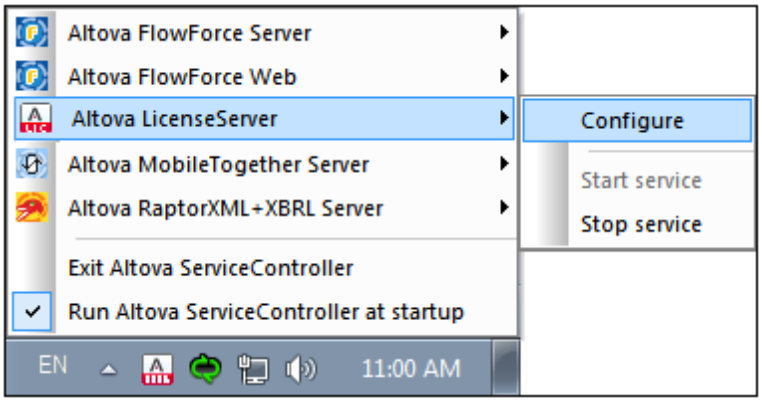
Cette section :

- Comment lancer le LicenseServer sur des systèmes Windows
- Comment lancer le LicenseServer sur des systèmes Linux
- Comment lancer le LicenseServer sur des systèmes macOS
- Note concernant la Connexion avec altova.com

Systèmes Windows

Vous pouvez lancer le LicenseServer par le biais du Altova ServiceController qui est disponible dans la zone de notification.

Tout d'abord, cliquer sur **Start | All Programs | Altova LicenseServer | Altova ServiceController** pour lancer Altova ServiceController et afficher son icône dans la zone de notification (*voir capture d'écran ci-dessous*). Si vous choisissez l'option *Run Altova ServiceController at Startup*, Altova ServiceController sera lancé au démarrage du système et son icône sera disponible dans la zone de notification à partir de ce moment.



Pour lancer LicenseServer, cliquer sur l'icône Altova ServiceController dans la zone de notification, planer au-dessus de **Altova LicenseServer** dans le menu qui s'ouvre (*voir capture d'écran ci-dessus*), puis sélectionner **Start Service** depuis le sous-menu de LicenseServer. Si LicenseServer est déjà en cours, l'option *Start Service* sera désactivée.

Systèmes Linux

Pour lancer LicenseServer en tant que service sur les systèmes Linux, exécuter la commande suivante dans une fenêtre de terminal.

```
[Debian 7]:          sudo /etc/init.d/licenseserver start
[Debian >=8]:        sudo systemctl start licenseserver
[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 vous devez arrêter LicenseServer, remplacer **start** avec **stop** dans la commande ci-dessus.)

Pour plus d'informations, voir Installation (Linux).

Systèmes macOS

Pour lancer le LicenseServer en tant que service sur les systèmes macOS, exécuter la commande suivante dans une fenêtre de terminal :

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

Si, à tout moment, vous devez arrêter LicenseServer, utiliser :

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

Pour plus d'informations, voir Installation (macOS).

Connexion au Serveur de licence principal sous altova.com

L'Altova LicenseServer doit pouvoir communiquer avec le Serveur de licence principal sous `altova.com` pour valider et authentifier les données relatives aux licences et pour assurer la conformité continue avec les accords de licence Altova. Cette communication a lieu par le biais de HTTPS avec le port 443. Si Altova LicenseServer, après avoir établi la vérification initiale avec le Serveur de licence principal `altova.com`, n'est pas capable de se reconnecter avec `altova.com` pour une durée de plus de 5 jours (= 120 heures), alors Altova LicenseServer ne pourra plus utiliser les produits de logiciels Altova connectés à cet Altova LicenseServer.

Toute perte de connexion de la sorte avec les serveurs principaux `altova.com` sera inscrite dans l'onglet Messages de la page de Configuration d'Altova LicenseServer. En outre, l'administrateur peut configurer Altova LicenseServer pour envoyer automatiquement un e-mail d'alerte lorsque la connexion à `altova.com` est perdue. Les paramètres des e-mails d'alerte sont disponibles dans l'onglet Paramètres de la page de Configuration.

 © 2018 Altova GmbH

Ouvrir la page de Config de LicenseServer (Windows)

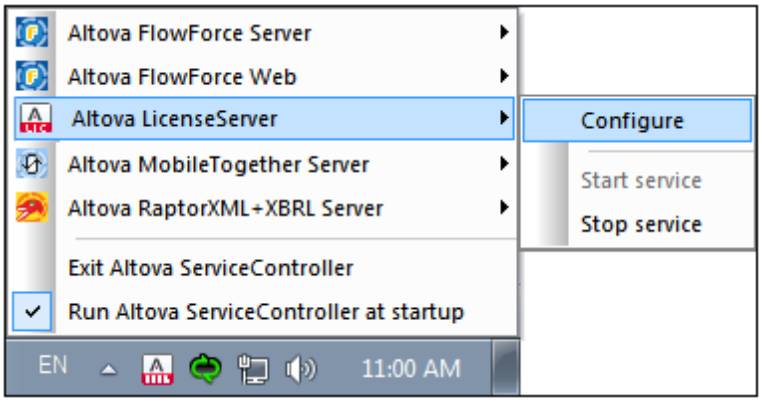
Cette section :

- Ouvrir la page de Configuration si LicenseServer se trouve sur la même machine
- Ouvrir la page de Configuration si LicenseServer se trouve sur une autre machine
- Se connecter avec le mot de passe initial
- Définir un port fixe pour la page de Configuration

Ouvrir la page de Configuration si LicenseServer se trouve sur la même machine

Sur les systèmes Windows, si LicenseServer se trouve sur la même machine, vous pouvez ouvrir la page de Configuration de LicenseServer de deux manières :

- Cliquer sur **Start | All Programs | Altova LicenseServer | LicenseServer Configuration Page**. La page de Configuration s'ouvre dans un nouvel onglet de votre navigateur Internet.
- Cliquer sur l'icône Altova ServiceController dans la zone de notification, placer la souris sur **Altova LicenseServer** dans le menu qui s'ouvre (*voir capture d'écran ci-dessous*) et choisir **Configurer** depuis le sous-menu LicenseServer



La page de Configuration s'ouvre dans une nouvelle fenêtre de navigateur et son masque de connexion s'affiche (*capture d'écran ci-dessous*).

Ouvrir la page de Configuration si LicenseServer se trouve sur une autre machine

Pour ouvrir la page de Configuration de LicenseServer à partir d'une autre machine Windows sur le réseau local (un autre que celui sur lequel LicenseServer est installé), saisir l'URL de la page de Configuration LicenseServer dans la barre d'adresse d'un navigateur et appuyer sur **Enter**. Par défaut, l'URL de la page de Configuration sera :

```
http://<serverIPAddressOrName>:8088/
```

L'URL est présente dans le code HTML de la page de Configuration. Elle est nommée

WebUI.html et se trouve sous :

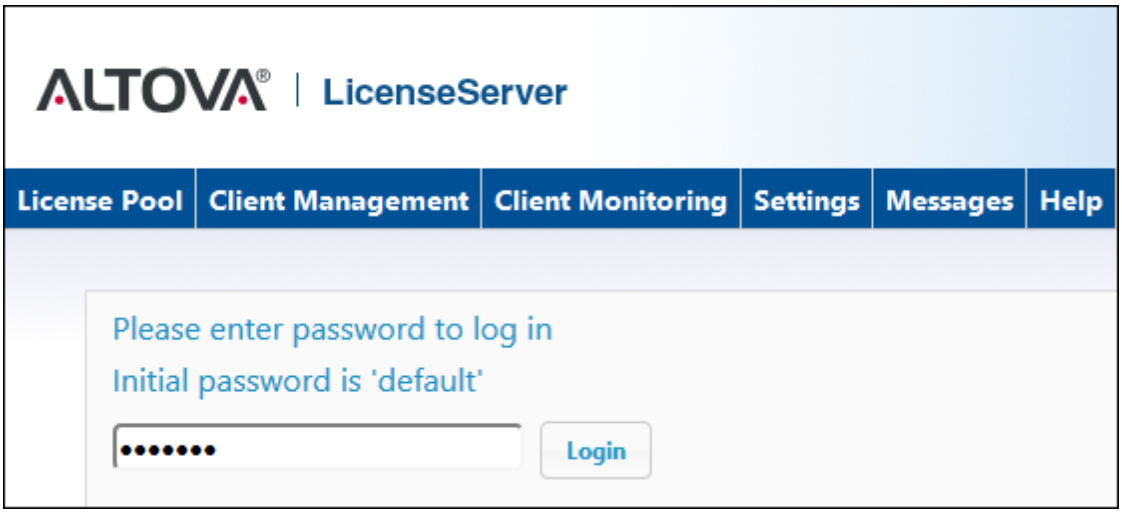
```
C:/ProgramData/Altova/LicenseServer/WebUI.html
```

Si vous avez configuré l'URL de la page de Configuration pour être généré dynamiquement (dans l'onglet Paramètres de la page de Configuration), alors une nouvelle URL sera générée à chaque fois que LicenseServer sera lancé. Vous devrez contrôler la version actuelle de **WebUI.html** pour trouver l'URL actuelle de la page de Configuration.

L'URL générée dynamiquement dans **WebUI.html** aura une forme telle que : `http://127.0.0.1:55541/optionally-an-additional-string`, et sera localisée dans la fonction `checkIfServiceRunning()` dans un script près de la fin de l'élément `<head>`. Alors que le numéro de port dans l'URL est attribué dynamiquement, la partie de l'adresse IP identifie le serveur sur lequel LicenseServer a été installé. Si vous souhaitez accéder à la page de Configuration de LicenseServer à partir d'une autre machine, assurez-vous que la partie de l'adresse IP de l'URL possède l'adresse IP correcte ou le nom du serveur sur lequel LicenseServer a été installé. Par exemple l'URL de la page de Configuration pourrait ressembler à : `http://SomeServer:55541`.

Se connecter avec le mot de passe initial

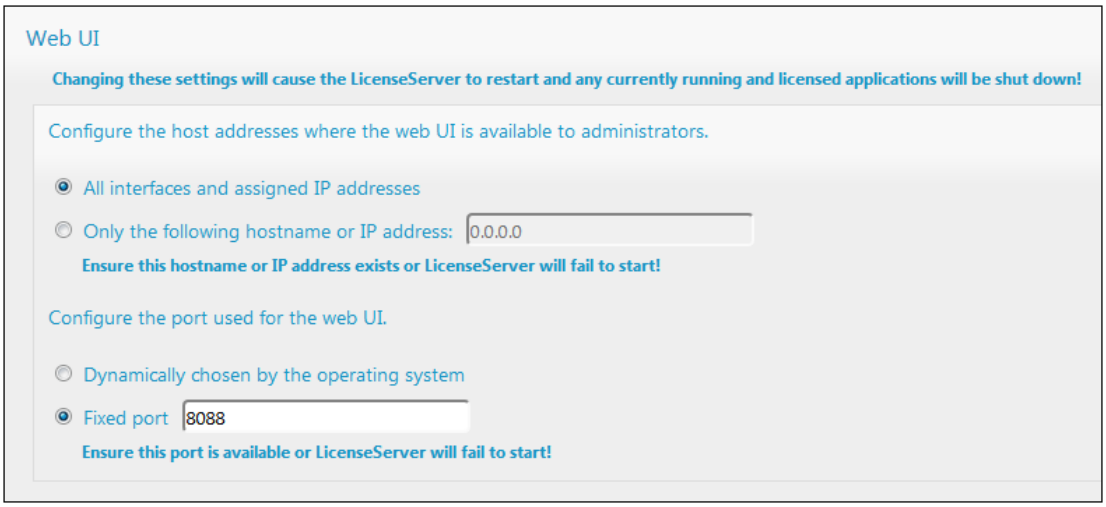
Après avoir effectué les étapes décrites ci-dessus, la page de Configuration s'ouvre et le masque de connexion s'affiche (*capture d'écran ci-dessous*). Vous pouvez vous connecter avec votre mot de passe initial `default`. Après vous être connecté, vous pourrez changer votre mot de passe dans l'onglet Paramètres.



Configurer un port fixe ou dynamique pour la page Configuration

Le port de la page de Configuration (UI Web) et donc son adresse, peut être spécifié dans la page Paramètres. Par défaut, le port est `8088`. Vous pouvez configurer tout autre port que vous

souhaitez pour la page de Configuration LicenseServer (*voir capture d'écran ci-dessous*). En alternative, vous pouvez permettre au port d'être sélectionné dynamiquement à chaque fois que LicenseServer est démarré. Dans ce cas, vous devrez trouver l'URL de la page de Configuration à partir du fichier `WebUI.html` (voir Ouvrir la page de Config de LicenseServer (Windows) et Ouvrir la page de Config de LicenseServer (Linux), et Ouvrir la page de Config de LicenseServer (macOS)).



L'avantage d'un port fixe est que la page URL est connue à l'avance et peut donc être accédée facilement. Si le port est attribué dynamiquement, la partie port de l'URL devra être consultée dans le fichier `WebUI.html` à chaque fois que LicenseServer est lancé.

Ouvrir la page de Config de LicenseServer (Linux)

Cette section :

- Ouvrir la page de Configuration pour la première fois avec l'URL retournée
- URL de la page de Configuration de LicenseServer
- Se connecter avec le mot de passe initial
- Configurer un port fixe pour la page de Configuration

Ouvrir la page de Configuration pour la première fois avec l'URL retournée

Sur les systèmes Linux, lorsque vous enregistrez votre produit de serveur Altova auprès de LicenseServer par le biais de la CLI, l'URL de la page de Configuration de LicenseServer est retournée. Lors de l'ouverture de l'URL dans un navigateur, vous serez invité à lire et à accepter le contrat de licence. Après avoir accepté le contrat de licence, le masque de connexion de la page de Configuration s'affiche (*capture d'écran ci-dessous*).

Note : Les produits de bureautique Altova sont uniquement disponibles pour Windows.

URL de la page de Configuration de LicenseServer

Pour ouvrir la page de Configuration LicenseServer à tout moment, saisie son URL dans la barre d'adresse d'un navigateur et appuyer sur **Entrée**. Par défaut, l'URL de la page de Configuration sera :

```
http://<serverIPAddressOrName>:8088/
```

L'URL se trouve dans le code HTML de la page de Configuration elle-même. Elle est nommée **`webUI.html`** et se trouve sous :

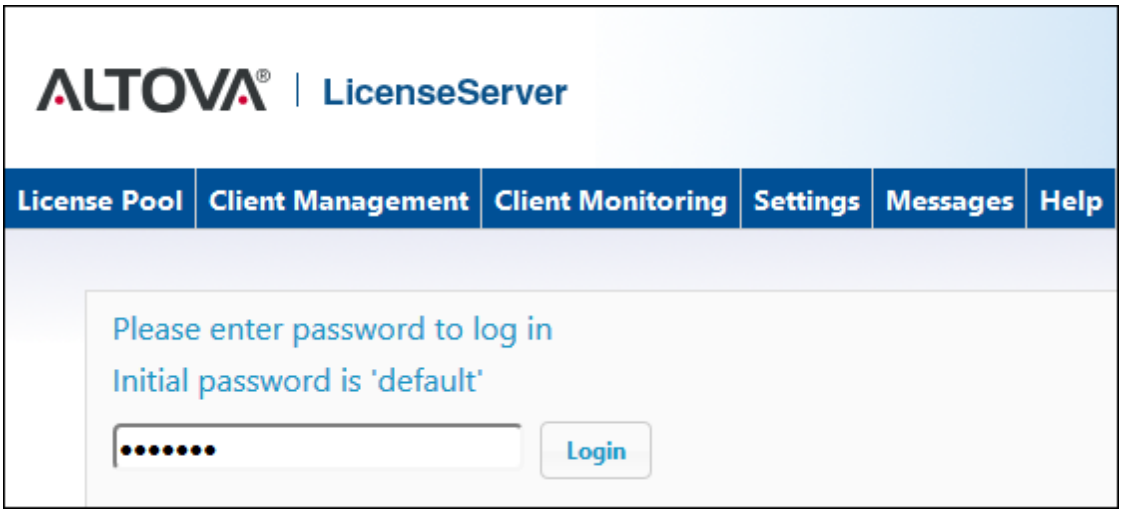
```
/var/opt/Altova/LicenseServer/webUI.html
```

Si vous avez configuré l'URL de la page de Configuration pour être générée dynamiquement (dans l'onglet Paramètres de la page de Configuration), alors une nouvelle URL sera générée à chaque fois que LicenseServer sera lancé. Vous devrez contrôler la version actuelle de **`WebUI.html`** pour trouver l'URL actuelle de la page de Configuration.

L'URL générée dynamiquement dans **`WebUI.html`** aura une forme telle que : `http://127.0.0.1:55541/`*`optionally-an-additional-string`*, et sera localisée dans la fonction `checkIfServiceRunning()` dans un script près de la fin de l'élément `<head>`. Alors que le numéro de port dans l'URL est attribué dynamiquement, la partie de l'adresse IP identifie le serveur sur lequel LicenseServer a été installé. Si vous souhaitez accéder à la page de Configuration de LicenseServer depuis une autre machine, assurez-vous que la partie de l'adresse IP de l'URL possède l'adresse IP correcte ou le nom du serveur sur lequel LicenseServer a été installé. Par exemple, l'URL de la page de Configuration pourrait ressembler à : `http://MyServer:55541`.

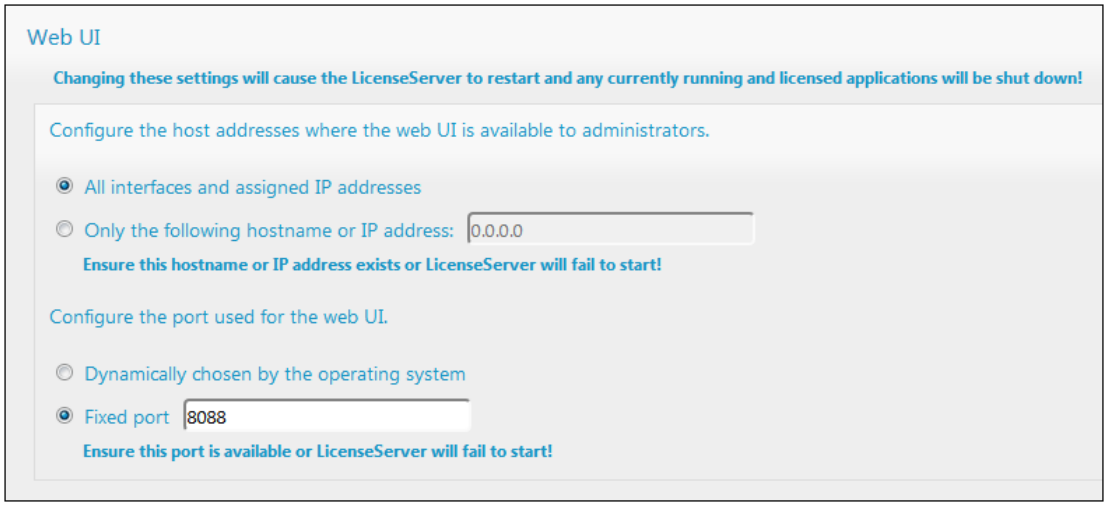
Se connecter avec le mot de passe initial

Après avoir effectué les étapes décrites ci-dessus, la page de Configuration s'ouvre et le masque de connexion s'affiche (*capture d'écran ci-dessous*). Vous pouvez vous connecter avec votre mot de passe initial `default`. Après vous être connecté, vous pourrez changer votre mot de passe dans l'onglet Paramètres.



Configurer un port fixe ou dynamique pour la page Configuration

Le port de la page de Configuration (UI Web) et donc son adresse, peut être spécifié dans la page Paramètres. Par défaut, le port est `8088`. Vous pouvez configurer tout autre port que vous souhaitez pour la page de Configuration LicenseServer (*voir capture d'écran ci-dessous*). En alternative, vous pouvez permettre au port d'être sélectionné dynamiquement à chaque fois que LicenseServer est démarré. Dans ce cas, vous devrez trouver l'URL de la page de Configuration à partir du fichier `WebUI.html` (voir Ouvrir la page de Config de LicenseServer (Windows) et Ouvrir la page de Config de LicenseServer (Linux), et Ouvrir la page de Config de LicenseServer (macOS)).



L'avantage d'un port fixe est que la page URL est connue à l'avance et peut donc être accédée facilement. Si le port est attribué dynamiquement, la partie port de l'URL devra être consultée dans le fichier `WebUI.html` à chaque fois que LicenseServer est lancé.

 © 2018 Altova GmbH

Ouvrir la page de Config de LicenseServer (macOS)

Cette section :

- Ouvrir la page de Configuration pour la première fois avec l'URL retournée
- URL de la page de Configuration de LicenseServer
- Se connecter avec le mot de passe initial
- Configurer un port fixe pour la page de Configuration

Ouvrir la page de Configuration pour la première fois avec l'URL retournée

Sur les systèmes macOS, lorsque vous enregistrez votre produit de serveur Altova auprès de LicenseServer par le biais de la CLI, l'URL de la page de Configuration de LicenseServer est retournée. Lors de l'ouverture de l'URL dans un navigateur, vous serez invité à lire et à accepter le contrat de licence. Après avoir accepté le contrat de licence, le masque de connexion de la page de Configuration s'affiche (*capture d'écran ci-dessous*).

Note : Les produits de bureautique Altova sont uniquement disponibles pour Windows.

URL de la page de Configuration de LicenseServer

Pour ouvrir la page de Configuration LicenseServer à tout moment, saisir son URL dans la barre d'adresse d'un navigateur et appuyer sur **Entrée**. Par défaut, l'URL de la page de Configuration sera :

```
http://<serverIPAddressOrName>:8088/
```

L'URL se trouve dans le code HTML de la page de Configuration elle-même. Elle est nommée **`webUI.html`** et se trouve sous :

```
/var/Altova/LicenseServer/webUI.html
```

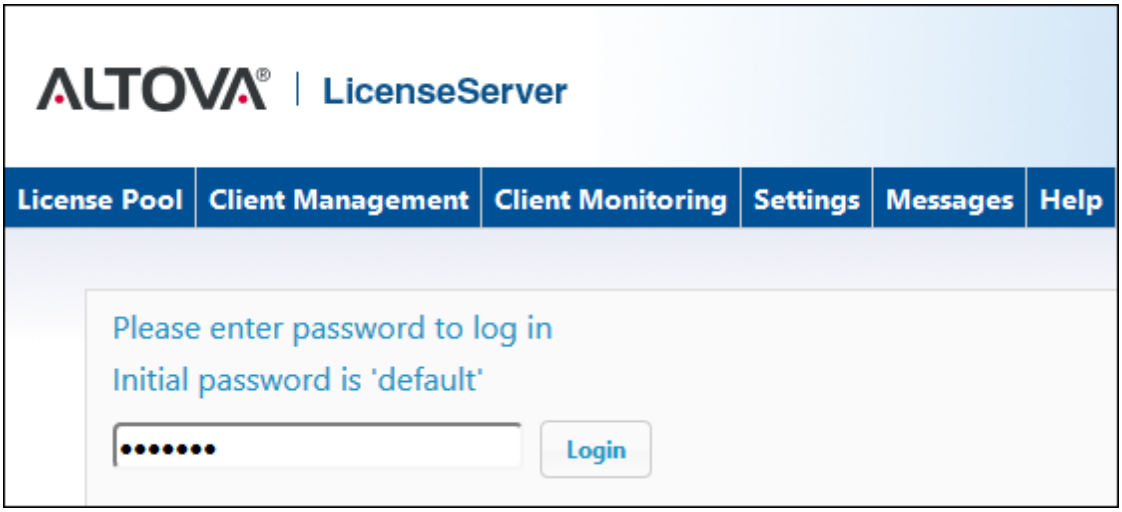
Si vous avez configuré l'URL de la page de Configuration pour être générée dynamiquement (dans l'onglet Paramètres de la page de Configuration), alors une nouvelle URL est générée à chaque fois que LicenseServer est lancé. Vous devrez contrôler la version actuelle de **`WebUI.html`** pour trouver l'URL actuelle de la page de Configuration.

L'URL générée dynamiquement dans **`WebUI.html`** aura une forme telle que : `http://127.0.0.1:55541/optionally-an-additional-string`, et sera localisée dans la fonction `checkIfServiceRunning()` dans un script près de la fin de l'élément `<head>`. Alors que le numéro de port dans l'URL est attribué dynamiquement, la partie de l'adresse IP identifie le serveur sur lequel LicenseServer a été installé. Si vous souhaitez accéder à la page de Configuration de LicenseServer depuis une autre machine, assurez-vous que la partie de l'adresse IP de l'URL possède l'adresse IP correcte ou le nom du serveur sur lequel LicenseServer a été installé. Par exemple, l'URL de la page de Configuration pourrait ressembler à : `http://MyServer:55541`

Note : La page de Configuration peut aussi être accédée directement depuis l'icône **Finder | Applications | Altova License Server**.

Se connecter avec le mot de passe initial

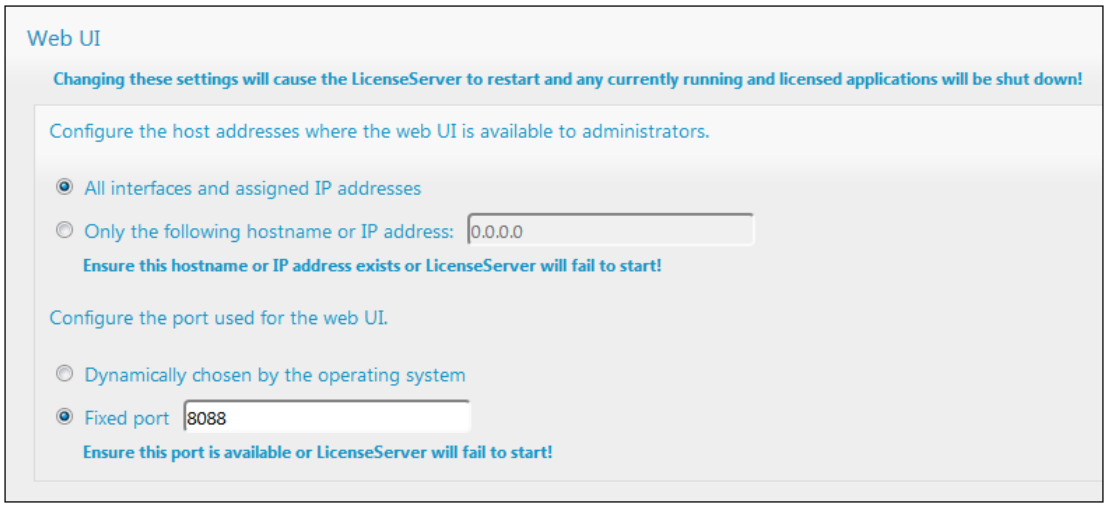
Après avoir effectué les étapes décrites ci-dessus, la page de Configuration s'ouvre et le masque de connexion s'affiche (*capture d'écran ci-dessous*). Vous pouvez vous connecter avec votre mot de passe initial `default`. Après vous être connecté, vous pourrez changer votre mot de passe dans l'onglet Paramètres.



Configurer un port fixe ou dynamique pour la page Configuration

Le port de la page de Configuration (UI Web) et donc son adresse, peut être spécifié dans la page Paramètres. Par défaut, le port est `8088`. Vous pouvez configurer tout autre port que vous souhaitez pour la page de Configuration LicenseServer (*voir capture d'écran ci-dessous*). En alternative, vous pouvez permettre au port d'être sélectionné dynamiquement à chaque fois que LicenseServer est démarré. Dans ce cas, vous devrez trouver l'URL de la page de Configuration à partir du fichier `WebUI.html` (voir Ouvrir la page de Config de LicenseServer (Windows) et Ouvrir la page de Config de LicenseServer (Linux), et Ouvrir la page de Config de LicenseServer (macOS)).

 © 2018 Altova GmbH



L'avantage d'un port fixe est que la page URL est connue à l'avance et peut donc être accédée facilement. Si le port est attribué dynamiquement, la partie port de l'URL devra être consultée dans le fichier `WebUI.html` à chaque fois que LicenseServer est lancé.

Charger des licences sur LicenseServer

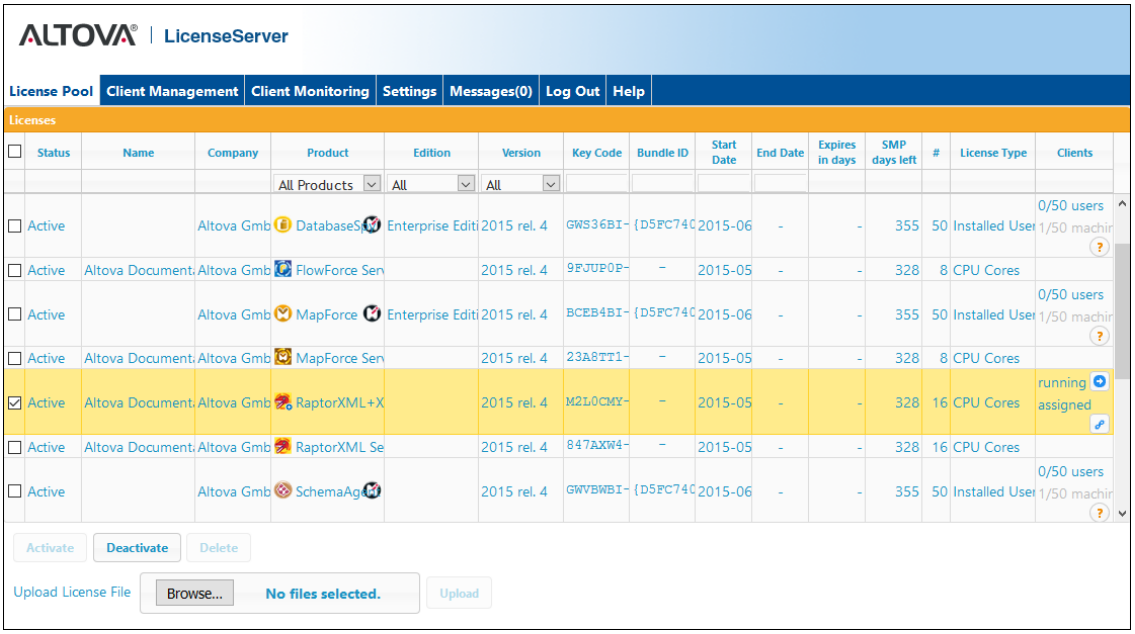
Cette section :

- Charger un fichier de licence dans le pool de licences de LicenseServer
- Statut de la licence
- Activer les licences que vous souhaitez utiliser
- Étapes suivantes

Charger un fichier de licence dans le pool de licences de LicenseServer

Après avoir obtenu un fichier de licence (extension de fichier `.altova_licenses)` de la part d'Altova, enregistrer ce fichier dans un endroit approprié puis le charger sur Altova LicenseServer. Chaque fichier de licence peut contenir une ou plusieurs licences, cela dépend de votre achat. Lorsque vous chargez un fichier de licence (un fichier `.altova_licenses`), toutes les licences qui s'y trouvent seront chargées dans le pool de licences du LicenseServer et pourront être attribuées à un produit Altova qui a été enregistré avec ce LicenseServer. Toutes les licences chargées, qu'il s'agisse d'une seule licence ou de plusieurs - et pour tous les produits Altova, sont rassemblées dans un pool de licences sur LicenseServer. Ce pool de licences est affiché dans l'onglet de License Pool de la page de Configuration LicenseServer (*capture d'écran*).

Les fichiers de licence sont chargés sur le LicenseServer en utilisant la fonction Upload de l'onglet License Pool (*voir capture d'écran*).



Les fichiers de licence sont chargés sur le LicenseServer à l'aide de la fonction *Upload License File* située sur l'onglet License Pool (*voir capture d'écran ci-dessus*). Cliquer sur le bouton **Browse** et choisir le fichier de licence que vous souhaitez. (Le fichier de licence à charger est le fichier que vous avez reçu en tant que pièce jointe dans votre e-mail de License de la part d'Altova ; il possède une extension de fichier `.altova_licenses)` . Le fichier de licence apparaîtra dans le champ de texte du Fichier de chargement de licence et le bouton **Upload** sera activé. Cliquer sur

le bouton **Upload** pour charger le fichier de licence. Toutes les licences dans le fichier sont chargées et affichées dans l'onglet License Pool. La capture d'écran ci-dessus montre plusieurs licences, chargées à partir de plusieurs fichiers de licence.

Statut de la licence

Les valeurs de statut de la licence sont les suivantes :

- *Activating :* (En cours d'Activation) lorsqu'une licence est chargée dans le pool de licences de LicenseServer, le serveur transmettra les données relatives aux licences au serveur de licences principal `altova.com` pour valider, authentifier et activer la licence fournie. Cela est nécessaire pour assurer la conformité avec les contrats de licence Altova. Au cours de cette activation initiale et de transaction d'authentification, qui dure généralement entre 30 secondes et quelques minutes (selon votre connexion Internet, la vitesse et le trafic réseau général), le statut de la licence indiqué sera *Activating...*.

- *Failed Verification :* (Échec de la vérification) si une connexion avec le serveur de licence principal `altova.com` ne peut pas être établie, le statut de la licence dans le pool affiché sera *Failed Verification*. Si cela se produit, vérifier votre connexion Internet et vos règles de pare-feu pour assurer que LicenseServer est en mesure de communiquer avec le serveur de licence principal `altova.com`.

- *Active :* (Activé) une fois que la licence a été authentifiée et activée, le statut passera à *Active*.

- *Inactive :* (Désactivé) si une licence a été vérifiée mais qu'elle est présente sur un autre LicenseServer dans le réseau, le statut affiché sera *Inactive*. Un statut *Inactive* apparaît aussi lorsqu'une licence est désactivée manuellement dans le pool de licences par l'administrateur.

- *Pending :* si la date de démarrage d'une licence se trouve dans l'avenir, la licence sera affichée en tant que *Pending.* Le statut passera à *Active* à 00:00 à la date configurée. Une licence *Pending* peut être attribuée à un produit et garantit que la licence d'un produit se poursuive sans interruption avant qu'une licence existante expire. Le passage à la nouvelle licence est conçu pour être une transition lisse sans interrompre les traitements de client.

- *Blocked :* (Bloqué) si une licence affiche *Blocked* dans le pool de licences, il y a eu un problème d'authentification dans la licence et le serveur de licences principal `altova.com` n'a pas donné son autorisation à LicenseServer d'utiliser cette licence. Cela pourrait être le résultat d'une violation d'un contrat de licence, d'une utilisation abusive d'une licence ou d'autres problèmes de conformité. Une fois le problème réglé, supprimer, recharger et réactiver la licence. Veuillez voir la table ci-dessous pour des informations complémentaires.

Ces statuts sont résumés dans le tableau ci-dessous :

Statut	Signification
Activating...	Lors du chargement, l'information de licence est envoyée à `altova.com` à

	des fins de vérification. Rafraîchir le navigateur pour consulter le statut mis à jour. La vérification et l'activation peuvent prendre quelques minutes.
Failed Verification	Impossible d'établir une connexion avec `altova.com`. Après avoir établi la connexion, soit redémarrer le service, soit activer la licence (avec le bouton **Activate**).
Active	La vérification a été effectuée avec succès, la licence est activée.
Inactive	La vérification a été effectuée avec succès, mais la licence se trouve aussi sur un autre LicenseServer dans le réseau. Les licences peuvent être rendues inactives avec le bouton **Deactivate**.
Pending	La date de démarrage d'une licence *Pending* se trouve dans l'avenir et passera au mode *Active* à cette date. Elle peut être attribuée à un produit et garantit le renouvellement automatique de la licence d'un produit avant que la licence existante expire.
Blocked	La vérification a échoué. La licence est invalide et elle est bloquée. Une fois que le problème responsable du blocage de la licence a été résolu, supprimer la licence, la recharger et la réactiver. Chaque fois qu'une licence est chargée, elle prendra contact avec le serveur maître Altova, qui lancera une nouvelle vérification. Si vous ne la rechargez pas, la vérification de la licence devra attendre jusqu'à la prochaine communication prévue avec le serveur maître Altova et cela peut durer jusqu'à 24 heures. Veuillez contacter votre Support Altova si le problème persiste.

Note : Après avoir envoyé une licence à `altova.com` à des fins de vérification, le navigateur doit être rafraîchi pour voir le statut mis à jour. La vérification et l'activation peuvent prendre quelques minutes.

Note : Si une connexion à `altova.com` n'a pas pu être établie, le statut affichera *Failed Verification*. Après avoir établi une connexion, soit redémarrer le service, soit essayer d'activer la licence avec le bouton **Activate**.

Note : Lorsqu'une licence affiche un statut *Inactive* ou *Blocked*, un message d'explication du statut sera également ajouté au journal de Messages.

Seule une licence active ou pending peut être attribuée à une installation de produit. Une licence inactive pourra être activée ou supprimée du pool de licences. Si une licence est supprimée du pool de licences, elle pourra être chargée à nouveau dans le pool en chargeant le fichier de licence le contenant. Lorsqu'un fichier de licence est mis à jour, seules les licences ne se trouvant pas déjà dans le pool seront chargées dans le pool. Pour activer, désactiver ou supprimer une licence, la sélectionner et cliquer sur le bouton **Activate**, **Deactivate**, ou **Delete**, respectivement.

Activer la/les licence/s que vous souhaitez utiliser

Avant de pouvoir attribuer une licence à un produit Altova, celle-ci doit être activée. Veuillez donc vous assurer qu'elle ait été activée. Si elle n'est pas activée, la sélectionner et cliquer sur **Activate**.

 © 2018 Altova GmbH

Étapes suivantes

Après avoir chargé le fichier de licence sur LicenseServer et d'avoir contrôlé que la licence que vous souhaitez est activée, procédez à l'étape suivante :

1. Enregistrer le produit de serveur Altova (FlowForce Server, MapForce Server, StyleVision Server) avec LicenseServer. (Si vous avez déjà enregistré le produit avant de charger le fichier de licence, vous pouvez maintenant commencer à attribuer des licences.)
2. Attribuer une licence au produit Altova qui a été enregistré avec le LicenseServer.

Enregistrer un/des produits

Avant de pouvoir attribuer une licence à un produit Altova, vous devez enregistrer l'installation de produit avec LicenseServer. L'enregistrement s'effectue à partir du produit Altova et le processus diffère selon le type de produit.

- *Produits de bureau :* enregistrement via le dialogue d'activation de logiciel du produit.
- *Produits de serveur qui détiennent une UI Web :* l'enregistrement de FlowForce Server et de MobileTogether Server s'effectue par le biais de l'onglet *Setup* de l'UI Web ou le CLI du produit.
- *Produits de serveur qui détiennent **pas** d'UI Web :* l'enregistrement de MapForceServer, RaptorXML(+XBRL) Server et StyleVisionServer s'effectue par le biais du CLI de ces produits. Vous aurez besoin du nom du serveur ou de l'adresse IP sur la machine sur laquelle LicenseServer est installé afin de procéder à l'enregistrement.

Cette section décrit comment enregistrer des produits de serveur Altova différents :

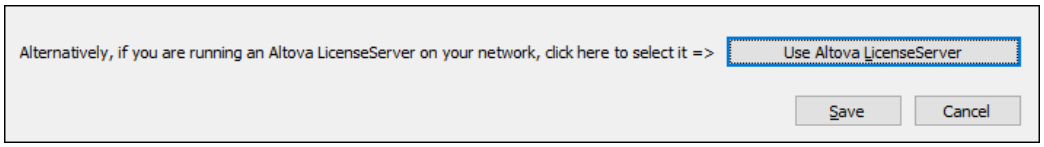
- Enregistrer les produits de bureau Altova
- Enregistrer FlowForce Server
- Enregistrer MapForce Server
- Enregistrer MobileTogether Server
- Enregistrer RaptorXML(+XBRL) Server
- Enregistrer StyleVision Server

 © 2018 Altova GmbH

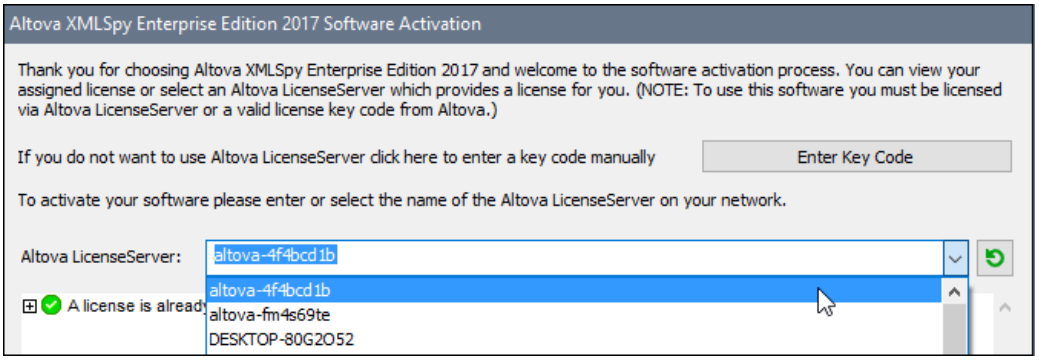
Enregistrer les produits de bureau Altova

Pour enregistrer un produit de bureau Altova avec un Altova LicenseServer, procéder comme suit :

1. Sélectionner la commande de menu **Help | Software Activation** pour vous rendre dans le dialogue d'Activation de logiciel. Vous pouvez activer votre logiciel soit (i) par le biais d'Altova LicenseServer, soit (ii) en saisissant les détails du code-clé de votre produit. Dans cette documentation, nous nous limitons à la description de la licence par le biais d'Altova LicenseServer.
2. Pour licencier votre produit Altova par le biais de LicenseServer, cliquer sur **Utiliser Altova LicenseServer** (situé en bas du dialogue; *voir capture d'écran*)



3. Le dialogue passe maintenant au mode d'activation de LicenseServer (*capture d'écran ci-dessous*). Dans la liste de choix *Altova LicenseServer*, sélectionner un LicenseServer depuis la liste déroulante.



Une fois la connexion vers le LicenseServer sélectionné a été effectuée, le produit est immédiatement enregistré avec le LicenseServer sélectionné, et, dans l'onglet Client Management, le produit est affiché dans la liste de produit de l'appareil de ce client.

Désenregistrer un produit de bureau

Afin de désenregistrer un produit de bureau, rendez-vous sur l'onglet Client Management du LicenseServer et, dans le panneau *Product licensing* situé à droite, cliquez sur le bouton **Unregister Product** du produit en question.

Enregistrer FlowForce Server

Cette section :

- Méthodes d'enregistrement de FlowForce Server avec LicenseServer
- Accéder à la page de configuration de FlowForce Server (Windows)
- Accéder à la page de configuration de FlowForce Server (Linux)
- Enregistrer FlowForce Server par le biais de la page de configuration
- Enregistrer FlowForce Server par le biais de la CLI FlowForce (Windows)
- Enregistrer FlowForce Server par le biais de la CLI FlowForce (Linux)
- Étapes suivantes

Méthodes d'enregistrement de FlowForce Server

FlowForce Server peut être enregistré avec LicenseServer en utilisant une des méthodes suivantes :

- Par le biais de la page de configuration FlowForce Server
- Par le biais de la CLI FlowForce (Windows)
- Par le biais de la CLI FlowForce (Linux)

Accéder à la page de configuration de FlowForce Server (Windows)

La page de configuration de FlowForce Server peut être accédée par l'une des manières suivantes :

- Par le biais du menu **Start** :
  Start | Altova FlowForce Server 2018 | FlowForce Server Setup Page

- Par le biais de Altova ServiceController : cliquer sur l'icône ServiceController dans la zone de notification. Dans le menu qui s'ouvre, sélectionner *Altova FlowForce Web | Setup*.

La page de configuration de FlowForce Server s'ouvre (*capture d'écran ci-dessous*).

Accéder à la page de configuration de FlowForce Server (Linux)

Après avoir installé FlowForce Server sur Linux (voir la documentation utilisateur de FlowForce Server pour plus d'informations concernant l'installation), lancer FlowForce Web Server en tant que service avec la commande suivante :

```
sudo /etc/init.d/flowforcewebserver start
```

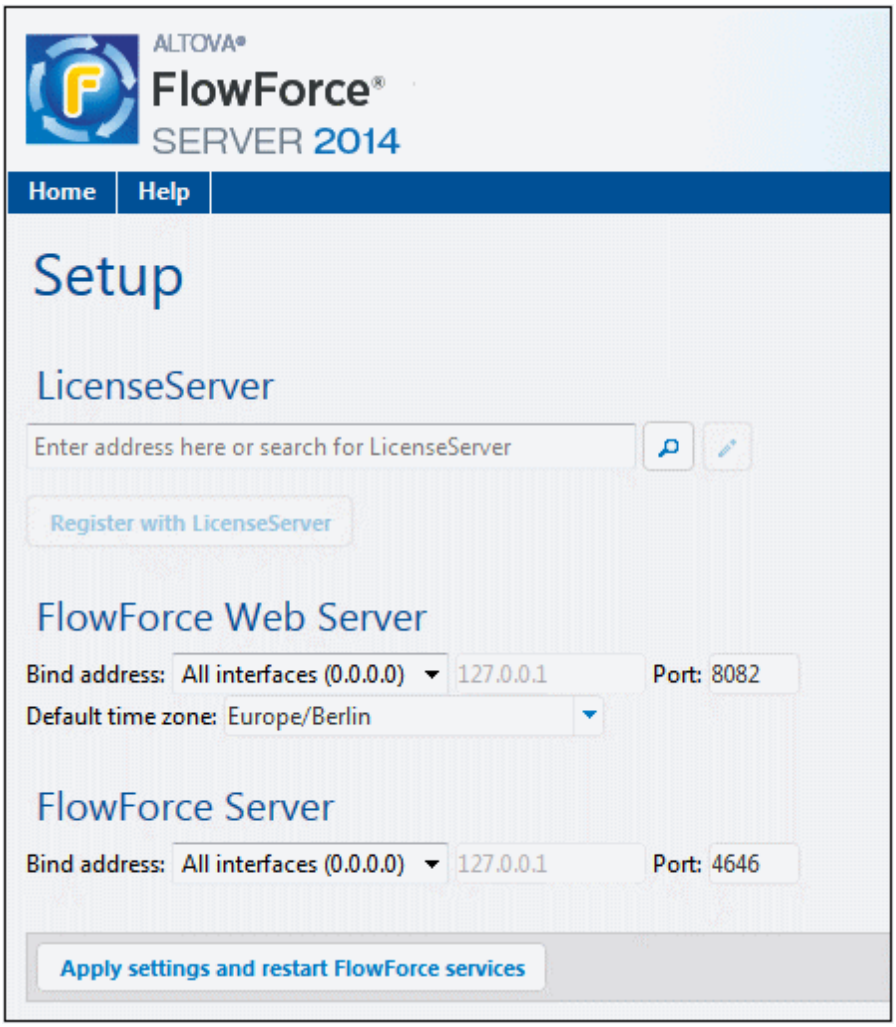
Un message contenant l'URL de la configuration FlowForce Server apparaît dans la fenêtre de terminal :

```
FlowForceWeb running on http://127.0.1.1:3459/setup?key=52239315203
```

Saisir l'URL dans le champ d'adresse d'un navigateur et appuyer sur **Enter** pour accéder à la page de configuration de FlowForce Server (*capture d'écran ci-dessous*).

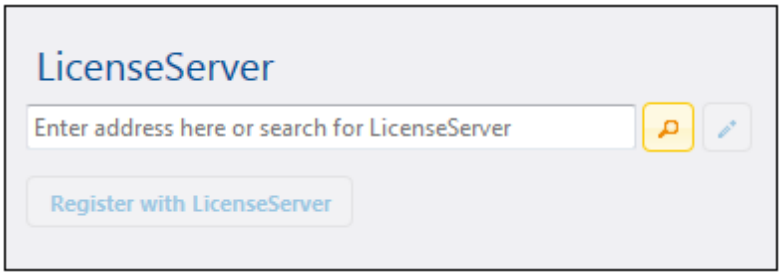
Enregistrer FlowForce Server par le biais de la page de configuration

Dans la page de configuration (*capture d'écran ci-dessous*) (ci-dessus, vous trouverez une description de l'accès), le champ LicenseServer spécifie l'utilisation de Altova LicenseServer pour l'enregistrement.

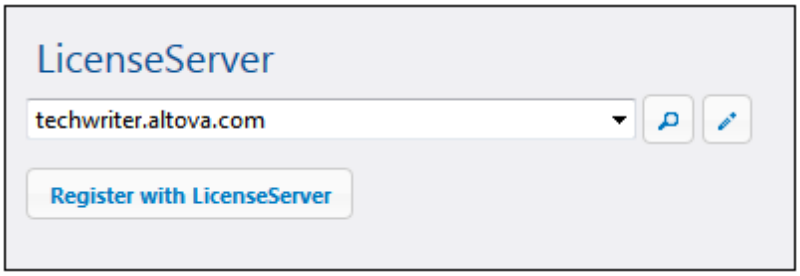


Le LicenseServer peut être spécifié d'une des deux manières disponibles.

- Vous pouvez chercher les Altova LicenseServers actuellement disponibles dans le réseau, donc ceux en cours de fonctionnement actuellement. Pour ce faire, cliquer sur le bouton **Search for Altova LicenseServers** (*marqué en jaune dans la capture d'écran ci-dessous*).



La recherche retourne une liste de Altova LicenseServers disponibles sur le réseau. Un LicenseServer sera sélectionné (*capture d'écran ci-dessous*) et les autres seront disponibles dans la liste déroulante de la liste de choix. Sélectionner le LicenseServer sur lequel votre licence de FlowForce est stockée.

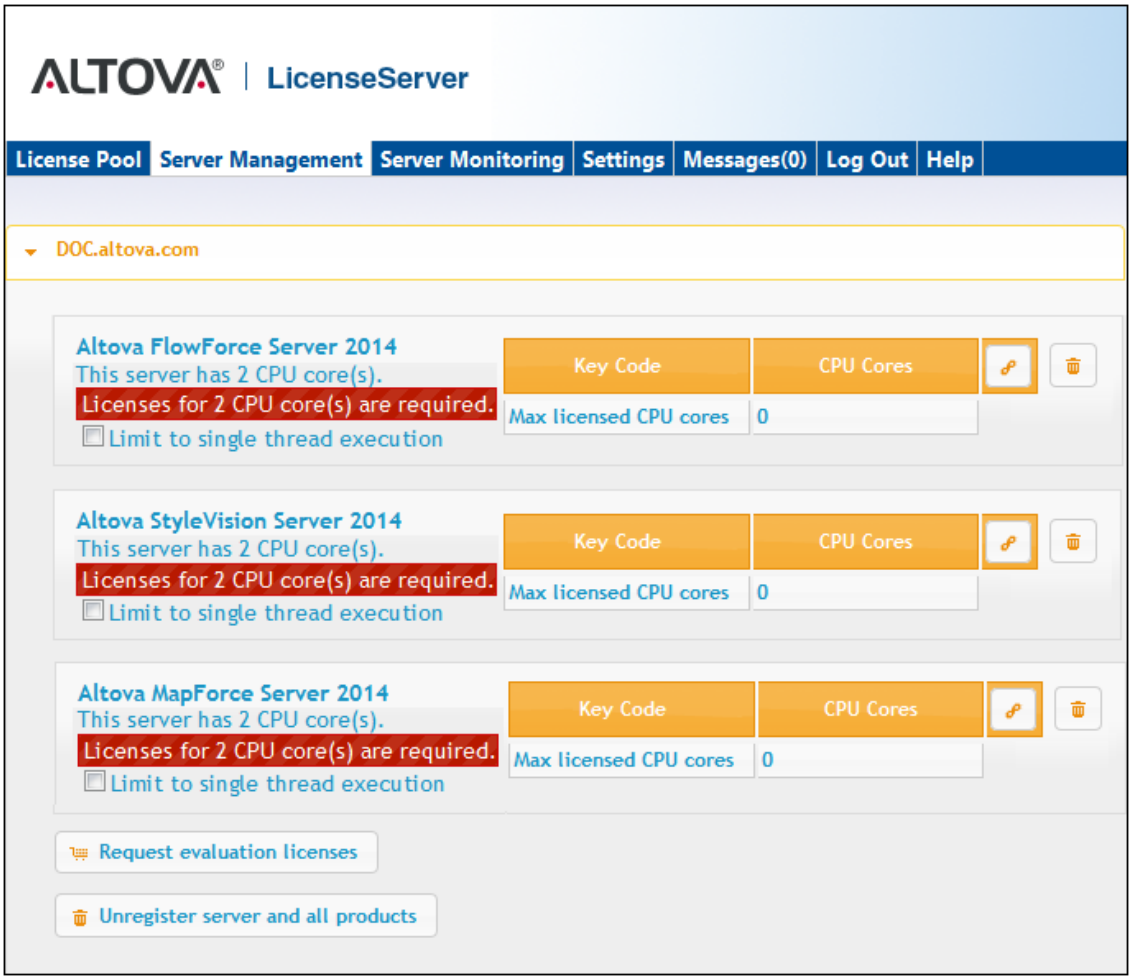


- En alternative, vous pouvez saisir l'adresse du LicenseServer dans le champ de LicenseServer. Si les LicenseServers actuellement en cours de fonctionnement sont disponibles sous la forme d'une liste déroulante, vous devez cliquer sur le bouton **Manually Enter Address** pour pouvoir saisir une adresse dans le champ LicenseServer.

Après avoir spécifié le LicenseServer, cliquer **Register with LicenseServer**. L'application de serveur Altova sera enregistrée avec le LicenseServer spécifié et cette page de Configuration de LicenseServer s'ouvrira dans un navigateur. L'onglet Client Management sera activé (*capture d'écran ci-dessous*).

Note : Vous devrez éventuellement permettre l'affichage de fenêtres pop-up pour permettre l'affichage de la page de Configuration LicenseServer.

 © 2018 Altova GmbH



Dans la capture d'écran ci-dessus, trois produits Altova ont été enregistrés avec Altova LicenseServer sous `DOC.altova.com`. La section suivante décrit comment attribuer des licences, Attribuer des licences aux produits enregistrés.

Enregistrer FlowForce Server par le biais de la CLI FlowForce (Windows)

Sur les machines Windows, FlowForce Server peut aussi être enregistré avec un Altova LicenseServer sur votre réseau par le biais de la ligne de commande (CLI) en utilisant la commande `licenseserver` :

```
FlowForceServer licenseserver Server-Or-IP-Address
```

Par exemple, si LicenseServer marche sur `http://localhost:8088`, enregistrer FlowForce Server avec :

```
FlowForceServer licenseserver localhost
```

Si FlowForce Server a été installé avec d'autres produits de serveur Altova, l'enregistrement de FlowForce Server enregistrera aussi automatiquement les produits de serveur Altova. Une fois avoir enregistré FlowForce Server avec succès, vous pouvez vous rendre sur LicenseServer et attribuer une licence à FlowForce Server. La section Attribuer des licences aux produits enregistrés décrit comment attribuer des licences.

Enregistrer FlowForce Server par le biais de la CLI FlowForce (Linux)

Sur les machines Linux, FlowForce Server peut être enregistré avec LicenseServer en utilisant la commande `licenseserver` dans la CLI de FlowForce Server. Veuillez noter que FlowForce Server doit être lancé avec des droits racine.

```
sudo /opt/Altova/FlowForceServer2018/bin/flowforceserver licenseserver
localhost
```

Dans la commande ci-dessus, `localhost` est le nom du serveur sur lequel LicenseServer est installé. Veuillez également noter que l'emplacement du programme d'exécution de FlowForce Server est :

```
/opt/Altova/FlowForceServer2018/bin
```

Une fois avoir enregistré FlowForce Server avec succès, vous pouvez vous rendre sur LicenseServer et attribuer une licence à FlowForce Server. La section Attribuer des licences aux produits enregistrés décrit comment attribuer des licences.

Étapes suivantes

Après avoir enregistré votre produit Altova avec LicenseServer, procéder comme suit :

1. Si vous n'avez pas encore chargé votre/s fichier/s de licence sur LicenseServer (voir section précédente, Charger la/les licence/s), charger maintenant le fichier de licence et vérifier que la licence que vous souhaitez est active. Si vous avez déjà procédé à cette étape, continuer avec l'étape suivante, Attribuer les licences.
2. Attribuer une licence à votre produit Altova qui a été enregistré avec le LicenseServer.

 © 2018 Altova GmbH

Enregistrer MapForce Server

Cette section :

- Enregistrer MapForce Server depuis FlowForce Server (Windows)
- Enregistrer un MapForce Server autonome (Windows)
- Enregistrer MapForce Server (Linux)
- Étapes suivantes

MapForce Server peut être installé dans le cadre du paquet FlowForce Server ou en tant qu'un produit de serveur autonome. Dans tous les cas, il doit être enregistré avec Altova LicenseServer. Après qu'il ait été enregistré avec LicenseServer, vous pouvez attribuer une licence depuis LicenseServer. Sur les systèmes Windows, vous pouvez installer MapForce Server dans le cadre de l'installation de FlowForce. Si MapForce Server a été installé dans le cadre du paquet FlowForce Server, il sera automatiquement enregistré auprès de LicenseServer lorsque FlowForce est enregistré avec le LicenseServer. Sur les systèmes Linux, les deux produits doivent être installés séparément. Si MapForce Server a été installé dans le cadre du paquet FlowForce Server, il sera automatiquement enregistré lorsque FlowForce est enregistré. Mais si MapForce Server est installé avant FlowForce Server, vous devrez enregistrer les deux produits séparément.

Enregistrer MapForce Server depuis FlowForce Server (Windows)

MapForce Server est contenu dans un pack avec FlowForce Server, donc lorsque FlowForce Server est enregistré avec un Altova LicenseServer sur votre réseau, MapForce Server sera également enregistré automatiquement avec LicenseServer. Vous trouverez une description de l'enregistrement de FlowForce Server dans la documentation de FlowForce Server dans la section Enregistrer FlowForce Server avec LicenseServer.

Après l'enregistrement, vous pouvez vous rendre au LicenseServer et attribuer une licence de MapForce Server sur le MapForce Server. Vous trouverez une description de l'attribution de licence dans la section Attribuer des licences aux produits enregistrés.

Enregistrer un MapForce Server autonome (Windows)

Si vous avez installé MapForce Server en tant que pack autonome, vous devrez l'enregistrer avec un Altova LicenseServer sur votre réseau puis lui attribuer une licence à partir de Altova LicenseServer. Vous pouvez enregistrer MapForce Server par le biais de son interface de ligne de commande (CLI) en utilisant la commande `licenseserver` :

```
MapForceServer licenseserver Server-Or-IP-Address
```

Par exemple, si LicenseServer marche sur `http://localhost:8088`, enregistrer MapForce Server avec :

```
MapForceServer licenseserver localhost
```

Après avoir enregistré MapForce Server avec succès, vous pouvez vous rendre sur LicenseServer et attribuer une licence au MapForce Server. Vous trouverez une description de l'attribution de licence dans la section Attribuer des licences aux produits enregistrés.

Enregistrer MapForce Server (Linux)

Sur les machines Linux, MapForce Server peut être enregistré avec LicenseServer en utilisant la commande `licenseserver` de la CLI de MapForce Server. Veuillez noter que MapForce Server doit être lancé avec des droits racine.

```
sudo /opt/Altova/MapForceServer2018/bin/mapforceserver licenseserver localhost
```

Dans la commande ci-dessus, `localhost` est le nom du serveur sur lequel LicenseServer est installé. Veuillez noter que l'emplacement du programme d'exécution de MapForce Server est :

```
/opt/Altova/MapForceServer2018/bin
```

Après avoir enregistré MapForce Server avec succès, vous pouvez vous rendre sur LicenseServer et attribuer une licence au MapForce Server. Vous trouverez une description de l'attribution de licence dans la section Attribuer des licences aux produits enregistrés.

Étapes suivantes

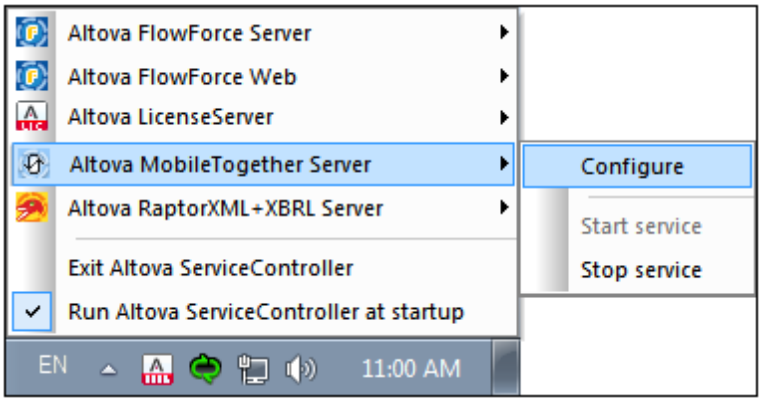
Après avoir enregistré votre produit Altova avec LicenseServer, procéder à l'étape suivante :

1. Si vous n'avez pas encore chargé votre/s fichier/s de licence sur LicenseServer (voir section précédente, Charger la/les licence/s), charger maintenant le fichier de licence et vérifier que la licence que vous souhaitez est active. Si vous avez déjà procédé à cette étape, continuer avec l'étape suivante, Attribuer les licences.
2. Attribuer une licence à votre produit Altova qui a été enregistré avec le LicenseServer

 © 2018 Altova GmbH

Enregistrer MobileTogether Server

Pour lancer MobileTogether Server, cliquer sur l'icône **ServiceController** dans la zone de notification, planer sur **Altova MobileTogether Server** dans le menu qui s'affiche (*voir capture d'écran ci-dessous*) puis sélectionner **Start Service** dans le sous-menu de MobileTogether Server. Si MobileTogether Server fonctionne déjà, l'option *Start Service* sera désactivée.



Enregistrer MobileTogether Server avec :

- L'onglet Paramètres de l'UI Web de MobileTogether : (i) lancer MobileTogether Server via le ServiceController (*voir le point précédent*); (ii) Saisir votre mot de passe pour accéder à la page de Configurations ; (iii) sélectionner l'onglet Paramètres, (iv) se rendre sur le panneau LicenseServer situé en bas de la page, sélectionner le nom ou l'adresse du LicenseServer et cliquer sur **Register with LicenseServer**.

- sa CLI, en utilisant la commande `licenseserver` :
  `MobileTogetherServer licenseserver [options] ServerName-Or-IP-Address`
  Par exemple, si `localhost` est le nom du serveur sur lequel LicenseServer est installé :
  `MobileTogetherServer licenseserver localhost`

Après avoir enregistré le produit avec succès, rendez-vous à l'onglet Client Management de la page de configuration de LicenseServer's pour attribuer une licence au MobileTogether Server.

Enregistrer RaptorXML(+XBRL) Server

Cette section :

- Enregistrer RaptorXML(+XBRL) Server (Windows)
- Enregistrer RaptorXML(+XBRL) Server (Linux)
- Étapes suivantes

RaptorXML(+XBRL) Server doit être installé sur la machine de serveur ou le réseau sur lequel LicenseServer est connecté, puis il doit être lancé en tant que service. Il doit ensuite être enregistré avec LicenseServer. Ce n'est qu'une fois l'enregistrement effectué qu'une licence peut être attribuée depuis le LicenseServer. Cette section décrit comment enregistrer RaptorXML(+XBRL) Server avec LicenseServer.

Enregistrer RaptorXML(+XBRL) Server (Windows)

Vous pouvez enregistrer RaptorXML(+XBRL) Server par le biais de son interface de ligne de commande (CLI) en utilisant la commande `licenseserver` :

RaptorXML Server:	**`RaptorXML licenseserver Server-Or-IP-Address`**
RaptorXML+XBRL Server:	**`RaptorXMLXBRL licenseserver Server-Or-IP-Address`**

Par exemple, si LicenseServer marche sur `http://localhost:8088`, enregistrer RaptorXML(+XBRL) Server avec :

RaptorXML Server:	**`RaptorXML licenseserver localhost`**
RaptorXML+XBRL Server:	**`RaptorXMLXBRL licenseserver localhost`**

Après avoir enregistré RaptorXML(+XBRL) Server avec succès, vous pouvez vous rendre sur LicenseServer et attribuer une licence au RaptorXML(+XBRL) Server. Vous trouverez une description de l'attribution de licence dans la section Attribuer des licences aux produits enregistrés.

Enregistrer RaptorXML(+XBRL) Server (Linux)

Sur les machines Linux, RaptorXML(+XBRL) Server peut être enregistré avec LicenseServer en utilisant la commande `licenseserver` de la CLI RaptorXML(+XBRL) Server. Veuillez noter que RaptorXML(+XBRL) Server doit être démarré avec des droits racine.

```
sudo /opt/Altova/RaptorXMLServer2018/bin/raptorxmlserver licenseserver
localhost
sudo /opt/Altova/RaptorXMLXBRLServer2018/bin/raptorxmlxbrlserver licenseserver
```

 © 2018 Altova GmbH

```
localhost
```

Dans la commande ci-dessus, `localhost` est le nom du serveur sur lequel LicenseServer est installé. Veuillez noter que l'emplacement du programme d'exécution de RaptorXML(+XBRL) Server est :

```
/opt/Altova/RaptorXMLServer2018/bin
/opt/Altova/RaptorXMLXBRLServer2018/bin
```

Après avoir enregistré RaptorXML(+XBRL) Server avec succès, vous pouvez vous rendre sur LicenseServer et attribuer une licence au RaptorXML(+XBRL) Server. Vous trouverez une description de l'attribution de licence dans la section Attribuer des licences aux produits enregistrés.

Étapes suivantes

Après avoir enregistré votre produit Altova avec LicenseServer, procéder à l'étape suivante :

1. Si vous n'avez pas encore chargé votre/s fichier/s de licence sur LicenseServer (voir section précédente, Charger la/les licence/s), charger maintenant le fichier de licence et vérifier que la licence que vous souhaitez est active. Si vous avez déjà procédé à cette étape, continuer avec l'étape suivante, Attribuer les licences.
2. Attribuer une licence à votre produit Altova qui a été enregistré avec le LicenseServer

Enregistrer StyleVision Server

Cette section :

- Enregistrer StyleVision Server depuis FlowForce Server (Windows)
- Enregistrer un StyleVision Server autonome (Windows)
- Enregistrer StyleVision Server (Linux)
- Étapes suivantes

StyleVision Server peut être installé dans le cadre du paquet FlowForce Server ou en tant qu'un produit de serveur autonome. Dans tous les cas, il doit être enregistré avec Altova LicenseServer. Après qu'il ait été enregistré avec LicenseServer, vous pouvez attribuer une licence depuis LicenseServer. Sur les systèmes Windows, vous pouvez installer StyleVision Server dans le cadre de l'installation de FlowForce. Si StyleVision Server a été installé dans le cadre du paquet FlowForce Server, il sera automatiquement enregistré auprès de LicenseServer lorsque FlowForce est enregistré avec le LicenseServer. Sur les systèmes Linux, les deux produits doivent être installés séparément. Si StyleVision Server a été installé dans le cadre du paquet FlowForce Server, il sera automatiquement enregistré lorsque FlowForce est enregistré. Mais si StyleVision Server est installé avant FlowForce Server, vous devrez enregistrer les deux produits séparément.

Enregistrer StyleVision Server depuis FlowForce (Windows)

StyleVision Server est contenu dans un pack avec FlowForce Server, donc lorsque FlowForce Server est enregistré avec un Altova LicenseServer sur votre réseau, StyleVision Server sera également enregistré automatiquement avec LicenseServer. Vous trouverez une description de l'enregistrement de FlowForce Server dans la documentation de FlowForce Server dans la section Enregistrer FlowForce Server avec LicenseServer.

Après l'enregistrement, vous pouvez vous rendre au LicenseServer et attribuer une licence de StyleVision Server sur le StyleVision Server. Vous trouverez une description de l'attribution de licence dans la section Attribuer des licences aux produits enregistrés.

Enregistrer un StyleVision Server (Windows)

Si vous avez installé StyleVision Server en tant que pack autonome, vous devrez l'enregistrer avec un Altova LicenseServer sur votre réseau puis lui attribuer une licence à partir de Altova LicenseServer. Vous pouvez enregistrer StyleVision Server par le biais de son interface de ligne de commande (CLI) en utilisant la commande `licenseserver` :

```
StyleVisionServer licenseserver Server-Or-IP-Address
```

Par exemple, si LicenseServer marche sur `http://localhost:8088`, enregistrer StyleVision Server avec :

```
StyleVisionServer licenseserver localhost
```

Après avoir enregistré StyleVision Server avec succès, vous pouvez vous rendre sur LicenseServer et attribuer une licence au StyleVision Server. Vous trouverez une description de l'attribution de licence dans la section Attribuer des licences aux produits enregistrés.

Enregistrer StyleVision Server (Linux)

Sur les machines Linux, StyleVision Server peut être enregistré avec LicenseServer en utilisant la commande `licenseserver` de la CLI de StyleVision Server. Veuillez noter que StyleVision Server doit être lancé avec des droits racine.

```
sudo /opt/Altova/StyleVisionServer2018/bin/stylevisionserver licenseserver
localhost
```

Dans la commande ci-dessus, `localhost` est le nom du serveur sur lequel LicenseServer est installé. Veuillez noter que l'emplacement du programme d'exécution de StyleVision Server est :

```
/opt/Altova/StyleVisionServer2018/bin
```

Après avoir enregistré StyleVision Server avec succès, vous pouvez vous rendre sur LicenseServer et attribuer une licence au StyleVision Server. Vous trouverez une description de l'attribution de licence dans la section Attribuer des licences aux produits enregistrés.

Étapes suivantes

Après avoir enregistré votre produit Altova avec LicenseServer, procéder à l'étape suivante :

1. Si vous n'avez pas encore chargé votre/s fichier/s de licence sur LicenseServer (voir section précédente, Charger la/les licence/s), charger maintenant le fichier de licence et vérifier que la licence que vous souhaitez est active. Si vous avez déjà procédé à cette étape, continuer avec l'étape suivante, Attribuer les licences.
2. Attribuer une licence à votre produit Altova qui a été enregistré avec le LicenseServer

Attribuer des licences à des produits enregistrés

Cette section :

- Avant d'attribuer une licence
- L'onglet Client Management
- Icônes dans l'onglet Server Management
- Note sur les cœurs et les licences
- Attribuer une licence
- Désenregistrer les produits du LicenseServer

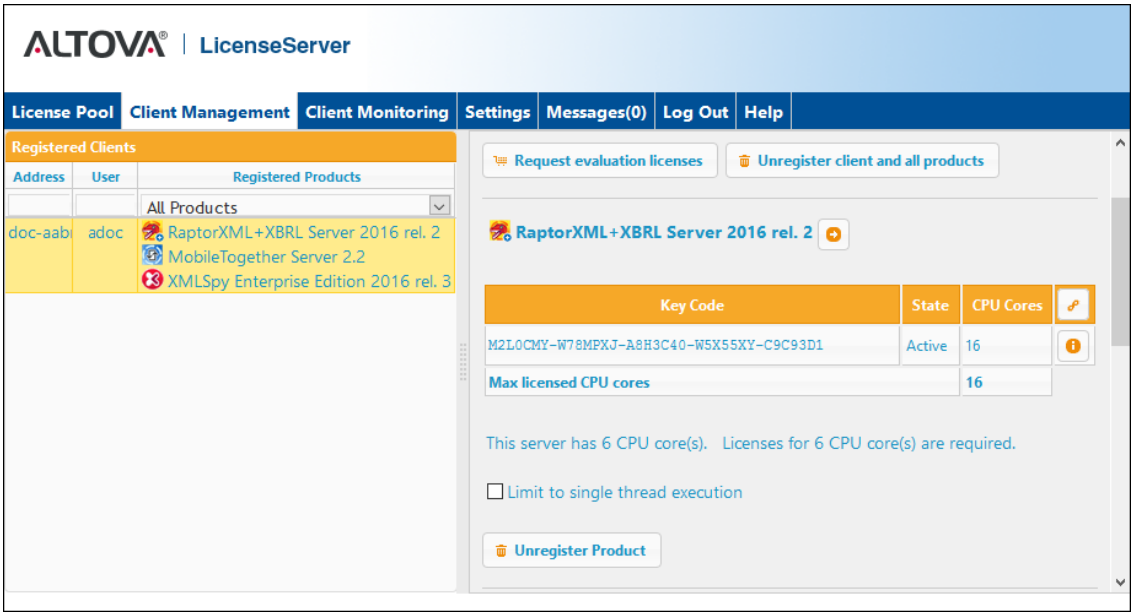
Avant d'attribuer une licence

Avant d'attribuer une licence à un produit Altova, veuillez vous assurer que :

- La licence pertinente a été mise à jour dans le pool de licences de LicenseServer et que cette licence est active.
- Votre produit Altova a été enregistré avec LicenseServer.

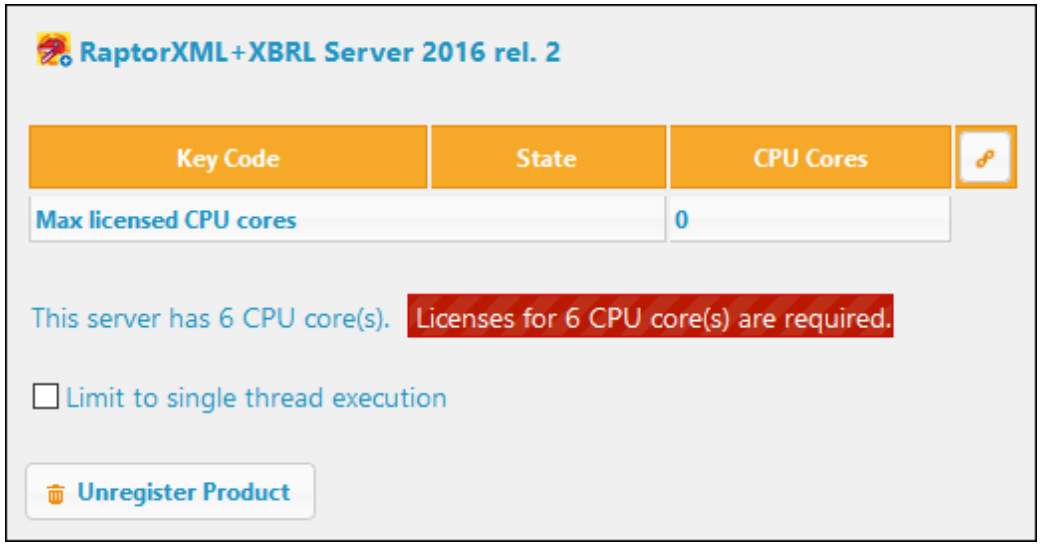
L'onglet Client Management

Les licences sont attribuées dans l'onglet Client Management de la page de Configuration de LicenseServer (*capture d'écran*). La capture d'écran montre dans le panneau de gauche, un appareil client comportant trois produits Altova qui ont été enregistrés auprès du LicenseServer.



Veuillez noter les points suivants concernant l'onglet Client Management :

- Dans le panneau de gauche, chaque produit est recensé sous le nom de son appareil client. La capture d'écran ci-dessus représente un appareil client. Cet appareil client comporte trois produits Altova enregistrés auprès du LicenseServer. Si un produit Altova sur une machine client différente est enregistré auprès de ce LicenseServer, alors cette machine client, avec ses produits enregistrés, figurera également dans le panneau de gauche.
- Lorsque vous sélectionnez un appareil client dans le panneau de gauche, les détails de licence des produits enregistrés de cet appareil sont affichés dans le panneau de droite. Ici, vous pouvez éditer les attributions de licence de chaque produit.
- Chaque produit Altova enregistré sur un appareil client a sa propre entrée *Key Code* (code-clé) qui prend le code-clé d'une licence. Une licence est attribuée à un produit enregistré en cliquant sur le bouton **Edit Assigned Licenses** (Éditer les licences attribuées) (*voir la liste d'icônes ci-dessous*) et en sélectionnant la licence nécessaire de celle disponible pour ce produit dans le pool de licences. Cette procédure est expliquée en détails ci-dessous.
- Les produits de serveur présentent également une ligne expliquant combien de cœurs CPU doivent être mis sous licence pour faire fonctionner ce produit sur ce client. Si le nombre de cœurs mis sous licence est moins important que le nombre nécessaire, alors l'information est marquée en rouge (*voir capture d'écran ci-dessous*). (Le nombre de cœurs CPU devant obtenir une licence est le nombre de cœurs CPU sur ce client. Il est obtenu depuis la machine client par le LicenseServer.)



- Si des **versions multiples** d'un seul produit (par exemple StyleVision Server 2013 et StyleVision Server 2014) ont été installées sur une machine et si chacune de ces installations ont été enregistrées avec un seul LicenseServer, alors les enregistrements multiples sont consolidés dans un seul enregistrement dans l'onglet Client Management et sont affichés en tant qu'un enregistrement unique. Lorsqu'une licence est attribuée à cet enregistrement unique, toutes les installations indiquées par cet enregistrement seront mises sous licence. Néanmoins, plusieurs instances d'une seule installation peuvent être exécutées simultanément, mais il n'est pas possible d'exécuter une instance de StyleVision Server 2013 et une instance de StyleVision Server 2014. Veuillez noter que toute version récemment installée doit être enregistrée pour qu'elle puisse être exécutée.
- Les nouvelles versions des produits de serveur Altova peuvent uniquement être mises sous licence avec la version de LicenseServer la plus récente au moment de la mise du produit sur le marché. Les produits de serveur Altova plus anciens fonctionneront avec les

versions plus récentes du LicenseServer. Donc si vous installez une nouvelle version d'un produit de serveur Altova et si votre version actuelle du LicenseServer version n'est pas la version la plus récente, désinstallez l'ancienne version du LicenseServer et installez la version la plus récente. Toutes les informations d'enregistrement et d'obtention de licence contenues dans votre ancienne version du LicenseServer seront enregistrées dans une base de données sur l'appareil client au moment de la désinstallation et seront importées automatiquement dans la nouvelle version. (Le numéro de version du LicenseServer qui est approprié pour toute version particulière d'un produit de serveur est affiché pendant l'installation du produit de serveur. Vous pouvez choisir d'installer cette version avec le produit de serveur. La version du LicenseServer actuellement installée est indiquée au bas de la page de configuration du LicenseServer .)

Icônes dans l'onglet Client Management

Edit Assigned Licenses. Disponible pour chaque liste de produit. Ouvre le dialogue Edit Assigned Licenses dans lequel des nouvelles licences peuvent être attribuées et les licences déjà attribuées peuvent être éditées.

Show Licenses. Apparaît avec chaque liste de licence. Passe à l'onglet License Pool et marque la licence sélectionnée permettant de lire les détails de la licence.

Unregister Product. Le produit sélectionné sera désenregistré depuis le LicenseServer.

Note sur les cœurs et les licences
L'obtention de la licence des produits de serveur Altova, est basée sur le nombre de cœurs de processeurs disponibles sur la machine de produit. Par exemple, un processeur double cœur a deux cœurs, un processeur quadricœur a quatre cœurs, un processeur hexacœurs a six cœurs, etc. Le nombre de cœurs pour lesquels une licence a été délivrée pour un produit doit être supérieur ou égal au nombre de cœurs disponibles sur cette machine de serveur, que ce serveur soit une machine physique ou virtuelle. Par exemple, si un serveur a huit cœurs (un processeur octacœur), vous devrez acheter au moins une licence octacœur. Vous pouvez aussi additionner les licences pour obtenir le nombre souhaité de cœurs. Ainsi, deux licences de quadricœurs peuvent être utilisées pour un serveur octacœur au lieu d'acheter une licence octacœur.

Si vous utilisez un serveur d'ordinateur avec un grand nombre de cœurs CPU, mais ne disposez que d'un faible volume à traiter, vous pouvez aussi créer une machine virtuelle qui disposera d'un plus petit nombre de cœurs et acheter une licence pour ce nombre de cœurs. Il va de soi que la vitesse de traitement d'un tel déploiement sera moins rapide que si tous les cœurs disponibles sur le serveur étaient utilisés.

Note: Chaque licence de produit de serveur Altova peut être utilisée pour une seule machine client à la fois (l'appareil sur lequel le produit de serveur Altova est installé), même si la licence a une capacité de licence n'est pas utilisée. Par exemple, si une licence de 10-cœurs est utilisée pour une machine client qui détient 6 cœurs CPU, les 4 cœurs

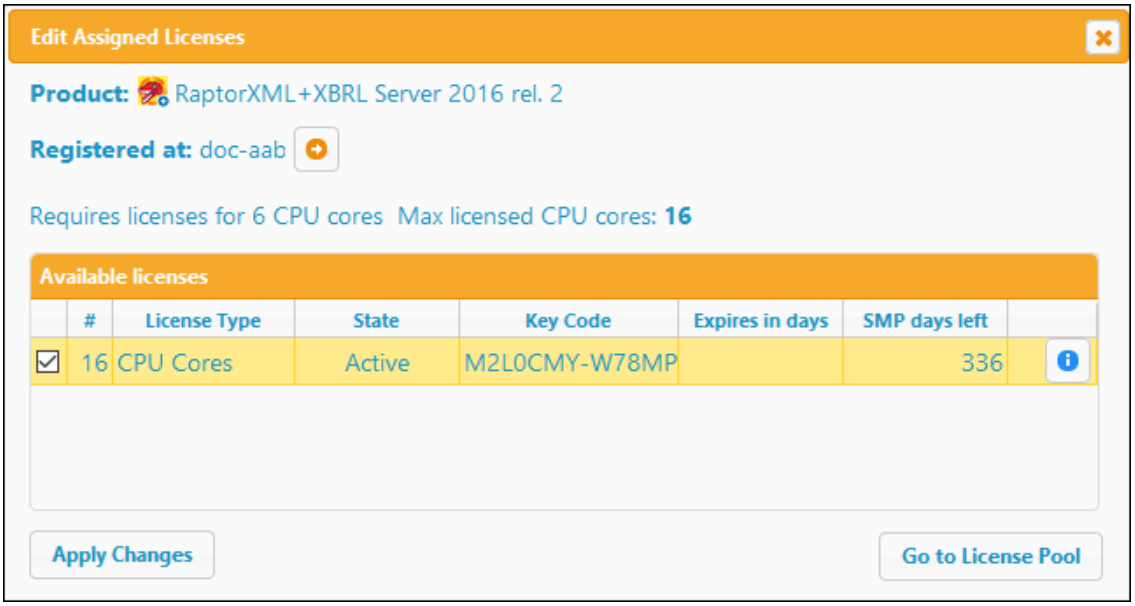
restants de la capacité de licence ne pourront pas être utilisés simultanément pour une autre machine client.

Note : En raison des fonctions de ses services, MobileTogether Server Advanced Edition ne sera exécuté que sur des machines comportant **deux ou plus de cœurs**.

Les licences MobileTogether Server
Les licences de MobileTogether Server sont basées sur le nombre des cœurs CPU dans l'appareil MobileTogether Server. Les licences cœur permettent à un nombre illimité d'appareils MobileTogether Client de se connecter au serveur. Néanmoins, si vous cochez la case *Limiter à une seule exécution thread*, un seul appareil mobile pourra se connecter au MobileTogether Server à tout moment. Cela est utile pour l'évaluation et les tests à petite échelle. Veuillez noter que, dans ce cas, un second appareil se connectera à MobileTogether Server, et prendra la licence. Le premier appareil ne pourra plus se connecter et recevra un message d'erreur.

Attribuer une licence

Pour attribuer une licence à un produit enregistré, cliquer sur le bouton **Edit Assigned Licenses** de ce produit. Le dialogue Manage Licenses s'ouvre (*capture d'écran ci-dessous*).



Veuillez noter les points suivants à propos des licences affichées dans le dialogue Manage Licenses :

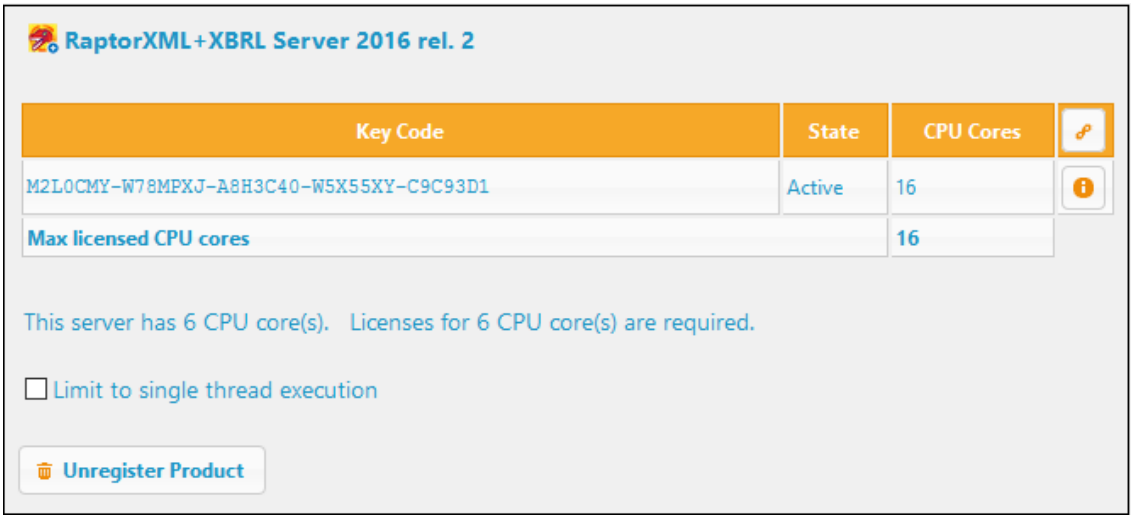
- Le produit pour lequel une licence est nécessaire est recensé dans la partie supérieure gauche du dialogue. Dans la capture d'écran ci-dessus, le produit est Altova RaptorXML +XBRL Server.
- La machine sur laquelle le serveur est installé (`Doc-aab` dans la capture d'écran ci-dessus) est affichée après.
- Le dialogue affiche toutes les licences actuellement actives pour ce produit dans le pool

de licences. Dans notre capture d'écran, une licence RaptorXML+XBRL Server actuellement active se trouve dans le pool de licences. (LicenseServer détectera automatiquement à partir de chaque licence dans le pool le produit pour lequel elle a été émise).

- Le type de licence peut être soit *Cores* (tous les produits de serveur Altova, y compris MobileTogether Server) ou *Users* (MobileTogether Server uniquement). Le type de licence est affiché dans la colonne *License Type.* La licence dans la capture d'écran ci-dessus a été obtenue pour 16 cœurs CPU.
- Vous devez connaître le nombre de cœurs de processeurs sur le serveur sur lequel le produit de serveur Altova a été installé. Si la machine possède un processeur double cœur, il vous faudra une licence à deux cœurs (le décompte des cœurs CPU). Le nombre de cœurs requis par le produit enregistré est indiqué sous le nom de la machine. La licence que vous attribuez à ce serveur doit être valide pour un nombre de cœurs suffisant. Veuillez noter que vous pouvez combiner des licences pour atteindre le nombre nécessaire de cœurs. Donc si le processeur de la machine est à octa-cœurs (huit cœurs), vous pouvez combiner deux licences à quatre cœurs.
- Le dialogue Edit Assigned Licenses recensera uniquement les licences actives pour **ce** produit. Les licences des autres produits Altova ne figureront pas dans la liste.
- Les cases à cocher des licences qui ont déjà été attribuées, par exemple, sur une autre installation du produit du réseau, seront cochées. Donc seules des licences non cochées peuvent être sélectionnées.
- La colonne *CPU Cores* (ou # pour les utilisateurs de MobileTogether Server) indique pour combien de cœurs CPU (ou MobileTogether Clients) une licence est valide.
- Si vous souhaitez procéder à des modifications du pool de licences, par exemple pour charger, activer, désactiver ou supprimer une licence, cliquer sur le bouton **Go to License Pool**.

Sélectionner la licence que vous souhaitez attribuer. La case à cocher de la licence sera cochée. De même, le nombre total de cœurs de CPU mis sous licence pour ce produit sur ce client est recensé près de la partie supérieur gauche du dialogue en tant que *Max licensed CPU cores (voir capture d'écran ci-dessus)*. Vous pouvez sélectionner plus de licences si vous souhaitez augmenter le nombre de cœurs de CPU mis sous licence pour ce produit sur ce client. Dans ce cas, le *Max licensed CPU cores* sera la somme des cœurs CPU sur toutes les licences sélectionnées.

Après avoir sélectionné la/les licence/s, cliquer sur **Apply Changes**. La/les licence/s sera/ont attribuée/s à ce produit et sera/ont affichée/s dans l'onglet Client Management (*voir capture d'écran ci-dessous*). La capture d'écran indique qu'une licence à 16 cœurs CPU pour Altova RaptorXML+XBRL a bien été attribuée à l'appareil client.



Désenregistrer les produits du LicenseServer

Chaque produit Altova enregistré avec LicenseServer est recensé dans le panneau de droite *(Product licensing)* sous son nom de machine client et présente un bouton **Unregister Product** en bas de son entrée (*voir capture d'écran ci-dessus)*. Cliquer sur ce bouton pour désenregistrer le produit du LicenseServer. Si une licence a été attribuée au produit, l'attribution sera terminée lors du désenregistrement du produit. Pour désenregistrer tous les produits, cliquer sur le bouton **Unregister Client and All Products** situé en haut du panneau de droite *(Product licensing)* (*voir d'abord la capture d'écran dans cette section*).

Pour réenregistrer un produit avec le LicenseServer, procéder comme suit :

- *Produits Serveur:* rendez-vous sur la page Settings dans l'UI Web du produit. Si le produit de serveur ne dispose pas d'une UI Web, ouvrez une fenêtre d'invite de commande et utilisez le CLI du produit pour l'enregistrer. Les étapes pour chaque produit de serveur sont décrites ici : Enregistrer FlowForce Server, Enregistrer MapForce Server, Enregistrer MobileTogether Server, Enregistrer StyleVision Server et Enregistrer RaptorXML(+XBRL) Server.
- *Produits de bureau* : Ouvrir le Dialogue d'activation de logiciel du produit (**Aide | Activation de logiciel)**, passer à l'Activation par le biais du mode LicenseServer et, dans le champ *Altova LicenseServer*, sélectionner le LicenseServer que vous souhaitez pour enregistrer le produit. Le produit sera enregistré et il apparaîtra dans l'onglet de Client Management du LicenseServer, dans la liste des produits enregistrés du client.

11.8 Référence de la page de Configuration

La page de Configuration LicenseServer est l'interface d'administrateur avec le LicenseServer (UI Web). Elle permet la gestion de LicenseServer et l'obtention de la licence pour les produits Altova qui ont été enregistrés avec LicenseServer. La page de Configuration de LicenseServer est visionnée dans un navigateur web. Pour une description de l'ouverture de la page de Configuration voir les sections, Ouvrir la Page de Config LicenseServer (Windows), Ouvrir la Page de Config LicenseServer (Linux) et Ouvrir la page de Config du LicenseServer (macOS).

Cette section est une référence utilisateur pour la page de Configuration et est organisée selon les onglets de la page de Configuration :

- License Pool (Pool de licences)
- Client Management (Gestion du client)
- Client Monitoring (Surveillance du client)
- Settings (Paramètres)
- Messages, Log Out (Messages, Déconnexion)

Pour des instructions étape par étape de l'attribution des licences avec LicenseServer, voir la section Comment attribuer les licences.

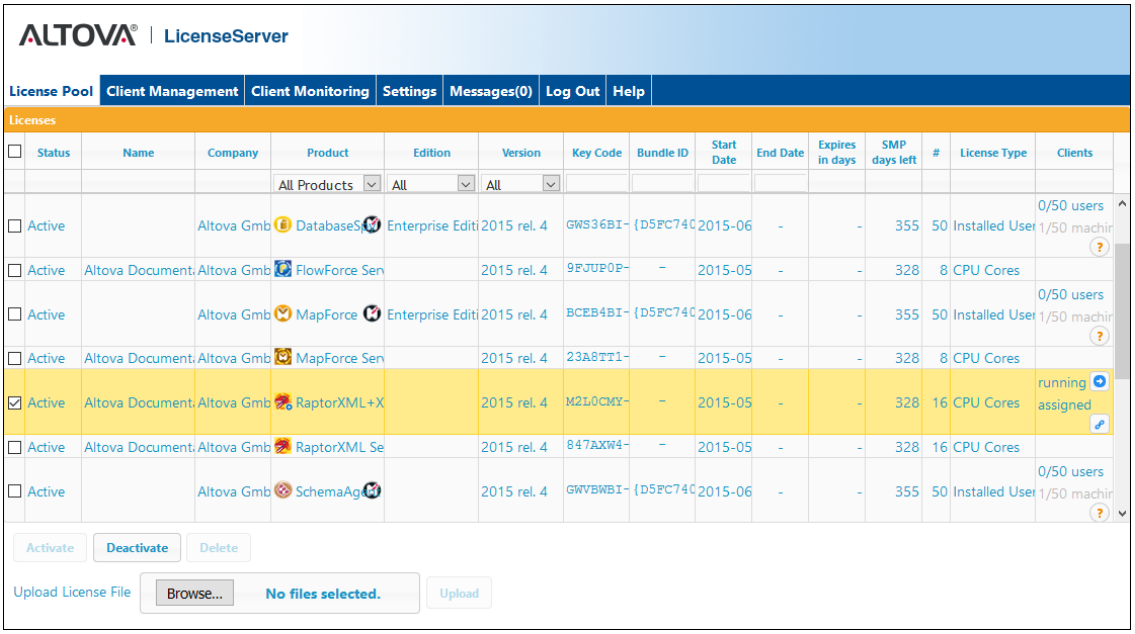
 © 2018 Altova GmbH

Pool de licences

Cette section :

- Charger une licence
- Statut de la licence
- Activer, désactiver et supprimer une licence
- Icônes dans l'onglet du Pool de licences
- Informations concernant la licence
- Note sur les licences de produit de bureau
- Note sur les cœurs et les licences

L'onglet **License Pool** affiche les informations concernant les licences actuellement disponibles sur LicenseServer (*voir capture d'écran ci-dessous*). Lorsqu'un fichier licence est chargé sur LicenseServer avec le bouton **Upload** sur cette page, toutes les licences contenues dans ce fichier licence sont placées dans le pool de licences sur LicenseServer sont affichées sur la page License Pool. La page License Pool affiche donc un aperçu de toutes les licences actuellement disponibles sur le LicenseServer et les détails de chacune de ces licences. Sur cette page, outre les fait de pouvoir charger des licences, vous pouvez également activer, désactiver et supprimer les licences sélectionnées.



Charger une licence

Pour charger un fichier de licence (un fichier `.altova_licenses` que vous recevrez de la part d'Altova GmbH pour votre produit Altova), cliquer sur le bouton **Browse**, chercher le fichier licence et le sélectionner. Cliquer sur **Upload**, pour placer toutes les licences contenues dans le fichier

licence dans le pool de licences et les afficher sur la page License Pool (*capture d'écran ci-dessus*).

Statut de la licence

Les valeurs de statut de la licence sont les suivantes :

- *Activating :* (En cours d'Activation) lorsqu'une licence est chargée dans le pool de licences de LicenseServer, le serveur transmettra les données relatives aux licences au serveur de licences principal `altova.com` pour valider, authentifier et activer la licence fournie. Cela est nécessaire pour assurer la conformité avec les contrats de licence Altova. Au cours de cette activation initiale et de transaction d'authentification, qui dure généralement entre 30 secondes et quelques minutes (selon votre connexion Internet, la vitesse et le trafic réseau général), le statut de la licence indiqué sera *Activating...*.

- *Failed Verification :* (Échec de la vérification) si une connexion avec le serveur de licence principal `altova.com` ne peut pas être établie, le statut de la licence dans le pool affiché sera *Failed Verification*. Si cela se produit, vérifier votre connexion Internet et vos règles de pare-feu pour assurer que LicenseServer est en mesure de communiquer avec le serveur de licence principal `altova.com`.

- *Active :* (Activé) une fois que la licence a été authentifiée et activée, le statut passera à *Active*.

- *Inactive :* (Désactivé) si une licence a été vérifiée mais qu'elle est présente sur un autre LicenseServer dans le réseau, le statut affiché sera *Inactive*. Un statut *Inactive* apparaît aussi lorsqu'une licence est désactivée manuellement dans le pool de licences par l'administrateur.

- *Pending :* si la date de démarrage d'une licence se trouve dans l'avenir, la licence sera affichée en tant que *Pending.* Le statut passera à *Active* à 00:00 à la date configurée. Une licence *Pending* peut être attribuée à un produit et garantit que la licence d'un produit se poursuive sans interruption avant qu'une licence existante expire. Le passage à la nouvelle licence est conçu pour être une transition lisse sans interrompre les traitements de client.

- *Blocked :* (Bloqué) si une licence affiche *Blocked* dans le pool de licences, il y a eu un problème d'authentification dans la licence et le serveur de licences principal `altova.com` n'a pas donné son autorisation à LicenseServer d'utiliser cette licence. Cela pourrait être le résultat d'une violation d'un contrat de licence, d'une utilisation abusive d'une licence ou d'autres problèmes de conformité. Une fois le problème réglé, supprimer, recharger et réactiver la licence. Veuillez voir la table ci-dessous pour des informations complémentaires.

Ces statuts sont résumés dans le tableau ci-dessous :

Statut	Signification

Activating...	Lors du chargement, l'information de licence est envoyée à `altova.com` à des fins de vérification. Rafraîchir le navigateur pour consulter le statut mis à jour. La vérification et l'activation peuvent prendre quelques minutes.
Failed Verification	Impossible d'établir une connexion avec `altova.com`. Après avoir établi la connexion, soit redémarrer le service, soit activer la licence (avec le bouton **Activate**).
Active	La vérification a été effectuée avec succès, la licence est activée.
Inactive	La vérification a été effectuée avec succès, mais la licence se trouve aussi sur un autre LicenseServer dans le réseau. Les licences peuvent être rendues inactives avec le bouton **Deactivate**.
Pending	La date de démarrage d'une licence *Pending* se trouve dans l'avenir et passera au mode *Active* à cette date. Elle peut être attribuée à un produit et garantit le renouvellement automatique de la licence d'un produit avant que la licence existante expire.
Blocked	La vérification a échoué. La licence est invalide et elle est bloquée. Une fois que le problème responsable du blocage de la licence a été résolu, supprimer la licence, la recharger et la réactiver. Chaque fois qu'une licence est chargée, elle prendra contact avec le serveur maître Altova, qui lancera une nouvelle vérification. Si vous ne la rechargez pas, la vérification de la licence devra attendre jusqu'à la prochaine communication prévue avec le serveur maître Altova et cela peut durer jusqu'à 24 heures. Veuillez contacter votre Support Altova si le problème persiste.

Note : Après avoir envoyé une licence à `altova.com` à des fins de vérification, le navigateur doit être rafraîchi pour voir le statut mis à jour. La vérification et l'activation peuvent prendre quelques minutes.

Note : Si une connexion à `altova.com` n'a pas pu être établie, le statut affichera *Failed Verification*. Après avoir établi une connexion, soit redémarrer le service, soit essayer d'activer la licence avec le bouton **Activate**.

Note : Lorsqu'une licence affiche un statut *Inactive* ou *Blocked*, un message d'explication du statut sera également ajouté au journal de Messages.

Seule une licence active ou pending peut être attribuée à une installation de produit. Une licence inactive pourra être activée ou supprimée du pool de licences. Si une licence est supprimée du pool de licences, elle pourra être chargée à nouveau dans le pool en chargeant le fichier de licence le contenant. Lorsqu'un fichier de licence est mis à jour, seules les licences ne se trouvant pas déjà dans le pool seront chargées dans le pool. Pour activer, désactiver ou supprimer une licence, la sélectionner et cliquer sur le bouton **Activate**, **Deactivate**, ou **Delete**, respectivement.

Connexion au Serveur de licence principal sous altova.com
L'Altova LicenseServer doit pouvoir communiquer avec le Serveur de licence principal sous `altova.com` pour valider et authentifier les données relatives aux licences et pour assurer la conformité continue avec les accords de licence Altova. Cette communication a lieu par le biais de HTTPS avec le port 443. Si Altova LicenseServer, après avoir établi la vérification

initiale avec le Serveur de licence principal `altova.com`, n'est pas capable de se reconnecter avec `altova.com` pour une durée de plus de 5 jours (= 120 heures), alors Altova LicenseServer ne pourra plus utiliser les produits de logiciels Altova connectés à cet Altova LicenseServer.

Toute perte de connexion de la sorte avec les serveurs principaux `altova.com` sera inscrite dans l'onglet Messages de la page de Configuration d'Altova LicenseServer. En outre, l'administrateur peut configurer Altova LicenseServer pour envoyer automatiquement un e-mail d'alerte lorsque la connexion à `altova.com` est perdue. Les paramètres des e-mails d'alerte sont disponibles dans l'onglet Paramètres de la page de Configuration.

Activer, désactiver et supprimer une licence

Une licence active peut être désactivée en sélectionnant la licence et en cliquant sur **Deactivate**. Une licence inactive (bouton **Activate**) ou supprimée (bouton **Delete**). Lorsqu'une licence est supprimée, elle est retirée du pool de licences. Une licence supprimée peut être ajoutée à nouveau dans le pool de licences en chargeant le fichier de licence le contenant. Si un fichier licence est re-chargé, seules les licences qui ne se trouvent pas encore dans le pool de licences y seront ajoutées ; les licences se trouvant déjà dans le pool de licences n'y seront pas rajoutées.

Icônes dans l'onglet du Pool de licences

Logo Altova MissionKit. Apparaît à côté des noms de produits de bureau individuels Altova lorsque la licence du produit de bureau fait partie d'une licence MissionKit. *Voir Note sur les licences de produit de bureau ci-dessous*.

Show Assigned Client (Afficher le client attribué). Apparaît dans la colonne *Clients* d'une licence qui a été attribuée. Va vers l'onglet Client Management, dans lequel les licences des produits enregistrés d'un client peuvent être gérés.

Show Running Client (Afficher le client en cours). Apparaît dans la colonne *Clients* d'une licence qui a été attribuée au logiciel actuellement en cours. Va vers l'onglet Client Management de l'appareil du client exécutant ce logiciel. Sont affichés ici, l'information concernant le client sélectionné et son logiciel enregistré.

Show Info (Affiche l'info). Apparaît dans la colonne *Clients* d'une licence qui n'a pas été attribuée. Affiche l'information concernant la licence, comme le décompte des utilisateurs et si la licence fait partie d'un paquet de licences.

Informations concernant la licence

Les informations suivantes concernant la licence sont affichées :

- *Status :* (Statut) peut être une des valeurs suivantes : *Activating (Activation)*| | *Failed*

Verification (Échec de la vérification)| *Active (Actif)*| *Inactive (Inactif)*| *Blocked (Bloqué)*. Voir Statut de licence ci-dessus.

- *Name, Company :* (Nom, Entreprise) le nom et l'entreprise du titulaire de licence. Cette information a été soumise au moment de l'achat.
- *Product, Edition, Version :* (Produit, Édition, Version) la version et l'édition des produits licenciés. Vous trouverez en haut de chaque colonne une liste de choix pour filtrer les licences par catégorie.
- *Key Code, Bundle ID: (Code de clé, Bundle ID)* La clé de licence pour déverrouiller le produit. Tous les produits contenus dans un seul paquet Altova MissionKit doivent présenter le même Bundle ID. Les produits non regroupés dans un paquet n'ont pas de Bundle ID.
- *Start Date, End Date: (Date de commencement, date d'expiration)* Ensemble ils indiquent la période de validité de la licence. Les licences permanentes n'ont pas de date butoir.
- *Expires in days, SMP (days left):* (Clé, Expire dans jours, SMP (jours restants)) le nombre de jours restants avant que la licence expire. Chaque licence achetée est fournie avec un Support & Maintenance Package, qui est valide pour un certain nombre de jours. La colonne *SMP* note combien d'assistance vous pouvez encode profiter.
- *#, License Type :* le nombre d'utilisateurs ou de cœurs de CPU permis par la licence est recensé dans la colonne #. Que la licence régule les utilisateurs ou les cœurs, cela est indiqué dans la colonne *License Type*. Dans le cas du produit MobileTogether Server Altova, les licences sont aussi attribuées sur la base du nombre **d'appareils** client qui se connectent à MobileTogether Server. Dans le cas de tous les autres produits de serveur Altova, les licences sont attribuées sur la base des **cœurs CPU** uniquement (*voir note ci-dessous*). Dans le cas des produits de bureau Altova, les licences sont attribuées sur la base des **utilisateurs**. *Voir Note sur les licence de produits de bureau ci-dessous*.
- *Clients:* Cette colonne n'a que des entrées pour les licences MobileTogether Server (MTS) et les licences de produits de bureau. Elle ne contient pas d'entrées pour les licences de produits de serveur non-MTS. En ce qui concerne les licences d'appareil MobileTogether Server, cette colonne affiche si la licence a été attribuée. En ce qui concerne les produits de bureau, la colonne affiche le décompte de la machine et le décompte des utilisateurs tel que décrit ci-dessous.

Produits de bureau : décompte des appareils et des utilisateurs

- Le décompte d'appareils (*machine count)* indique le nombre d'appareils disposant actuellement d'une licence pour exécuter le logiciel avec une licence donnée. Par exemple, `7/10 machines` signifie que la licence peut être utilisée pour des instances de logiciel sur 10 appareils et est actuellement utilisée pour des instances de logiciel sur 7 appareils. Cliquer sur le bouton **Show Assigned Client** pour vous rendre sur l'onglet Client Management et voir les détails des licences de l'appareil du client.
- Le décompte d'utilisateur (*user count)* indique le nombre d'utilisateurs utilisant actuellement la licence parmi le nombre total d'utilisateurs permis. Seuls les installations de logiciel sous licence et actuellement exécutées sont comptées. Par exemple, , `3/10 users` signifie que la licence est actuellement utilisée par 3 utilisateurs pour un total de 10 utilisateurs autorisés. Si une installation de logiciel sous licence est actuellement en cours, vous pouvez cliquer sur le bouton **Show Running Client** pour ouvrir l'onglet Client Monitoring et voir les détails des produits Altova qui sont exécutés sur les appareils des clients sur ce réseau.
- Les décomptes *user count* et *machine count* vous indiquent la capacité de licence et l'utilisation d'une licence donnée. Par exemple, si le décompte d'appareil d'une licence d'utilisateur installée est `7/10` et si le décompte utilisateur est `3/10`, alors nous pouvons déduire les informations suivantes : (i) le produit de logiciel peut être mis sous licence sur 10 appareils ; (ii) le logiciel a été mis sous licence sur 7 appareils ; (iii) 3 des 7

installations de logiciel mises sous licence sont actuellement exécutées.

Désattribuer une licence
Afin de désattribuer une licence depuis une installation de logiciel sur un appareil, rendez-vous sur l'onglet Client Management. Sélectionner l'appareil, puis le logiciel que vous souhaitez désattribuer. Ensuite, cliquer sur le bouton **Edit Assigned Licenses**, décocher la licence et cliquer sur **Apply Changes**.

Note sur les licences de produit de bureau

Il existe trois types de licences d'utilisateurs de produits bureau :

- *Installed User (Utilisateur installé) :* les licences ont été achetées pour un nombre bien précis d'ordinateurs sur lesquels vous aimeriez installer le logiciel. Par exemple, si vous achetez une licence *10-Installed-User*, vous pourrez installer et utiliser le logiciel sur jusqu'à 10 ordinateurs. Sur chaque appareil mis sous licence, vous pourrez démarrer autant d'instances de logiciel simultanées que vous le souhaitez. La licence, pour chaque "utilisateur installé", permet d'utiliser le produit de logiciel sur cet appareil.
- *Concurrent User (Utilisateur parallèles) :* cette licence vous permet d'installer le logiciel sur un nombre d'ordinateurs allant jusqu'au nombre d'appareils égal à 10 fois le nombre autorisé d'utilisateurs parallèles. Toutes les installations doivent se trouver sur le même réseau physique. À tout moment, le logiciel peut uniquement être utilisé par le nombre autorisé d'utilisateurs parallèles. Par exemple, si vous achetez une licence *20-Concurrent-User*, le logiciel peut être installé sur jusqu'à 200 ordinateurs se trouvant dans le même réseau physique et utilisé sur jusqu'à 20 ordinateurs simultanément à tout moment. Si vous souhaitez utiliser les licences *Concurrent User* sur des réseaux physiques différents, vous devrez acheter une licence séparée pour chaque réseau. Veuillez noter que vous ne pouvez pas utiliser une licence *Concurrent User* pour plusieurs réseaux.
- *Named User (Utilisateur nommé) :* pour chaque licence *Named-User,* vous pouvez installer le logiciel sur jusqu'à 5 ordinateurs mais seul l'utilisateur nommé dans la licence pourra utiliser le logiciel. Avec cette licence, un utilisateur peut travailler sur des ordinateurs différents à la condition que **seule une instance** du logiciel soit utilisé à tout moment.

Note à propos des licences Altova MissionKit
Altova MissionKit est une suite de produits de bureau Altova. Une licence Altova MissionKit comprend des licences individuelles pour chacun des produits de bureau dans la suite MissionKit. Chacune de ces licences de produit détient un code clé différent et unique mais le même MissionKit Bundle ID. Si vous chargez une licence Altova MissionKit sur le pool de licences, les licences individuelles de chaque produits de MissionKit est recensé dans le License Pool (affichant le logo Altova MissionKit). Si vous attribuez une de ces licences de produit à un utilisateur particulier, tous les autres produits de ce paquet MissionKit seront aussi attribués à cet utilisateur. Par conséquence, aucun autre produit dans ce paquet MissionKit particulier ne pourra être attribué à un autre utilisateur.

Extractions de licence
Vous pouvez extraire une licence depuis le pool de licences pour une période de jusqu'à 30 jours afin de stocker la licence sur la machine du produit. Cela vous permet de travailler hors ligne, ce qui peut être utile si, par exemple, vous souhaitez travailler dans un environnement où vous

n'aurez pas accès à votre Altova LicenseServer (si par exemple votre produit Altova est installé sur un laptop et que vous êtes en déplacement). Lorsque la licence est extraite, le LicenseServer affiche la licence comme étant utilisée et celle-ci ne peut pas être utilisée par une autre machine. La licence retourne automatiquement à l'état archivé lorsque la période d'extraction expire. En alternative, une licence extraite peut être archivée à tout moment par le biais du bouton **Archiver** dans le dialogue Activation du logiciel. Pour extraire une licence depuis le pool de licence sur votre machine de produit, allez sur le menu **Aide** de votre produit de bureau Altova et choisir **Activation du logiciel**. Veuillez consulter le manuel de votre produit Altova pour plus d'informations.

Note sur les cœurs et les licences
L'obtention de la licence des produits de serveur Altova, est basée sur le nombre de cœurs de processeurs disponibles sur la machine de produit. Par exemple, un processeur double cœur a deux cœurs, un processeur quadricœur a quatre cœurs, un processeur hexacœurs a six cœurs, etc. Le nombre de cœurs pour lesquels une licence a été délivrée pour un produit doit être supérieur ou égal au nombre de cœurs disponibles sur cette machine de serveur, que ce serveur soit une machine physique ou virtuelle. Par exemple, si un serveur a huit cœurs (un processeur octacœur), vous devrez acheter au moins une licence octacœur. Vous pouvez aussi additionner les licences pour obtenir le nombre souhaité de cœurs. Ainsi, deux licences de quadricœurs peuvent être utilisées pour un serveur octacœur au lieu d'acheter une licence octacœur.

Si vous utilisez un serveur d'ordinateur avec un grand nombre de cœurs CPU, mais ne disposez que d'un faible volume à traiter, vous pouvez aussi créer une machine virtuelle qui disposera d'un plus petit nombre de cœurs et acheter une licence pour ce nombre de cœurs. Il va de soi que la vitesse de traitement d'un tel déploiement sera moins rapide que si tous les cœurs disponibles sur le serveur étaient utilisés.

Note: Chaque licence de produit de serveur Altova peut être utilisée pour une seule machine client à la fois (l'appareil sur lequel le produit de serveur Altova est installé), même si la licence a une capacité de licence n'est pas utilisée. Par exemple, si une licence de 10-cœurs est utilisée pour une machine client qui détient 6 cœurs CPU, les 4 cœurs restants de la capacité de licence ne pourront pas être utilisés simultanément pour une autre machine client.

Note : En raison des fonctions de ses services, MobileTogether Server Advanced Edition ne sera exécuté que sur des machines comportant **deux ou plus de cœurs**.

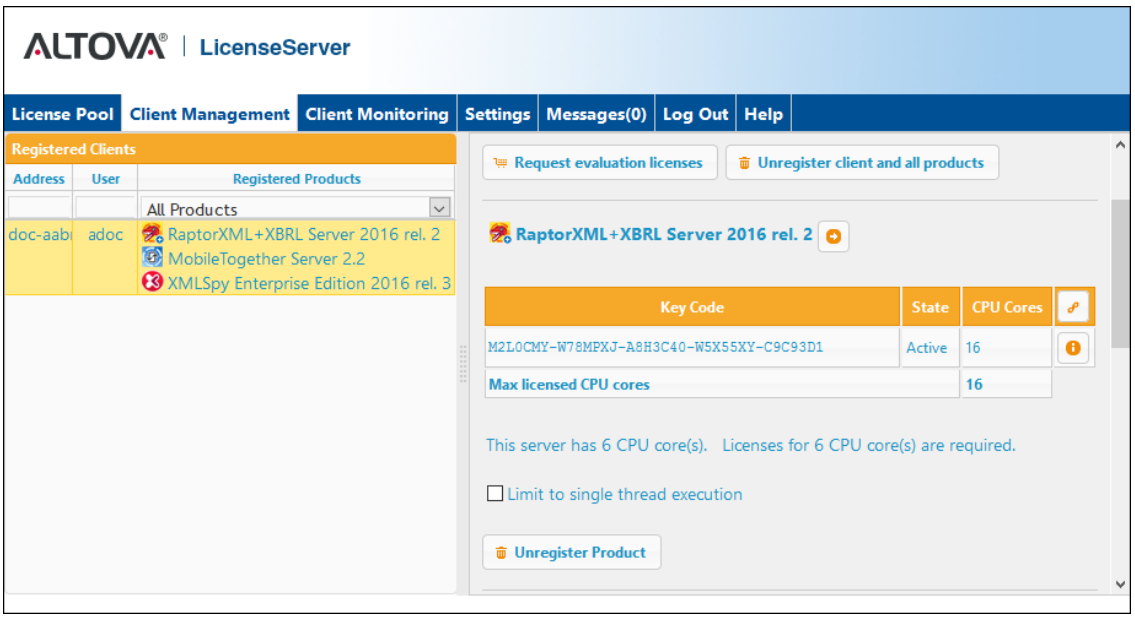
Les licences MobileTogether Server
Les licences de MobileTogether Server sont basées sur le nombre des cœurs CPU dans l'appareil MobileTogether Server. Les licences cœur permettent à un nombre illimité d'appareils MobileTogether Client de se connecter au serveur. Néanmoins, si vous cochez la case *Limiter à une seule exécution thread*, un seul appareil mobile pourra se connecter au MobileTogether Server à tout moment. Cela est utile pour l'évaluation et les tests à petite échelle. Veuillez noter que, dans ce cas, un second appareil se connectera à MobileTogether Server, et prendra la licence. Le premier appareil ne pourra plus se connecter et recevra un message d'erreur.

Gestion des clients

Cette section :

- Icônes dans l'onglet Client Management
- Gestion des licences dans le panneau Product Licensing
- Attribution des licences
- Exécution à traitement unique
- Une machine client sous plusieurs noms différents
- Demande d'une licence d'évaluation
- Désenregistrement de produits

L'onglet **Client Management** (*capture d'écran ci-dessous*), est divisé en deux panneaux :



- *Registered clients :* le panneau de gauche affiche une liste tabulaire des appareils sur le réseau dont au moins un produit Altova est enregistré avec LicenseServer. Ce genre d'appareil est appelé *registered clients*. Chaque client enregistré est recensé dans le panneau de gauche avec tous ses produits enregistrés. Afin de découvrir comment enregistrer des produits avec LicenseServer. consulter Enregistrer les Produit/s. L'affichage dans ce panneau peut être filtré en sélectionnant ou en saisissant un filtre au haut d'une des colonnes du panneau.
- *Product licensing:* le panneau de droite. Lorsqu'un client enregistré est sélectionné dans le panneau de gauche (*Registered Clients*), les détails de licence des produits de ce client enregistré sont affichés dans le panneau de droite. Ici, vous pouvez gérer les licences de chaque produit enregistré (*description ci-dessous*).

Icônes dans l'onglet Server Management

Edit Assigned Licenses. (Éditer les licences attribuées) Disponible pour chaque produit. Le dialogue Manage Licenses (Gérer les licences) s'ouvre dans lequel des nouvelles licences peuvent être attribuées au produit et les licences déjà attribuées peuvent être éditées.

Show Licenses. (Afficher les informations de licence) Apparaît avec chaque licence. Passe à l'onglet License Pool et marque la licence sélectionnée pour que l'utilisateur puisse lire les détails de licence.

Unregister This Product. (Désenregistrer ce produit) Disponible pour chaque produit. Le produit sélectionné (sur la machine client sélectionnée) sera désenregistré du LicenseServer.

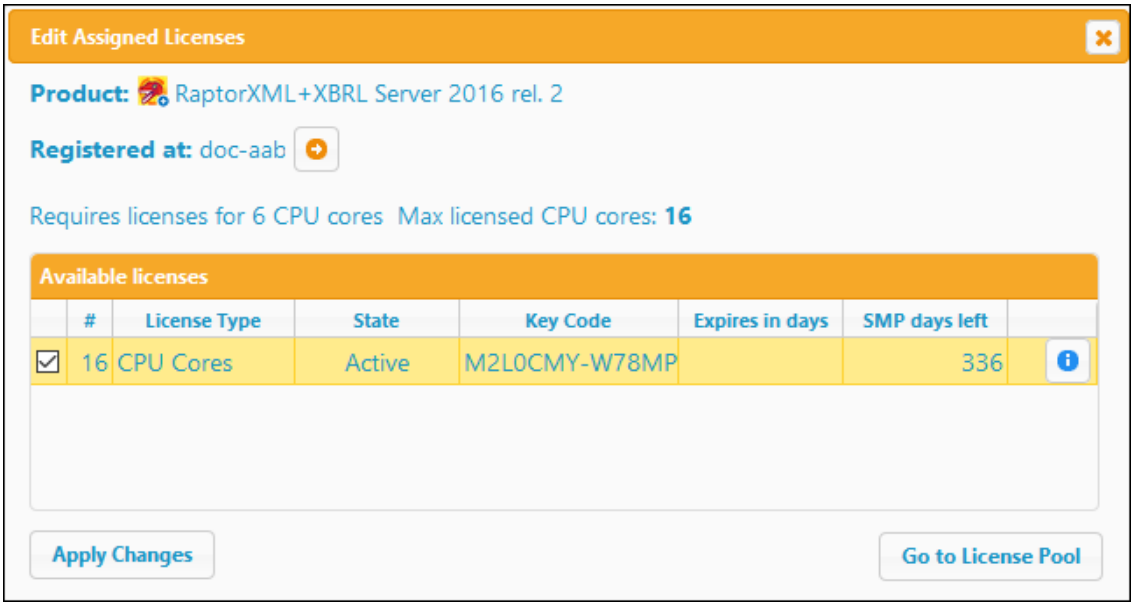
Gestion des licences dans le panneau de Product Licensing

Le panneau de droite *Product licensing*, vous permet de procéder aux étapes suivantes :

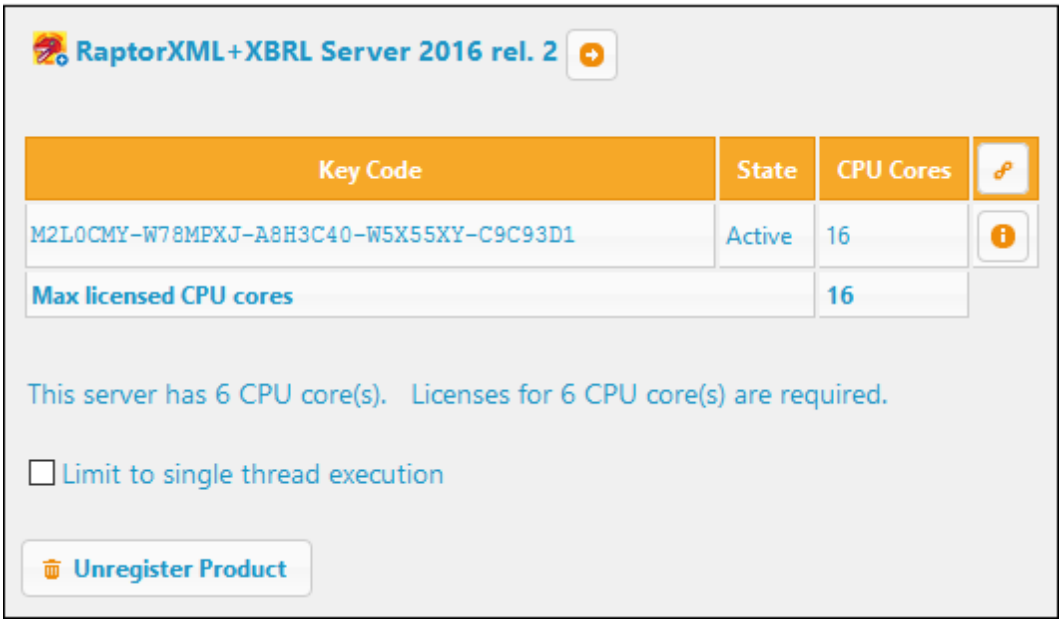
- *Assign, unassign, change a product's license: (Attribuer, désattribuer, changer une licence d'un produit)* cliquer sur le bouton **Edit Assigned Licenses** du produit pour ce faire. Voir *Attribution des licences* ci-dessous. Veuillez noter que chaque produit de serveur a une ligne établissant combien de cœurs de CPU doivent être mis sous licence pour exécuter ce produit sur ce client. Si le nombre de cœurs de licence est moins important que le nombre exigé, l'information sera marquée en rouge. (Le nombre de cœurs de CPU qui doit être mis sous licence est le nombre de cœurs de CPU sur ce client et qui est obtenu depuis l'appareil du client par LicenseServer.)
- *Set up a single-core, server-product license to use only one core of a client (Définir une licence à cœur unique, server-product à utiliser uniquement sur le cœur d'un client):* voir *Exécution à traitement unique* ci-dessous.
- *Unregister a product from LicenseServer (Désenregistrer un produit de LicenseServer) :* pour ce faire utiliser un bouton **Unregister Product** du produit. Voir *Désenregistrer des produits* ci-dessous.

Attribuer une licence

Pour attribuer une licence à un produit enregistré, cliquer sur le bouton **Edit Assigned Licenses** de ce produit. Le dialogue Edit Assigned Licenses s'ouvre (*capture d'écran ci-dessous*).



Sélectionner la/les licence/s que vous souhaitez attribuer, ensuite, cliquer sur **Apply Changes (Appliquer les changements)**. Les licences seront attribuées à ce produit et seront affichées dans l'onglet *Product Licensing* de l'onglet Client Management (*voir capture d'écran*).



Exécution à traitement unique
Si une licence de produit de serveur Altova pour un seul cœur est disponible dans le pool de licences, une machine multi-cœurs peut se voir attribuer cette licence à cœur unique. Dans ce cas de figure, la machine exécutera ce produit sur un seul cœur. Le traitement sera donc plus lent puisque le traitement unique (ce qui est possible sur des cœurs multiples) ne sera

 © 2018 Altova GmbH

pas disponible. Le produit sera exécuté dans le mode de traitement unique sur cette machine.

Pour attribuer une licence à cœur unique sur une machine à multicœurs, cocher la case *Limit to single thread execution* pour ce produit.

Dans le cas de **MobileTogether Server (MTS)**, si une exécution à traitement unique est sélectionnée pour une licence à cœurs MTS, alors un seul appareil mobile pourra se connecter au MobileTogether Server à tout moment. Veuillez noter que si, dans ce cas, un deuxième appareil se connecte au MobileTogether Server, il reprendra la licence. Le premier appareil ne pourra plus se connecter et recevra un message d'erreur.

Une machine client sous des noms différents

Si une machine client est enregistrée plus d'une fois sur le LicenseServer, il peut apparaître dans l'onglet Client Management sous plusieurs nom, c. à. d. avec plusieurs entrées. Cela peut se produire par exemple si une machine est réenregistrée avec le nom d'hôte donné dans une forme différente.

Afin d'assurer que des licences supplémentaires ne sont pas attribuées en redondance à la même machine sous ses noms différents, vous devriez désenregistrer les entrées de la machine client redondante en cliquant sur le bouton **Unregister client and all products** dans le panneau en haut à droite (Product Licensing). De même, si la même licence est attribuée plusieurs fois sur le même appareil sous ses noms différents, des conflits de licence surgiront. Ainsi, pour éviter ces deux situations (licences redondantes et attributions multiples d'une seule licence), il est recommandé que les entrées redondantes d'une seule machine client soient désenregistrées.

Ci-dessous, vous trouverez les formes qu'un nom d'appareil peut prendre dans l'onglet Client Management :

- *Nom d'hôte avec le nom de domaine (le nom de domaine complètement qualifié, FQDN), comme : "win80-x64_1.my.domain.com" ou "Doc3.my.domain.com".* Cela arrive lorsque le nom d'hôte de la machine (avec ou sans les informations de domaine) est passé en tant que l'argument de la commande CLI `licenseserver` qui est utilisée pour enregistrer le produit de serveur avec LicenseServer. Par exemple : `<AltovaServerProduct> licenseserver Doc3`. Cela produit un FQDN comme : `Doc3.my.domain.com`.

  Un FQDN est également produit lorsque `localhost` est fourni sur les systèmes Windows 7 et 10 en tant que le nom d'hôte.

- *Nom d'hôte sans nom de domaine. Par exemple : "win80-x64_1" ou "Doc3".* Cela se produit sur les systèmes Windows 8 lorsque `localhost` est indiqué en tant que le nom de la machine.

- *localhost*. Dans certains cas, `localhost` est également affiché en tant que le nom de la machine.

Note : Si, pendant l'installation d'un produit de serveur Altova sur les machines Windows la machine est automatiquement enregistrée avec LicenseServer, `localhost` est utilisé par

le programme d'installation en tant que le nom de la machine.

Connexion à un LicenseServer par VPN
Si un appareil client se connecte à votre réseau par le biais d'un service Virtual Private Network (VPN), l'appareil client peut se voir adresse une adresse IP dynamiquement, ce qui signifie qu'il peut être identifié en tant qu'un appareil différent à chaque fois qu'il se connecte. Vous trouverez un moyen de résoudre ce problème dans la rubrique Information de réseau.

Demande d'une licence d'évaluation

Vous pouvez obtenir une licence d'évaluation de 30 jours pour chacun des produits Altova installés d'un client qui a été enregistré avec LicenseServer. Cliquer sur le bouton **Request Evaluation Licenses** en haut du panneau de droite (*Product Licensing)*. Un dialogue apparaît contenant une liste des produits de serveur Altova (sur cette machine client) qui ont été enregistrés avec LicenseServer. Veuillez vous assurer que les produits pour lesquels vous souhaitez obtenir une licence d'évaluation sont cochés puis remplissez les champs d'enregistrement et envoyez la demande. Altova vous enverra un e-mail contenant la/les licence/s d'évaluation pour une période de 30 jours. Pour les produits de serveur, le nombre des cœurs pour lesquels la licence est valide sera exactement le nombre requis par le produit au moment de l'envoi de la requête. Enregistrer la/les licence/s sur disque et la/les charger sur le pool de licences.

Désenregistrement de produits

Chaque produit Altova enregistré avec LicenseServer est recensé dans le panneau de droite (*Product Licensing)* sous son nom de machine client et affiche un bouton **Unregister Product** en bas de son entrée. Cliquer sur ce bouton pour désenregistrer le produit depuis LicenseServer. Si une licence a été attribuée au produit, l'attribution sera terminée lorsque le produit sera désenregistré. Pour désenregistrer tous les produits, cliquer sur le bouton **Unregister Client and All Products** situé en haut du panneau de droite (*Product Licensing)* (*voir première capture d'écran dans cette section*).

Pour réenregistrer un produit avec le LicenseServer, procéder comme suit :

- *Produits serveur :* rendez-vous à la page Paramètres dans l'UI Web du produit de serveur. Si le produit de serveur n'a pas d'UI Web, ouvrir une ligne de commande et utiliser la CLI du produit pour l'enregistrer. Les étapes pour chaque produit de serveur sont décrites ici : Enregistrer FlowForce Server, Enregistrer MapForce Server, Enregistrer MobileTogether Server, Enregistrer StyleVision Server et Enregistrer RaptorXML(+XBRL) Server.
- *Produits de bureau* : ouvrir le dialogue d'activation de logiciel (**Help | Software Activation**), passer à l'Activation par le biais du mode LicenseServer et, dans le champ *Altova LicenseServer*, choisir le LicenseServer avec lequel vous souhaitez enregistrer le produit. Le produit sera enregistré et apparaîtra dans l'onglet Client Management du LicenseServer, dans la liste des produits enregistrés de ce client.

,
Pour plus d'informations, voir la section Attribuer des licences à des produits enregistrés.

 © 2018 Altova GmbH

Surveillance des clients

L'onglet **Client Monitoring** fournit un aperçu de l'appareil client sélectionné. L'onglet affiche les éléments suivants :

Clients extraits
Les utilisateurs finaux d'un produit de bureau Altova (pas un produit de serveur), comme XMLSpy ou MapForce, peuvent extraire une licence qui est enregistrées auprès de LicenseServer. Cela peut se produire dans des situations où l'on peut penser que l'appareil de l'utilisateur final sera hors ligne pendant une certaine période. La licence peut être extraire du LicenseServer pour la période pendant laquelle l'appareil sera hors ligne. Pour cette période, l'utilisateur final peut continuer à utiliser le produit de bureau Altova sans prendre contact avec LicenseServer. Les licences et les utilisateurs qui sont actuellement extraits, ainsi que la période d'extraction sont recensés sous ce titre.

Note : L'extraction de licence est effectuée par l'utilisateur final par le biais du dialogue Activation de logiciel (**Aide | Activation de logiciel**) du produit de bureau Altova.

Clients en cours
Une liste des produits Altova actuellement en cours d'exécution sur ce client. Si plusieurs instances d'un produit sont exécutées, chacune de ces instances est recensée.

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	

Note : Failover LicenseServers travaille avec des applications client de version v2015rel3 ou plus récente (dans le cas d'Altova MobileTogether Server, version 1.5 ou plus récente) ; les clients plus vieux sont marqués.

Note : Si une licence pour un produit de bureau est extraite, elle sera affichée dans une section séparée *Checked-out Clients* de l'onglet Surveillance des clients. La licence sera indiquée comme en cours d'exécution et ne pourra pas être utilisée pour une autre machine sur le réseau.

Icônes dans l'onglet Client Monitoring

Show Licenses. (Afficher les informations de licence) Disponible pour chaque instance de produit. Passe à l'onglet License Pool et marque la licence de l'instance de produit sélectionnée.

Manage Client. (Gérer le client) Disponible pour chaque instance de produit. Passe à l'onglet Client Management, et souligne le client de l'instance de produit sélectionné.

 © 2018 Altova GmbH

Paramètres

Cette section :

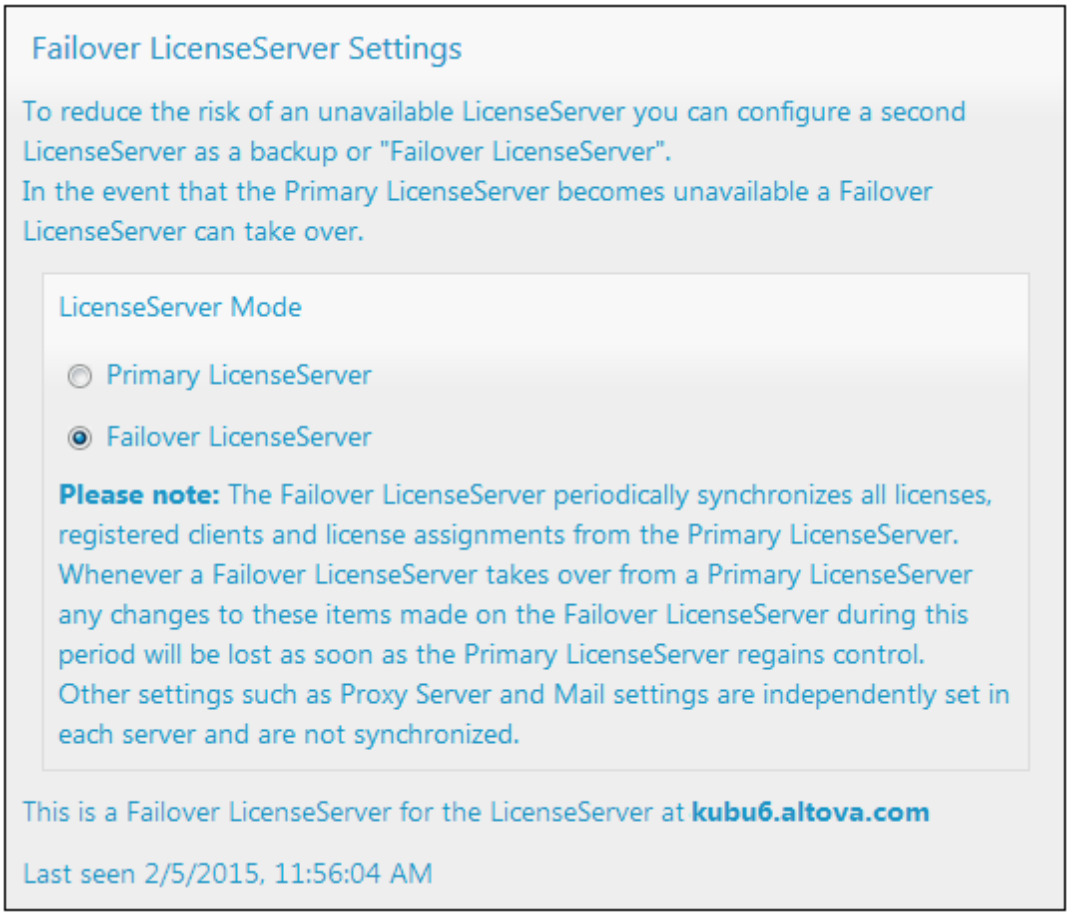
- Paramètres Failover LicenseServer
- Paramètres de réseau
- Paramètres e-mail d'alerte
- Paramètres divers

Ci-dessous vous trouverez la description de l'onglet **Settings** (Paramètres) ci-dessous. Vous pouvez définir les éléments suivants :

- Une **période d'attente jusqu'à la fermeture du LicenseServer**. Une fermeture sera généralement implémentée pour l'entretien du serveur. Le temps jusqu'à la fermeture peut être utilisée pour économiser du temps pour les clients exécutant des produits de bureau Altova. Le temps de fermeture que vous sélectionnez est le temps maximum de fermeture. LicenseServer se fermera plus tôt, dès que LicenseServer ne sera plus connecté à un client exécutant un produit de bureau. Le compte à rebours de la fermeture démarrera lorsque vous cliquerez sur **Shutdown**. Pour annuler la fermeture, cliquer sur **Abort Shutdown**. Afin de permettre aux clients un fonctionnement pendant la fermeture de LicenseServer, configurer un **Failover LicenseServer**.
- Un second LicenseServer peut être configuré pour prendre en charge depuis le Primary LicenseServer sur celui-ci n'est plus disponible. Le second LicenseServer est appelé le **Failover LicenseServer**. Vous trouverez une description de la configuration des paramètres ici.
- Le mot de passe pour la connexion au LicenseServer. Saisir le mot de passe désiré et cliquer sur **Change Password (Changer mot de passe)**.
- Tester la connectivité à Altova en cliquant sur **Test Connection to Altova (Connexion test à Altova)**. Noter que vous devez enregistrer les nouveaux paramètres (en cliquant sur le bouton **Save (Enregistrer)** au bas du panneau) avant de tester la connexion. Le bouton **Test Connection to Altova** est désactivé pendant que le test est en cours et se réactive lorsque le test est terminé.
- Les paramètres de réseau pour la page de configuration (UI Web) à base web, pour le serveur proxy (le cas échéant) utilisé pour établir la connexion à Internet (le cas échéant), et pour la disponibilité de LicenseServer (License Service). Ces paramètres sont décrits dans les Paramètres de réseau ci-dessous.
- Les paramètres de serveur d'e-mail et le destinataire d'e-mail d'alerte à contacter en cas d'une occurrence de LicenseServer particulière. Ces paramètres sont décrits dans les Paramètres de mail d'alerte ci-dessous.
- Après avoir modifié un paramètre, cliquer sur **Save**, en bas du panneau. Veuillez noter que les modification des paramètres ne prendront effet que s'ils sont enregistrés.

Paramètres de LicenseServer Failover

Un second LicenseServer peut être configuré pour reprendre les tâches du LicenseServer primaire si celui-ci n'est plus disponible. Ce second LicenseServer est appelé **Failover LicenseServer**.



Pour configurer un LicenseServer en tant que Failover LicenseServer, procéder comme suit :

1. Installer le LicenseServer comme décrit dans la section d'installation.
2. Configurer le mode de LicenseServer sur *Failover LicenseServer* en activant le bouton radio correspondant (*voir capture d'écran ci-dessus*). (Par défaut, le mode de LicenseServer Mode est défini sur *Primary LicenseServer*.)
3. Dans le dialogue Find the Primary LicenseServer (Trouver le LicenseServer Primaire) qui s'ouvre (*capture d'écran ci-dessous*), saisir le LicenseServer Primaire que vous souhaitez sauvegarder avec ce LicenseServer Failover. Vous pouvez procéder de deux manières : (i) cliquer sur **Search for LicenseServers** puis sélectionner dans la liste de choix le LicenseServer que vous souhaitez sauvegarder depuis la liste des LicenseServers trouvés ; (ii) cliquer sur **Manually Enter Address**, et saisir l'adresse du LicenseServer que vous souhaitez sauvegarder. Après avoir saisi le LicenseServer Primaire, cliquer sur **Connect to Primary LicenseServer**.

 © 2018 Altova GmbH

4. Un dialogue de confirmation apparaît vous demandant si vous souhaitez définir le LicenseServer actuel en tant que le LicenseServer Failover du LicenseServer Primaire que vous venez de sélectionner. Si vous continuez avec la confirmation, vous supprimerez toute licence installée et les clients enregistrés. Cliquer sur **Yes** si vous souhaitez confirmer.

Une fois qu'un LicenseServer Failover a été configuré, le LicenseServer Primaire et LicenseServer Failover présenteront des notifications à propos de leurs modes respectifs dans la partie supérieure de la page de Configuration. Dans les deux captures d'écran ci-dessous, le LicenseServer Failover est montré tout d'abord, suivi du LicenseServer Primaire.

Veuillez noter les points suivants :

- Après la configuration d'un LicenseServer Failover, il synchronisera périodiquement toutes les licences, les clients enregistrés et les accords de licence du Primaire. Si celui-ci n'est plus disponible, le Failover reprend le rôle du LicenseServer. Une fois que le Primaire est à nouveau disponible, il reprend le contrôle à partir du Failover. Toute modification concernant la licence réalisée sur le Failover au cours de cette période sera perdue lorsque le Primaire reprend le service.
- Le LicenseServer Failover fournira des licences uniquement aux clients de version 2015 rel 3 ou plus récente. Les clients plus anciens sont marqués en tant que tels dans l'onglet Client Monitoring du LicenseServer Primaire (*capture d'écran ci-dessous*). Nous recommandons de mettre vos applications client au niveau de la version 2015 rel 3 ou plus récente (dans le cas d'Altova MobileTogether Server, version 1.5 ou plus récente) si vous souhaitez utiliser la fonction LicenseServer Failover (dans le cas d'Altova MobileTogether Server, version 1.5 ou plus récente).

Extraction de licence

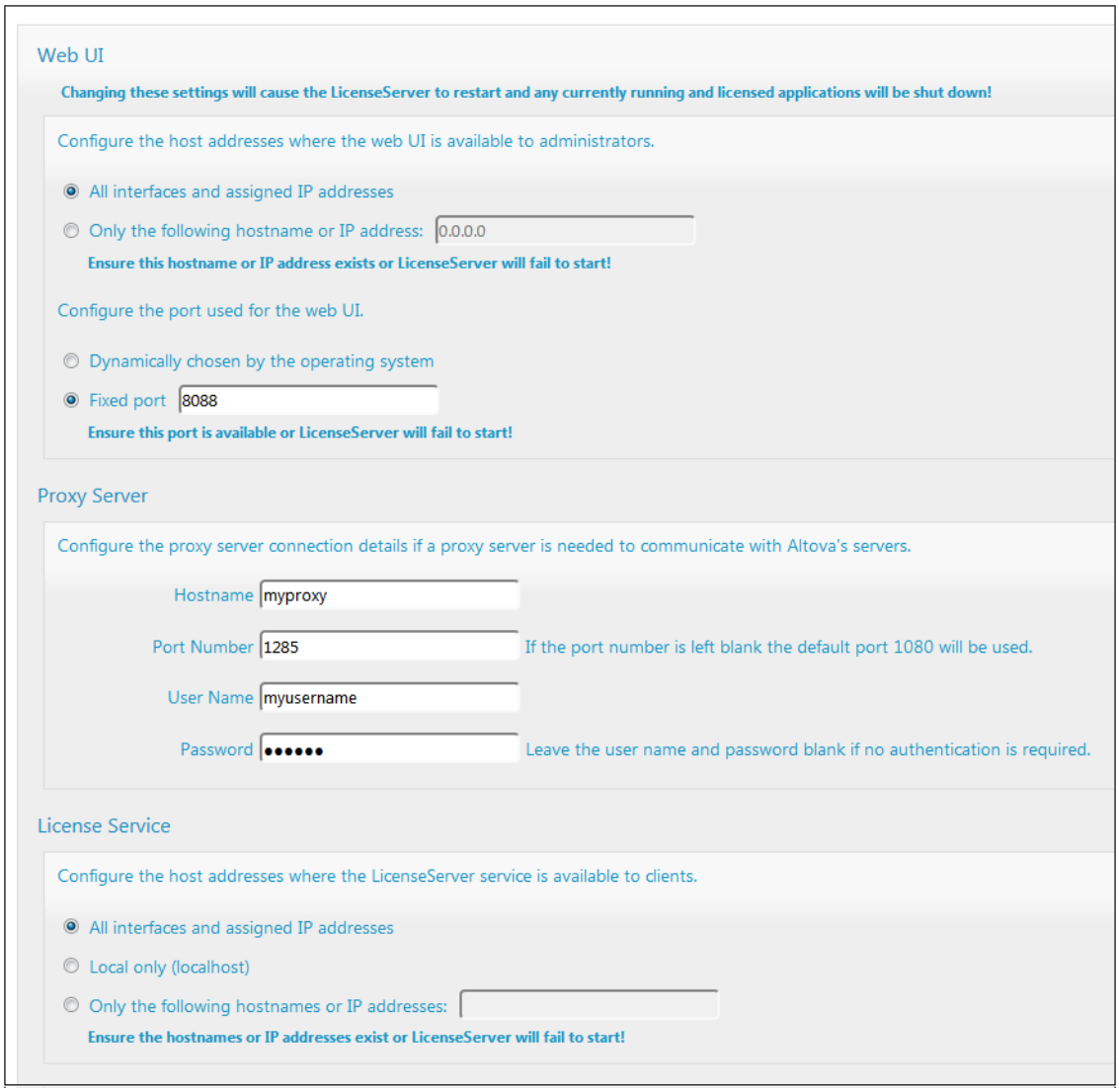
Vous pouvez choisir si vous voulez permettre aux licences de bureau d'être extraites du pool de licences vers l'appareil sur lequel le produit de bureau Altova est installé. Si vous autorisez cette opération, un client qui a acquis une licence depuis votre LicenseServer peut extraire cette licence et demeurer sans contrôle pour une période pouvant aller jusqu'à la période maximum que vous avez définie dans ce paramètre. La période maximum autorisée pour les extractions est de 30 jours. (Cela permet à un client de continuer à utiliser le produit de bureau même lorsque l'accès au LicenseServer n'est pas possible (par ex. pendant les voyages) ou n'est pas souhaité.) Le client peut effectuer l'extraction par le biais de la commande **Aide | Activation du logiciel** de l'application bureau Altova.

Une fois qu'une licence a été extraite, elle est déplacée vers la section *Clients extraits* de l'onglet Surveillance des clients. Elle sera considérée être en utilisation et ne sera pas disponible pour une utilisation par d'autres clients sur le réseau.

Paramètres de réseau

Les administrateurs peuvent spécifier des points d'accès de réseau à la page de configuration de LicenseServer et sur LicenseServer.

 © 2018 Altova GmbH



- *Web UI:* (UI Web) les adresses IP autorisées peuvent varier de toutes les interfaces et adresses IP sur cette machine à une adresse fixe et les ports peuvent être soit calculés dynamiquement soit fixes. Cela permet une gamme étendue d'adresses IP autorisées : Paramètres de port. Le paramètre de port par défaut est **8088**.
- *Proxy Server (Disponible depuis la version v1.3):* (Serveur proxy (disponible à partir de v1.3 et plus) Si un serveur proxy est utilisé pour une connexion à Internet, les détails du serveur proxy doivent être saisis dans le panneau Proxy Server (*voir capture d'écran ci-dessus*). Sinon, les paramètres du serveur proxy peuvent être laissés en blanc. Pour configurer LicenseServer pour utiliser un serveur proxy, saisir le nom d'hôte du serveur proxy et, si nécessaire, un numéro de port. Si le serveur proxy ne nécessite pas d'authentification, alors vous pourrez laisser les champs *User Name (Nom utilisateur)* et *Password (Mot de passe)* en blanc.
- *License Service:* (Service de licence) la machine sur laquelle License Server est installée peut être connectée à plusieurs réseaux par le biais d'une ou plusieurs interfaces de réseau. Sur chaque réseau, la machine License Server est connue par un nom d'hôte et une adresse IP. Le paramètre de *License Service* vous permet de configurer sur quels

réseaux le service de licence est disponible. L'option `localhost` permet un service uniquement sur la machine locale. Si vous établissez une liste des noms d'hôte et/ou des adresses IP, utilisez une liste séparée par des virgules sans aucun espace (par exemple : `hostname1,IPAddress1,hostname2`). Le numéro de port du service est fixé à **35355**.

Les paramètres par défaut permettent un accès sans restrictions au LicenseServer et à sa page de configuration à partir de laquelle les réseaux vers LicenseServer sont connectés. Si vous souhaitez restreindre l'accès soit à LicenseServer soit à sa page de configuration, saisie les paramètres appropriés et cliquer sur **Save**.

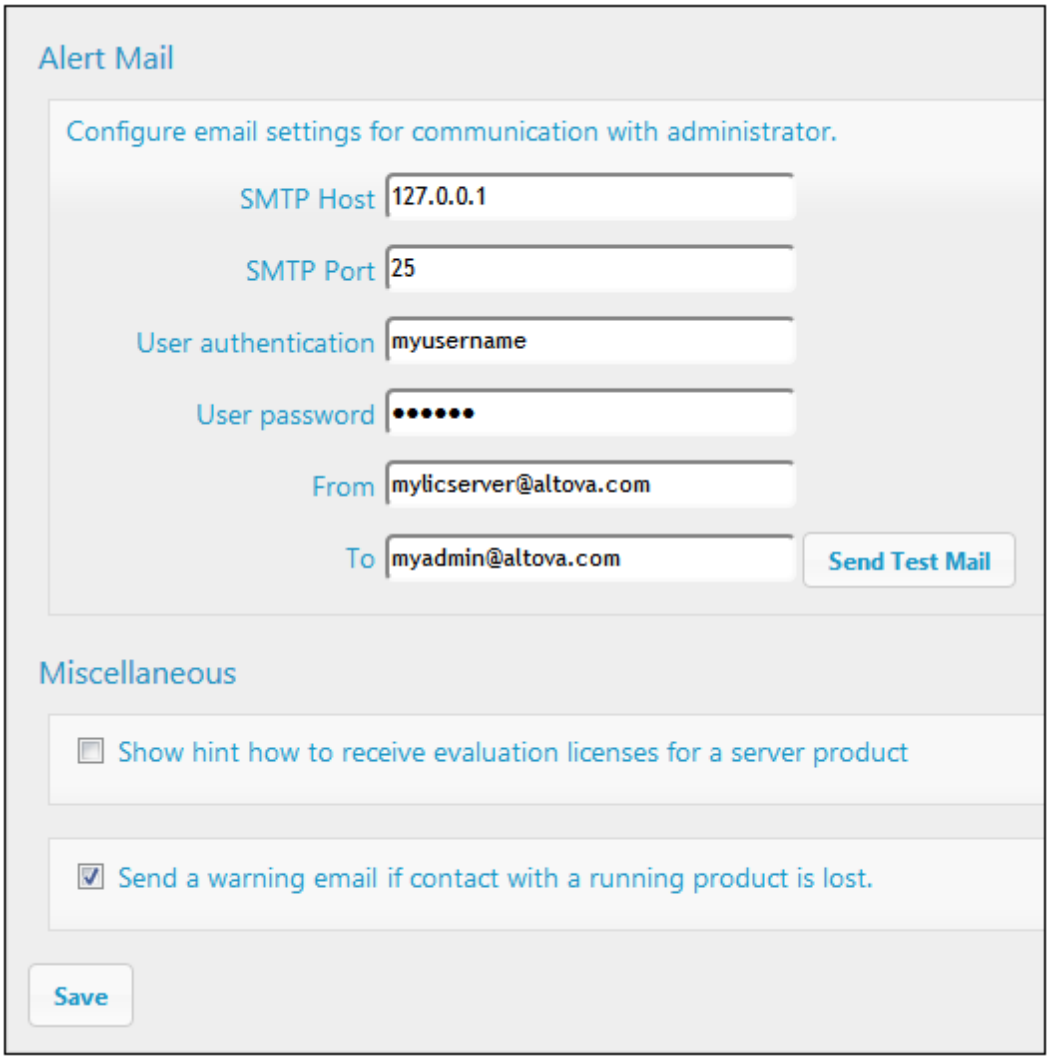
Exécuter un test de connectivité (*voir ci-dessus*) pour vérifier que les paramètres sont corrects.

Paramètres de mail d'alerte

Altova LicenseServerdoit être connecté au serveur `altova.com`. Si la connexion est interrompue pendant plus de 24*5 heures (5 jours), LicenseServer n'autorisera pas les licences. En conséquence, les sessions de travail avec les produits Altova sous licence par LicenseServer pourraient être sujettes à des interruptions.

Il est possible d'envoyer un mail d'alerte à une adresse e-mail pour avertir l'administrateur qu'une connexion est interrompue. Le panneau Alert Mail (*voir capture d'écran ci-dessous*) vous permet de saisir les paramètres pour envoyer les messages d'alerte à une adresse e-mail d'un administrateur.

 © 2018 Altova GmbH



SMTP Host et *SMTP Port* sont les détails d'accès du serveur d'e-mail à partir duquel l'alerte d'e-mail sera envoyée. *User Authentication* et *User Password* sont les identifiants d'utilisateur pour accéder au serveur d'e-mails. Le champ *From (De)* prend l'adresse du compte e-mail à partir duquel l'e-mail sera envoyé. Le champ *To (À)* prend l'adresse e-mail du destinataire.

Cliquer **Save** lorsque vous avez terminé. Après avoir enregistré les paramètres de mail d'alerte, les alertes par e-mail seront envoyées à l'adresse spécifiée à chaque fois qu'un événement particulier se produit, par exemple, lors de la perte d'une connexion à `altova.com`. Veuillez noter que de tels événements sont également enregistrés dans l'onglet Messages où ils peuvent y être consultés.

Paramètres divers

Show hints for receiving and deploying evaluation licenses (Montrer des indices de réception et de déploiement des licences d'évaluation)
Cocher cette case (*voir capture d'écran ci-dessus*) pour afficher en haut de la page de configuration de brèves instructions concernant comment évaluer et déployer les licences d'évaluation.

Send a warning email if contact with a running product is lost (Envoyer un e-mail d'avertissement si un produit en cours d'exécution est perdu)
Un message d'avertissement est envoyé depuis l'adresse *From (De)* à l'adresse *To (À)* si une connexion avec un produit mis sous licence en cours d'exécution est perdue.

 © 2018 Altova GmbH

Messages, Log Out

L'onglet **Messages** affiche tous les messages pertinents aux licences dans le pool de licences du LicenseServer. Chaque message contient un bouton **Delete** (Supprimer) qui vous permet de supprimer ce message.

L'onglet **Log Out** (Déconnexion) sert de bouton de déconnexion. Cliquer sur l'onglet pour vous déconnecter immédiatement, le masque de connexion s'affiche.

11.9 Réinitialisation du mot de passe

Si vous oubliez votre mot de passe LicenseServer, vous pouvez utiliser la commande **`passwordreset`** depuis la CLI pour réinitialiser le mot de passe à `default`.

1. Ouvrir une fenêtre de ligne de commande.
2. Passer au répertoire où le pack d'application de LicenseServer ou le programme d'exécution est installé.
3. Saisir la commande : **`licenseserver passwordreset`**
   Cela permet de réinitialiser le mot de passe d'administrateur de LicenseServer à `default`.
4. Vous pouvez maintenant vous connecter à la UI Web Administrateur avec le mot de passe `default`.

 © 2018 Altova GmbH

Index

A

Adresse IP des clients, 564
Altova LicenseServer,
(voir LicenseServer), 562
Altova ServiceController, 574
API .NET Framework, 317
API de RaptorXML+XBRL Server, 308
API Python, 310
Attribuer des licences, 606, 620

B

bibliothèque Python,
de RaptorXML+XBRL Server, 314

C

Catalogues,
Catalogues XML, 39
Charger les licences, 588, 613
commande pip, 314
Commandes Aide sur CLI, 228
Commandes licence sur CLI, 230
Commandes XQuery, 152
Commandes XSLT, 137
Configuration,
sur Linux, 22
sur Mac OS X, 31
sur Windows, 15
Configuration serveur, 272
Connexions VPN sur LicenseServer, 564

D

Déconnexion, 635
Données graphique,
exemple, 528

E

E-mails d'alerte, 627
Enregistrer FlowForce Server avec LicenseServer, 594
Enregistrer les produits de bureau avec LicenseServer, 593
Enregistrer MapForce Server avec LicenseServer, 599
Enregistrer StyleVision Server avec LicenseServer, 604
Exécution XQuery, 161
Extensions Altova,
fonctions graphiques (voir fonctions graphique), 464

F

fichier de config JSON,
pour module Python RaptorXMLXBRLServer, 314
FlowForce Server,
enregistrer avec LicenseServer, 594
Fonctions d'extension .NET,
aperçu, 548
constructeurs, 551
conversions de type de données .NET en XPath/XQuery, 556
conversions de type de données XPath/XQuery en .NET, 554
méthodes d'instance, champs d'instance, 553
méthodes statiques, champs statiques, 552
pour XSLT et XQuery, 548
Fonctions d'extension dans .NET pour XSLT et XQuery,
voir sous Fonctions d'extension .NET, 548
Fonctions d'extension dans Java pour XSLT et XQuery,
voir sous les fonctions d'extension Java, 536
Fonctions d'extension dans les scripts MSXSL, 558
Fonctions d'extension Java,
aperçu, 536
constructeurs, 542
conversions de type de données XPath/XQuery en Java, 545
conversions de type de données, Java en Xpath/XQuery, 547
fichiers de classe définis par l'utilisateur, 538
fichiers JAR définis par l'utilisateur, 541
méthodes d'instance, champs d'instance, 544
méthodes statique, champs statiques, 543

© 2018 Altova GmbH

Fonctions d'extension Java,
pour XSLT et XQuery, 536
Fonctions d'extension pour XSLT et XQuery, 535
Fonctions graphique,
liste, 517
structure de donnée graphique pour, 522

I

Information de réseau,
connexions via VPN, 564
et adresse IP du client, 564
procédure de connexion, 564
Installation sur Linux, 23
Installation sur Mac OS X, 32
Installation sur Windows, 16
installer module Python RaptorXMLXBRLServer, 314
Interface .NET, 5
Interface administrateur, 612
Interface COM, 5
Interface HTTP, 5, 266
configuration serveur, 268, 272
problèmes de sécurité, 49
requêtes client, 283
Interface Java, 5
Interface Python, 5
Interfaces,
aperçu de, 5

L

Licence sur Linux, 27
Licence sur Mac OS X, 35
Licence sur Windows, 18
Licences,
attribuer, 606, 620
charger, 588, 613
LicenseServer,
enregistrer FlowForce Server avec, 594
enregistrer les produits de bureau avec, 593
enregistrer MapForce Server avec, 599
enregistrer StyleVision Server avec, 604
étapes d'attribution des licences, 575
installation sur Linux, 569
installation sur Mac OS X, 572
installation sur Windows, 567
interface avec, 612
lancer, 577
Page de configuration, 612
paramètres, 627
Ligne de commande,
et XQuery, 152
options, 239
Résumé d'utilisation, 52
Linux,
installation sur, 23
licence sur, 27
Localisation, 236

M

Mac OS X,
installation sur, 32
licence sur, 35
MapForce Server,
enregistrer avec LicenseServer, 599
Messages, 635
Mise à jour, 566
module Python,
deRaptorXML+XBRL Server, 314
Mot de passe,
par défaut au démarrage, 579
Mot de passe par défaut, 579
msxsl:script, 558

N

Nom d'hôte des clients, 564

O

Onglet Server Management, 606, 620
Onglet Server Monitoring, 626

P

Page de configuration, 612
ouvrir sur Linux, 582
ouvrir sur Mac OS X, 585
ouvrir sur Windows, 579
URL de, 579
URL de (Linux), 582
URL de (Mac OS X), 585
Page de configuration LicenseServer,
(voir Configuration page), 579
(voir page de configuration), 582, 585
Paramètres, 627
Paramètres de réseau, 627
Pool de licences, 588, 613
Problèmes de sécurité, 49
Produits de bureau,
enregistrer avec LicenseServer, 593
Python,
problèmes de sécurité, 49

R

RaptorXML,
éditions et interfaces, 5
exigences de système, 7
fonctions, 8
interface HTTP, 5
interface ligne de commande, 5
Interface Python, 5
interfaces avec COM, Java, .NET, 5
introduction, 3
spécifications prises en charge, 10
Ressources globales, 47
RootCatalog.xml, 314

S

Scripts dans XSLT/XQuery,
voir sous Fonctions d'extension, 535
ServiceController, 574
StyleVision Server,
enregistrer avec LicenseServer, 604

T

Transformation XSLT, 138

V

Validation,
de document XSLT, 146
de DTD, 68
de taxonomie XBRL, 118
de XSD, 72
d'instance et de taxonomie XBRL, 88
d'instance XBRL, 89
d'instance XML avec DTD, 57
d'instance XML avec XSD, 61
of XQuery document, 169
Validation de document XSLT, 146
Validation XBRL,
voir Validation, 88
Vérification de la bonne formation, 78

W

Windows,
installation sur, 16
licence sur, 18

X

XQuery, 153
Fonctions d'extension, 535
XQuery document validation, 169
XSLT,
Fonctions d'extension, 535

© 2018 Altova GmbH