Altova Authentic 2023 Desktop

Manuel de l'utilisateur et de référence

Altova Authentic 2023 Desktop
Manuel de l'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: 2022

© 2016-2022 Altova GmbH

Table des matières

1 À propos d'Authentic Desktop et de cette documentation

Altova Authentic 2023 Desktop est une approche visuelle innovante pour permettre à l'utilisateur final de créer des documents XML sans devoir s'occuper des aspects techniques de XML. Authentic Desktop fonctionne sur Windows 7 SP1 avec mise à jour de la plateforme, Windows 8, Windows 10, Windows 11 et Windows Server 2008 R2 SP1 avec mise à jour de la plateforme ou plus récent. Authentic Desktop Enterprise Edition est disponible sur des appareils 64-bit et 32-bit.

Dernière mise à jour : 17.10.2022

© 2016-2022 Altova GmbH

1.1 Chemins de fichier Windows

Chemins de fichiers dans Windows

Les chemins de fichiers dans cette documentation ne seront pas les mêmes pour tous les systèmes d'exploitation. Veuillez les correspondances suivantes :

- (Mon) dossier de documents : Situé par défaut aux emplacements suivants. Les fichiers d'exemple sont situés dans un sous-dossier de ce dossier.

Windows 7/8/10/11	`C:\Users\<username>\Documents`

- *Dossier d'application :* Le dossier d'application est le dossier où votre application d'Altova est situé. Le chemin du dossier d'application est, par défaut, le suivant.

Windows 7/8/10/11	`C:\Program Files\Altova\`
version 32-bit sur 64-bit OS	`C:\Program Files (x86)\Altova\`

Note: Authentic Desktop est aussi pros en charge sur Windows Server 2008 R2 SP1 avec mise à jour de la plateforme ou plus récent.

1.2 À propos de cette documentation

Ce manuel de l'utilisateur contient un tutoriel et une explication sur des fonctions diverses de Authentic View pour que vous puissiez vous initier. Il contient également une section de référence complète qui décrit les fonctions de l'interface. Il consiste en les sections suivantes :

- Une introduction qui décrit la GUI et l'environnement Authentic Desktop.
- Un tutoriel pour vous aider à vous lancer en utilisant Authentic Desktop.
- Une description d'Authentic View, qui est un affichage WYSIWYG d'un document XML. Authentic View permet aux utilisateurs d'écrire et éditer des documents XML comme s'ils étaient de simples documents de texte ou des formulaires interactifs. Le balisage XML est dissimulé des utilisateurs, leur permettant ainsi de se concentrer sur le contenu du document. Authentic View est l'affichage principal de Authentic Desktop.
- Une description du Mode Navigateur, dans lequel le document XML est transformé immédiatement et présenté dans une fenêtre de navigateur.
- Une explication de la fonction Ressources globales, qui permet de passer rapidement d'une ressource à l'autre.
- Des explications pour comprendre comment Authentic Desktop peut être utilisé dans Visual Studio et Eclipse.
- Une Référence de commande de menu qui contient une description des fenêtres et des commandes de menu disponibles dans Authentic Desktop.

© 2016-2022 Altova GmbH

2 GUI et environnement

Cette section décrit :

- L'application GUI, et
- L'environnement d'application.

La section GUI démarre en présentant un aperçu de la GUI, puis décrit chacune des fenêtres de GUI en détail. Elle montre également comment redimensionner, déplacer et travailler avec les fenêtres et la GUI.

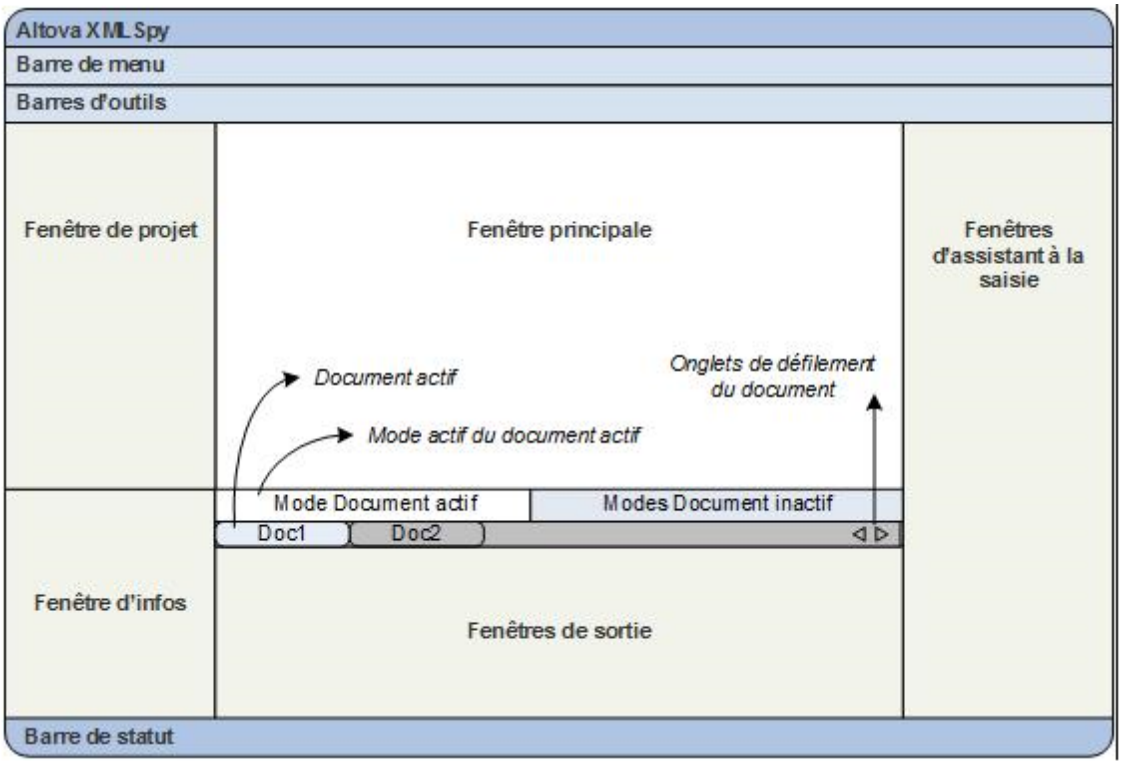
Les points de la section d'environnement de l'application détaillent différents paramètres qui contrôlent comment les fichiers sont affichés et comment ils peuvent être édités. Elle explique aussi comment et où vous pouvez personnaliser votre application. Dans cette section, vous apprendrez où les fichiers d'exemples et de tutoriels ont été installés sur votre appareil et, dans la section, vous pourrez suivre un lien vers le site Internet d'Altova, où vous pourrez explorer la matrice des fonctions de votre application, en savoir plus sur les différents formats de votre manuel, découvrir les options de prises en charge variées à votre disposition et découvrir les autres produits contenus dans la gamme Altova.

2.1 L'Interface graphique utilisateur (GUI)

L'interface graphique utilisateur (GUI) consiste en une fenêtre principale et plusieurs barres latérales (*voir l'illustration ci-dessous*). Par défaut, les barres latérales sont situées autour de la fenêtre principale et sont réparties dans les groupes suivants :

- Fenêtre de projet
- Fenêtre d'info
- Assistants à la saisie : Éléments, attributs, entités, etc. (dépendant du type de document actuellement actif)
- Fenêtre de sortie : Messages

Le fenêtre principale et les barres latérales sont décrites dans les sous-sections de cette section.



La GUI contient aussi une barre de menu, une barre de statut et des barres d'outils, dont toutes sont décrites dans la sous-section de cette section.

Activer et désactiver l'affichage des barres latérales.

Les groupes latérales fenêtres (Fenêtre Projet, Fenêtre Info, Aides à la saisie, Fenêtre de sortie) peuvent être déplacés ou dissimulés en les allumant ou en les éteignant à l'aide des commandes dans le menu **Fenêtre**. Une barre latérale affichée (ou un groupe de barres latérales à onglets) peut être dissimulée en cliquant avec la touche de droite de la souris sur la barre de titre de la barre latérale affichée (ou du groupe de barres latérales à onglets) et en sélectionnant la commande **Dissimuler**.

© 2016-2022 Altova GmbH

Faire flotter et ancrer les barres latérales

Une fenêtre de barre latérale individuelle peut soit flotter librement de la GUI, soit être ancrée à l'intérieur de la GUI. Lorsqu'une fenêtre flottante est ancrée, elle vient se placer à sa dernière position d'ancrage. Une fenêtre peut également être ancrée en tant qu'onglet dans une autre fenêtre.

Pour ancrer ou détacher une fenêtre veuillez suivre l'une des méthodes suivantes :

- Cliquer avec le bouton de droite sur la barre de titre d'une fenêtre puis choisir la commande requise (**Flottant** ou **Ancrage**).
- Double-cliquer la barre de titre de la fenêtre. Si elle était ancrée, la fenêtre flotte à présent. Si elle flotte, la fenêtre s'ancrera sur la position à laquelle elle était fixée précédemment.
- Glisser la fenêtre (en utilisant sa barre de titre en guise de poignée) hors de sa position d'ancrage pour la faire flotter. Faire glisser une fenêtre flottante (par sa barre de titre) vers l'emplacement d'ancrage. Deux groupes de flèches bleues apparaissent. Le groupe extérieur de quatre flèches permet un ancrage par rapport à la fenêtre d'application (le long du rebord supérieur, droit, bas ou gauche de la GUI). Le groupe intérieur de flèches permet un ancrage par rapport à la fenêtre au-dessus de laquelle le curseur est placé actuellement. Si vous déposez une fenêtre glissée sur le bouton au centre du groupe intérieur de flèches (ou sur la barre de titre d'une fenêtre) permet d'ancrer la fenêtre glissée en tant que fenêtre à onglet dans le cadre de la fenêtre dans laquelle elle a été déposée.

Pour détacher une fenêtre à onglet, double-cliquez sur son onglet. Pour faire glisser une fenêtre à onglet hors d'un groupe de fenêtres à onglets, faites-la glisser par son onglet.

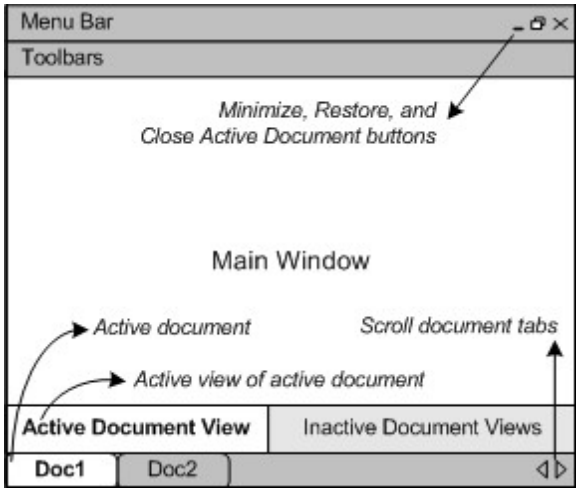
Dissimulation automatique de barres latérales

La fonction dissimulation automatique vous permet de minimiser les barres latérales ancrées à des boutons situés le long de la fenêtre d'application. Vous obtenez plus de place d'écran pour la fenêtre principale et les autres barres latérales. Faire défiler cette barre latérale en faisant défiler une barre latérale minimisée.

Pour dissimuler automatiquement et rétablir les barres latérales, cliquer sur l'icône de la punaise dans la barre de titre de la fenêtre de la barre latérale (ou cliquer avec la barre de titre et choisir **Dissimulation automatique**).

2.1.1 Fenêtre principale

La Fenêtre principale (*capture d'écran ci-dessous*) est l'endroit où vous consultez et éditez les documents.



Fichiers dans la fenêtre principale

- Vous pouvez ouvrir autant de fichiers que vous le souhaitez et les éditer tous à la fois.
- Chaque document ouvert a sa propre fenêtre et un onglet (contenant le nom de fichier du document) en bas de la fenêtre principale. Pour activer un document ouvert, cliquez sur son onglet.
- Si plusieurs fichiers sont ouverts, certains onglets de document peuvent ne pas être visibles ou manquer de place dans la barre des onglets du document. Vous pouvez faire apparaître les onglets en : (i) utilisant les boutons de défilement situés à droite de la barre d'onglet du document, ou (ii) en sélectionnant le document requis depuis la liste en bas du menu Fenêtre.
- Lorsque le document actif est maximisé, ses touches **Minimiser**, **Restaurer**, et **Fermer** sont situés du côté droit de la barre de menu. Lorsqu'un document est ouvert en cascade, en mosaïque ou réduit, les boutons Maximiser, Restaurer et Fermer sont situés dans la barre de titre de la fenêtre du document.
- Lorsque vous maximisez un fichier, tous les fichiers ouverts seront maximisés.
- Les fichiers ouverts peuvent être mis en cascade ou en mosaïque dans le menu Fenêtre.
- Vous pouvez aussi activer les fichiers ouverts dans la séquence dans laquelle ils ont été ouverts en utilisant **Ctrl+Tab** ou **Ctrl+F6**.
- Cliquer de la touche droite de la souris ouvre un menu contextuel avec une sélection de commandes de fichiers, telles que **Imprimer** et **Fermer**.

Modes dans la fenêtre principale

Le document actif peut être affiché et édité dans des modes variés. Les modes disponibles sont affichés dans une barre située au-dessus des onglets de document (*voir illustration ci-dessus*) et le mode actif est marqué. Un mode est sélectionné en cliquant sur le bouton de mode requis ou en utilisant les commandes dans le menu Mode.

Les modes disponibles sont soit des modes d'édition, soit de navigateur :

- Mode Authentic: Pour éditer des documents XML qui sont basés sur StyleVision Power Stylesheet dans l'interface graphique.
- Mode Navigateur: Un mode de navigateur intégré qui prend en charge les deux feuilles de style CSS et XSL.

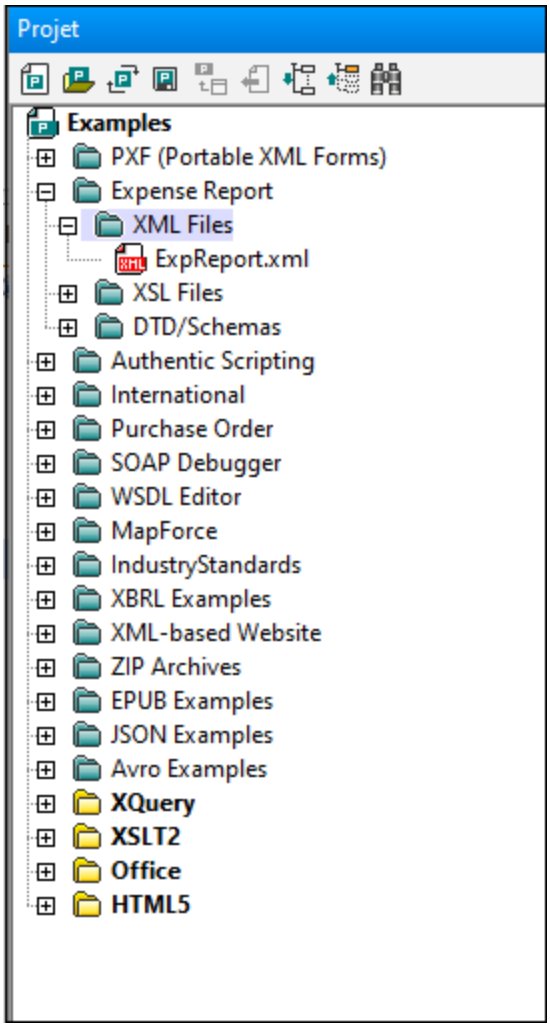
© 2016-2022 Altova GmbH

Note : Le mode par défaut pour les extensions de fichiers individuels peut être personnalisé dans le dialogue **Outils | Options** : dans le panneau Mode par défaut de l'onglet Types de fichiers.

2.1.2 Fenêtre Projet

Un projet est une collection de fichiers qui sont liés les uns aux autres d'une manière que vous déterminez. Par exemple, dans la capture d'écran ci-dessous, un projet nommé `Examples` rassemble les fichiers pour les exemples variés dans des dossiers d'échantillons séparés, qui peuvent eux-mêmes être organisés dans des sous-dossiers. Dans le cadre du projet `Examples`, par exemple, le dossier d'exemple `OrgChart` est encore organisé en sous-dossiers pour les fichiers XML, XSL et Schéma.

Note: La fenêtre Projet de Authentic Desktop contiendra initialement le projet Examples par défaut de l'application. Pour charger le projet par défaut Exemples, rendez-vous dans le dossier **`Examples`** de l'application dans le *dossier (Mes) Documents* , et double-cliquer sur le fichier **`Examples.spp`**.



Les projets vous permettent donc de rassembler des fichiers qui sont utilisés ensemble et d'y accéder plus rapidement. De plus, vous pouvez définir des schémas et des fichiers XSLT pour des dossiers individuels, permettant ainsi le traitement par lot de fichiers dans un dossier.

Opérations du projet

Les commandes permettant l'exploitation du projet sont disponibles dans le menu **Projet**, et certaines commandes sont disponibles dans les menus contextuels du projet et de ses dossiers (cliquer avec la touche de droite pour y accéder). Un sous-ensemble de commandes du menu **Projet** sont également disponibles dans la barre d'outils de la fenêtre de projet, car elles sont fréquemment utilisées (*capture d'écran ci-dessous*).

Les commandes de la barre d'outils sont, de gauche : *Nouveau projet, Ouvrir projet, Recharger projet, Enregistrer projet, Ajouter fichier actif au projet, Sélectionner fichier actif, Élargir tout, Réduire tout, Chercher*. Les noms de ces commandes sont explicites et expliquées dans le menu de projet.

Les opérations clefs liées à la fenêtre de projet sont recensées ci-dessous.

- Un projet à la fois est ouvert dans la Fenêtre Projet. Lorsqu'un nouveau projet est créé ou qu'un projet existant est ouvert, il remplace le projet actuellement ouvert dans la Fenêtre Projet.
- Une fois que des changements ont été effectués à un projet, celui-ci doit être enregistré (en cliquant sur la commande **Projet | Enregistrer projet**). Un projet avec des changements non enregistrés est indiqué avec un astérisque placé à côté de son nom (*voir capture d'écran ci-dessus*).
- Le projet présente une structure arborescente composée de dossier, de fichiers et d'autres ressources. Les autres ressources peuvent être ajoutées à tous les niveaux et à une profondeur illimitée.
- Les fichiers de projet sont des fichiers *sémantiques* qui représente un regroupement logique de fichiers. Ils **ne doivent pas** nécessairement correspondre à une organisation hiérarchique de fichiers sur votre disque dur.
- Les dossiers peuvent correspondre à et avoir une relation directe à des répertoires physiques dans votre système de fichier. Nous appelons de tels dossiers des *dossiers externes*, et ils sont indiqués dans la Fenêtre de Projet par une icône de dossier jaune (contrairement aux dossiers de projet normaux qui sont verts). Les dossiers de projet externes doivent être synchronisés explicitement en utilisant la commande **Réinitialiser**.
- Un dossier peut contenir un mélange arbitraire de types de fichier. En alternative, vous pouvez définir des extensions de type de fichier pour chaque dossier (dans le dialogue de Propriétés de ce dossier) pour garder les fichiers communs dans un endroit pratique. Lorsqu'un fichier est ajouté au dossier parent, il est automatiquement ajouté au sous-dossier qui a été défini pour contenir des fichiers de cette extension de fichier.
- Lorsque vous passez au-dessus d'un fichier d'image file qui a été placé dans le dossier de projet, une prévisualisation de l'image est affichée (formats `.png`, `.jpeg`, `.gif`, `.bmp`, `.tiff`, et `.ico`). Double-cliquez sur l'image pour l'ouvrir dans l'aperçu de l'image par défaut/du programme d'édition du système.
- Dans la Fenêtre de Projet, un dossier peut être glissé vers un autre dossier ou vers un autre emplacement dans le cadre du même dossier alors qu'un fichier peut être glissé vers un autre dossier mais ne peut pas être déplacé à l'intérieur de ce dossier (dans lequel les fichiers sont classés alphabétiquement). De plus, les fichiers et dossiers peuvent être glissés depuis Windows File Explorer vers la fenêtre de projet.
- Chaque dossier a un ensemble de propriétés qui sont définis dans le dialogue Propriétés de ce dossier. Ces propriétés comprennent des extensions de fichier pour le dossier, le schéma avec lequel valider les fichiers XML, le fichier XSLT avec lequel transformer des fichiers XML, etc.

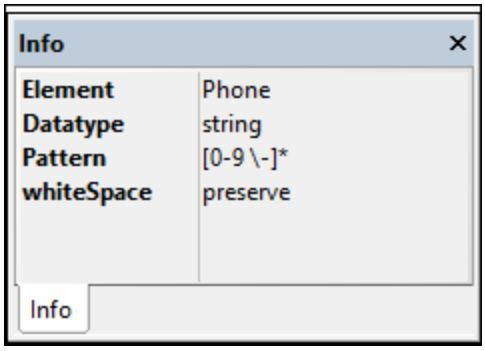
© 2016-2022 Altova GmbH

- Le traitement par lot des fichiers dans un dossier est effectué en cliquant avec la touche de droite sur le dossier et en sélectionnant la commande pertinente depuis le menu contextuel (par exemple, **Valider XML** ou **Vérifier la bonne formation**).

Note : L'affichage de la fenêtre Projet peut être allumé et éteint dans le menu **Fenêtre**.

2.1.3 Fenêtre Info

La Fenêtre Info (*capture d'écran ci-dessous*) montre des informations concernant l'élément ou l'attribut sur lequel le curseur se trouve actuellement.



L'affichage de la Fenêtre Info peut être allumé ou éteint dans le menu **Fenêtre**.

2.1.4 Assistants de saisie

Les assistants à la saisie consistent en une fonction intelligente d'édition qui vous aide à créer des documents XML valides rapidement. Lorsque vous éditez un document, les assistants à la saisie affichent des options d'édition structurelles conformément à l'emplacement actuel du curseur. Les assistants à la saisie obtiennent l'information requise depuis le DTD, Schéma XML sous-jacents et/ou StyleVision Power Stylesheet, etc. Si, par exemple, vous éditez un document de données XML, alors les éléments, attributs et entités pouvant être insérés dans l'emplacement actuel du curseur sont affichés dans les fenêtres des assistants de saisie pertinents.

Note: Vous pouvez allumer ou éteindre l'affichage des assistants de saisie avec l'option de menu **Fenêtre | Assistants de saisie**.

2.1.5 Fenêtre de sortie : Messages

La Fenêtre Messages affiche des messages concernant les actions exécutées dans Authentic Desktop ainsi que les erreurs et autres sorties. Par exemple, si un document XML est validé et est valide, un message de réussite de la validation sera affiché dans la fenêtre Messages. Sinon, un message qui décrit l'erreur est affiché. Veuillez noter les liens (texte de lien noir) vers les nœuds et le contenu de nœud dans le document XML, ainsi que les liens (texte de lien bleu) menant aux sections dans la spécification pertinente pour la règle en question sur Internet.

Valider les dossiers et les fichiers dans la fenêtre Projet

La commande **Valider** (dans le menu XML) est normalement appliquée au document actif. Mais vous pouvez aussi appliquer la commande à un fichier, dossier ou groupe de fichiers dans le projet actif. Choisir le fichier ou le dossier requis dans la fenêtre Projet (en cliquant dessus), et cliquer sur **XML | Valider XML** ou sur **F8**. Les fichiers invalides dans un projet seront ouverts et rendus actifs dans la Fenêtre principale et le message d'erreur *Le fichier n'est pas valide* sera affiché.

Note : Vous pouvez aussi exécuter le contrôle de bonne formation (**Contrôle de bonne formation** ou **F7**) dans la fenêtre Projet.

2.1.6 Barre de menu, barres d'outils, barre de statut

Barre de menu

La barre de menu (*voir illustration*) contient les menus d'application divers. Les conventions suivantes s'appliquent :

- Si les commandes dans un menu ne **s'**appliquent pas dans d'un mode ou dans un emplacement particulier dans le document, elles ne sont pas disponibles.
- Certaines commandes de menu font apparaître un sous-menu avec une liste d'options disponibles. Les sous-commandes de menus sont indiquées par une flèche orientée vers la droite.
- Certaines commandes de menu font apparaître un dialogue qui vous invite à donner des informations supplémentaires requises pour exécuter la commande sélectionnée. De telles commandes sont indiquées par une ellipse (...) après le nom de la commande.
- Pour accéder une commande de menu, cliquez sur le nom du menu et ensuite sur la commande. Si un sous-menu est valable pour un élément de menu, le sous-menu s'ouvrira lorsque vous ferez passer la souris au-dessus du nom de l'élément de menu. Cliquez sur l'item du sous-menu requis.
- Un menu peut être ouvert depuis le clavier en appuyant sur la combinaison de clé appropriée. La combinaison de clé pour chaque menu est **Alt+*KEY***, où ***KEY*** est la lettre soulignée dans le nom de menu. **Par exemple, la combinaison de touches pour le menu** F**ichier** est **Alt+F**.
- Une commande de menu (c'est-à-dire une commande dans un menu) peut être sélectionnée en choisissant consécutivement (i) le menu avec sa combinaison de touches (voir point précédent), et puis (ii) la combinaison de touches pour la commande spécifique (**Alt+*TOUCHE, où*** TOUCHE est la lettre soulignée dans le nom du menu). Par exemple, pour créer un nouveau fichier (**File | New**), appuyez sur **Alt+F**, puis **Alt+N**.
- Certaines commandes de menu peuvent être sélectionnées **directement** en appuyant sur un **raccourci clavier** spécial ou une combinaison de clé (**Ctrl+*KEY***). Les commandes qui ont des raccourcis y associés sont indiquées par un raccourci clavier ou une combinaison de clé recensée à droite de la commande. Par exemple, vous pouvez utiliser la combinaison de touches de raccourci **Ctrl+N** pour créer un nouveau fichier ; ou bien la touche de raccourci **F8** pour valider un fichier XML. Vous pouvez créer vos propres raccourcis dans l'onglet Clavier du dialogue Personnaliser (**Outils | Personnaliser**).

Barres d'outils

Les barres d'outils (*voir illustration*) contiennent des icônes qui sont des raccourcis à la sélection des commandes de menu. Le nom de la commande apparaît en plaçant la souris sur l'icône. Pour exécuter la commande, cliquer sur l'icône.

© 2016-2022 Altova GmbH

Les boutons de barre d'outils sont organisés en groupes. Dans le dialogue **Outils | Personnaliser | Barres d'outils**, vous pouvez spécifier quels groupes de barre d'outils vous souhaitez afficher. Ces paramètres s'appliquent au mode actuel. Pour configurer un paramètre pour un autre mode, veuillez passer à ce mode-là, puis définir les paramètres dans **Outils | Personnaliser | Barres d'outils**. Dans la GUI, vous pouvez glisser les groupes de barre d'outils par leurs poignées (ou barres de titre) vers des emplacements alternatifs à l'écran. Double-cliquer sur la poignée fait que la barre d'outils se détache et flotte ; double-cliquer sur sa barre de titre fait que la barre d'outils s'ancre à son emplacement précédent.

Barre de statut

La barre de statut est située au bas de la fenêtre de l'application (*voir illustration*) et affichera (i) les informations de statut concernant le chargement des fichiers, et (ii) des informations concernant les commandes de menu et les raccourcis de commande dans les barres d'outils si vous placez la souris dessus. Si vous utilisez la version 64-bit de Authentic Desktop, cela est indiqué dans la barre de statut avec le suffixe (x64) placé après le nom de l'application. Il n'y a pas de suffixe pour la version 32-bit.

2.2 L'environnement d'application

Dans cette section, nous décrivons les différents aspects de l'application qui sont importants pour se lancer. La lecture de cette section vous permettra de vous familiariser avec Authentic Desktop et de maîtriser les fonctions les plus importantes. Elle contient des informations importantes concernant les paramètres et la personnalisation que nous vous recommandons de lire pour obtenir un aperçu général de la palette des paramètres et options de personnalisation disponibles et comment les modifier.

Cette section est organisée comme suit :

- *Paramètres et personnalisation :* Décrit comment et où les paramètres et options de personnalisation importants peuvent être définis.
- *Tutoriels, Projets, Exemples :* Note l'emplacement des fichiers non-programmes divers contenus dans le paquet d'application.
- *Fonctions de produit et documentation, et les produits Altova :* Fournit des liens vers le site Internet Altova, où vous pouvez trouver des informations concernant les fonctions du produit, des formats d'aide supplémentaires et d'autres produits Altova.

2.2.1 Paramètres et personnalisation

Cette section fournit un bref aperçu des aspects qui vous permettent de personnaliser Authentic Desktop.

Paramètres
Plusieurs paramètres importants Authentic Desktop sont définis dans différents onglets du dialogue Options. Nous vous recommandons de parcourir les options pour vous familiariser avec les options disponibles.

Personnalisation
Vous pouvez aussi personnaliser les différents aspects de Authentic Desktop, y compris l'apparence de la GUI. Ces options de personnalisations sont disponibles dans le dialogue Personnaliser (**Outils | Personnaliser**. Les différentes options de personnalisation sont décrites dans la section Référence du Menu.

2.2.2 Tutoriels, Projets, Exemples

Le paquet d'installation de Authentic Desktop contient des fichiers de tutoriels, de projets et d'exemple.

Emplacement des fichiers de tutoriels, de projets et d'exemple

Les fichiers de tutoriels, de projets et d'exemple de Authentic Desktop sont installés dans le dossier :

```
C:\Users\<username>\Documents\Altova\Authentic2023\AuthenticExamples\
```

Le dossier `My Documents\Altova\Authentic2023` sera installé pour chaque utilisateur enregistré sur un PC dans le dossier `<username>` de cet utilisateur. Ainsi, sous ce système d'utilisation, chaque utilisateur aura son propre dossier `AuthenticExamples` dans une zone de travail séparée.

© 2016-2022 Altova GmbH

Emplacement des fichiers de tutoriel, de projets et d'exemples

Tous les fichiers de tutoriel, de projet et d'exemple sont situés dans le dossier `AuthenticExamples`.

3 Tutoriel Mode Authentic

Dans le Authentic View, vous pouvez éditer les documents XML dans une interface graphique WYSIWYG (*capture d'écran ci-dessous*), comme dans les applications de traitement de texte de style Microsoft Word. De fait, il vous suffit de saisir des données. Aucun besoin de vous soucier du formatage du document, le formatage est déjà défini dans la feuille de style qui contrôle le Authentic View du document XML. La feuille de style (StyleVision Power Stylesheet, abrégée en SPS dans ce tutoriel) est créée par un designer de feuille de style utilisant le produit StyleVision d'Altova.

L'édition d'un document XML dans le Authentic View exige deux actions de la part de l'utilisateur : (i) l'édition de la structure du document (par exemple, l'ajout ou la suppression de parties du document, comme des paragraphes et des en-têtes) ; et (ii) la saisie de données (le contenu de parties de documents).

Ce tutoriel vous entraîne dans les étapes suivantes :

- Ouvrir un document XML dans le Authentic View. L'exigence principale pour l'édition dans le Authentic View est que le document XML soit associé à un fichier SPS.
- Un aperçu de l'interface du Authentic View et une description générale des mécanismes d'édition centraux.
- Éditer la structure de document en insérant et en supprimant les nœuds.
- Saisir des données dans le document XML.
- Saisir (i) des valeurs d'attribut par le biais de l'Assistant à la saisie Attributs, et les (ii) valeurs d'entité.
- Imprimer le document.

© 2016-2022 Altova GmbH

Ce tutoriel a été conçu pour vous aider à vous lancer, il a donc été rédigé de manière intentionnellement simple. Vous trouverez des références supplémentaires et des descriptions de fonctions plus poussées dans la section interface du Mode Authentic.

Exigences du tutoriel

Tous **les fichiers** dont vous aurez besoin pour le tutoriel se trouvent dans le dossier `AuthenticExamples` de votre dossier d'application Altova. Ces fichiers sont :

- `NanonullOrg.xml` (le document XML que vous allez ouvrir)
- `NanonullOrg.sps` (la StyleVision Power Stylesheet à laquelle le document XML est lié)
- `NanonullOrg.xsd` (le Schéma XML sur lequel le document XML et la StyleVision Power Stylesheet sont basés et auxquels ils sont liés)
- `nanonull.gif` and `Altova_right_300.gif` (deux fichiers d'image utilisés dans le tutoriel)

Note : au cours du tutoriel, nous vous demanderons de regarder le texte du document XML (contrairement au Authentic View du document). Si l'édition de produit Altova que vous utilisez ne comprend pas un Mode Texte (comme c'est le cas avec Authentic Desktop et Authentic Browser), utiliser un **éditeur de texte** courant comme Wordpad ou Notepad pour consulter le texte du document XML.

Attention : nous vous recommandons d'utiliser une copie de `NanonullOrg.xml` pour le tutoriel, ainsi vous pourrez toujours récupérer l'original, le cas échéant.

3.1 Ouvrir un document XML dans le Mode Authentic

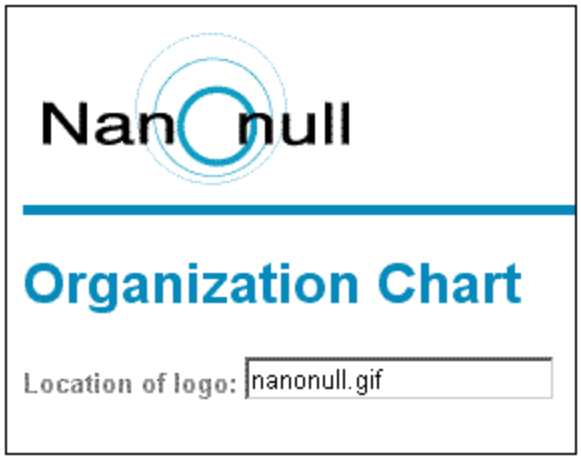
Dans le Authentic View, vous pouvez éditer un document XML existant ou créer et éditer un nouveau document XML. Dans ce tutoriel, vous allez ouvrir un document XML existant dans le Authentic View (décrit dans cette section) et apprendre comment l'éditer (sections suivantes). En outre, vous trouverez dans cette section une description pour créer un nouveau document XML pour une édition dans le Authentic View.

Ouvrir un document XML existant

Le fichier que vous allez ouvrir est nommé `NanonullOrg.xml`. Il se trouve dans le dossier `AuthenticExamples` de votre application Altova. Vous pouvez ouvrir `NanonullOrg.xml` de deux manières :

- Cliquer sur **Fichier | Ouvrir** dans votre produit Altova, puis chercher `NanonullOrg.xml` dans le dialogue qui apparaît et cliquer sur **Ouvrir**.
- Utiliser Windows Explorer pour localiser le fichier, cliquer avec la touche de droite et choisir votre produit Altova en guise d'application avec laquelle ouvrir le fichier.

Le fichier `NanonullOrg.xml` s'ouvre directement dans Authentic View (*capture d'écran ci-dessous).*



Attention : c'est la SPS qui définit et contrôle comment un document XML est affiché dans le Authentic View. Sans une SPS, il ne peut pas y avoir de Authentic View du document.

Créer un nouveau document XML sur la base d'une SPS

Vous pouvez aussi créer un nouveau document XML qui se base sur une SPS. Vous avez deux possibilités : par le biais de la commande de menu **Fichier | Nouveau** et par le biais de la commande de menu **Authentic | Nouveau document**. Dans les deux cas, une SPS est sélectionnée.

Via Fichier | Nouveau

1. Sélectionner **Fichier | Nouveau**.
2. Dans le dialogue Créer un nouveau document, chercher la SPS désirée.

Si un fichier de modèle XML a été attribué à la SPS, les données dans le Fichier de modèle XML sont utilisées en tant que données de départ du modèle de document XML créé dans Authentic View.

© 2016-2022 Altova GmbH

Via Authentic | Nouveau document

1. Sélectionner **Authentic | Nouveau document.**
2. Dans le dialogue Créer un nouveau document, chercher la SPS désirée.

Si un Fichier de modèle XML a été attribué à la SPS, les données dans le Fichier de modèle XML sont utilisées en tant que données de départ du modèle de document XML créé dans Authentic View.

3.2 L'interface du Mode Authentic

L'interface d'édition du Authentic View consiste en une fenêtre principale dans laquelle vous saisissez et éditez les données de document, et trois assistants à la saisie. L'édition d'un document est très simple. Si vous souhaitez voir les balises du document, activez les balises. Ensuite, vous pouvez commencer à saisir du contenu dans votre document. Pour modifier la structure du document, vous pouvez utiliser soit le menu contextuel soit l'Assistant à la saisie Éléments.

Afficher les balises de nœud XML (balises de document)

Un document XML est essentiellement une hiérarchie de nœuds. Par exemple :

```
<DocumentRoot>
   <Person id="ABC001">
       <Name>Alpha Beta</Name>
       <Address>Some Address</Address>
       <Tel>1234567</Tel>
   </Person>
</DocumentRoot>
```

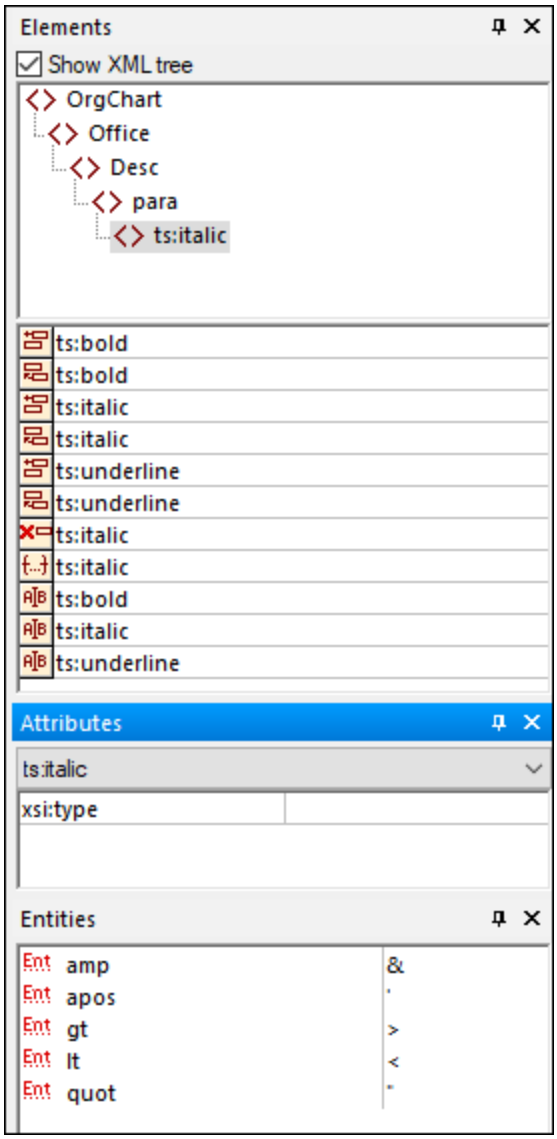
Par défaut, les balises de nœud ne sont pas affichées dans le Authentic View. Vous pouvez activer les balises de nœud en sélectionnant l'item de menu **Authentic | Afficher grande balise** (ou l'icône de barre d'outils ). Les grandes balises contiennent les noms des nœuds respectifs. En alternative, vous pouvez sélectionner des petites balises (sans nom de code dans la balise) et des balises mixtes (un mélange de grandes balises, de petites balises et de balises sans balises. Cela est défini par le designer de la feuille de style ; la balise mixte par défaut pour le document est "sans balise").

Vous pouvez consulter le texte du document XML dans le Mode Texte de votre produit Altova ou dans un éditeur de texte.

Assistant à la saisie

Il existe trois assistants à la saisie dans l'interface (*capture d'écran ci-dessous*), situés par défaut le long de la bordure droite de la fenêtre d'application. Il s'agit des assistants à la saisie Éléments, Attributs et Entité.

© 2016-2022 Altova GmbH



Assistant à la saisie Éléments

L'assistant à la saisie Éléments affiche les éléments qui peuvent être insérés et supprimés avec référence à l'emplacement actuel du curseur ou à la sélection dans la Fenêtre principale. Veuillez noter que l'assistant à la saisie est sensible au contexte ; son contenu change selon l'emplacement du curseur ou de la sélection. Le contenu de l'assistant à la saisie peut être modifié d'une autre manière : lorsqu'un autre nœud est sélectionné dans l'arborescence XML de l'assistant à la saisie Éléments, les éléments pertinents pour ce nœud sont affichés dans l'assistant à la saisie. L'assistant à la saisie Éléments peut être agrandi pour montrer l'arborescence XML en cochant la case Afficher Arborescence XML située en haut de l'assistant à la saisie (*voir capture d'écran ci-dessus*). L'arborescence XML montre la hiérarchie des nœuds depuis le nœud d'élément de niveau supérieur jusque dans le nœud sélectionné dans la Fenêtre principale.

Assistant à la saisie Attributs

L'assistant à la saisie Attributs affiche les attributs de l'élément sélectionné dans la Fenêtre principale et les valeurs de ces attributs. Les valeurs de l'attribut peuvent être saisies dans l'assistant à la saisie Attributs. Les nœuds de l'élément depuis l'élément de niveau supérieur jusqu'à l'élément sélectionné sont disponibles pour la sélection dans la liste de choix de l'assistant à la saisie Attributs. La sélection d'un élément depuis la liste

déroulante de la liste de choix entraîne l'affichage des attributs de cet élément dans l'assistant à la saisie, où ils peuvent ensuite être édités.

Assistant à la saisie Entités
L'assistant à la saisie Entités n'est pas sensible au contexte et affiche toutes les entités déclarées pour le document. Double-cliquer sur une entité pour l'insérer au niveau de l'emplacement du curseur. La description de l'ajout d'entités à un document est traitée dans la section interface du Mode Authentic.

Menu contextuel

Cliquer avec la touche de droite sur un emplacement dans le document du Authentic View pour ouvrir un menu contextuel pertinent à cet emplacement (du nœud). Le menu contextuel propose des commandes qui vous permettent de :

- Insérer des nœuds à cet emplacement, ou avant ou après le nœud sélectionné. Les sous-menus affichent les listes des nœuds qui sont autorisés au niveau des emplacements d'insertion respectifs.
- Supprimer le nœud sélectionné (si cela est autorisé par le schéma) ou tout élément d'ancêtre amovible. Les nœuds qui peuvent être supprimés (conformément au schéma) sont recensés dans un sous-menu.
- Insérer des entités et des sections CDATA. Les entités déclarées pour le document sont recensées dans un sous-menu. Les sections CDATA peuvent uniquement être insérées dans du texte.
- Couper, copier, coller (y compris coller en tant que XML ou texte), et supprimer du contenu de document.

Note: pour plus de détails concernant l'interface, voir interface du Mode Authentic

© 2016-2022 Altova GmbH

3.3 Opérations de nœud

Il existe deux types principaux de nœuds que vous aurez l'occasion de rencontrer dans un document XML Authentic View : **les nœuds d'élément** et **les nœuds d'attribut**. Ces nœuds sont marqués par des balises que vous pouvez activer. Il existe d'autres nœuds dans le document, comme les nœuds de texte (qui ne sont pas balisés) et des nœuds de section CDATA (qui sont balisés, afin de les délimiter du texte environnant).

Les opérations de nœud décrites dans cette section se réfèrent uniquement aux nœuds d'élément et aux nœuds d'attribut. Lorsque vous tentez les opérations décrites dans cette section, il est conseillé d'activer les grandes balises.

Note : il est important de garder en tête que seuls des **éléments de même niveau ou de niveau plus élevé** peuvent être insérés avant ou après l'élément sélectionné. Les éléments de même niveau sont des **frères**. Les frères d'un élément paragraphe seront d'autres éléments paragraphes, mais ils pourraient aussi être des listes, une table, une image, etc. Des frères peuvent se produire avant ou après un élément. Les éléments d'un niveau plus élevé sont des éléments **ancêtres** et les frères d'ancêtres. Pour un élément paragraphe, des éléments ancêtres peuvent être une section, un chapitre, un article, etc. Un paragraphe dans un fichier XML valide aurait déjà des ancêtres. C'est pourquoi l'ajout d'un élément de niveau supérieur dans le Authentic View crée le nouvel élément en tant que frère de l'ancêtre pertinent. Par exemple, si un élément section est inséré après un paragraphe, il est créé en tant que frère de la section qui contient l'élément de paragraphe actuel.

Effectuer des opérations de nœud

Les opérations de nœud peuvent être effectuées en sélectionnant une commande dans le menu contextuel ou en cliquant sur l'entrée d'opération de nœud dans l'Assistant à la saisie Éléments. Dans certains cas, un élément ou un attribut peut être ajouté en cliquant sur le lien Ajouter le nœud dans le Authentic View du document. Dans les cas particuliers des éléments définis en tant que paragraphes ou d'item de liste, si vous appuyez sur la touche Entrée lorsque vous vous trouvez dans un tel élément, vous créerez un nouvel élément frère de ce type. Cette section décrit aussi comment les nœuds peuvent être créés et supprimés en utilisant les mécanismes Appliquer l'élément, Supprimer le nœud, et Effacer l'élément.

Insérer les éléments

Des éléments peuvent être insérés dans les emplacements suivants :

- L'emplacement de curseur dans un nœud d'élément. Les éléments disponibles pour l'insertion à cet endroit sont recensés dans un sous-menu de la commande **Insérer** du menu contextuel. Dans l'assistant à la saisie Éléments, les éléments qui peuvent être insérés dans un emplacement sont indiqués par l'icône  . Dans le document `NanonullOrg.xml`, placer le curseur dans l'élément `para`, et créer des éléments `bold` et `italic` à l'aide du menu contextuel et de l'assistant à la saisie Éléments.
- Avant ou après l'élément sélectionné ou un de ses ancêtres, si autorisé par le schéma. Sélectionner l'élément requis depuis le/s sous-menu/s qui apparait/ssent. Dans l'assistant à la saisie Éléments, les éléments qui peuvent être insérés avant ou après l'élément sélectionné sont indiqués avec les icônes  et  , respectivement. Veuillez noter que dans l'assistant à la saisie Éléments, vous pouvez uniquement insérer des éléments avant/après l'élément sélectionné ; vous ne pouvez pas insérer avant/après un élément ancêtre. Pour essayer cette commande, placez tout d'abord le curseur dans l'élément `para` puis dans la table recensant les employés.

Ajouter le lien de nœud

Si un élément ou un attribut est inclus dans le design de document, et qu'il n'est pas présent dans le document XML, un lien `Ajouter lien` s'affichera dans l'emplacement du document à l'endroit où ce nœud est spécifié. Pour voir ce lien, sélectionner, dans la ligne contenant le texte *Emplacement du logo*, le nœud `@href` dans l'élément `CompanyLogo` et le supprimer (en appuyant sur la touche **Supprimer**). Le lien `add @href` apparaît dans l'élément `CompanyLogo` qui a été édité (*capture d'écran ci-dessous*). Cliquer sur le lien pour ajouter le nœud `@href` au document XML. La fenêtre de saisie dans les balises `@href` apparaît parce que le design spécifie que le nœud `@href` sera ajouté comme cela. Vous devrez tout de même saisir la valeur (ou le contenu) du nœud `@href`. Saisir le texte `nanonull.gif`.

Si le modèle de contenu d'un élément est ambigu, par exemple s'il spécifie qu'une séquence des éléments enfant peut apparaître dans n'importe quel ordre, alors le lien `add...` apparaît. Veuillez noter qu'aucun nom de nœud n'est spécifié. Cliquer sur le lien pour faire apparaître une liste des éléments qui peuvent être insérés validement.

Note : le lien Ajouter nœud apparaît directement dans le modèle du document ; il n'y a pas d'entrée correspondante dans le menu contextuel ou dans l'assistant à la saisie Éléments.

Créer de nouveaux éléments avec la touche Entrée

Dans les cas où un élément a été formaté en tant que paragraphe ou item de liste (par le designer de feuille de style), appuyer sur la touche Entrée lorsque vous vous trouvez dans un tel nœud pour insérer un nouveau nœud de ce type après le nœud actuel. Vous pouvez essayer ce mécanisme dans le document `NanonullOrg.xml` en vous rendant à la fin d'un nœud `para` (juste avant sa balise de fin) et en appuyant sur **Entrée**.

Appliquer des éléments

Dans les éléments de contenu mixte (ceux qui contiennent du texte et des éléments enfant), certains contenus de texte peuvent être sélectionnés et un élément enfant autorisé peut y être appliqué. Le texte sélectionné devient le contenu de l'élément appliqué. Pour appliquer des éléments, dans le menu contextuel, sélectionner **Appliquer** puis sélectionner depuis les éléments applicables. (Si aucun élément ne peut être appliqué au texte sélectionné, la commande **Appliquer** n'apparaît pas dans le menu contextuel). Dans l'assistant à la saisie Éléments, les éléments qui peuvent être appliqués pour une sélection sont indiqués par l'icône . Dans le document `NanonullOrg.xml`, sélectionner le texte se trouvant dans l'élément `para` de contenu mixte et faire des essais en appliquant les éléments `bold` et `italic`.

Le designer de feuille de style peut aussi avoir créé une icône de barre d'outils à appliquer à un élément. Dans le document `NanonullOrg.xml`, les éléments `bold` et `italic` peuvent être appliqués en cliquant sur les icônes grasses et italiques dans la barre d'outils Authentic de l'application.

© 2016-2022 Altova GmbH

Supprimer les nœuds

Un nœud peut être supprimé si sa suppression ne rend pas le document invalide. La suppression d'un nœud entraîne également la suppression de tout son contenu. Un nœud peut être supprimé à l'aide de la commande **Supprimer** dans le menu contextuel. Lorsque la commande Supprimer est marquée, un sous-menu contenant tous les nœuds pouvant être supprimés s'ouvre, en commençant par le nœud sélectionné et jusqu'au nœud de niveau supérieur du document. Pour sélectionner un nœud à supprimer, placer le curseur dans le nœud, ou bien marquer le nœud (ou une de ses parties). Dans assistant à la saisie Éléments, les nœuds qui peuvent être supprimés sont indiqués par l'icône  . Il est également possible de supprimer un nœud en le sélectionnant et en appuyant sur la touche **Supprimer**. Dans le document `NanonullOrg.xml`, faites des essais en supprimant quelques nœuds à l'aide des mécanismes décrits. Vous pouvez annuler vos changements avec **Ctrl+Z**.

Effacer les éléments

Les nœuds d'élément qui sont les enfants d'éléments à contenu mixte (aussi bien enfants de texte et d'élément) peuvent être effacés. L'élément entier peut être effacé lorsque le nœud est sélectionné ou lorsque le curseur est placé dans le nœud en tant qu'un point d'insertion. Un fragment de texte dans l'élément peut être effacé de la balise de l'élément en marquant le fragment de texte. Une fois la sélection effectuée, sélectionner **Effacer** dans le menu contextuel et puis l'élément à supprimer. Dans l' assistant à la saisie Éléments, les éléments qui peuvent être effacés pour une sélection particulière sont indiqués avec l'icône  (sélection de point d'insertion) et l'icône  (sélection de la plage). Dans le document `NanonullOrg.xml`, essayer le mécanisme d'effacement avec les élément enfants `bold` et `italic` de `para` (qui présente le contenu mixte).

Tables et structure de table

Il y a deux types de table Authentic View :

- *Tables SPS (statique et dynamique)*. La structure générale de la table SPS est déterminée par le designer de feuille de style. Dans le cadre de cette structure large, les seuls changements structurels que vous êtes autorisé à effectuer sont les changements orientés vers le contenu. Par exemple, vous pouvez ajouter de nouvelles lignes à une table SPS dynamique.
- *Tables XML*, dans lesquelles vous décidez de présenter les contenus d'un nœud particulier (par ex. un nœud pour les détails spécifiques aux personnes) en tant que table. Si le designer de feuille de style a activé la création de ce nœud en tant qu'une table XML, vous pouvez déterminer la structure de la table et éditer ses contenus. Les tables XML sont discutées en détail dans la section Tables dans le Mode Authentic.

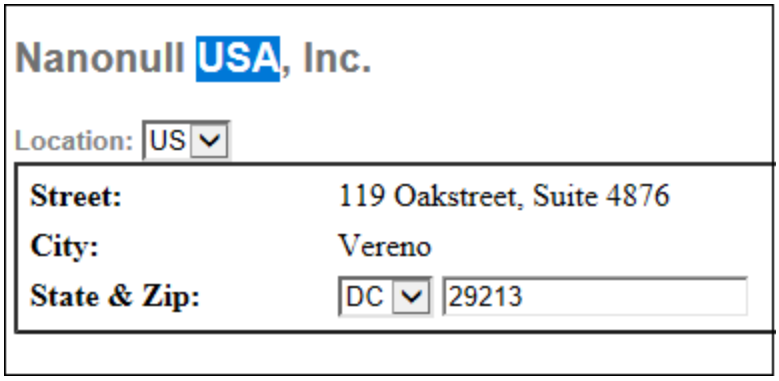
3.4 Saisir des données dans le Mode Authentic

Les données sont saisies dans le document XML, directement dans la fenêtre principale du Authentic View. De plus, des données (la valeur de l'attribut) peuvent être saisies dans l'assistant à la saisie Attributs pour les attributs. Les données sont saisies (i) directement en tant que texte, ou (ii) en sélectionnant une option dans un appareil de saisie des données qui est ensuite mappé à une entrée de texte prédéfinie.

Ajouter le contenu de texte

Vous pouvez entrer le contenu d'élément et les valeurs d'attribut directement en tant que texte dans la fenêtre principale du Authentic View. Pour insérer du contenu, placer le curseur à l'emplacement où vous souhaitez insérer le texte, puis le saisir. Vous pouvez aussi copier du texte depuis le presse-papiers dans le document. Le contenu peut aussi être édité à l'aide des mécanismes d'édition standard, comme les touches **Verr.** et **Supprimer**. Par exemple, vous pouvez marquer le texte à éditer et saisir un texte de remplacement avec la touche **Verr.** activée.

Par exemple, pour changer le nom de l'entreprise, dans le champ `Name` de `Office`, placer le curseur après Nanonull, et saisir `USA` pour changer le nom de Nanonull, Inc. en Nanonull USA, Inc.



Si le texte est éditable, vous pourrez placer votre curseur dessus et le marquer. Essayez de modifier l'un des **noms de champ** (pas les valeurs de champ), comme "Street", "City", ou "State/Zip," dans le bloc d'adresse. Vous ne pourrez pas placer le curseur dans ce texte car celui-là ne fait pas partie du contenu XML ; il est dérivé de la StyleVision Power Stylesheet.

Insérer des caractères spéciaux et des entités

Lorsque vous saisissez des données, le type suivant de contenu est géré de manière spéciale :

- *Les caractères spéciaux qui sont utilisés pour des balises XML* (esperluette (&), apostrophe ('), supérieur à (>), inférieur à (<) et guillemets (")). Ces caractères sont disponibles en tant qu'entités intégrées et peuvent être saisies dans le document en double-cliquant l'entité respective dans l'assistant à la saisie Entités. Si ces caractères se produisent fréquemment (par exemple dans les listes de code de programme), ils peuvent être saisis dans les sections CDATA. Pour insérer une section CDATA, cliquer avec la touche de droite dans l'emplacement où vous souhaitez saisir la section CDATA, et sélectionner **Insérer Section CDATA** depuis le menu contextuel. Le processeur XML ignore tous les caractères de balise dans les sections CDATA. Cela signifie également que si vous souhaitez insérer un caractère spécial dans une section CDATA, vous devrez saisir ce caractère et non pas sa référence d'entité.

© 2016-2022 Altova GmbH

- *Les caractères spéciaux qui ne peuvent pas être saisis par le biais du clavier* doivent être saisis en les copiant depuis le mappage de caractères de votre système vers l'emplacement souhaité dans le document.
- *Une chaîne de texte fréquemment utilisée* peut être définie en tant qu'entité, qui apparaît dans l'Assistant à la saisie Entités. L'entité est insérée dans l'emplacement requis en plaçant le curseur dans chaque emplacement requis et en double-cliquant l'entité dans l'assistant à la saisie. Cela est utile pour l'entretien parce que la valeur de la chaîne de texte est maintenue dans un emplacement ; si la valeur doit être changée, il suffit de changer la définition de l'entité.

Note : Lorsque la balise est dissimulée dans le Authentic View, un élément vide peut facilement être ignoré. Pour vous assurer de ne pas oublier un élément vide, activer les grandes ou les petites balises.

Essayez d'utiliser chaque type de contenu de texte décrit ci-dessus.

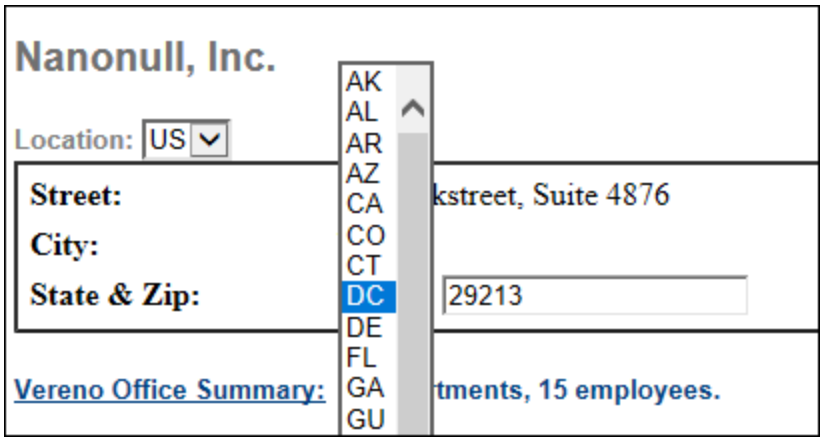
Ajouter un contenu par le biais d'un appareil de saisie de données

Dans l'édition de contenu que vous avez appris ci-dessus, le contenu est ajouté en saisissant le texte directement en tant que contenu. Vous disposez également d'un autre moyen pour saisir un **contenu d'élément** (ou de valeurs d'attribut) dans Authentic View : par le bais d'appareils de saisie de données.

Ci-dessous, vous trouverez une liste des appareils de saisie de données dans le Authentic View, complétée d'une explication pour saisir les données dans le fichier XML pour chaque appareil.

Appareil de saisie de données	Données dans le fichier XML
Champ de saisie (fenêtre Texte)	Texte saisi par l'utilisateur
Champ de saisie multiligne	Texte saisi par l'utilisateur
Liste de choix	Sélection de l'utilisateur mappé à la valeur
Case à cocher	Sélection de l'utilisateur mappé à la valeur
Bouton radio	Sélection de l'utilisateur mappé à la valeur
Bouton	Sélection de l'utilisateur mappé à la valeur

Dans la table statique contenant les champs d'adresse (*affiché ci-dessous*), il y a deux appareils de saisie de données : un champ d'entrée pour le champ `Zip` et une liste de choix pour le champ State. Les valeurs que vous saisissez dans les champs de texte sont saisies directement en tant que le contenu XML des éléments respectifs. Pour tous les autres appareils de saisie des données, votre sélection est mappée à une valeur.



Voici le texte XML correspondant au Authentic View affiché ci-dessus :

```
<Address>
  <ipo:street>119 Oakstreet, Suite 4876</ipo:street>
  <ipo:city>Vereno</ipo:city>
  <ipo:state>DC</ipo:state>
  <ipo:zip>29213</ipo:zip>
</Address>
```

Veuillez noter que la sélection de liste de choix `DC` est mappée à une valeur de `DC`. La valeur du champ `Zip` est saisie directement en tant que le contenu de l'élément `ipo:zip`.

© 2016-2022 Altova GmbH

3.5 Saisir des valeurs d'attribut

Un attribut est la propriété d'un élément et un élément peut présenter un nombre quelconque d'attributs. Les attributs ont des valeurs. Vous serez parfois amené à saisir des données XML en tant que valeur d'attribut. Dans le Authentic View, vous pouvez saisir des valeurs d'attribut de deux manières :

- En tant que contenu dans la fenêtre principale si l'attribut a été créé pour accepter sa valeur de cette manière
- Dans l'assistant à la saisie Attributs

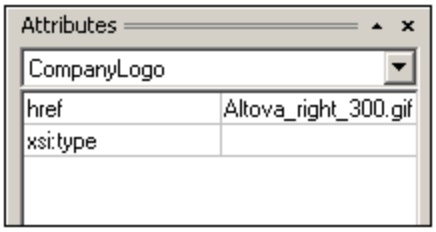
Valeurs d'attribut dans la fenêtre principale

Les valeurs d'attribut peuvent être saisies en tant que texte normal ou en tant que sélection par l'utilisateur qui sera mappée à une valeur XML. Elles seront saisies de la même manière que le contenu d'élément est saisi : voir Saisir des données dans le Authentic View. Dans ces cas, la distinction entre le contenu d'élément et la valeur d'attribut est réalisée par la StyleVision Power Stylesheet et les données sont gérées de manière appropriées.

Valeurs d'attribut dans l'assistant à la saisie Attributs

Si vous souhaitez saisir ou modifier une valeur d'attribut, vous pouvez aussi le faire dans l'Assistant à la saisir Attributs. Tout d'abord, le nœud d'attribut est sélectionné dans le Authentic View, ensuite la valeur de l'attribut est saisie ou éditée dans l'assistant à la saisie Attributs. Dans le document `NanonullOrg.xml`, l'emplacement du logo est stocké en tant que la valeur de l'attribut `href` de l'élément `CompanyLogo`. Pour changer le logo à utiliser :

1. Sélectionner l'élément `CompanyLogo` en cliquant sur une balise CompanyLogo. Les attributs de l'élément `CompanyLogo` sont affichés dans l'assistant à la saisie Attributs.
2. Dans l'assistant à la saisie Attributs, changer la valeur de l'attribut `href` de `nanonull.gif` à `Altova_right_300.gif` (une image dans le dossier `AuthenticExamples`).



Cela entraîne le remplacement du logo Nanonull par le logo Altova.

Note : les entités ne peuvent pas être saisies dans l'assistant à la saisie Attributs.

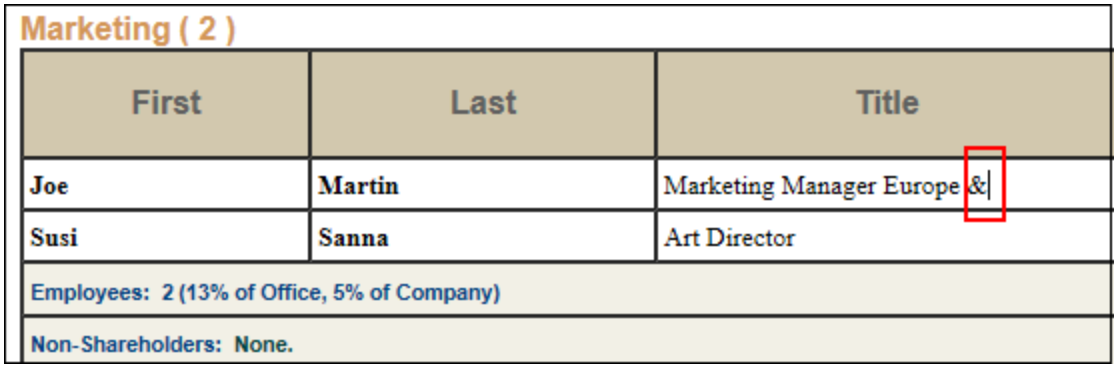
3.6 Ajouter des entités

Une entité contenue dans le Authentic View est généralement une donnée XML (mais pas nécessairement), comme un caractère unique ; une chaîne de texte ; et même un fragment de document XML. Une entité peut aussi être un fichier binaire, comme un fichier d'image. Toutes les entités disponibles pour un document particulier sont affichées dans l'assistant à la saisie Entités (*capture d'écran ci-dessous*). Pour insérer une entité, placer le curseur à l'endroit du document où vous souhaitez l'insérer puis double-cliquer sur l'entité dans l'assistant à la saisie Entités. Veuillez noter que vous ne pouvez pas saisir des entités dans l'assistant à la saisie Entités.

Le caractère esperluette (&) a une signification particulière dans XML (tout comme les apostrophes, les symboles inférieur à et supérieur à et les guillemets doubles). Pour insérer ces caractères, les entités sont utilisées de manière à ce qu'elles ne sont pas confondues avec des caractères réservés à XML. Ces caractères sont disponibles en tant qu'entités dans Authentic View.

Dans `NanonullOrg.xml`, changer le titre de Joe Martin (dans Marketing) en Marketing Manager Europe & Asia. Pour ce faire :

1. Placer le curseur à l'endroit où vous souhaitez insérer l'esperluette.
2. Double-cliquer sur l'entité listée en tant que "amp". Cela permet d'insérer une esperluette (*capture d'écran ci-dessous*).



Note : l'assistant à la saisie Entités n'est pas sensible au contexte. Toutes les entités disponibles sont affichées quel que soit l'endroit où le curseur est placé. Cela ne signifie pas qu'une entité peut être insérée à tout endroit du document. Si vous n'êtes pas sûr, valider le document après avoir inséré l'entité : **XML | Valider** (**F8**).

© 2016-2022 Altova GmbH

Définir vos propres entités

En tant qu'éditeur de document, vous pouvez définir vos propres entités de document. Pour ce faire, voir la description dans la section Définir des entités dans le Authentic View.

3.7 Imprimer le document

Une impression depuis le Authentic View d'un document XML préserve le formatage vu dans le Authentic View.

Pour imprimer `NanonullOrg.xml`, procéder comme suit :

1. Passer au mode Dissimuler balise si vous n'y êtes pas déjà. Vous devrez procéder à cette étape si vous ne voulez pas imprimer les balises.
2. Sélectionner **Fichier | Imprimer aperçu** pour voir un aperçu de toutes les pages. Ci-dessous, vous verrez une partie de l'impression d'une page d'aperçu, réduite de 50%. Veuillez noter que le formatage de la page est le même que celui dans Authentic View.

3. Pour imprimer le fichier, cliquer sur **Fichier | Imprimer**.

Veuillez noter que vous pouvez aussi imprimer une version du document qui affiche la balise. Pour ce faire, passer au Authentic View pour afficher les petites et les grandes balises, puis imprimer.

© 2016-2022 Altova GmbH

4 Interface Mode Authentic

Le Authentic View est activé en cliquant sur l'onglet Authentic du document actif. Si aucune SPS n'a été attribuée au document XML, vous serez invité à en attribuer un.

Cette section fournit :

- Un aperçu de l'interface
- Une description des icônes de la barre d'outils spécifiques à Authentic View
- Une description des modes de consultation disponibles dans la fenêtre principale du Authentic View
- Une description des assistants à la saisie et de la manière de les utiliser
- Une description des menus contextuels disponibles dans des points variés dans le Authentic View du document XML

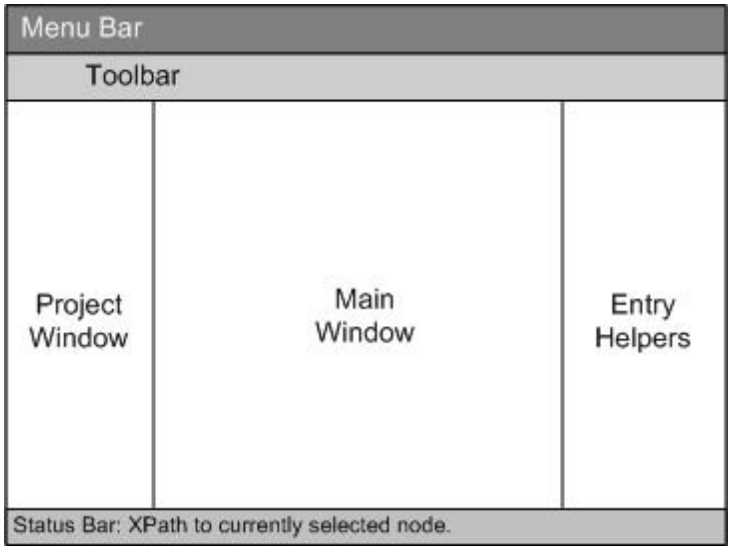
Sources supplémentaires des informations du Authentic View :

- Un tutoriel Authentic View qui vous montre comment utiliser l'interface Authentic View. Ce tutoriel est disponible dans la documentation des produits Altova XMLSpy et Altova Authentic Desktop (voir la section Tutoriel), et en ligne.
- Pour une description détaillée des commandes de menu du Authentic View, voir la section Référence de l'utilisateur de votre documentation de produit.

Site Web Altova : Édition du contenu XML, Création de contenu XML

4.1 Aperçu de la GUI

Le Authentic View présente en haut de l'écran une barre de menu et une barre d'outils, et trois volets qui couvrent le reste de l'interface : la fenêtre Projet, la fenêtre Principale et la fenêtre Assistants à la saisie. La disposition est représentée ci-dessous.



Barre Menu

Les menus disponibles dans la barre de menus sont décrits en détail dans la section Référence de l'utilisateur de votre documentation de produit.

Barre Outils

Les symboles et icônes affichés dans la barre outils sont décrits dans la section, icônes de la barre d'outils du Authentic View.

Fenêtre Projet

Vous pouvez grouper dans un seul projet des fichiers XML, XSL, schéma XML et Entité. Pour créer et modifier la liste des fichiers de projet, utiliser les commandes dans le menu **Projet** (décrit dans la section Référence de l'utilisateur de votre documentation de produit). La liste des fichiers de projet est affichée dans la fenêtre Projet. Un fichier dans la fenêtre Projet peut être accédé en le double-cliquant.

Fenêtre d'info

Cette fenêtre fournit des informations sur le nœud actuellement sélectionné dans le Mode Authentic.

Fenêtre Principale

Il s'agit de la fenêtre dans laquelle le document XML est affiché et édité. Il est décrit dans la section, fenêtre principale Authentic View.

© 2016-2022 Altova GmbH

Assistants à la saisie

Vous trouverez trois volets d'assistants à la saisie dans cette fenêtre : Éléments, Attributs et Entités. Les entrées apparaissant dans ces fenêtres sont sensibles au contexte (assistants à la saisie Éléments et Attributs), c.à.d. qu'elles dépendent de l'endroit où le curseur se trouve dans le document. Vous pouvez saisir un élément ou une entité dans le document en double-cliquant son assistant à la saisie. La valeur d'un attribut est saisie dans le champ de valeur de cet attribut dans assistant à la saisie Attributs. Voir la section Assistants à la saisie Authentic View pour plus de détails.

Barre de Statut

La barre de statut affiche le XPath vers le nœud actuellement sélectionné.

Menus contextuels

Il s'agit des menus qui apparaissent lorsque vous cliquez sur la touche de droite dans la fenêtre principale. Les commandes d'édition disponibles sont sensibles au contexte, c.à.d. qu'elles vous permettent de manipuler de la structure et du contenu afférent au nœud sélectionné. De telles manipulations comprennent l'insertion, ajout ou la suppression d'un nœud, l'ajout d'entités ou la découpe et le collage de contenu.

4.2 Icônes de la barre outils du Mode Authentic

Les icônes de la barre outils dans le Authentic View sont des raccourcis de commande. Certaines icônes utilisées dans les applications Windows ou les autres produits Altova vous seront déjà familières, d'autres seront nouvelles. Cette section décrit les icônes uniques au Authentic View. Dans la description ci-dessous, les icônes de la même famille sont regroupées.

Afficher/dissimuler les balises XML

Dans Authentic View, les balises pour tous, certains ou aucun des éléments ou attributs XML peuvent être affichés, soit avec leur nom (grandes balises) soit sans nom (petites balises). Les quatre icônes de balise apparaissent dans la barre outils et les commandes correspondantes sont disponibles dans le menu **Authentic**.

	Dissimuler les balises. Toutes les balises XML sont dissimulées sauf celles qui ont été réduites. Double-cliquer sur une balise réduite (qui est la manière habituelle de l'augmenter) dans le mode Dissimuler les balises pour afficher le contenu du nœud et dissimuler les balises.
	Afficher les petites balises. Les balises élément/attribut XML sont affichées sans leurs noms.
	Afficher les grandes balises. Les balises élément/attribut XML sont affichées avec leurs noms.
	Affichage des balises mixtes. Dans la StyleVision Power Stylesheet, chaque élément ou attribut XML peut être spécifié pour être affiché (soit en tant que balise grande ou petite), ou pas. Cela s'appelle un mode de balise mixte étant donné que certains éléments peuvent être spécifiés pour être affichés avec des balises et d'autres sans balises. Ainsi, dans le mode de balise mixte, l'utilisateur du Authentic View voit une balise personnalisée. Veuillez noter, néanmoins que cette personnalisation est créée par la personne qui a conçu la StyleVision Power Stylesheet. Elle ne peut pas être définie par l'utilisateur du Authentic View.

Éditer des structures de tables dynamiques

Les lignes dans une **table SPS dynamique** sont des répétitions d'une structure de données. Chaque ligne représente une occurrence d'un seul élément. Chaque ligne, a donc la même sous-structure XML que la suivante.

Les commandes d'édition des tables dynamiques manipulent les lignes d'une table dynamique SPS. Cela signifie que vous pouvez modifier le nombre et l'ordre des occurrences d'élément. Néanmoins, vous ne pouvez pas éditer les colonnes d'une table SPS dynamique puisque cela signifierait changer la sous-structure des occurrences de l'élément individuel.

Les icônes pour les commandes d'édition de la table dynamique apparaissent dans la barre d'outils, et sont aussi disponibles dans le menu **Authentic**.

© 2016-2022 Altova GmbH

	Apposer une ligne à la table
	Insérer une ligne dans la table
	Reproduire une ligne de table actuelle (des contenus de cellule sont dupliqués)
	Déplacer la ligne actuelle vers le haut d'une ligne
	Déplacer la ligne actuelle vers le bas d'une ligne
	Supprimer la ligne actuelle

Note : ces commande s'appliquent uniquement aux **tables SPS dynamiques**. Elles ne devraient pas être utilisées dans les tables SPS statiques. Les différents types de tables utilisées dans le Authentic View sont décrites dans la section Utiliser des tables dans le Mode Authentic de cette documentation.

Créer et éditer des tables XML

Si vous souhaitez présenter vos données sous la forme d'une table, vous pouvez insérer vos propres tables. Celles-ci sont insérées en tant que tables XML. Vous pouvez modifier la structure d'une table XML et formater la table. Les icônes pour créer et éditer les tables XML sont disponibles dans la barre outils, et sont affichées ci-dessous. Elles sont décrites dans la section icônes d'édition de tables XML.

Les commandes correspondant à ces icônes **ne sont pas disponibles en tant qu'items de menu**. Veuillez aussi noter que pour que vous puissiez utiliser des tables XML, cette fonction doit être activée et configurée convenablement dans la StyleVision Power Stylesheet. Une description détaillée des types de tables utilisés dans le Authentic View et de la manière dont créer et éditer les tables XML est indiquée dans la section Utiliser les tables dans le Mode Authentic.

Icônes de formatage de texte

Le texte dans le Authentic View est formaté en lui appliquant un élément ou un attribut XML qui a le formatage requis. Si ce type de formatage a été défini, le designer de la StyleVision Power Stylesheet peut fournir des icônes dans la barre outils du Authentic View pour appliquer le formatage. Pour appliquer un formatage de texte à l'aide d'une icône de formatage de texte, marquer le texte que vous souhaitez formater, et cliquer sur l'icône appropriée.

Icônes de navigation de ligne de BD

 Les icônes fléchées sont, de gauche à droite, Aller au premier enregistrement dans la BD ; Aller à l'enregistrement précédent ; Ouvrir le dialogue Aller à l'enregistrement n° ; Aller à l'enregistrement suivant et Aller au dernier enregistrement.

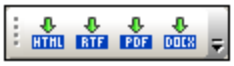
	Cette icône ouvre le dialogue Éditer la requête de base de données dans laquelle vous pouvez saisir une requête. Le Authentic View affiche les enregistrements requêtés.

Édition de la base de données XML

La commande **Sélectionner Nouvelle ligne avec les données XML pour l'édition** vous permet de sélectionner une nouvelle ligne depuis la table pertinente dans une BD XML, comme dans une IBM DB2. Cette ligne apparaît dans le Authentic View, peut y être éditée puis enregistrée dans la BD.

Touches de la barre d'outils Portable XML Form (PXF)

Les touches de barre d'outils PXF suivants sont disponibles dans le Authentic View de XMLSpy et Authentic Desktop:



Cliquer sur les touches individuelles génère une sortie HTML, RTF, PDF et/ou DocX.

Ces touches sont activées lorsqu'un fichier PXF est ouvert dans le Authentic View. Les touches individuelles sont activées si le fichier PXF a été configuré pour contenir la feuille de style XSLT pour ce format de sortie spécifique. Par exemple, si le fichier PXF était configuré pour contenu les feuilles de style XSLT pour HTML et RTF, alors seules les touches de barre d'outils pour les sorties HTML et RTF seront activées alors que celles pour les sorties PDF et DocX (Word 2007+) seront désactivées.

© 2016-2022 Altova GmbH

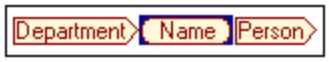
4.3 Fenêtre principale du Mode Authentic

Il y a quatre types de modes dans Authentic View: Grandes balises ; petites balises ; balises mixtes et dissimuler toutes les balises. Ces modes vous permettent de visualiser le document avec des niveaux variables d'information de marquage. Pour sauter entre les modes, utiliser les commandes dans le menu **Authentic** ou les icônes dans la barre d'outils (voir la section précédente, icônes de la barre d'outils du Mode Authentic).

Grandes balises

Cette option permet de montrer les balises de démarrage et de fin des éléments et des attributs avec les noms d'élément/attribut dans les balises :

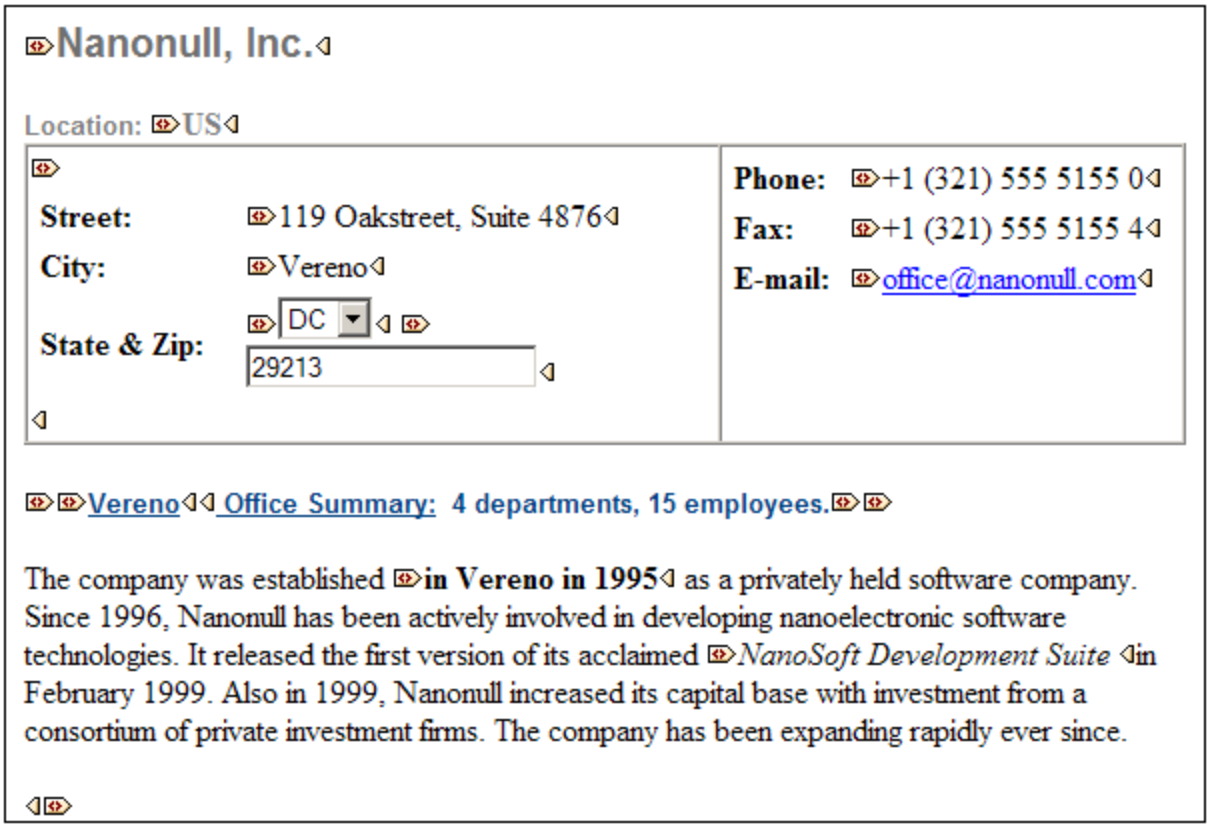
L'élément `Name` dans la figure ci-dessus est **agrandi**, donc la balise de démarrage et de fin sont affichés ainsi que le contenu de l'élément. Un élément/attribut peut être **contracté** en double-cliquant soit sur son balisage de démarrage soit sur son balisage de fin. Pour agrandir l'élément/attribut contracté, double-cliquer sur la balise contractée.



Dans les grandes balises, les attributs sont reconnus par le symbole égale à contenu dans les balises de démarrage et de fin de l'attribut :

Petites balises

Cette option permet de montrer les balises de démarrage et de fin des éléments et des attributs sans les noms :



Veuillez noter que les balises de démarrage contiennent un symbole alors que les balises de fin sont vides. De même, les balises d'élément ont un symbole <> alors que les onglets d'attributs contiennent un signe égal à comme leur symbole (*capture d'écran ci-dessous*).

Pour réduire ou agrandir un élément / attribut, double-cliquer sur la balise appropriée. L'exemple ci-dessous montre un élément réduit (marqué en bleu). Veuillez noter que la forme de la balise de l'élément réduit et celle de la balise de démarrage de l'élément agrandi à sa gauche.

Balisage mixte

Le balisage mixte constitue un balisage personnalisé. La personne ayant conçu la StyleVision Power Stylesheet peut spécifier l'affichage de grandes balises, de petites balises, ou d'aucune balise pour des éléments/attributs individuels dans le document. L'utilisateur du Authentic View peut consulter ce balisage personnalisé dans le mode de consultation Balisage mixte.

Dissimuler toutes les balises

Toutes les balises XML sont dissimulées. Puisque le formatage vu dans le Authentic View est le formatage du document imprimé, ce mode de consultation est un mode WYSIWYG du document.

© 2016-2022 Altova GmbH

Affichage du contenu

Dans le Authentic View, le contenu est affiché de deux manières :

- Texte simple. Vous saisissez le texte et celui-ci devient le contenu de l'élément ou la valeur de l'attribut.

- Appareils de saisie de données. L'affichage contient soit un champ de saisie (fenêtre de texte), un champ d'entrée multiligne, une liste de choix, une case à cocher ou un bouton radio. Dans le cas des champs de saisie et des champs d'entrée multiligne, le texte que vous saisissez dans le champ devient le contenu XML de l'élément ou de la valeur de l'attribut.

  Dans le cas des autres appareils de saisie de données, votre sélection produit une valeur de XML correspondante, qui est spécifiée dans la StyleVision Power Stylesheet. Ainsi, dans une liste de choix, une sélection de type, par exemple, "autorisé" (qui pourrait être disponible dans la liste déroulante de la liste de choix) pourrait mapper vers une valeur XML de "1", ou bien vers "autorisé", ou tout autre chose ; alors que "non autorisé" pourrait mapper vers "0", ou "non autorisé", ou tout autre chose.

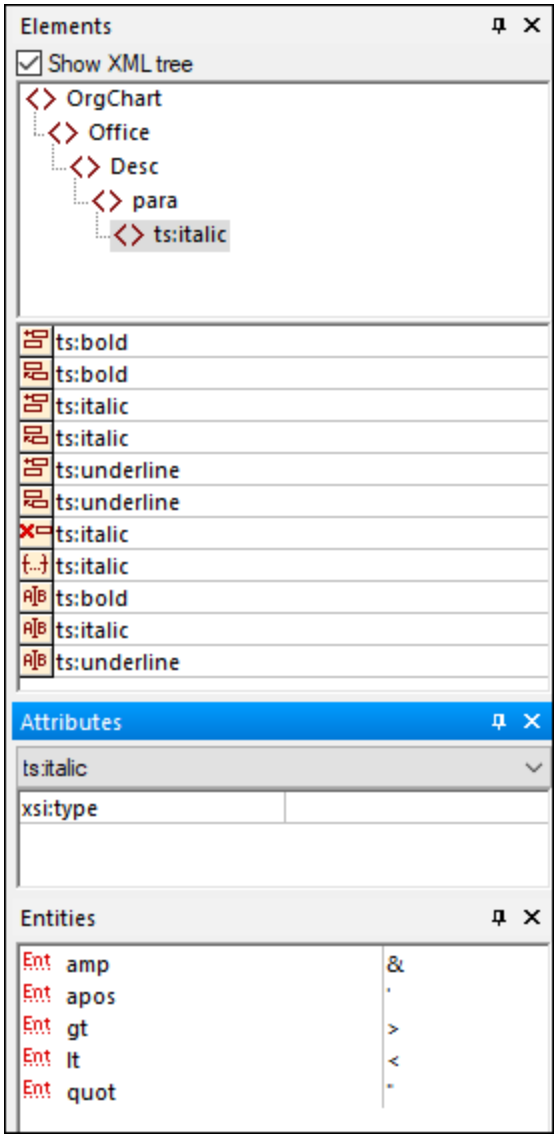
Nœuds optionnels

Lorsqu'un élément ou un attribut est **optionnel** (conformément au schéma référencé), une invite de type `add [element/attribute]` est affiché :

Cliquer sur l'invite permet d'ajouter l'élément et de placer le curseur pour l'entrée des données. S'il y a plusieurs nœuds optionnels, l'invite de type `add...` est affiché. Cliquer sur l'invite pour afficher un menu des nœuds optionnels.

4.4 Assistants à la saisie du Mode Authentic

Il y a trois assistants à la saisie dans le Authentic View : pour Éléments, Attributs et Entités. Ils sont affichés en tant que fenêtre le long du côté droit de l'interface du Authentic View (*voir capture d'écran ci-dessous*).



Les assistants à la saisie Éléments et Attributs sont sensibles au contexte, c. à.d. que ce qui apparaît dans les assistants à la saisie dépend de l'endroit où se trouve le curseur dans le document. Les entités affichées dans les assistants à la saisie Entités ne sont pas sensibles au contexte ; toutes les entités autorisées pour le document sont affichées où que soit le curseur.

Chacun des assistants à la saisie est décrit séparément ci-dessous.

Assistants à la saisie Éléments

L'assistant à la saisie Éléments consiste en deux parties :

© 2016-2022 Altova GmbH

- La partie supérieure, contenant une arborescence XML qui peut être activée et désactivée avec la case à cocher **Afficher arborescence XML**. L'arborescence XML montre les ancêtres jusqu'à l'élément de racine du document pour l'élément actuel. Lorsque vous cliquez sur un élément dans l'arborescence XML, les éléments correspondant à cet élément (tel que décrit dans l'item suivant dans cette liste) apparaissent dans la partie inférieure de l'assistant à la saisie Éléments.
- La partie inférieure, contenant une liste qui peut être insérée à l'intérieur, avant et après ; supprimée ; appliquée ou effacée de l'élément sélectionné ou la gamme de texte dans le Authentic View. Vous trouverez une description des choses à faire avec un élément recensé dans l'assistant à la saisie dans l'icône située à gauche du nom d'élément dans l'assistant à la saisie. Les icônes qui se produisent dans les assistants à la saisie Éléments sont recensées ci-dessous avec une description.

Pour utiliser un nœud depuis l'assistant à la saisie, cliquer sur cette icône.

Insérer après Élément

L'élément contenu dans l'assistant à la saisie est inséré après l'élément sélectionné. Veuillez noter qu'il est apposé au niveau hiérarchique correct. Par exemple, si votre curseur se trouve dans un élément `//sect1/para` et que vous apposez un élément `sect1`, le nouvel élément `sect1` sera apposé non en tant qu'un frère suivant `//sect1/para` mais en tant que le frère suivant de l'élément `sect1` qui est le parent de cet élément `para`.

Insérer avant Élément

L'élément contenu dans l'assistant à la saisie est inséré avant l'élément sélectionné. Veuillez noter que, tout comme la commande d'élément Insérer après, l'élément est inséré au niveau hiérarchique correct.

Supprimer Élément

Supprime l'élément et son contenu.

Insérer Élément

Un élément provenant de l'assistant à la saisie peut aussi être inséré dans un élément. Quand un curseur est placé dans un élément, les éléments enfants autorisés de cet élément peuvent être insérés. Veuillez noter que les éléments enfants autorisés peuvent faire partie d'un modèle de contenu éléments uniquement ainsi qu'un modèle de contenu mixte (texte plus éléments enfant).

Un élément enfant autorisé peut être inséré soit lorsqu'une plage de texte est sélectionnée soit lorsque le curseur est placé à un point d'intersection dans le texte.

- Lorsqu'une plage de texte est sélectionnée et qu'un élément a été inséré, la plage de texte devient le contenu de l'élément inséré.
- Lorsqu'un élément est inséré à un point d'insertion, l'élément est inséré à ce point.

Une fois qu'un élément a été inséré, il peut être effacé en cliquant sur une des deux icônes Effacer Élément qui apparaît (dans l'assistant à la saisie Éléments) pour ces éléments inline. Laquelle des deux icônes apparaît dépend du fait que vous sélectionnez une plage de texte ou que vous placiez le curseur dans le texte en tant qu'un point insertion (voir ci-dessous).

Appliquer Élément

Si vous sélectionnez un élément dans votre document (en cliquant sur sa balise de démarrage ou de fin dans le Mode Afficher grandes balises) et que cet élément peut être remplacé par un autre élément (par exemple dans un élément de contenu mixte comme `para`, un élément `italic` peut être remplacé par l'élément `bold`), cette

icône indique que l'élément contenu dans l'Assistant à la saisie peut être appliqué à l'élément sélectionné (original). La commande **Appliquer élément** peut aussi être appliquée à une plage de texte se trouvant dans le cadre d'un élément de contenu mixte ; la plage de texte sera créée en tant que le contenu de l'élément appliqué.

- Si l'élément appliqué possède un **élément enfant portant le même nom** qu'un enfant de l'élément original et qu'une instance de cet élément enfant existe dans l'élément original, alors l'élément enfant de l'original est retenu dans le contenu de l'élément nouveau.
- Si l'élément appliqué ne possède **pas d'élément enfant portant le même nom** qu'un enfant instancié de l'élément original, alors l'enfant instancié de l'élément original est apposé en tant que frère de tout élément enfant ou d'éléments que le nouvel élément pourrait avoir.
- Si l'élément appliqué possède un **élément enfant pour lequel aucun équivalent n'existe** dans le modèle de contenu de l'élément original, alors cet élément enfant n'est pas créé directement mais une option dans le Authentic View vous permet de l'insérer.

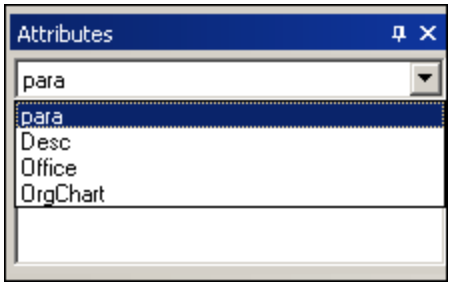
Si une plage de texte est sélectionnée au lieu d'un élément, l'application d'un élément à la sélection créera l'élément appliqué à cet emplacement avec la plage de texte sélectionnée en tant que son contenu. L'application d'un élément lorsque le curseur se trouve à un point d'insertion n'est pas autorisée.

 *Effacer Élément*
Cette icône apparaît lorsque le texte se trouvant dans un élément de contenu mixte est sélectionné. Cliquer sur l'icône pour effacer l'élément des alentours de la plage de texte sélectionnée.

 *Effacer Élément (lorsque le point d'insertion est sélectionné)*
Cette icône apparaît lorsque le curseur est placé dans un élément qui est un enfant d'un élément de contenu mixte. Cliquer sur l'icône pour effacer l'élément inline.

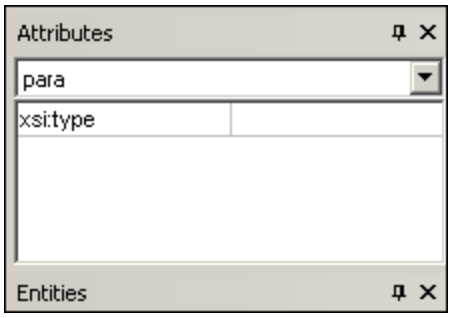
Assistants à la saisie Attributs

L'assistant à la saisie Attributs consiste en une liste de choix déroulante et une liste des attributs. L'élément que vous avez sélectionné (vous pouvez cliquer sur la balise de démarrage ou de fin, ou placer le curseur n'importe où dans le contenu de l'élément pour le sélectionner) apparaît dans la liste de choix. L'assistant à la saisie Attributs affiché dans les graphiques ci-dessous a un élément `para` dans la liste de choix. Cliquer sur la flèche dans la liste de choix pour faire dérouler une liste de tous les **ancêtres remontant jusqu'à l'élément de racine du document** de l'élément `para`, qui, dans ce cas est `OrgChart`.



En dessous de la liste de choix, vous verrez s'afficher une liste des attributs valides pour cet élément, dans ce cas, pour `para`. Si un attribut est obligatoire sur un élément donné, il apparaîtra en gras. (Dans l'exemple ci-dessous, il n'y a pas d'attributs obligatoires sauf l'attribut intégré `xsi:type`.)

© 2016-2022 Altova GmbH



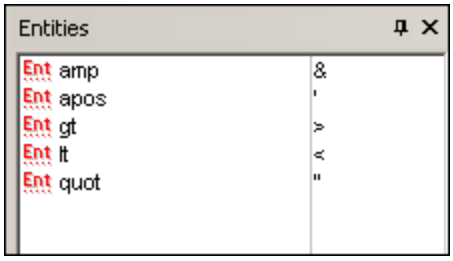
Afin de saisir une valeur pour un attribut, cliquer dans le champ de valeur de l'attribut et saisir la valeur. Cela permet de créer l'attribut et sa valeur dans le document XML.

Notez les points suivants :

- Dans le cas de l'attribut `xsi:nil`, qui apparaît dans l'assistant à la saisie Attributs lorsqu'un élément nullable a été sélectionné, la valeur de l'attribut `xsi:nil` peut uniquement être saisi en sélectionnant une des valeurs autorisées (`true` ou `false`) depuis la liste déroulante pour la valeur de l'attribut.
- L'attribut `xsi:type` peut être changé en cliquant dans le champ de valeur de l'attribut puis en sélectionnant, depuis la liste déroulante qui apparaît, une des valeurs recensées. Ces valeurs sont les types abstraits disponibles définis dans le Schéma XML sur lequel le document Mode Authentic se base.

Assistants à la saisie Entités

L'assistant à la saisie Entités vous permet d'insérer une entité dans votre document. Les entités peuvent être utilisées pour insérer des caractères spéciaux ou des fragments de texte qui se produisent souvent dans un document (comme le nom d'une entreprise). Pour insérer une entité, placer le curseur à l'endroit du texte où vous souhaitez insérer l'entité, puis double-cliquer l'entité dans l'assistant à la saisie Entités.



Note : Une entité interne est une entité dont la valeur est définie dans le DTD. Une entité externe est une entité dont la valeur est contenue dans une source externe, par ex. un autre fichier XML. Les entités internes et externes sont recensées dans l'assistant à la saisie Entités. Lorsque vous insérez une entité, qu'elle soit interne ou externe, l'entité, pas sa valeur, est insérée dans le texte XML. Si l'entité est une entité interne, le Authentic View affiche **la valeur de l'entité**. Si l'entité est une entité externe, le Authentic View affiche l'entité, et pas sa valeur. Cela signifie, par exemple, qu'un fichier XML qui est une entité externe sera affichée dans l'affichage du Authentic View en tant qu'une entité ; son contenu ne remplace pas l'entité dans l'affichage Authentic View.

Vous pouvez aussi **définir vos propres entités** dans le Authentic View et celles-ci seront aussi affichées dans l'assistant à la saisie : voir Définir des entités dans la section Éditer dans le Authentic View.

© 2016-2022 Altova GmbH

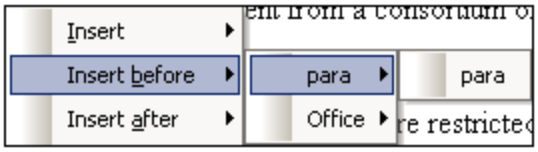
4.5 Menus contextuels Mode Authentic

Cliquer avec la touche de droite sur des contenus de document sélectionnés ou sur un nœud pour faire apparaître un menu contextuel avec des commandes pertinentes pour la sélection de l'emplacement du curseur.

Insérer les éléments

La figure ci-dessous montre le sous-menu **Insérer**, qui est une liste de tous les éléments qui peuvent être insérés à l'emplacement actuel du curseur. Le sous-menu **Insérer avant** recense une liste de tous les éléments qui peuvent être insérés avant l'élément actuel. Le sous-menu **Insérer après** recense une liste de tous les éléments qui peuvent être insérés après l'élément actuel. Dans la figure ci-dessous, l'élément actuel est l'élément `para`. Les éléments `bold` et `italic` peuvent être insérés dans l'élément `para` actuel.

Comme indiqué ci-dessous, les éléments `para` et `Office` peuvent être insérés avant l'élément `para` actuel.



Les commandes d'insertion de nœud, de remplacement (**Appliquer**), et la suppression de balise (**Effacer**) qui sont disponibles dans le menu contextuel sont aussi disponibles dans les Assistants à la saisie Mode Authentic et sont décrites en détail dans cette section.

Insérer une entité

Positionner le curseur sur la commande Insérer Entité pour faire apparaître un sous-menu contenant une liste de toutes les entités déclarées. Cliquer sur une entité pour l'insérer dans la sélection. Voir Définir des Entités pour une description de la définition des entités pour le document.

Insérer une section CDATA

Cette commande est activée lorsque le curseur est placé dans du texte. Cliquer dessus pour insérer une section CDATA au niveau d'insertion du curseur. La section CDATA est délimitée par des balises de démarrage et de fin ; pour voir ces balises, vous devez passer à un affichage des grandes ou des petites balises. Dans les sections CDATA, les balises XML et le parsage est ignoré. Les caractères de balisage XML (l'esperluette, l'apostrophe, le signe supérieur à et inférieur à et les guillemets) ne sont pas traités en tant que balises, mais en tant que littéraux. Ainsi, des sections CDATA sont utiles pour des textes comme des listes de code de programme, qui contiennent des caractères de balisage XML.

Supprimer le nœud

Positionner le curseur de la souris sur la commande **Supprimer** pour ouvrir une liste de menu consistant en un nœud sélectionné et tous ses ancêtres inamovibles (ceux qui n'invalideront pas le document) jusqu'à l'élément du document. Cliquer sur l'élément à supprimer. Il s'agit là d'une méthode rapide pour supprimer un élément ou

un ancêtre amovible. Veuillez noter qu'en cliquant sur un élément ancêtre, vous supprimerez tous ses descendants, y compris l'élément sélectionné.

Effacer

La commande **Effacer** efface la balise élément autour de la sélection. Si le nœud entier est sélectionné, la balise élément sera effacée pour le nœud entier. Si un segment de texte est sélectionné, la balise élément sera uniquement effacée autours de ce segment de texte.

Appliquer

La commande **Appliquer** applique un élément sélectionné à votre sélection dans la fenêtre principale. Pour plus de détails, voir Assistants à la saisie Mode Authentic.

Copier, Couper, Coller

Il s'agit des commandes Windows standard. Veuillez noter, néanmoins que la commande **Coller** colle du texte soit en tant que XML ou en tant que Texte, selon les spécifications du designer de la feuille de style pour la SPS complète. Pour plus d'informations concernant le fonctionnement des commandes **Copier en tant que XML** et **Copier en tant que Texte**, voir la description de la commande **Coller en tant que** juste en-dessous.

Coller comme

La commande **Coller en tant que** permet de coller en tant que XML ou en tant que texte un fragment XML de Authentic View (qui a été copié dans le presse-papiers). Si le fragment copié est collé en tant que XML, il est collé avec sa balise XML. S'il est collé en tant que texte, seul le contenu de texte du fragment copié est collé (pas la balise XML, si existante). Les situations suivantes sont possibles :

- Un ***nœud entier et ses balises de marquage*** sont marqués dans Authentic View et copiés dans le presse-papiers. (i) Le nœud peut être collé en tant que XML dans n'importe quel endroit où ce nœud peut être placé validement. Il ne sera pas collé dans un emplacement invalide. (ii) Si le nœud est collé en tant que texte, seul le *contenu du texte* de ce nœud sera collé (pas la balise) ; le contenu suivant peut être collé dans n'importe quel endroit du document XML où il est possible de coller du texte.
- Un ***fragment de texte*** est marqué dans le Authentic View et copié dans le presse-papiers. (i) Si ce fragment est collé en tant que XML, alors les balises de marquage XML du texte, bien que celles-ci n'ont pas été copiées explicitement avec le fragment de texte, seront collées avec le texte, mais uniquement si le nœud XML est valide à l'endroit où le fragment est collé. (ii) Si le fragment est collé en tant que texte, il peut être collé n'importe où dans le document XML où il est possible de coller du texte.

Note : Ie texte sera copié dans les nœuds dans les endroits où le texte est autorisé, vous devrez donc veiller à ce que le texte copié n'invalide pas le document. Le texte copié devra donc être : (i) lexicalement valide dans le nouvel emplacement (par exemple des caractères non-numériques dans un nœud numérique serait un exemple invalide), et (ii) ne pas invalider le nœud d'une autre manière (par exemple, quatre chiffres dans un nœud qui n'accepte que des nombres à trois chiffres serait un exemple d'invalidation du nœud).

Note : si le texte collé invalide le document de quelque manière, le texte sera affiché en rouge.

Supprimer

La commande **Supprimer** supprime le nœud sélectionné et son contenu. Un nœud est considéré être sélectionné pour cet objectif en plaçant le curseur dans le nœud ou en cliquant soit sur la balise de démarrage ou de fin du nœud.

© 2016-2022 Altova GmbH

5 Éditer dans le Mode Authentic

Cette section décrit les fonctions importantes du Authentic View en détail. Les fonctions décrites dans cette section y ont été incluses soit parce qu'elles sont fréquemment utilisées soit parce que les mécanismes ou les concepts impliqués nécessitent des explications.

La section explique les fonctions suivantes :

- Il existe trois types distincts de tables utilisées dans le Authentic View. La section Utiliser des tables dans le Authentic View explique les trois types de tables (SPS statique, SPS dynamique et XML), et la manière et le moment de les utiliser. Elle commence avec l'aspect général et conceptuel de la fonction et continue ensuite vers les détails de son utilisation.
- Le sélecteur de date est un calendrier graphique qui saisit les dates dans le format XML correct lorsque vous cliquez sur une date. Voir le Sélecteur de date.
- Une entité est une abréviation pour un caractère spécial ou une chaîne de texte. Vous pouvez définir vos propres entités, qui vous permettent d'insérer ces caractères spéciaux ou chaînes de texte en insérant les entités correspondantes. Voir Définir des entités pour plus de détails.
- Dans les éditions Enterprise et Professional des produits Altova, les utilisateurs du Authentic View peuvent signer des documents XML avec des signatures XML numériques et vérifier ces signatures.
- Quels types de formats d'image peut être affiché dans le Authentic View.

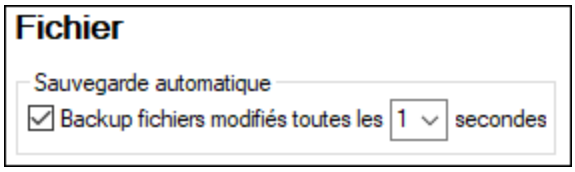
Site Web Altova : Édition du contenu XML, Création de contenu XML

© 2016-2022 Altova GmbH

5.1 Sauvegarde automatique des fichiers

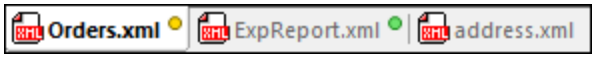
Les fichiers qui sont modifiés dans Authentic Desktop sont automatiquement sauvegardés à des intervalles réguliers. Dans l'onglet *Fichier* du dialogue Options (**Outils | Options | Fichier**) affiché dans la capture d'écran, vous pouvez :

- Allumer/éteindre des sauvegardes automatiques
- Spécifier la fréquence des sauvegardes (5 secondes à 300 secondes)



Indicateurs

Les onglets de fichier en bas de la fenêtre principale contiennent des symboles à droite du nom de fichier qui indiquent l'état enregistré/non-enregistré et l'état de sauvegarde du fichier *(capture d'écran ci-dessous)*.



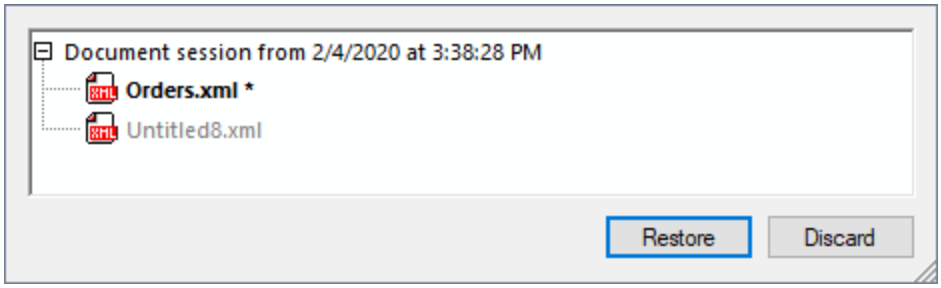
Enregistré / Non-enregistré
Un cercle coloré apparaît si un fichier a été modifié. Si vous ne voyez pas apparaître de symbole, cela signifie que le fichier n'a pas été modifié depuis la dernière ouverture ou depuis le dernier enregistrement. Dans la capture d'écran ci-dessus, par exemple, `address.xsd` .

État de sauvegarde
Les couleurs des cercles indiquent l'état de sauvegarde du fichier.

- *Jaune :* Le fichier a été modifié mais la dernière modification n'a pas été sauvegardée (ou enregistrée).
- *Vert :* Le fichier a été sauvegardé et il n'a pas été modifié depuis la dernière sauvegarde. Néanmoins, le fichier n'a pas été enregistré. (S'il avait été enregistré, il n'y aurait pas de cercle.)
- *Rouge :* La sauvegarde n'est pas prise en charge pour ce fichier ou la sauvegarde a échoué.
- *Gris :* La fonction de sauvegarde automatique a été désactivée (par le biais du dialogue Options ; *voir ci-dessus*). La présence du symbole, néanmoins, indique que le fichier n'a pas été enregistré depuis la dernière modification. (S'il avait été enregistré, il n'y aurait pas de cercle.)

Restaurer depuis les sauvegardes

Si Authentic Desktop cesse de manière imprévue, alors, au prochain démarrage de l'application, un dialogue Restaurer Document est affiché qui contient une liste de tous les documents qui ont été ouverts au moment de l'interruption de l'application *(capture d'écran ci-dessous)*. Vous pouvez planer sur chaque fichier pour voir son chemin. Dans le cas des fichiers temporaires qui n'ont pas encore été enregistrés, le chemin de fichier sera le chemin par défaut actuel où un dialogue Enregistrer sous s'est ouvert pour ce fichier.



Pour chaque fichier dans la liste, le style de police et la présence/absence d'astérisques propose l'information suivante :

- Un style gras et une astérisque indiquent que le fichier contient des modifications non enregistrées. Ce type de fichier sera restauré à son dernier état de sauvegarde.
- Un style normal indique que le fichier a été enregistré et qu'il n'y a pas de modifications non enregistrées. Les fichiers seront restaurés à leur état enregistré.
- Un style grisé indique que le fichier n'a pas été enregistré ni sauvegardé (par exemple, parce qu'il s'agit d'un nouveau fichier qui n'a pas été édité). Ces fichiers ne seront pas restaurés.

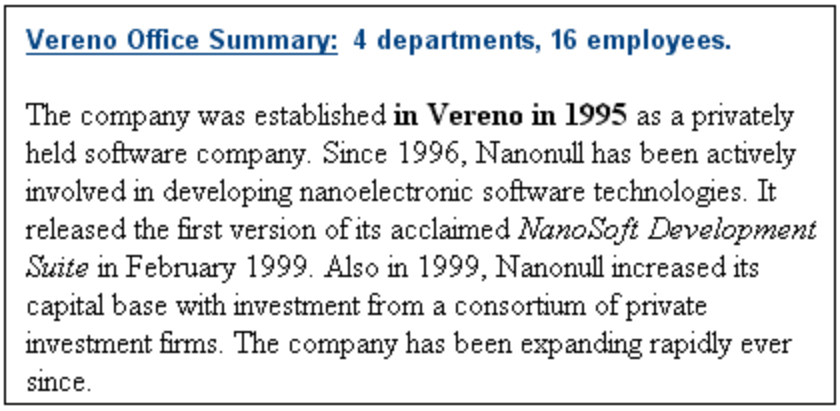
Vous pouvez à présent suivre une des étapes suivantes :

- Cliquer sur **Restaurer** pour restaurer les fichiers dans la GUI depuis leur dernier état sauvegardé.
- Cliquer sur **Éliminer** pour ne pas ouvrir un des fichiers listé et pour éliminer toute sauvegarde disponible.

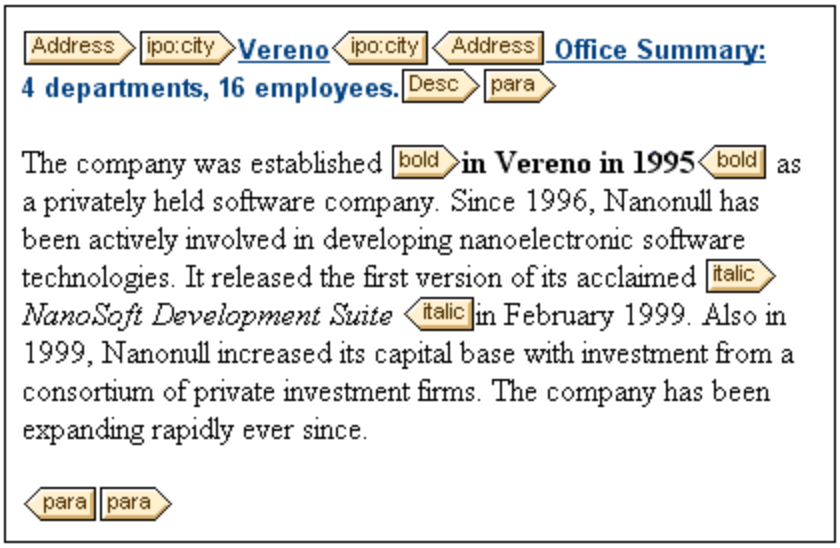
© 2016-2022 Altova GmbH

5.2 Édition de base

Lorsque vous éditez dans le Authentic View, vous éditez un document XML. Néanmoins, le Authentic View peut dissimuler le balisage structurel du XML du document, affichant ainsi uniquement le contenu du document (*première capture d'écran ci-dessous*). Vous n'aurez donc pas à traiter avec l'aspect technique de XML, et vous pourrez éditer le document comme si vous éditeriez un document texte normal. Si vous le souhaitez, vous pouvez passer au balisage à tout moment pendant l'édition (*seconde capture d'écran ci-dessous*).



Un document éditable Authentic View sans balises XML.



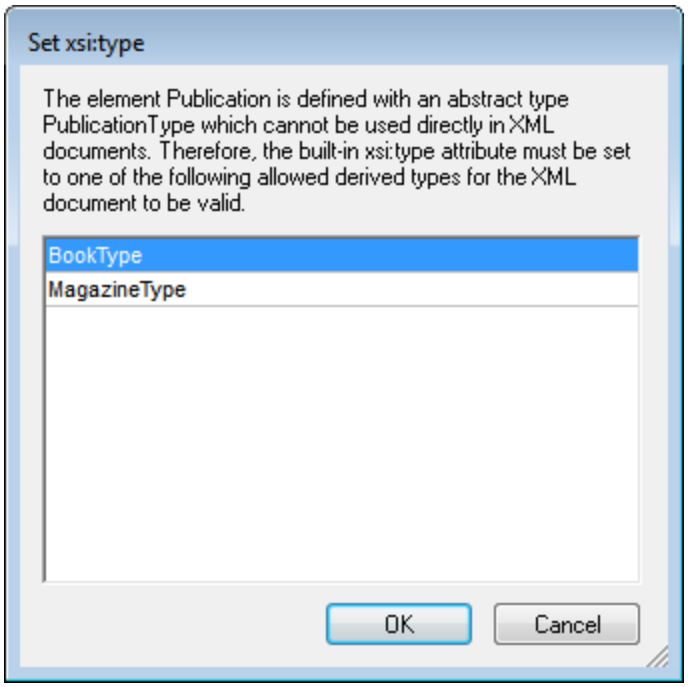
Un document éditable Authentic View avec des balises de marquage XML.

Insérer les nœuds

Très souvent, vous devrez ajouter un nouveau nœud au document XML Authentic. Par exemple, un nouvel élément `Person` peut devoir être ajouté à un document de type carnet d'adresses. Dans ces cas, le Schéma XML permettrait l'ajout du nouvel élément. Il vous suffit de cliquer avec la touche de droite sur le nœud dans le document Mode Authentic avant ou après celui auquel vous souhaitez ajouter le nouveau nœud. Dans le menu contextuel qui apparaît, sélectionner **Insérer avant** ou **Insérer après**, vos besoins. Les nœuds disponibles

pour l'insertion à cet endroit dans le document sont recensés dans un sous-menu. Cliquer sur le nœud requis pour l'insérer. Le nœud sera inséré. Tous les nœuds descendants obligatoires sont aussi insérés. Si un nœud descendant est optionnel, un lien cliquable, `Add NodeName`, apparaît afin de vous permettre d'ajouter le nœud optionnel si vous le souhaitez.

Si le nœud ajouté est un élément avec un type abstrait, alors un dialogue (*par exemple comme celui ci-dessous*) apparaîtra contenant une liste des types dérivés disponibles dans le Schéma XML.



La fenêtre dans la capture d'écran ci-dessus s'ouvre lorsqu'un élément `Publication` est ajouté. L'élément `Publication` est de type `PublicationType`, qui est un type complexe abstrait. Les deux types complexes `BookType` et `MagazineType` sont dérivés depuis le `PublicationType` abstrait. C'est pourquoi, lorsqu'un élément `Publication` est ajouté dans le document XML, un de ces deux types concrets dérivés du type abstrait de `Publication` doit être spécifié. Le nouvel élément `Publication` sera ajouté avec un attribut `xsi:type` :

```
<Publication xsi:type="BookType"> ... </Publication>
<Publication xsi:type="MagazineType"> ... </Publication>
...
<Publication xsi:type="MagazineType"> ... </Publication>
```

Sélectionner un des types dérivés disponibles et cliquer sur **OK** permet de :

- Définir le type dérivé sélectionné en tant que la valeur de l'attribut `xsi:type` de l'élément
- Insérer l'élément avec les nœuds descendants définis dans le modèle de contenu du type dérivé sélectionné.

Le type dérivé sélectionné peut être changé ultérieurement en changeant la valeur de l'attribut `xsi:type` de l'élément dans l'Assistant à la saisie Attributs. Lorsque le type de l'élément est changé de cette manière, tous les nœuds du modèle de contenu du type précédent sont supprimés et les nœuds du modèle de contenu du nouveau type sont insérés.

© 2016-2022 Altova GmbH

Éditer du texte

Un document Authentic View consiste essentiellement en du texte et des images. Pour éditer le texte dans le document, placer le curseur à l'emplacement où vous souhaitez insérer le texte et effectuer votre saisie. Vous pouvez copier, déplacer et supprimer du texte à l'aide des touches familières (comme la touche **Supprimer**) et les mécanismes glisser et déposer. Une exception est la touche **Entrée**. Puisque le document du Authentic View est préformaté, vous ne devez pas (et ne pourrez pas) ajouter des lignes supplémentaires ou des espaces entre les items. La touche **Entrée** dans le Authentic View sert donc à apposer une autre instance de l'élément actuellement en cours d'édition et devrait être utilisé exclusivement à cette fin.

Copier en tant que XML ou en tant que texte

Le texte peut être copié et collé en tant que XML ou en tant que texte.

- Si du texte est collé en tant que XML, le balisage XML est collé avec le contenu de texte des nœuds. Le balisage XML est collé même si uniquement une partie des contenus d'un nœud a été copié. La balise peut uniquement être collée aux endroits autorisés par le schéma.
- Si le texte est collé en tant que texte, la balise XML n'est pas collée.

Afin de coller du texte en tant que XML ou en tant que texte, copier tout d'abord le texte (**Ctrl+C**), cliquer avec la touche de droite à l'emplacement où vous souhaitez coller le texte puis sélectionner la commande de menu contextuel **Coller en tant que | XML** ou **Coller en tant que | Texte**. Si le raccourcis **Ctrl+V** est utilisé, le texte sera collé dans le Mode Coller par défaut de la SPS. Le Mode Coller par défaut aura été spécifié par le concepteur de la SPS. Pour plus de détails, voir la section Menus contextuels.

En alternative, il est possible de déplacer un texte marqué à l'emplacement où il doit être collé. Lorsque le texte est déposé, une fenêtre popup apparaît vous demandant si le texte doit être collé en tant que texte ou en tant que XML. Sélectionner l'option désirée.

Formater du texte

Un des principes de base des systèmes de document XML est que le contenu est gardé séparément de la présentation. Le document XML contient le contenu, alors que la feuille de style contient la présentation (formatage). Dans le Authentic View, le document XML est présenté par le biais de la feuille de style. Cela signifie que tout le formatage que vous verrez dans le Authentic View est produit par la feuille de style. Si vous voyez du texte gras, ce formatage gras aura été défini dans la feuille de style. Si vous voyez une liste ou une table, ce format de liste ou de table aura été défini dans la feuille de style. Le document XML dont vous vous servez pour l'édition dans le Authentic View ne recèle que du contenu ; il ne contient aucune information concernant le formatage. Le formatage est contenu dans la feuille de style. Cela signifie pour vous, l'utilisateur du Authentic View, que vous n'avez pas à (et ne pourrez pas) formater le texte que vous éditez. Vous éditez le contenu. Le formatage qui s'applique automatiquement au contenu que vous éditez est lié à la valeur sémantique et/ou structurelle des données que vous éditez. Par exemple, une adresse e-mail (qui peut être considérée comme une unité sémantique) sera formatée automatiquement d'une certaine manière parce qu'il s'agit d'un e-mail. De la même manière, un titre doit se produire à un emplacement particulier du document (unité aussi bien structurelle que sémantique) et sera formaté automatiquement selon la manière spécifiée par le concepteur dans la feuille de style pour les titres. Vous ne pouvez pas modifier le formatage de l'adresse e-mail ou du titre. Tout ce que vous pouvez faire est d'éditer le contenu de l'adresse e-mail ou du titre.

Dans certains cas, le contenu peut nécessiter une présentation particulière ; par exemple, une chaîne de texte doit être présentée en gras. Dans ces cas, la présentation doit être liée avec un élément structurel du document. Par exemple, une chaîne de texte qui doit être présentée en gras, sera séparée structurellement du contenu environnant par des balises que le concepteur de la feuille de style formatera en gras. Si vous, en tant

que l'utilisateur du Authentic View souhaitez utiliser une telle chaîne de texte, vous devrez contenir la chaîne de texte dans l'élément de balise approprié. Pour plus d'information concernant la procédure à suivre, voir la commande Insérer l'élément dans la section Assistant à la saisie Éléments de la documentation.

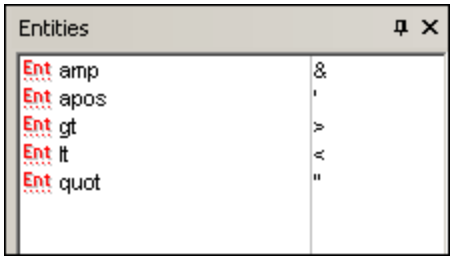
Utiliser RichEdit dans Authentic View

Dans le Authentic View, lorsque le curseur est placé dans un élément qui a été créé en tant qu'un composant RichEdit, les touches et les commandes dans la barre d'outils RichEdit (*capture d'écran ci-dessous*) sont activées. Dans le cas contraire, elles sont grisées.

Sélectionner le texte auquel vous souhaitez appliquer le style et spécifier le style avec les touches et les commandes de la barre d'outils RichEdit. RichEdit permet aux utilisateurs du mode Authentic View de spécifier la police, le poids, la décoration, la taille de la police, la couleur, la couleur de l'arrière-plan et l'alignement du texte. Le texte qui aura été stylisé sera contenu dans les balises de l'élément de style.

Insérer des entités

Dans les documents XML, certains caractères sont réservés pour le balisage et ne peuvent pas être utilisés dans du texte normal. Il s'agit de l'esperluette (`&`), de l'apostrophe (`'`), du caractère inférieur à (`<`), supérieur à (`>`) et des guillemets (`"`). Si vous souhaitez utiliser ces caractères dans vos données, vous devrez les insérer en tant que références d'entité, par le biais de l'Assistant à la saisie Éléments (*capture d'écran ci-dessous*).



XML offre aussi la possibilité de créer des entités personnalisées. Celles-ci peuvent prendre la forme de : (i) caractères spéciaux qui ne sont pas disponibles sur votre clavier, (ii) des chaînes de texte que vous souhaitez réutiliser dans le contenu de votre document, (iii) des fragments de données XML, ou (iv) d'autres ressources comme des images. Vous pouvez définir vos propres entités dans le cadre de l'application Authentic View. Une fois définies, ces entités apparaissent dans l'Assistant à la saisie Éléments et peuvent être insérées comme dans le document.

Insérer des sections CDATA

Les sections CDATA sont des sections de texte dans un document XML que le parseur XML ne traite pas en tant que données XML. Elles peuvent être utilisées pour échapper de grandes sections de texte si vous ne souhaitez pas remplacer des caractères spéciaux par des références d'entités ; cela peut être le cas, par exemple, avec un code de programme ou un fragment XML qui doit être reproduit avec ses balises de marquage. Les sections CDATA peuvent se produire dans un contenu d'élément et sont délimitées par `<![CDATA[` et `]]>` au début et à la fin, respectivement. Par conséquence, la chaîne de texte `]]>` ne devrait pas se produire dans une section CDATA car elle marquerait la fin prématurée de la section. Dans ce cas, le caractère "supérieur à" devrait être échappé par sa référence d'entité (`>`). Pour insérer une section CDATA dans le

© 2016-2022 Altova GmbH

cadre d'un élément, placer le curseur à l'emplacement désiré, cliquer avec la touche de droite et sélectionner **Insérer Section CDATA** depuis le menu contextuel. Pour voir la balise de section CDATA dans le Authentic View, passer à l'affichage des balises. En alternative, vous pouvez marquer le texte devant être inséré dans une section CDATA, puis sélectionner la commande **Insérer la section CDATA**.

Note : Les sections CDATA ne peuvent pas être insérées dans des champs d'entrée (c'est à dire dans des champs de texte et des champs à texte multiligne). Les sections CDATA peuvent uniquement être saisies dans des éléments qui sont affichés dans le Authentic View en tant que composants de contenu de texte.

Éditer et suivre les liens

Un hyperlien consiste en deux parties : le texte de lien et la cible du lien. Vous pouvez éditer le texte de lien en cliquant dans le texte et en l'éditant. Mais vous ne pouvez pas éditer la cible du lien. (La cible du lien est définie par le concepteur de la feuille de style (soit en saisissant une adresse cible statique soit en dérivant l'adresse cible depuis les données contenues dans le document XML)). Vous pouvez vous rendre à la cible du lien depuis le Authentic View en appuyant sur **Ctrl** et en cliquant sur le texte du lien. (Souvenez-vous : si vous cliquez simplement sur le lien, vous aboutirez à l'édition du texte de lien.)

5.3 Tables dans le Mode Authentic

Les trois types de table se répartissent dans deux catégories : les tables SPS (statiques et dynamiques) et les tables CALS/HTML.

Les tables SPS sont de deux types : statiques et dynamiques. Les tables SPS sont conçues par le designer de la StyleVision Power Stylesheet à laquelle votre document XML est lié. Vous ne pouvez pas insérer de table SPS dans le document XML, mais vous pouvez saisir des données dans des champs de table SPS et ajouter et supprimer les lignes des tables SPS dynamiques. La section ci-dessous concernant les tables SPS explique les fonctions de ces tables.

Les tables CALS/HTML sont insérées par vous, l'utilisateur du Authentic View. Leur objectif est de vous permettre d'insérer des tables à tout emplacement autorisé dans la hiérarchie du document, le cas échéant. Les fonctions d'édition des Tables CALS/HTML et des icônes d'édition de la table CALS/HTML sont décrites ci-dessous.

5.3.1 Tables SPS

Deux types de tables SPS sont utilisées dans Authentic View : les tables statiques et les tables dynamiques.

Tables statiques

Les tables statiques ont une structure et un type de contenu des cellules fixes. Vous, en tant que l'utilisateur de Authentic View, pouvez saisir des données dans les cellules de table, mais vous ne pouvez pas changer la structure de ces tables (donc ajouter des lignes ou des colonnes, etc.) ou modifier le type de contenu d'une cellule. Vous saisissez des données soit en entrant du texte soit en sélectionnant à partir d'options présentées sous la forme de cases à cocher ou de boutons radio ou en tant que liste dans une liste de choix. Une fois avoir saisi les données, vous pouvez les éditer.

Nanonull, Inc.

Street:	119 Oakstreet, Suite 4876	**Phone:**	+1 (321) 555 5155
City:	Vereno	**Fax:**	+1 (321) 555 5155 - 9
State & Zip:	DC 29213	**E-mail:**	office@nanonull.com

Note : Les icônes ou les commandes d'édition des tables dynamiques **ne doivent pas** être utilisées pour éditer des tables statiques.

Tables dynamiques

Les tables dynamiques ont des lignes qui représentent une structure de données répétitives, donc chaque ligne a une structure de données identique (ce qui n'est pas le cas avec des tables statiques). C'est pourquoi vous pouvez effectuer des opérations de ligne : apposer une ligne, insérer une ligne, déplacer une ligne vers le haut, déplacer une ligne vers le bas, et supprimer des lignes. Ces commandes sont disponibles sous le menu **Authentic** et en tant qu'icônes dans la barre outils (affiché ci-dessous).

© 2016-2022 Altova GmbH

Pour utiliser ces commandes, placer le curseur n'importe où dans la ligne appropriée et sélectionner la commande nécessaire.

Administration

First	Last	Title	Ext	EMail	Shares	Leave		
						Total	Used	Left
Vernon	**Callaby**	Office Manager	581	v.callaby@nanonull.com	1500	25	4	21
Frank	Further	Accounts Receivable	471	f.further@nanonull.com	0	22	2	20
Loby	Matise	Accounting Manager	963	l.matise@nanonull.com	add Shares	25	7	18
Employees: 3 (20% of Office, 9% of Company)					**Shares: 1500 (13% of Office, 6% of Company)**			
Non-Shareholders: Frank Further, Loby Matise.								

Pour vous déplacer parmi les cellules de la table, utiliser les touches fléchées Haut, Bas, Gauche, Droite. Pour vous déplacer d'une cellule à une autre, utiliser la touche **Tab**. Appuyer sur la touche **Tab** dans la dernière cellule de la dernière ligne crée une nouvelle ligne.

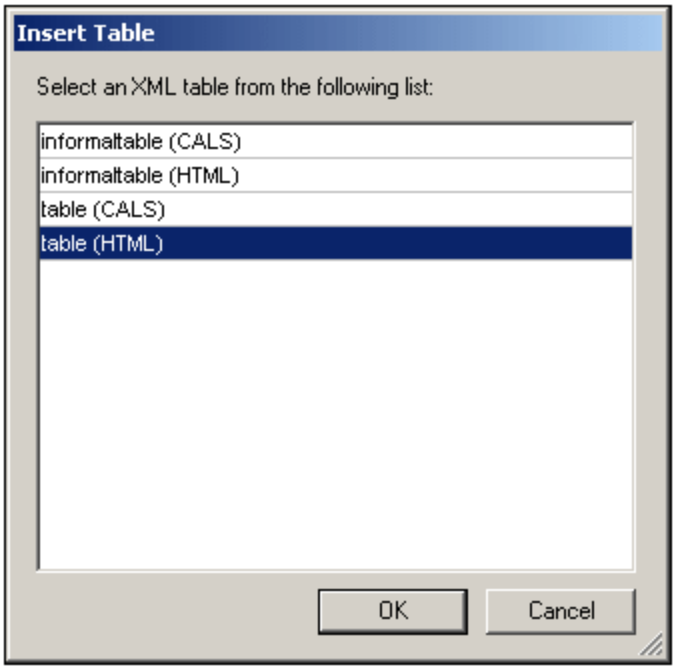
5.3.2 Tables CALS/HTML

Les tables CALS/HTML peuvent être insérées par vous, l'utilisateur du Authentic View pour certaines structures de données XML qui ont été spécifiées pour afficher un format de table. Trois étapes doivent être effectuées pour travailler avec des tables CALS/HTML : insérer la table ; la formater ; et saisir les données. Les commandes pour travailler avec les tables CALS/HTML sont disponibles en tant qu'icônes dans la barre d'outils (voir icônes d'édition de table CALS/HTML).

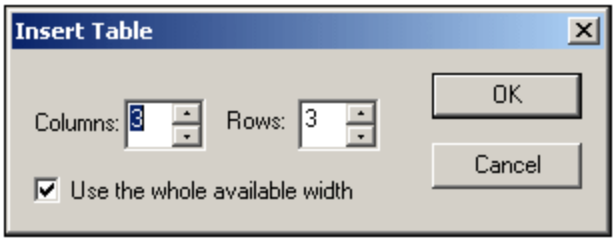
Insérer les tables

Pour insérer une table CALS/HTML, procéder comme suit :

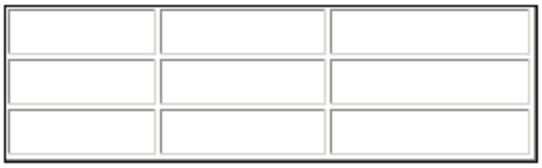
1. Placer votre curseur à l'endroit où vous souhaitez insérer la table, et cliquer sur l'icône  . (Veuillez noter que l'endroit où vous pouvez insérer des tables est déterminé par le schéma.) Le dialogue Insérer la table (*capture d'écran ci-dessous*) apparaît. Ce dialogue recense toutes les structures de données d'éléments XML pour lesquelles une structure de table a été définie. Par exemple, dans la capture d'écran ci-dessous, l'élément `informaltable` et l'élément `table` ont tous les deux été définis en tant que table CALS et en tant que table HTML.



2. Sélectionner l'entrée contenant l'élément et le modèle de table que vous souhaitez insérer et cliquer sur **OK**.
3. Dans le dialogue suivant (*capture d'écran ci-dessous*), sélectionner le numéro des colonnes et des lignes et spécifier si un en-tête et/ou un pied de page doit être ajouté à la table et si toute la table doit être étendue sur la largeur totale disponible. Cliquer **OK** une fois terminé.



La table suivante a été créée pour afficher les spécifications données dans le dialogue ci-dessus.



En utilisant les commandes de menu **Table**, vous pouvez ajouter et supprimer des colonnes et créer des joints et des divisions de colonnes et de lignes. Mais pour commencer, vous devez créer une structure large.

© 2016-2022 Altova GmbH

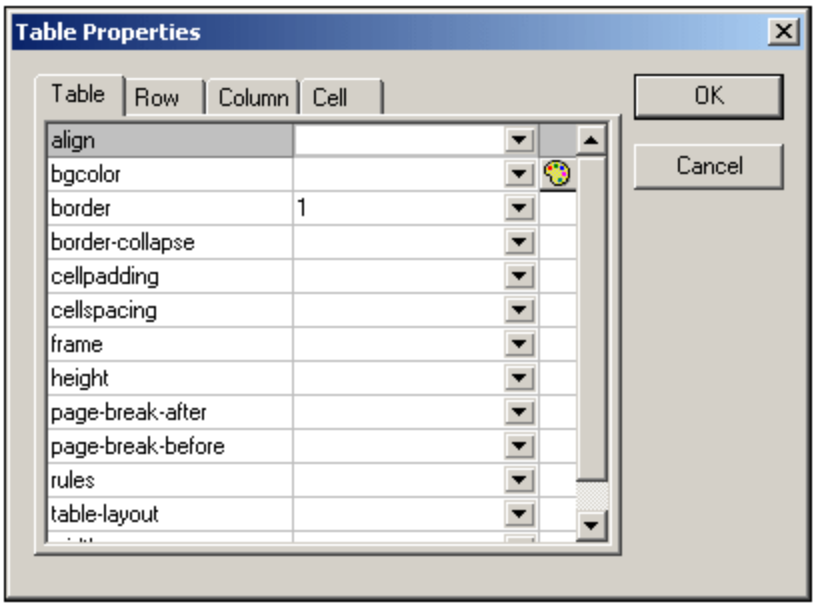
Formater les tables et la saisie les données

Le formatage de table aura déjà été attribué au design du document. Néanmoins, vous pourrez, dans certaines circonstances, modifier le formatage de table. Ces circonstances sont les suivantes :

- Les éléments correspondants aux éléments de structure de table divers doivent avoir les propriétés de table CALS ou HTML pertinentes définies en tant qu'attributs (dans le schéma XML sous-jacent). Seuls les attributs qui sont définis seront disponibles pour le formatage. Si, dans le design, les valeurs ont été définies pour ces attributs, vous pourrez substituer ces valeurs dans le Authentic View.
- Dans le design aucun attribut `style` contenant les styles CSS doivent avoir été définis. Si un attribut de style contenant des styles CSS a été spécifié pour un élément, l'attribut `style` a une précédence au-dessus de tout autre attribut de formatage défini sur cet élément. En résultat, tout formatage spécifié dans Authentic View sera substitué.

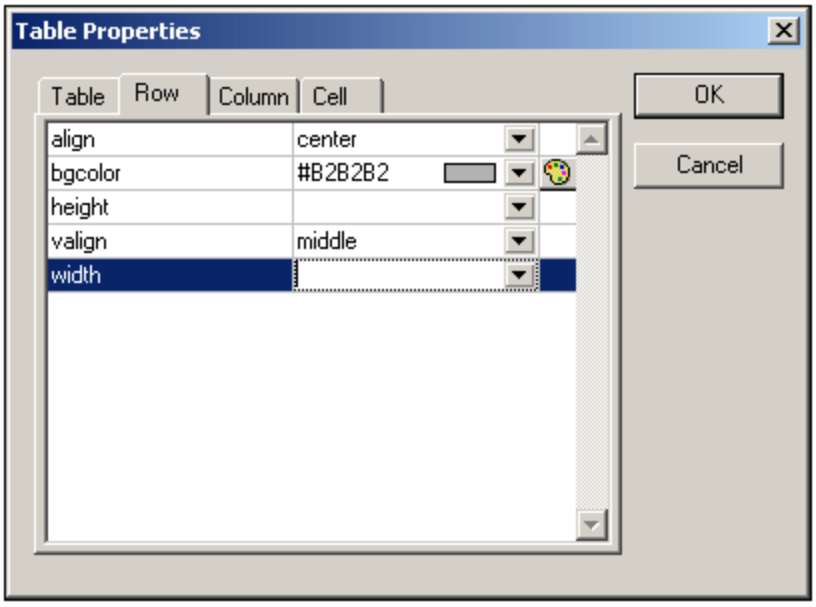
Pour formater une table, une ligne, une colonne ou une cellule, procéder comme suit :

1. Placer le curseur dans n'importe quel endroit de la table et cliquer sur l'icône  (Propriétés de table). Le dialogue Propriétés de table s'ouvre (*capture d'écran ci-dessous*), dans lequel vous pouvez spécifier le formatage pour la table, pour une ligne, une colonne ou une cellule.



2. Définir les propriétés d'espacement de cellule et de remplissage de la cellule sur "0". Votre table ressemble maintenant à celle ci-dessous :

3. Placer le curseur dans la première ligne pour la formater et cliquer sur l'icône  (Propriétés de table). Cliquer sur l'onglet **Ligne**.



Puisque la première ligne sera la ligne d'en-tête, définir une couleur d'arrière-plan pour faire ressortir cette ligne par rapport aux autres. Veuillez noter les propriétés Ligne qui ont été définies dans la figure ci-dessus. Ensuite, saisir le texte de l'en-tête de colonne. Votre table devrait ressembler à la figure ci-dessous :

Name	Telephone	Email

Veuillez noter que l'alignement est centré, tel que spécifié.

4. À présent, si, par exemple, vous souhaitez diviser la colonne "Telephone" en deux sous-colonnes "Office" et "Home", dans lequel cas vous devrez diviser la largeur de la colonne "Telephone" en deux colonnes. Tout d'abord, néanmoins, nous allons diviser la portée verticale de la cellule de l'en-tête pour en faire une ligne de sous-en-tête. Placer le curseur dans la cellule "Telephone" et cliquer sur l'icône  (Partager verticalement). Votre table devrait ressembler à la figure ci-dessous :

Name	Telephone	Email

5. À présent, placer le curseur dans la cellule contenant "Telephone", et cliquer sur l'icône  (Partager horizontalement). Ensuite saisir dans les en-têtes de colonne "Office" et "Home". Votre table devrait ressembler à la figure ci-dessous :

© 2016-2022 Altova GmbH

Name	Telephone		Email
	Office	Home	

À présent, vous devrez partager la largeur horizontale de chaque cellule dans la colonne "Telephone".

Vous pouvez aussi ajouter et supprimer des colonnes et des lignes et aligner verticalement du contenu de cellule, en utilisant les icônes d'édition de table. Les icônes d'édition de table CALS/HTML sont décrites dans la section titrée Icônes d'édition de table CALS/HTML.

Se déplacer dans les cellules de la table

Pour vous déplacer dans les cellules de la table CALS/HTML, utiliser les touches fléchées Haut, Bas, Gauche, Droite.

Saisir les données dans une cellule

Pour saisir les données dans une cellule, placer le curseur dans la cellule et saisir les données.

Formater un texte

Les textes contenus dans une table CALS/HTML, comme pour l'autre texte dans le document XML, doivent être formatés à l'aide d'élément ou d'attributs XML. Pour ajouter un élément, marquer le texte et double-cliquer sur l'élément nécessaire dans l'Assistant à la saisie Éléments. Pour spécifier une valeur d'attribut, placer le curseur dans le fragment de texte et saisir la valeur d'attribut nécessaire dans l'Assistant à la saisie Éléments. Une fois avoir formaté les textes d'en-tête en gras, votre table ressemblera à cela :

Name	Telephone		Email
	Office	Home	

Le texte ci-dessus a été formaté en marquant le texte puis en double-cliquant l'élément `strong`, pour lequel un modèle global existe qui spécifie "gras" en tant que le poids de la police. Le formatage de texte est visible immédiatement.

Note : pour afficher le formatage de texte dans Authentic View, un modèle global, avec le formatage de texte nécessaire doit avoir été créé dans StyleVision pour l'élément en question.

5.3.3 Icônes d'édition de table CALS/HTML

Les commandes nécessaires pour éditer les tables CALS/HTML sont disponibles sous la forme d'icônes dans la barre outils. Vous trouverez la liste ci-dessous. Veuillez noter que des commandes de menu correspondantes existent pour chacune de ces icônes. Pour une description complète de l'utilisation de Tables CALS/HTML, voir Tables CALS/HTML.

Insérer table

La commande "Insérer la table" insère une **table CALS/HTML** à la position actuelle du curseur.

Supprimer table

La commande "Supprimer la table" supprime la table active actuellement.

Apposer une ligne

La commande "Apposer une ligne" appose une ligne à la fin de la table active actuellement.

Apposer une colonne

La commande "Apposer une colonne" appose une colonne à la fin de la table active actuellement.

Insérer une ligne

La commande "Insérer une ligne" insère une ligne au-dessus de la position actuelle du curseur dans la table active actuellement.

Insérer une colonne

La commande "Insérer une colonne" insère une colonne à gauche de la position actuelle du curseur dans la table active actuellement.

Joindre cellule gauche

La commande "Joindre cellule gauche" joint la cellule actuelle (position actuelle du curseur) avec la cellule à sa gauche. Les balises des deux cellules demeurent dans la nouvelle cellule, les en-têtes de colonne demeurent inchangés et sont concaténés.

Joindre cellule droite

La commande "Joindre cellule droite" joint la cellule actuelle (position actuelle du curseur) avec la cellule à sa droite. Les contenus des deux cellules sont concaténés dans la nouvelle cellule.

Joindre cellule en-dessous

La commande "Joindre cellule en-dessous" joint la cellule actuelle (position actuelle du curseur) avec la cellule en-dessous. Les contenus des deux cellules sont concaténés dans la nouvelle cellule.

Joindre cellule au-dessous

La commande "Joindre cellule au-dessus" joint la cellule actuelle (position actuelle du curseur) avec la cellule au-dessus. Les contenus des deux cellules sont concaténés dans la nouvelle cellule.

© 2016-2022 Altova GmbH

Partager la cellule horizontalement

La commande "Partager la cellule horizontalement" crée une nouvelle cellule à droite de la cellule active actuellement. La taille des deux cellules est la même que la cellule originale.

Partager la cellule verticalement

La commande "Partager la cellule verticalement" crée une nouvelle cellule en-dessous de la cellule active actuellement.

Aligner en haut

Cette commande aligne les contenus de la cellule dans le haut de la cellule.

Centrer verticalement

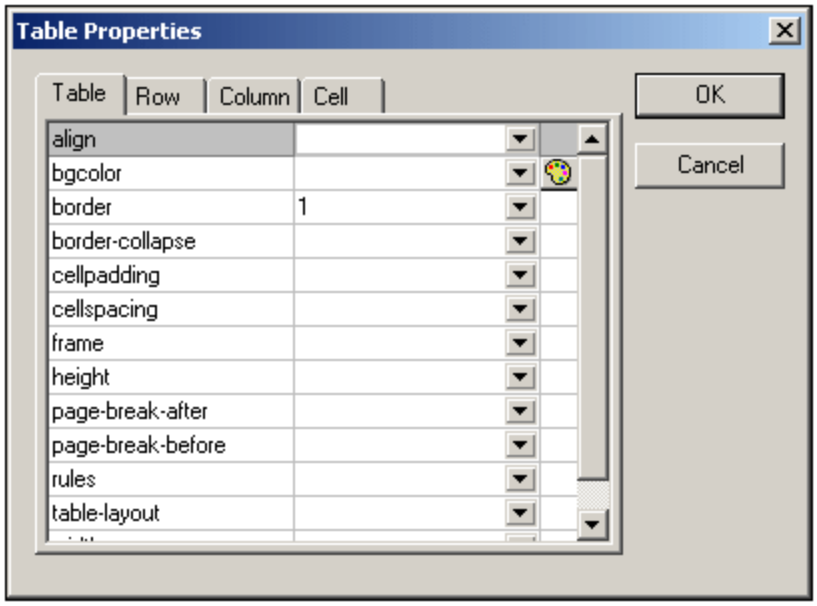
Cette commande recentre les contenus de la cellule.

Aligner en bas

Cette commande aligne les contenus de la cellule dans le bas de la cellule.

Propriétés de table

La commande "Propriétés de table" ouvre le dialogue Propriétés de table. Cette icône est uniquement active pour les tables HTML, elle ne peut pas être cliquée pour les tables CALS.



5.4 Éditer une BD

Dans le Authentic View, vous pouvez éditer des tables de bases de données (BD) et réenregistrer les données dans une BD. Cette section contient une description complète des fonctions d'interface à votre disponibilité lorsque vous éditez une table de BD. Les points généraux suivants doivent être notés :

- Le nombre d'enregistrements dans une table de BD qui est affiché dans Authentic View peut avoir été limité délibérément par le concepteur de la StyleVision Power Stylesheet pour rendre le design plus compact. Dans ces cas, seul ce nombre limité d'enregistrements sera chargé initialement dans Authentic View. À l'aide des icônes de navigation de ligne de table de BD (*voir* Parcourir une table de BD), vous pouvez charger et afficher les autres enregistrements dans la table de BD.
- Vous pouvez effectuer une requête de la BD pour afficher certains enregistrements.
- Vous pouvez ajouter, modifier et supprimer des enregistrements BD et réenregistrer vos changements dans la BD. Voir Modifier une table de BD.

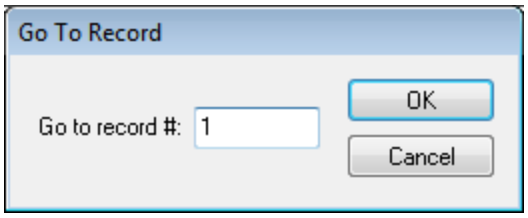
Pour ouvrir une StyleVision Power Stylesheet basée sur BD dans le Mode Authentic, cliquez sur **Authentic** | **Éditer les données de base de données**, et chercher la StyleVision Power Stylesheet requise.

Note : Dans le Mode Authentic, les données provenant d'une base de données SQLite ne peuvent pas être éditées. Lorsque vous tentez d'enregistrer des données SQLite depuis le Mode Authentic, une fenêtre de messages vous informera de cette limitation.

5.4.1 Parcourir une table de BD

Les commandes permettant de parcourir des lignes de table de BD sont disponibles sous la forme de touches dans le document Authentic View. Généralement, un volet de navigation comportant quatre ou cinq touches accompagne chaque table de BD.

Les icônes fléchées sont, de gauche à droite, Aller au premier enregistrement dans la table de BD ; Aller à l'enregistrement précédent ; Ouvrir le dialogue Aller à l'enregistrement (*capture d'écran ci-dessous*) ; Aller à l'enregistrement suivant et Aller au dernier enregistrement.



Pour parcourir une table de BD, cliquer sur la touche requise.

Bases de données XML

Dans le cas des BD XML, comme les IBM DB2, une cellule (ou ligne) contient un seul document XML, donc une seule ligne est chargée dans le Authentic View à la fois. Pour charger un document XML qui se trouve

© 2016-2022 Altova GmbH

dans une autre ligne, utiliser la commande de menu **Authentic | Sélectionner Nouvelle ligne avec les données XML pour l'édition**

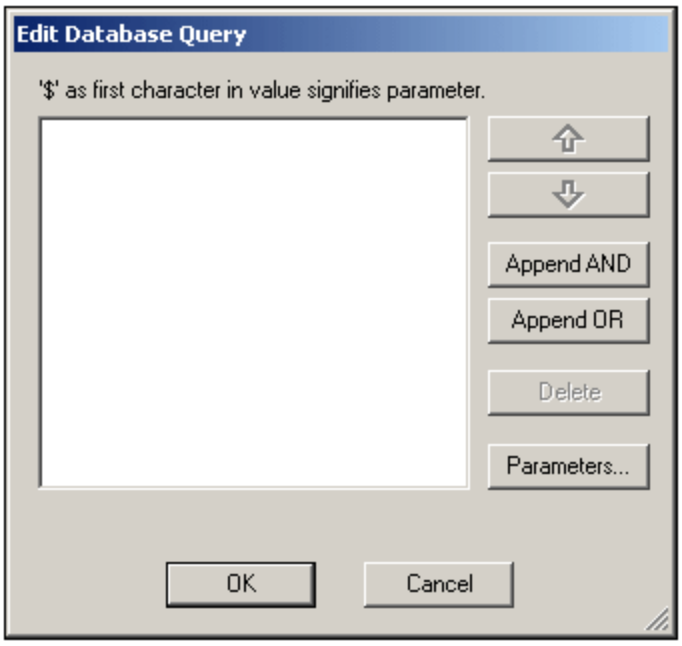
5.4.2 Requêtes BD

Une requête BD vous permet de requêter les enregistrements d'une table affichée dans le Authentic View. Une requête est effectuée pour une table individuelle et seule une requête peut être effectuée pour chaque table. Vous pouvez effectuer une requête à tout moment pendant l'édition. Si des modifications non enregistrées se trouvent dans votre document Authentic View au moment de la soumission de la requête, vous serez invité à enregistrer **tous** les changements effectués dans le document ou bien d'éliminer **tous** les changements. Veuillez noter que même les changements effectués dans d'autres tables seront enregistrés/éliminés. Une fois avoir soumis la requête, la table est rechargée à l'aide des conditions de requête.

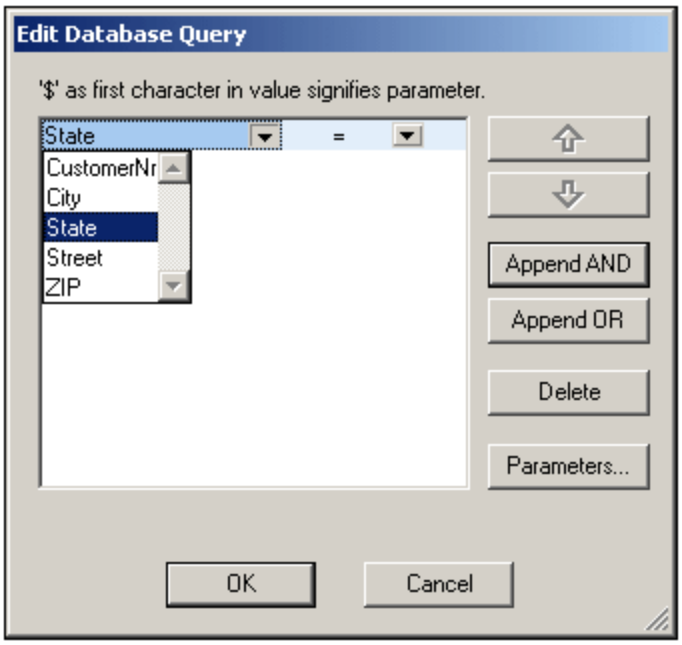
Note : Si vous voyez apparaître un message vous indiquant que trop de tables sont ouvertes, vous pourrez réduire le nombre de tables ouvertes en utilisant une requête pour filtrer certaines des tables.

Pour créer et soumettre une requête :

1. Cliquer sur la touche Requête  pour la table désirée afin d'ouvrir le dialogue Éditer la requête de base de données (*voir capture d'écran*). Ce bouton apparaît généralement en haut de chaque table BD ou en-dessous. Si la touche Requête n'apparaît pas pour une table, c'est que le concepteur de la StyleVision Power Stylesheet n'a pas activé la fonction de requête BD pour cette table.



2. Cliquer sur la touche **Apposer ET** ou **Apposer OU**. Cela vous permettra d'apposer un critère vide pour la requête (affiché ci-dessous).



3. Saisir l'expression pour le critère. Une expression consiste en : (i) un nom de champ (disponible depuis la liste de choix associée) ; (ii) un opérateur (disponible depuis la liste de choix associée) ; et (iii) une valeur (à saisir directement). Pour plus de détails concernant la construction d'expressions voir la section Expressions dans les critères.
4. Si vous souhaitez ajouter encore un critère, cliquer sur la touche **Apposer ET** ou **Apposer OU** selon l'opérateur logique (ET et OU) que vous souhaitez utiliser pour joindre les deux critères. Ensuite, ajouter le nouveau critère. Pour plus de détails concernant les opérateurs logiques, voir la section Réordonner les critères dans les requêtes BD.

Expressions dans les critères

Les expressions dans les critères de requête BD consistent en un nom de champ, un opérateur et une valeur. Les **noms de champ disponibles** sont les éléments enfants de la table de données de niveau supérieur sélectionnée ; les noms de ces champs sont recensés dans une liste de choix (*voir capture d'écran ci-dessus*). Les **opérateurs** que vous pouvez utiliser sont regroupés ci-dessous :

`=`	Égale à
`<>`	N'est pas égale à
`<`	Inférieur à
`<=`	Inférieur ou égale à
`>`	Supérieur à
`>=`	Supérieur ou égale à
`LIKE`	Phonétiquement semblable à
`NOT LIKE`	N'est pas phonétiquement semblable à
`IS NULL`	Est vide
`NOT NULL`	N'est pas vide

Si IS NULL ou NOT NULL a été sélectionné, le champ de valeur est désactivé. Les **Valeurs** doivent être saisies sans guillemets (ou tout autre délimiteur). Les valeurs doivent présenter le même formatage que celui du champ de BD correspondant ; sinon l'expression évaluera à `FALSE`. Par exemple, si un critère pour un

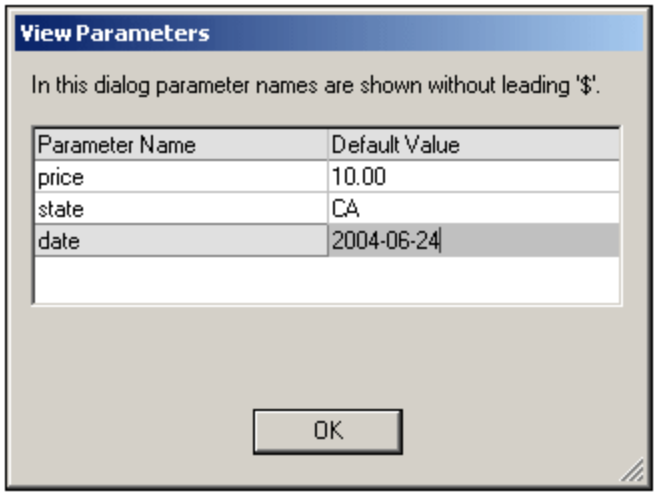
© 2016-2022 Altova GmbH

champ de type de données `date` dans une BD MS Access a une expression `StartDate=25/05/2004`, l'expression évaluera à `FALSE` parce que le type de données `date` dans une BD MS Access a un format `YYYY-MM-DD`.

Utiliser des paramètres avec les requêtes BD

Vous pouvez saisir le nom d'un **paramètre** en tant que la valeur d'une expression lorsque vous créez des requêtes. Les paramètres sont des variables qui peuvent être utilisées au lieu des valeurs littérales dans des requêtes. Lorsque vous les saisissez dans une expression, sa valeur est utilisée dans l'expression. Les paramètres qui sont disponibles ont été définis par le concepteur de SPS dans la SPS et peuvent être consultés dans le dialogue Afficher paramètres (*voir capture d'écran ci-dessous*). Une valeur par défaut a été attribuée aux paramètres dans la SPS. Celle-ci peut être contournée en passant une valeur au paramètre par le biais de la ligne de commande (si et quand le document de sortie est compilé par le biais de la ligne de commande).

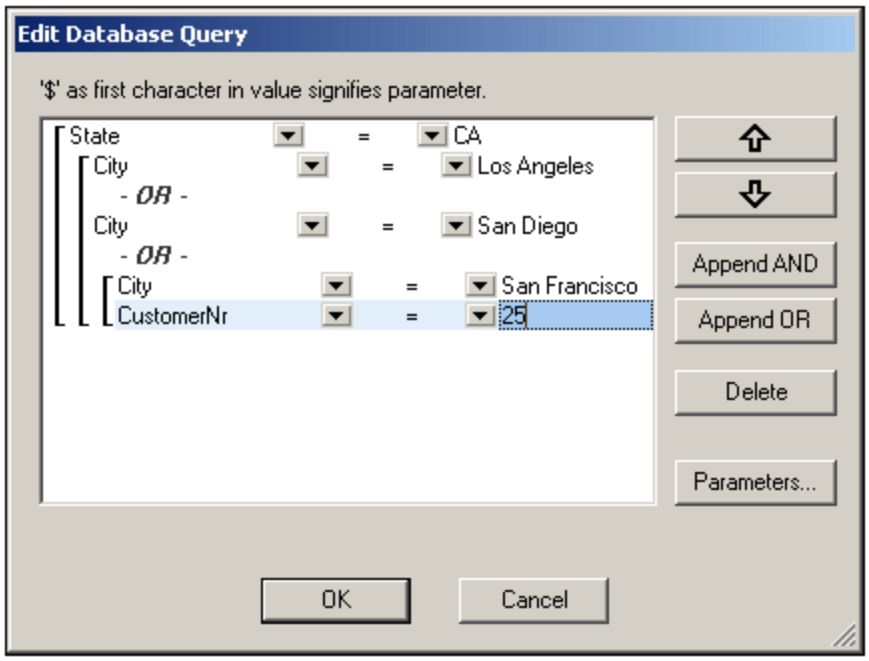
Afin de consulter les paramètres définis pour la SPS, cliquer sur la touche **Paramètres** dans le dialogue Éditer la requête de base de données. Le dialogue **Afficher les paramètres** s'ouvre (*voir capture d'écran*).



Le dialogue Afficher paramètres contient **tous** les paramètres qui ont été définis pour la feuille de style dans la SPS et les paramètres doivent être édités dans le design de la feuille de style.

Réordonner les critères dans les requêtes BD

La structure logique de la Requête de BD et les relations entre deux critères ou ensembles de critères est indiquée graphiquement. Chaque niveau de la structure logique est indiquée par un crochet. Deux critères ou ensembles de critères adjacents indiquent l'opérateur ET, alors que si deux critères sont séparés par le mot `OU`, alors l'opérateur OU sera indiqué. Les critères sont présentés en retrait, afin de permettre une lecture plus claire de la structure logique de la requête de BD.



La requête de BD affichée dans la capture d'écran ci-dessus peut être représentée dans le texte sous la forme suivante :

```
State=CA AND (City=Los Angeles OR City=San Diego OR (City=San Francisco AND
CustomerNr=25))
```

Vous pouvez réorganiser la requête de BD en déplaçant un critère ou un ensemble de critères vers le haut ou vers le bas par rapport aux autres critères dans la requête de BD. Pour déplacer un critère ou un ensemble de critères, procéder comme suit :

1. Sélectionner le critère en cliquant dessus, ou sélectionner un niveau entier en cliquant sur le crochet qui représente ce niveau.
2. Cliquer sur la touche Haut ou Bas dans le dialogue.

Les points suivants doivent être notés :

- Si le critère adjacent dans la direction du mouvement se trouve au même niveau, les deux critères échangent leur place.
- Un ensemble de critères (c.à.d. un critère dans un crochet) change de position dans le même niveau ; il ne change pas de niveau.
- Un critère individuel change de position dans le même niveau. Si le critère adjacent se trouve plus en avant ou en arrière (donc, pas au même niveau), alors le critère sélectionné se déplacera vers l'extérieur/intérieur, **un niveau à la fois**.

Pour supprimer un critère dans une requête de BD, sélectionner le critère et cliquer sur **Supprimer**.

Modifier une requête BD

Pour modifier une requête de BD:

© 2016-2022 Altova GmbH

1. Cliquer sur la touche Requête  . Le dialogue Éditer la requête de base de données s'ouvre. Vous pouvez maintenant éditer les expressions dans un des critères recensés, ajouter le nouveau critère, réordonner le critère ou supprimer le critère dans la requête de BD.
2. Cliquer sur **OK**. Les données provenant de la BD sont automatiquement rechargées dans Authentic View pour réfléchir les modifications de la requête de BD.

5.4.3 Modifier une table de BD

Ajouter un enregistrement
Pour ajouter un enregistrement dans une table de BD :

1. Placer le curseur dans la ligne de table de BD et cliquer sur l'icône  (pour apposer une ligne) ou sur l'icône  (pour insérer une ligne). Cela crée un nouvel enregistrement dans le fichier temporaire XML.
2. Cliquer sur la commande **Fichier | Enregistrer** pour ajouter le nouvel enregistrement dans la BD. Dans Authentic View, une ligne pour le nouvel enregistrement est apposée à l'affichage de table de BD. L'item `AltovaRowStatus` pour cet enregistrement est défini sur `A` (pour Ajouter).

Lorsque vous saisissez des données pour le nouvel enregistrement, celui-ci sera saisi en gras et souligné. Cela vous permet de différencier entre les enregistrements ajoutés et les enregistrements existants ; si les enregistrements existants n'ont pas été formatés avec ces propriétés de formatage de texte. Les erreurs de type de données sont marquées par un affichage rouge.

Le nouvel enregistrement est ajouté à la BD lorsque vous cliquez sur **Fichier | Enregistrer**. Une fois qu'un nouvel enregistrement a été enregistré sur la BD, son champ `AltovaRowStatus` est initialisé (indiqué par `---`) et l'enregistrement est affiché dans Authentic View en tant qu'un enregistrement normal.

Modifier un enregistrement

Pour modifier un enregistrement, placez le curseur à l'endroit nécessaire dans la table de BD et éditez l'enregistrement comme requis. Si le nombre d'enregistrements affichés est limité, vous devrez éventuellement chercher l'enregistrement requis (voir Parcourir une table BD).

Lorsque vous modifiez un enregistrement, les entrées dans tous les champs de l'enregistrement sont soulignées et l'item `AltovaRowStatus` de toutes les instances primaires de cet enregistrement est configuré sur `U` (pour Updated (mis à jour)). Toutes les instances secondaires de cet enregistrement ont leur `AltovaRowStatus` configuré sur `u` (minuscule). Les instances primaires et secondaires d'un enregistrement sont définies par la structure de la BD et correspondent au schéma XML généré. Par exemple, si une table Adresse est contenue dans une table Client, alors la table Adresse peut se produire dans le Document de Design dans deux types d'instanciation : en tant que la table Adresse elle-même et dans le cadre des instanciations de la table Client. Quel que soit l'un des deux types qui est modifié, il s'agira donc du type qui a été modifié primairement. Les autres types (il peut y avoir plus qu'un seul autre type) sont les types secondaires. Les erreurs de type de données sont marquées par un affichage en rouge.

Les modifications sont enregistrées dans la BD en cliquant sur **Fichier | Enregistrer**. Après qu'un enregistrement modifié ait été enregistré sur la BD, son champ `AltovaRowStatus` est initialisé (indiqué avec `---`) et l'enregistrement est affiché dans Authentic View en tant qu'un enregistrement normal.

Notez les points suivants :

- Si un seul champ d'un enregistrement est modifié dans Authentic View, c'est tout l'enregistrement qui sera mis à jour lorsque les données seront enregistrées sur la BD.
- La valeur de date `0001-01-01` est définie en tant que valeur `NULL` pour certaines BD et peuvent déclencher un message d'erreur.

Supprimer un enregistrement

Pour supprimer un enregistrement :

1. Placer le curseur dans la ligne représentant l'enregistrement à supprimer et cliquer sur l'icône  . L'enregistrement à supprimer est marqué par un texte barré. `AltovaRowStatus` est configuré comme suit : les instances primaires de l'enregistrement sont définies sur `D` ; les instances secondaires sur `d` ; et les enregistrements indirectement supprimés sur `X`. Les enregistrements indirectement supprimés sont des champs dans l'enregistrement supprimé qui sont conservés dans une table séparée. Par exemple, une table Adresse peut être incluse dans une table Client. Si un enregistrement Client est supprimé, son enregistrement Adresse correspondant sera supprimé indirectement. Si un enregistrement Adresse dans la table Client est supprimé, l'enregistrement Adresse dans la table Client sera supprimé primairement, mais le même enregistrement sera supprimé secondairement dans une table Adresse indépendante si cela a été instancié.
2. Cliquer sur **Fichier | Enregistrer** pour enregistrer des modifications dans la BD.

Note : Enregistrer les données sur la BD réinitialise la commande Annuler, vous ne pourrez donc pas annuler des actions qui ont été effectuées avant l'enregistrement.

© 2016-2022 Altova GmbH

5.5 Travailler avec des dates

Vous disposez de deux manières pour éditer les dates dans Authentic View:

- Les dates sont saisies ou modifiées à l'aide du Sélectionneur de date.
- Les dates sont saisies ou modifiées en saisissant la valeur.

La méthode que l'utilisateur d'Authentic View utilisera est définie dans la SPS. Les deux méthodes sont décrites dans les deux sous-sections de cette section.

Note sur les formats de date

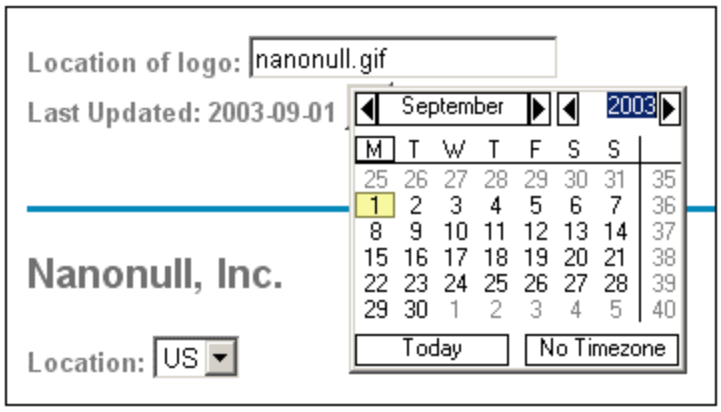
Dans le document XML, les dates peuvent être stockées dans un des divers types de données. Chacun de ces types de ces types de données exige que la date soit stockée dans un format lexical particulier pour que le document XML soit valide. Par exemple, le types de données `xs:date` exige un format lexical de type `YYYY-MM-DD`. Si la date dans un nœud `xs:date` est saisie dans un autre format que celui-ci, le document XML sera invalide.

Afin de vous assurer que la date est saisie dans le format correct, le concepteur de la SPS peut inclure le Sélecteur de date graphique dans son design. Cela permet d'assurer que la date sélectionnée dans le Sélecteur de date est saisie dans le format graphique correct. S'il n'y a pas de Sélecteur de données, le Authentic View devra veiller à saisir la date dans le format lexical correct. Valider le document XML peut fournir des astuces utiles concernant le format lexical requis.

5.5.1 Sélectionneur de date

Le Sélectionneur de date est un calendrier graphique utilisé pour saisir des dates dans un format standard dans le document XML. Le fait de disposer d'un format standard est important pour le traitement des dates dans le document. L'icône du Sélectionneur de date apparaît près du champ de date qu'elle modifie (*voir capture d'écran*).

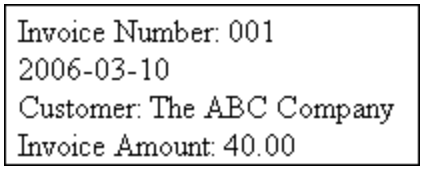
Afin d'afficher le Sélectionneur de date (*voir capture d'écran*), cliquer sur l'icône de Sélectionneur de date.



Pour sélectionner une date, cliquer sur la date, le mois, ou l'année désirée. La date est saisie dans le document XML et la date dans l'affichage est modifiée conformément. Vous pouvez aussi saisir un fuseau horaire le cas échéant.

5.5.2 Entrée de texte

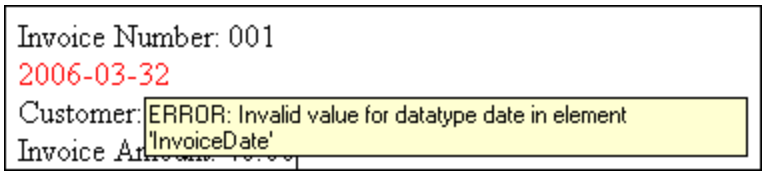
Pour les champs de date qui ne possèdent pas de Sélectionneur de date (*voir capture d'écran*), vous pouvez éditer la date directement en saisissant la nouvelle valeur.



Erreurs

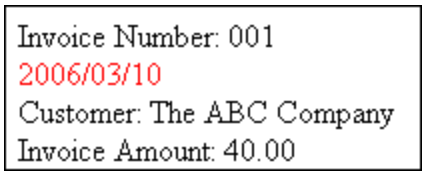
Les types d'erreur suivants seront marqués d'un indicateur :

- Si vous éditez une date et que vous la modifiez de manière à ce qu'elle sorte de la plage valide, la date passe au rouge pour vous avertir de l'erreur. Si vous placez le curseur de la souris sur la date invalide, un message d'erreur apparaît (*voir capture d'écran*).



- Si vous essayez de modifier le format de la date, la date passe en rouge pour vous avertir de l'erreur. (Dans la capture d'écran ci-dessous, les barres obliques sont utilisées au lieu des traits d'union).

© 2016-2022 Altova GmbH


Invoice Number: 001
2006/03/10
Customer: The ABC Company
Invoice Amount: 40.00

5.6 Définir les entités

À propos des entités

Vous pouvez définir des entités à utiliser dans le Authentic View, que votre document soit basé sur un DTD ou un Schéma XML. Une fois définies, ces entités sont affichées dans l'Assistant à la saisie Entités et dans le sous-menu **Insérer entité** du menu contextuel. Lorsque vous double-cliquez sur une entité dans l'Assistant à la saisie Entités, cette entité est insérée à l'endroit du point d'insertion du curseur.

Une entité est utile si vous utilisez une chaîne de texte, un fragment XML, ou certaines des autres ressources externes dans plusieurs emplacements de votre document. Vous définissez l'entité, constituée en principe d'un nom bref qui symbolise les données requises dans le dialogue Définir les entités. Une fois avoir défini une entité, vous pourrez l'utiliser dans plusieurs endroits de votre document. Cela vous permet d'économiser du temps et facilite considérablement l'entretien.

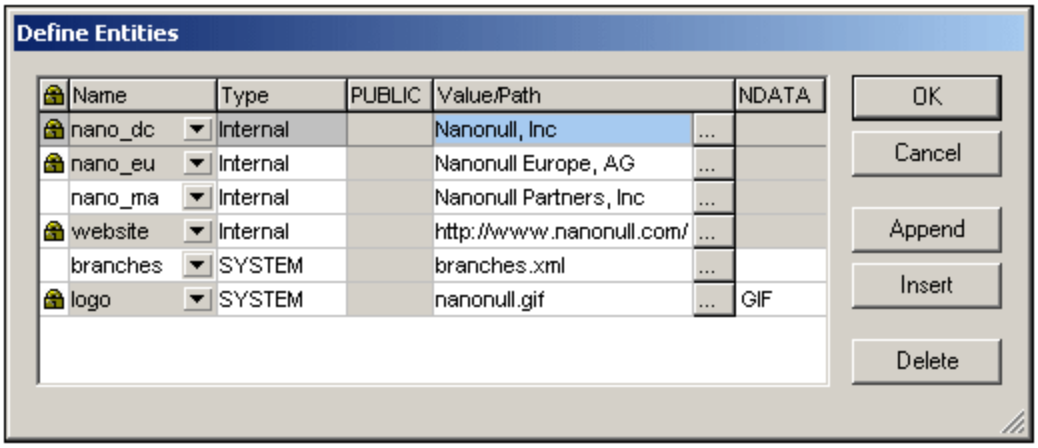
Types d'entités

Il existe deux types d'entités que vous pouvez utiliser dans votre document : une **entité parsée,** soit des données XML (soit une chaîne de texte ou un fragment d'un document XML), ou une **entité non parsée**, soit des données non-XML comme un fichier binaire (généralement un graphique, un son ou un objet multimédia). Chaque entité porte un nom et une valeur. Dans le cas des entités parsées, l'entité est un espace réservé pour les données XML. La valeur de l'entité est soit les données XML elles-mêmes ou une URI qui pointe vers un fichier `.xml` qui contient les données XML. Dans le cas des entités non parsées, la valeur de l'entité est une URI qui pointe vers le fichier de données non-XML.

Définir les entités

Pour définir une entité :

1. Cliquer sur **Authentic | Définir les entités XML**. Cela permet d'ouvrir le dialogue Définir les entités (*capture d'écran ci-dessous*).



2. Saisir le nom de votre entité dans le champ Nom. Il s'agit du nom qui apparaîtra dans l'Assistant à la saisie Entité.
3. Saisir le type d'entité depuis la liste déroulante dans le champ Type. Les types suivants sont possibles : une entité **Interne** est une entité pour laquelle le texte à utiliser est stocké dans le document XML lui-même. La sélection de **PUBLIC** ou de **SYSTEM** spécifie que la ressource est située en-dehors du

© 2016-2022 Altova GmbH

fichier XML, et qu'elle sera située avec l'utilisation d'un identificateur public ou un identificateur de système respectivement. Un identificateur de système est une URI qui donne l'emplacement de la ressource. Un identificateur public est un identificateur indépendant de l'emplacement qui permet à certains processeurs d'identifier la ressource. Si vous spécifiez aussi bien un identificateur public que système, l'identificateur public résout à l'identificateur système, et l'identificateur système est utilisé.

4. Si vous avez utilisé PUBLIC en tant que Type, saisir l'identificateur public de votre ressource dans le champ PUBLIC. Si vous avez sélectionné Internal ou SYSTEM en tant que votre Type, le champ PUBLIC est désactivé.
5. Dans le champ Valeur/Chemin, vous pouvez saisir un des items suivants :

   - Si le type d'entité est Interne, saisir la chaîne de texte que vous souhaitez en tant que la valeur de votre entité. Ne pas saisir de guillemets pour délimiter la saisie. Les guillemets que vous saisissez seront traités en tant que partie de la chaîne de texte.
   - Si le type d'entité est SYSTEM, saisir l'URI de la ressource ou sélectionner une ressource sur votre réseau local en utilisant la touche Parcourir. Si la ressource contient des données parsées, il doit s'agir d'un fichier XML (il doit donc avoir une extension `.xml`). En alternative, la ressource peut être un fichier binaire, comme un fichier GIF.
   - Si le type d'entité est PUBLIC, vous devrez saisir en plus un identificateur dans ce champ.

6. L'entrée NDATA indique au processeur que cette entité ne doit pas être parsée mais envoyée au processeur approprié. Le champ NDATA doit donc contenir certaines valeurs pour indiquer que l'entité est une entité non-parsée.

Fonctions de dialogue

Le dialogue Définir entités vous permet d'effectuer les tâches suivantes :

- Apposer des entités
- Insérer des entités
- Supprimer des entités
- Trier des entités par la valeur alphabétique de toute colonne en cliquant sur l'en-tête de colonne ; cliquer une fois pour trier dans l'ordre ascendant, cliquer deux fois pour trier dans l'ordre descendant.
- Redimensionner la fenêtre de dialogue et la largeur des colonnes.
- Verrouillage. Une fois qu'une entité est utilisée dans le document XML, elle est verrouillée et ne peut pas être éditée dans le dialogue Définir entités. Les entités verrouillées sont indiquées par un symbole de cadenas dans la première colonne. Le verrouillage d'une entité permet de garantir que le document XML soit valide par rapport aux entités. (Le document serait invalide si une entité est référencée mais pas définie.)
- Les entrées double sont marquées.

Limitations des entités

- Une entité contenue dans une autre entité n'est pas résolue, soit dans le dialogue, le Authentic View, ou la sortie XSLT, et le caractère esperluette d'une telle entité est affiché dans sa forme échappée : `&`.
- Les entités non-parsées qui ne sont pas des fichiers d'image ne sont pas résolues dans le Authentic View. Si une image dans le design est définie pour lire une entité non parsée externe et dont l'URI est configurée pour être un nom d'entité (par exemple : `'logo'`), alors ce nom d'entité peut être défini dans le dialogue Définir entités (*capture d'écran ci-dessus*) en tant qu'une entité non-parsée externe avec une valeur qui résout à l'URI du fichier image (comme cela a été fait pour l'entité `logo` dans la capture d'écran ci-dessus).

© 2016-2022 Altova GmbH

5.7 Signatures XML

Une SPS peut être conçue avec une signature XML configurée pour le Authentic View. Lorsque les signatures XML sont activées dans la SPS, l'utilisateur du Authentic View peut signer numériquement le fichier Authentic XML par le biais de l'activation de la signature. Une fois que le document a été signé, toute modification entraînera une vérification de la signature. Lorsqu'un document Authentic XML signé est ouvert dans le Authentic View de tout produit Altova, le processus de vérification sera exécuté sur le document et le résultat de la vérification sera affiché dans une fenêtre.

Note : les signatures XML peuvent être utilisées et seront vérifiées, dans le Mode Authentic des éditions Enterprise et Professional des produits Altova suivants : Authentic Desktop, Authentic Browser, XMLSpy et StyleVision.

Actions de signature XML

Les actions d'utilisateur du Mode Authentic suivantes sont possibles :

- *Choisir le certificat/mot de passe :* les signatures sont authentifiées avec un certificat ou un mot de passe. L'objet de l'authentification (certificat ou mot de passe) est exigé tout d'abord lorsque la signature est créée et à nouveau lorsqu'il est vérifié. Si un document XML Authentic a une SPS avec activation de la signature qui lui est attribuée, la SPS peut spécifier un certificat ou un mot de passe par défaut pour la signature. Qu'un certificat ou un mot de passe par défaut ait été spécifié ou pas, la signature peut être configurée pour permettre à l'utilisateur du Authentic View de sélectionner son propre certificat ou un mot de passe. L'utilisateur du Authentic View peut y recourir à tout moment dans le dialogue Signature XML (*capture d'écran ci-dessous*). La sélection d'un certificat ou d'un mot de passe personnel annule le certificat ou le mot de passe par défaut. Le certificat ou le mot de passe personnel est stocké dans la mémoire et est utilisé pour la session actuelle. Si, après qu'un certificat ou un mot de passe personnel ait été sélectionné, l'utilisateur du Authentic View ferme le fichier ou l'application, la SPS repasse à ses paramètres par défaut pour le certificat ou le mot de passe.

- *Signer le document:* le document XML Authentic peut être signé automatiquement ou manuellement. La signature automatique aura été spécifiée dans la configuration de la signature par le concepteur de la SPS et entraîne la signature automatique du document XML Authentic lors de l'enregistrement. Si l'option de signature automatique n'a pas été activée, le document peut être signé manuellement. Pour ce faire, cliquer sur l'icône  dans la barre d'outils Signature XML ou avec la commande **Authentic | Signature XML**, et, dans le dialogue Signature XML qui apparaît (*capture d'écran ci-dessus*), cliquer sur la touche **Signer Document**. Veuillez noter que la signature du document avec une signature intégrée nécessiterait que le schéma permette l'élément `Signature` en tant que le dernier élément enfant de l'élément racine (document). Sinon, le document sera invalide par rapport au schéma. Lors de la signature du document, l'objet d'authentification et le placement de la signature sont déterminés conformément à la configuration de la signature. Vous devez vous assurer que vous avez accès à l'information d'authentification. Pour plus d'information à ce propos, consulter votre concepteur de SPS.

- *Vérifier le document XML Authentic :* si une les Signatures XML sont activées dans la SPS, le processus de vérification sera exécuté sur la signature à chaque fois que le document XML du Authentic View est chargé. Si le mot de passe ou l'information de la clé de certificat n'est pas enregistrée avec la SPS et la signature, respectivement, l'utilisateur du Authentic View sera invité à saisir le mot de passe ou à sélectionner un certificat pour la vérification. Veuillez noter que si une signature intégrée est générée, elle sera enregistrée avec le fichier XML lorsque celui-ci est enregistré. La signature générée doit être supprimée explicitement (par le biais de la touche **Supprimer la signature** du dialogue Signature XML ; *voir capture d'écran ci-dessus*) si vous ne souhaitez pas

l'enregistrer avec le fichier XML. De même, si une signature détachée est générée, elle aussi doit être supprimée si cela n'est pas requis.

© 2016-2022 Altova GmbH

5.8 Images dans le Mode Authentic

Authentic View vous permet de spécifier des images qui seront utilisées dans le document de sortie final (HTML, RTF, PDF et Word 2007). Vous devriez noter que certains formats d'image ne sont pas toujours pris en charge dans certains formats ou certaines applications. Par exemple, le format SVG est pris en charge dans PDF, mais pas dans RTF et nécessiterait un add-on de navigateur pour pouvoir être consultés dans HTML. Ainsi, lorsque vous sélectionnez un format d'image, veuillez vous assurer de sélectionner un format qui est pris en charge dans les formats de sortie de votre document. La plupart des formats d'image sont pris en charge sur tous les formats de sortie (*voir la liste ci-dessous*).

Authentic View est basé sur Internet Explorer, et est capable d'afficher la plupart des formats d'image que votre version d'Internet Explorer peut afficher. Les formats d'image utilisés communs suivants sont pris en charge :

- GIF
- JPG
- PNG
- BMP
- WMF (Microsoft Windows Metafile)
- EMF (Enhanced Metafile)
- SVG (uniquement pour la sortie PDF)

Chemins relatifs

Les chemins relatifs sont résolus relativement au fichier SPS.

5.9 Séquences de touche dans le Mode Authentic

Touche Entrée

Dans le Authentic View la touche **Entrée** est utilisée pour ajouter des éléments supplémentaires lorsqu'elle se trouve dans certains emplacements de curseur. Par exemple, si le chapitre d'un livre peut (conformément au schéma) contenir certains paragraphes, une pression sur **Entrée** dans le texte du paragraphe entraîne l'ajout immédiat d'un nouveau paragraphe après le paragraphe actuel. Si un chapitre peut contenir un titre et plusieurs paragraphes, appuyer **Entrée** dans le chapitre, mais en dehors de tout élément de paragraphe (y compris dans l'élément du titre) entraînera l'ajout d'un nouveau paragraphe après le chapitre actuel (dans le cas où plusieurs chapitres sont autorisés par le schéma).

Note : la touche **Entrée ne permet pas** d'insérer une nouvelle ligne. Cela est le cas lorsque le curseur se trouvant dans un nœud de texte, comme un paragraphe.

Utiliser le clavier

Le clavier peut être utilisé de la manière standard, pour la saisie et la navigation. Veuillez noter les points spéciaux suivants :

- La touche **Tab** déplace le curseur vers l'avant, s'arrêtant avant et après les nœuds, et soulignant les contenus de nœud ; elle passe au-dessus des contenus statiques.
- Les hyperliens `add...` et `add Node` sont considérés comme des contenus de nœud et sont soulignés lorsque vous les parcourez avec la touche Tab. Ils peuvent être activés soit en appuyant sur la barre espace soit sur la clé **Entrée**.

© 2016-2022 Altova GmbH

6 Création de scripts Authentic

La fonction **Création de script Authentic** permet une plus grande flexibilité et interactivité pour les designs SPS. Ces designs peuvent être créés ou édités dans les éditions StyleVision Enterprise et Professional et peuvent être consultés dans le Mode Authentic des éditions Enterprise et Professional des produits Altova.

Une liste complète de prise en charge pour cette fonction dans les produits Altova est indiquée dans la table ci-dessous. Veuillez noter, néanmoins, que dans la version trusted du plug-in Authentic Browser, la création de script interne est éteinte pour des raisons de sécurité.

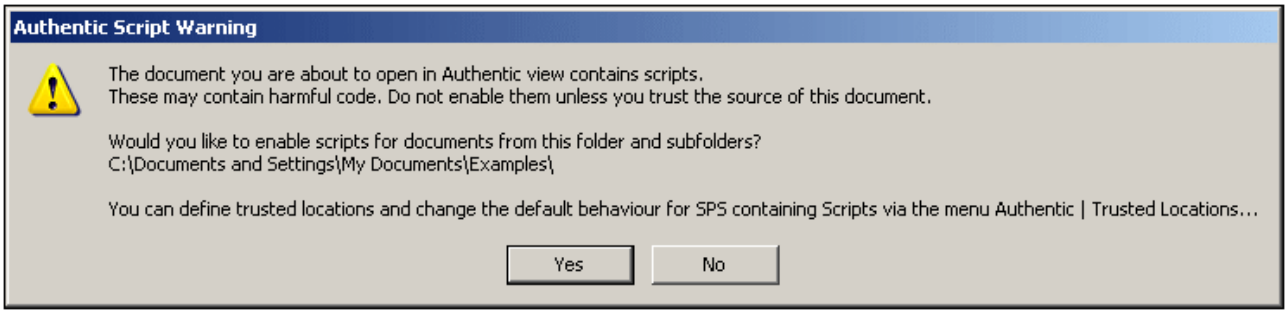
Produit Altova	Création de scripts Authentic	Scripts Authentic activés
StyleVision Enterprise	Oui	Oui
StyleVision Professional	Oui	Oui
StyleVision Basic *	Non	Non
XMLSpy Enterprise	Non	Oui
XMLSpy Professional	Non	Oui
AuthenticDesktop Enterprise	Non	Oui
Authentic Browser Plug-in Enterprise Trusted **	Non	Oui
Authentic Browser Plug-in Enterprise Untrusted	Non	Oui

* *Pas de Mode Authentic*
** *Les designs scriptés sont affichés. Pas d'exécution de macro interne ou de gestion des événements. Les événements externes sont déclenchés.*

Les Scripts Authentic se comportent de la même manière que dans tous les produits Altova, aucun code spécifique à un produit ou paramètres ne sont donc requis.

Dialogue d'avertissement Script Authentic

Si un fichier PXF, ou un fichier XML lié à une SPS, contient un script et que le fichier est ouvert ou passé au Mode Authentic, alors un dialogue d'avertissement (*capture d'écran ci-dessous*) s'ouvre.



Vous pouvez choisir parmi une des options suivantes :

- Cliquer sur **Oui** pour ajouter le dossier contenant le fichier à la liste Trusted Locations pour les scripts Authentic. Par conséquent, tous les fichiers dans le dossier trusted seront ouverts dans le Mode Authentic sans que ce message d'avertissement s'ouvre tout d'abord. La liste Trusted Locations est accessible par le biais de la commande de menu **Authentic | Trusted Locations**, et peut être modifiée.
- Cliquer sur **Non** pour ne pas ajouter le dossier contenant le fichier à la liste Trusted Locations. Le fichier sera affiché dans le Mode Authentic et les scripts seront désactivés. Le dialogue d'avertissement Script Authentic apparaîtra à chaque fois que ce fichier sera ouvert dans le Mode Authentic. Pour ajouter ultérieurement le dossier du fichier à la liste Trusted Locations, ouvrir le dialogue Trusted locations par le biais de la commande de menu **Authentic | Trusted Locations**, et ajouter le dossier ou le modifier le cas échéant.

Pour une description du dialogue Trusted Locations, voir la description de la commande de menu **Authentic | Trusted Locations** dans la Référence de l'utilisateur.

Note : lorsque Authentic Desktop est accédé par le biais de son interface COM (voir Référence du programmeur pour plus d'instructions), **le contrôle de sécurité n'est pas effectué** et le **dialogue d'avertissement du script Authentic n'est pas affiché**.

Comment fonctionne la Création de script Authentic

Le concepteur du design SPS peut utiliser la Création de script Authentic de deux manières pour rendre les documents Authentic interactifs :

- En attribuant des scripts pour les actions définies par l'utilisateur (macros) pour concevoir des éléments, des touches de barre d'outils, et des items de menu contextuel.
- En ajoutant au design des gestionnaires d'événement qui réagissent aux événements de Mode Authentic.

Toutes les créations de script nécessaires pour rendre interactifs les documents Authentic sont effectuées dans le cadre de la GUI StyleVision (éditions Enterprise et Professional). Les formulaires, les macros et les gestionnaires d'événement sont créés dans le cadre de l'interface de l'éditeur de création de script de StyleVision et ces scripts sont enregistrés dans la SPS. Ensuite, dans le Mode Design de StyleVision, les scripts enregistrés sont attribués aux éléments de design, aux boutons de la barre d'outils et aux menus contextuels. Lorsqu'un document XML basé sur la SPS est ouvert dans un produit Altova qui prend en charge la Création de script Authentic (*voir table ci-dessus*), le document présentera la flexibilité supplémentaire et l'interactivité qui auront été créées pour lui.

Documentation pour la Création de script Authentic

La documentation pour la Création de script Authentic est disponible dans la documentation de StyleVision. Elle peut être consultée en ligne dans la page de Documentation du produit sur le site Internet Altova.

© 2016-2022 Altova GmbH

7 Mode Navigateur

Le Mode Navigateur est généralement utilisé pour consulter :

- des fichiers XML qui possèdent un fichier associé XSLT. Lorsque vous passez au Mode Navigateur, le fichier XML est transformé immédiatement à l'aide de la feuille de style XSLT et le résultat est affiché directement dans le Mode Navigateur.
- Les fichiers HTML qui sont créés directement en tant que HTML ou créés par le biais d'une transformation XSLT d'un fichier XML.

Pour consulter les fichiers XML et HTML dans le Mode Navigateur, cliquez sur l'onglet **Navigateur**.

Moteurs de navigation dans le Mode Navigateur

Par défaut, le Mode Navigateur utilise actuellement Microsoft's Internet Explorer comme son moteur de navigation. Si vous souhaitez utiliser le moteur de navigation Microsoft's newer Edge WebView2 pour le Mode Navigateur, vous pouvez sélectionner cette option dans la section Affichage du dialogue des Options.

Note : Puisque Microsoft Edge WebView2 utilise le projet de logiciel Chromium, sur lequel le navigateur Chrome de Google est basé, utiliser WebView2 pour le Mode navigateur fournit également un bon aperçu de l'affichage Chrome de la page web.

Notes relatives à Microsoft Internet Explorer

Le Mode Navigateur requiert Microsoft's Internet Explorer 5.0 ou plus, ou Microsoft Edge WebView2 (*voir ci-dessus*).

Notez les points suivants d'Internet Explorer dans le Mode Navigateur :

- Si vous souhaitez utiliser le Mode Navigateur pour consulter des fichiers XML transformés par une feuille de style XSLT, nous vous recommandons vivement d'utiliser Internet Explorer 6.0 ou plus, qui utilise MSXML 3.0, un parseur XML prenant en charge le standard XSLT 1.0. Vous pouvez également installer MSXML 4.0.
- La prise en charge pour XSLT en IE 5 n'est pas compatible à 100 % avec la recommandation officielle XSLT. Donc si vous rencontrez un problème dans le Mode Navigateur avec IE 5, essayez de passer à IE 6 ou plus.
- En général, Vous devriez contrôler la prise en charge de XSLT de votre version d'Internet Explorer.
- Si vous rencontrez des problèmes avec l'affichage correct de HTML dans Internet Explorer, incluez l'onglet **`meta`** suivant dans l'élément **`head`** de votre document HTML :

```
<head>
... <meta http-equiv="X-UA-Compatible" content="ie=edge">...
</head>
```

Outils de développeurs dans le Mode Navigateur

Vous pouvez utiliser les Outils de développeur du navigateur Edge por inspecter, dégoguer et tester votre code HTML. Pour ouvrir les outils, cliquez avec la touche de droite dans le volet du Mode Navigateur et sélectionnez **Open Developer Tools**.

Fonctions Mode Navigateur

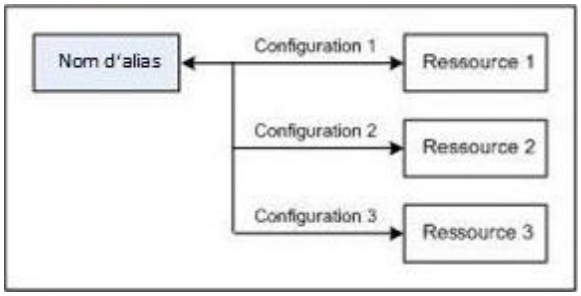
Les fonctions suivantes sont disponibles dans le Mode Navigateur. Elles peuvent être accédées via le menu **Navigateur**, le menu **Fichier** et le menu **Éditer**.

- *Ouvrir dans une fenêtre séparée :* Lorsque le Mode Navigateur est une fenêtre séparée, elle peut être positionnée côte à côte avec un mode d'édition du même document. À cette fin, cliquez sur la commande du menu **Navigateur | Fenêtre séparée.** Il s'agit d'une commande à bascule qui bascule le Mode Navigateur entre deux fenêtres : (i) une fenêtre séparée, et (ii) un affichage tabulaire dans la fenêtre principale. Ces commandes sont aussi disponibles dans le menu déroulant du bouton **Mode navigateur** (à la fin de la fenêtre principale).
- *Avant et arrière :* Le navigateur commun envoie la commande de naviguer à travers les pages qui ont été chargées dans le Mode Navigateur. Ces commandes sont dans le menu **Navigateur**.
- *Taille de police :* Peut être ajustée par le biais du menu **Navigateur**.
- *Arrêter, Actualiser, Imprimer :* Plus de commandes de navigateur standard, celles-ci peuvent être trouvées dans les menus **Navigateur** et **Fichier**.
- *Recherche :* Permet des recherches pour des strings de texte. Cette commande se trouve dans le menu **Éditer**.
- *Fenêtre Info :* Il existe des options ici pour afficher la page HTML active avec tout navigateur web installé sur l'appareil et pour ouvrir ou déplacer les navigateurs installés.

© 2016-2022 Altova GmbH

8 Ressources Globales Altova

Les ressources globales Altova sont une collection d'alias pour les ressources de fichier, dossiers et bases de données. Chaque alias peut posséder plusieurs configurations et chaque configuration mappe vers une ressource unique (*voir capture d'écran ci-dessous*). C'est pourquoi, lorsqu'une ressource globale est utilisée en tant qu'entrée, la ressource globale peut être changée dans le cadre de ses configurations. Pour ce faire, il suffit d'utiliser les commandes dans la GUI qui vous permettent de sélectionner la configuration active.



L'utilisation des Ressources globales Altova implique deux processus :

- Définir les Ressources globales : les ressources sont définies et les définitions sont stockées dans un fichier XML. Ces ressources peuvent être partagées avec plusieurs applications Altova.
- Utiliser les Ressources globales : dans le cadre de Authentic Desktop, les fichiers peuvent être situés par le biais d'une ressource globale et non par le biais d'un chemin de fichier. L'avantage est que la ressource peut être modifiée en changeant la configuration active dans Authentic Desktop.

Les ressources globales dans d'autres produits Altova

Actuellement, les ressources globales peuvent être définies et utilisées dans les produits Altova suivants : XMLSpy, StyleVision, MapForce, Authentic Desktop, MobileTogether Designer, et DatabaseSpy.

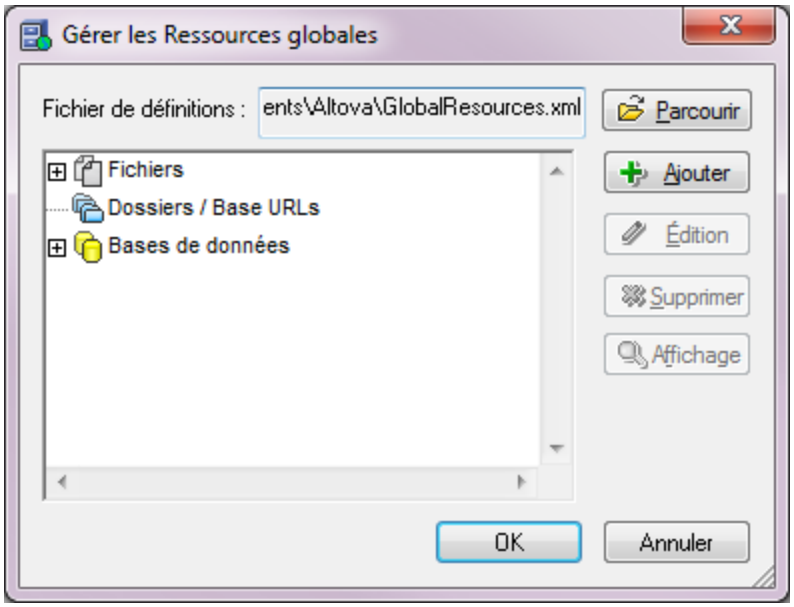
8.1 Définir les Ressources globales

Les Ressources globales Altova sont définies dans le dialogue de Gestion des Ressources globales, qui peut être accédé de deux manières :

- Cliquer sur la commande de menu **Outils | Ressources globales**.
- Cliquer sur l'icône **Gérer les Ressources globales** dans la barre outils des Ressources globales (*capture d'écran ci-dessous*).

Le fichier de Définitions des Ressources globales

Les informations concernant les ressources globales sont stockées dans un fichier XML appelé le Fichier de Définitions des Ressources globales. Ce fichier est créé lorsque la première ressource globale est définie dans le dialogue Gérer les Ressources globales (*capture d'écran ci-dessous*) et enregistré.



Lorsque vous ouvrez le dialogue Gérer les Ressources globales pour la première fois, l'emplacement par défaut et le nom du fichier Définitions des Ressources globales est spécifié dans la fenêtre de texte *Fichier de définitions* (*voir capture d'écran ci-dessous*) :

```
C:\Users\<username>\My Documents\Altova\GlobalResources.xml
```

Ce fichier est configuré en tant que le fichier de Définitions des ressources globales par défaut pour toutes les applications Altova. Donc une ressource globale peut être enregistrée depuis n'importe quelle application Altova sur ce fichier et elle sera immédiatement disponible pour toutes les autres applications Altova en tant que ressource globale. Afin de définir et enregistrer une ressource globale dans le fichier Définition des Ressources globales, ajouter la ressource globale dans le dialogue Gérer les Ressources globales et cliquer sur **OK** pour enregistrer.

© 2016-2022 Altova GmbH

Pour sélectionner un fichier de Définitions des ressources globales existant qui deviendra le fichier de définitions actif d'une application Altova particulière, chercher le fichier à l'aide du bouton **Parcourir** de la fenêtre de texte de *Fichier de définitions* (*voir capture d'écran ci-dessus*).

Note : Vous pouvez renommer le fichier de Définitions des ressources globales comme vous le souhaitez et l'enregistrer à tout endroit accessible de vos applications Altova. Tout ce que vous devez faire dans chaque application est de spécifier ce fichier en tant que le fichier de Définitions des ressources globales pour cette application (dans la fenêtre de texte *Fichier de définitions*). Les ressources deviennent globales pour tous les produits Altova si vous utilisez un seul fichier de définitions pour tous les produits Altova.

Note : Vous pouvez aussi créer des fichiers de Définitions des ressources globales multiples. Néanmoins, seul un d'entre eux peut être actif à tout moment dans une application Altova donnée, et seules les définitions contenues dans ce fichier seront disponibles pour l'application. La disponibilité des ressources peut donc être restreinte ou peut toucher plusieurs produits, tel que vous le souhaitez.

Gérer les ressources globales : ajouter, éditer, supprimer, enregistrer

Dans le dialogue Gérer les Ressources globales (*capture d'écran ci-dessus*), vous pouvez ajouter une ressource globale au fichier de Définitions des Ressources globales sélectionné ou bien éditer ou supprimer une ressource globale sélectionnée. Le fichier de Définitions des Ressources globales organise les ressources globales que vous ajoutez dans des groupes différents : fichiers, dossiers et bases de données (*voir capture d'écran ci-dessus*).

Pour ***ajouter une ressource globale***, cliquer sur le bouton **Ajouter** et définir la ressource globale dans le dialogue appropriée **Ressource globale** qui s'affiche (*voir les descriptions des fichiers, dossiers, et bases de données dans les sous-sections de cette section*). Après avoir défini une ressource globale et l'avoir enregistrée (en cliquant sur **OK** dans le dialogue Gérer les Ressources globales), la ressource globale est ajoutée à la bibliothèque des définitions globales dans le fichier des Définitions des Ressources globales. La ressource globale sera identifiée au moyen d'un alias.

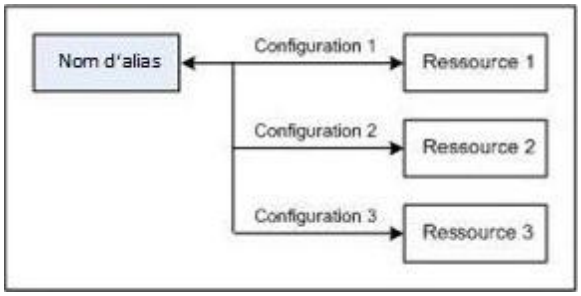
Pour ***éditer une ressource globale***, la sélectionner et cliquer sur **Éditer,** le dialogue **Ressource globale** pertinent s'affiche dans lequel vous pourrez procéder aux changements nécessaires (*voir les descriptions des fichiers, dossiers et bases de données dans les sous-sections de cette section*).

Pour ***supprimer une ressource globale***, la sélectionner et cliquer sur **Supprimer**.

Une fois que vous avez terminé l'ajout, l'édition et la suppression, n'oubliez pas de cliquer sur **OK** dans le dialogue de Gestion des Ressources globales et ***d'enregistrer vos modifications*** dans le fichier de Définition des Ressources globales.

Lier des ressources globales à des noms d'alias par le biais de configuration

La définition d'une ressource globale requiert le mappage d'un alias à une ressource (fichier, dossier ou base de données). Un seul nom d'alias peut être mappé à plusieurs ressources. Chaque mappage est appelé une configuration. Un seul nom d'alias peut donc être associé avec plusieurs ressources par le biais de configurations différentes (*capture d'écran ci-dessous*).



Dans une application Altova, vous pouvez ensuite attribuer des alias au lieu de fichiers. Pour chaque alias, vous pouvez passer entre les ressources mappées à cet alias en changeant tout simplement la configuration de la Ressource globale active de l'application (configuration active). Par exemple, dans l'application XMLSpy d'Altova, si vous souhaitez exécuter une transformation XSLT sur le document XML `MyXML.xml`, vous pouvez lui attribuer l'alias `MyXSLT` en tant que la ressource globale à utiliser pour les transformations XSLT. Dans XMLSpy, vous pouvez ensuite changer la configuration active pour utiliser plusieurs fichiers XSLT. Si `Configuration-1` mappe `First.xslt` à `MyXSLT` et que `Configuration-1` est sélectionné en tant que la configuration active, `First.xslt` sera utilisé pour la transformation. Cela vous permet d'utiliser des configurations multiples pour accéder à des ressources multiples par le biais d'un seul alias. Ce mécanisme peut être utile lors des tests et en ce qui concerne les ressources de comparaison. De plus, puisque les ressources globales peuvent être utilisées avec les produits Altova, les ressources peuvent aussi être testées et comparées sur plusieurs produits Altova.

8.1.1 Fichiers

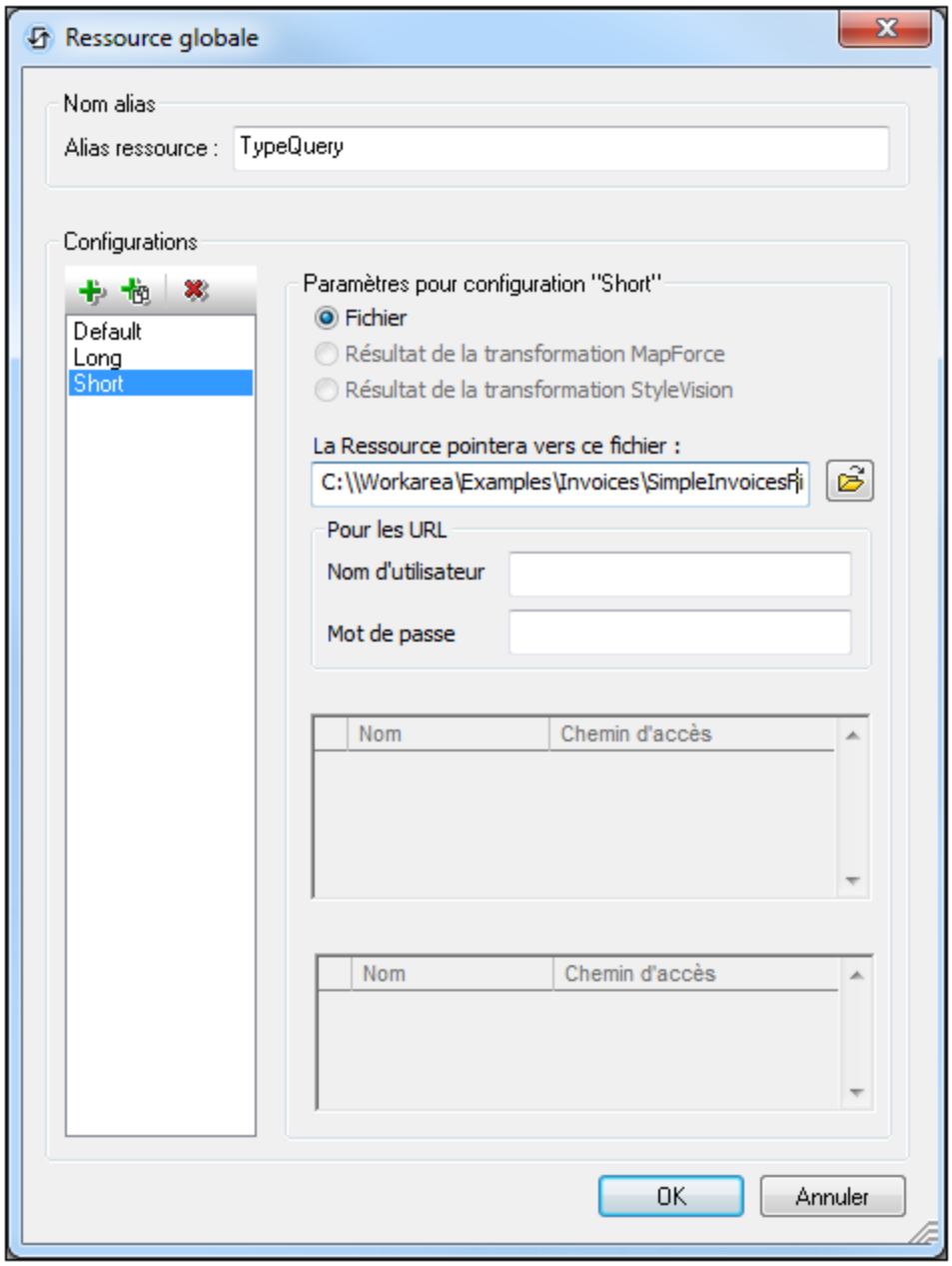
Le dialogue de Ressources globales pour Fichiers (*capture d'écran ci-dessous*) peut être accédé par le biais de la commande **Ajouter | Fichiers** dans le dialogue Gérer les Ressources globales. Dans ce dialogue, vous pouvez définir les configurations de l'alias nommé dans la fenêtre de texte *Alias de ressource*. Après avoir spécifié les propriétés des configurations tel qu'expliqué ci-dessous, enregistrer la définition de l'alias en cliquant sur **OK**.

Après avoir enregistré une définition d'alias, vous pouvez ajouter un autre alias en répétant les étapes indiquées ci-dessus (en commençant avec la commande **Ajouter | Fichiers** dans le dialogue Gérer les Ressources globales).

Dialogue Ressources globales

Un alias est défini dans le dialogue de Ressources globales (*capture d'écran ci-dessous*).

© 2016-2022 Altova GmbH



Icônes de dialogue des Ressources globales

Ajouter une configuration : ouvre le dialogue Ajouter configuration dans lequel vous pouvez saisir le nom de la configuration à ajouter.

Ajouter une configuration en tant que copie : ouvre le dialogue Ajouter configuration dans lequel vous pouvez saisir le nom de la configuration à créer en tant que copie de la configuration sélectionnée.

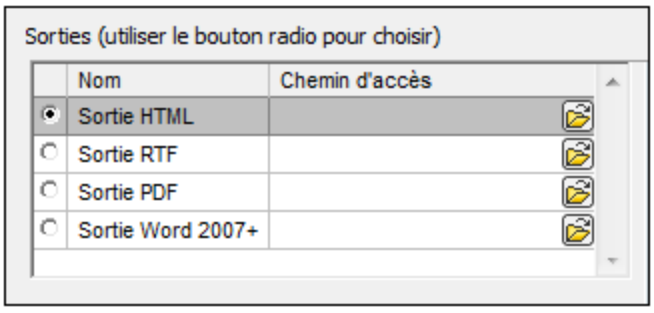
Supprimer : supprime la configuration sélectionnée.

Ouvrir : chercher le fichier à créer en tant que ressource globale.

Définir l'alias

Définir l'alias (son nom et ses configurations) comme suit :

1. *Nommer l'alias :* saisir le nom de l'alias dans le champ de saisie *Alias de ressource*.
2. *Ajouter les configurations :* le panneau Configurations contiendra par défaut une configuration nommée `Default` (*voir capture d'écran ci-dessus*), qui ne peut pas être supprimée ou renommée. Vous pouvez ajouter autant de configurations supplémentaires que vous le souhaitez en : (i) cliquant sur les icônes **Ajouter configuration** ou **Ajouter configuration en tant que copie**, et (ii) en donnant à la configuration un nom dans le dialogue qui apparaît. Chaque configuration ajoutée sera affichée dans la liste des Configurations. Dans la capture d'écran ci-dessus, deux configurations supplémentaires, nommées `Long` et `Short`, ont été ajoutées à la liste Configurations. La commande Ajouter Configuration en tant que copie vous permet de copier la configuration sélectionnée puis de la modifier.
3. *Sélectionner un type de ressource pour chaque configuration :* sélectionner une configuration depuis la liste Configurations, et, dans le panneau *Paramètres pour la configuration*, spécifier une ressource pour la configuration : (i) Fichier, (ii) Résultat d'une transformation Altova MapForce, ou (iii) Résultat d'une transformation Altova StyleVision. Sélectionner le bouton radio approprié. Si une option de transformation MapForce ou StyleVision est choisie, une transformation sera effectuée par MapForce ou StyleVision en utilisant, respectivement, le fichier `.mfd` ou `.sps` et le fichier d'entrée respectif. Le résultat de la transformation sera la ressource.
4. *Sélectionner un fichier pour le type de ressource :* si la ressource est un fichier directement sélectionné, chercher le fichier dans la fenêtre de texte *Sélection du fichier de ressource*. Si la ressource est le résultat d'une transformation, dans le champ de saisie *Sélection du fichier*, chercher le fichier `.mfd` (pour des transformations MapForce) ou `.sps` (pour les transformations StyleVision). Lorsque des entrées ou sorties multiples pour la transformation sont possibles avec MapForce, une sélection des options sera présentée. Par exemple, les options de sortie d'une transformation StyleVision sont affichées conformément à l'édition de StyleVision installée (*la capture d'écran ci-dessous montre les sorties pour l'Édition Enterprise*).



Choisir le bouton radio de l'option désirée (dans la capture d'écran ci-dessus, 'Sortie HTML' est sélectionnée). Si la ressource est le résultat d'une transformation, la sortie peut être enregistrée en tant qu'un fichier ou bien elle-même peut constituer une ressource globale. Cliquer sur l'icône et sélectionner, respectivement, Ressource globale (pour enregistrer la sortie en tant que ressource globale) ou Chercher (pour enregistrer la sortie en tant que fichier). Si aucune de ces options d'enregistrement n'est sélectionnée, le résultat de transformation sera chargé en tant que fichier temporaire lorsque la ressource globale sera invoquée.

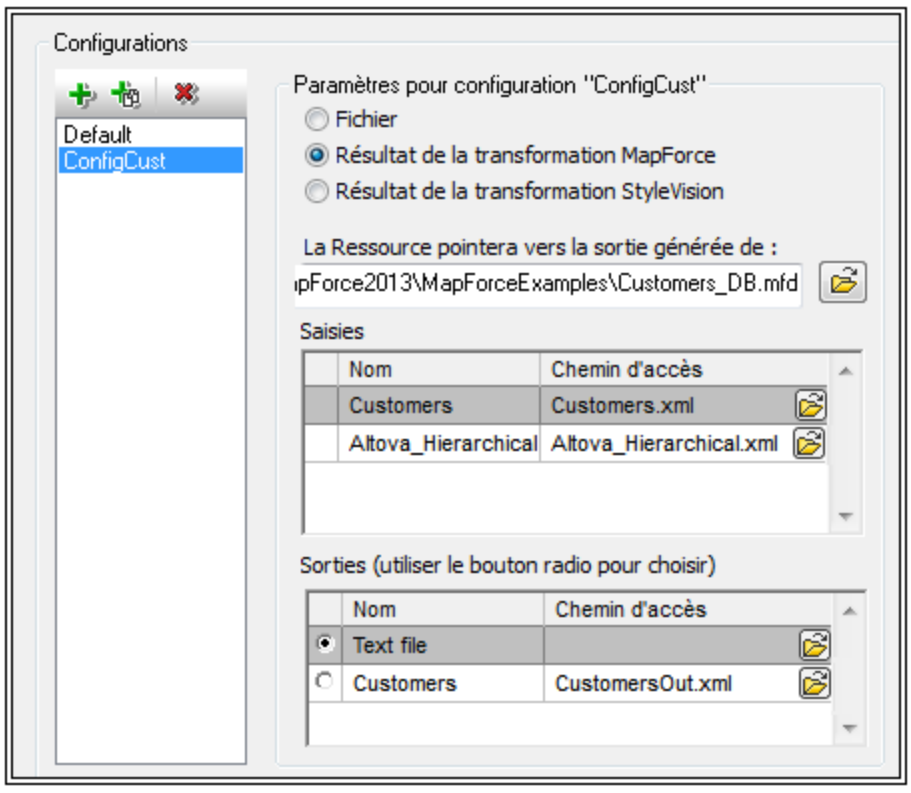
© 2016-2022 Altova GmbH

5. *Définir des configurations multiples le cas échéant :* vous pouvez ajouter plus de configurations et spécifier une ressource pour chacune d'entre elles. Pour ce faire, répéter les étapes 3 et 4 ci-dessus pour chaque configuration. Vous pouvez ajouter une nouvelle configuration à la définition d'alias à tout moment.
6. *Enregistrer la définition d'alias :* cliquer sur **OK** pour enregistrer l'alias et toutes ses configurations en tant que ressource globale. La ressource globale sera recensée sous Fichiers dans le dialogue Gérer les Ressources globales.

Résultat de la transformation MapForce

Altova MapForce mappe un ou plusieurs schémas de document d'entrée (existant) vers un ou plusieurs (nouveaux) schémas de document de sortie. Ce mappage, créé par un utilisateur de MapForce, est connu sous la désignation de MapForce Design (MFD). Les fichiers XML, fichiers de texte, bases de données, etc., qui correspondent au/x schéma/s d'entrée peuvent être utilisés en tant que sources de données. MapForce génère des fichiers de données de sortie qui correspondent au schéma de document de sortie. Ce document de sortie est le fichier *Résultat de la transformation MapForce* qui deviendra une ressource globale.

Si vous souhaitez configurer un fichier de données généré par MapForce en tant qu'une ressource globale, les points suivants doivent être spécifiés dans le dialogue de ressource globale (*voir capture d'écran ci-dessous*) :



- ***Un fichier `.mfd` (MapForce Design).*** Vous devez spécifier ce fichier dans la fenêtre *La ressource va pointer vers la sortie générée de* (*voir capture d'écran ci-dessus*).
- ***Un ou plusieurs fichiers de données d'entrée.*** Une fois le fichier MFD spécifié, il est analysé et, sur la base de l'information de schéma d'entrée qui y est contenue, le/ fichier/s de données par défaut

est/sont saisi/s dans le panneau *Entrées* (*voir capture d'écran ci-dessus*). Vous pouvez modifier la sélection du fichier par défaut pour chaque schéma d'entrée en spécifiant un autre fichier.

- ***Un fichier de sortie.*** Si le document MFD comporte plusieurs schémas de sortie, ils sont tous recensés dans le panneau *Sorties* (*voir capture d'écran ci-dessus*). Vous devez en choisir un. Si l'emplacement de fichier de sortie d'un schéma de sortie individuel est spécifié dans le document MFD, cet emplacement de fichier est saisi pour ce schéma de sortie dans le panneau *Sorties*. À partir de la capture d'écran ci-dessus, nous constatons que le document MFD spécifie que le schéma de sortie `Customers` comprend un fichier de données XML (`CustomersOut.xml`), alors que le schéma de sortie `Text file` n'a pas d'association de fichier dans le fichier MFD. Vous pouvez utiliser l'emplacement de fichier par défaut dans le panneau Sorties ou en spécifier un vous-même. Le résultat de la transformation MapForce sera enregistré dans l'emplacement de fichier du schéma de sortie sélectionné. Il s'agit du fichier qui sera utilisé en tant que ressource globale.

Note : L'avantage de cette option (Résultat de la transformation MapForce) est que la transformation est effectuée au moment où la ressource globale est invoquée. Cela signifie que la ressource globale contiendra les données les plus récentes (depuis le/s fichier/s d'entrée).

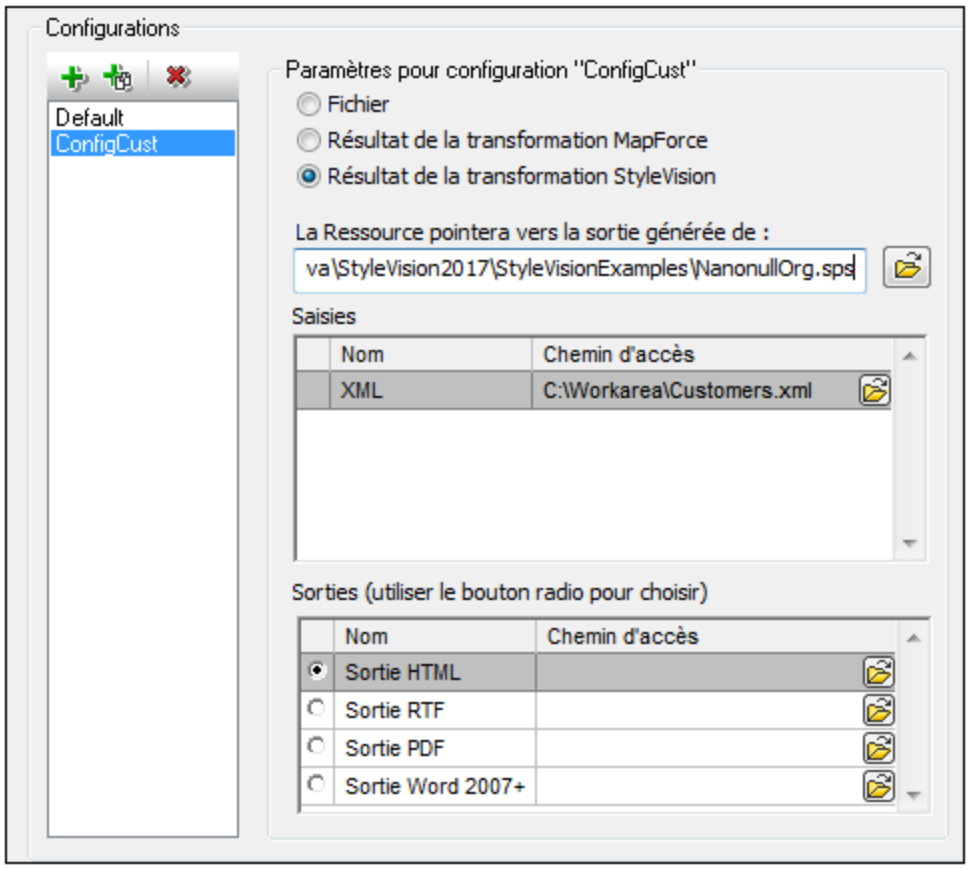
Note : Puisque MapForce est utilisé pour exécuter la transformation, vous devrez avoir préalablement installé Altova MapForce.

Résultat de la transformation StyleVision

Altova StyleVision est utilisé pour créer des fichiers StyleVision Power Stylesheet (SPS). Ces fichiers SPS génèrent des feuilles de style XSLT utilisées pour transformer des documents XML en documents de sortie dans des formats variés (HTML, PDF, RTF, Word 2007+, etc.). Si vous sélectionnez l'option *Résultat de la transformation StyleVision*, le document de sortie créé par StyleVision sera la ressource globale associée avec la configuration sélectionnée.

Pour l'option *Transformation StyleVision* dans le dialogue Ressource globale (*voir capture d'écran ci-dessous*), les fichiers suivants doivent être spécifiés.

© 2016-2022 Altova GmbH



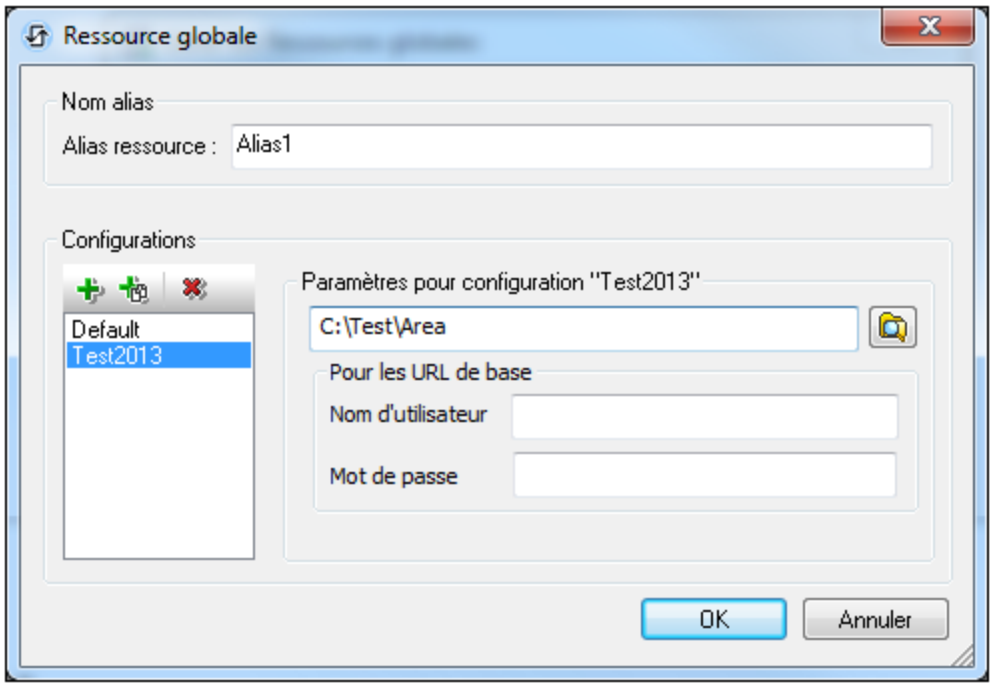
- ***Un fichier .sps (SPS).*** Vous devez spécifier ce fichier dans la fenêtre *La ressource va pointer vers la sortie générée de* (*voir capture d'écran ci-dessus*).
- ***Fichier/s d'entrée.*** Le fichier d'entrée est peut-être déjà spécifié dans le fichier SPS. Si c'est le cas, il apparaîtra automatiquement dans le panneau *Entrées* une fois que le fichier SPS a été sélectionné. Vous pouvez modifier cette entrée. S'il n'y a pas d'entrée, vous devrez en ajouter une.
- ***Fichier/s de sortie.*** Sélectionner le format de sortie dans le panneau *Sorties*, et spécifier un emplacement de fichier de sortie pour ce format.

Note : L'avantage de cette option (Résultat de la transformation StyleVision) est que la transformation sera effectuée lors de l'invocation de la ressource globale. Cela signifie que la ressource globale contiendra les données les plus récentes (depuis le/s fichier/s d'entrée).

Note : Puisque StyleVision est utilisé pour exécuter la transformation, vous devrez avoir préalablement installé Altova StyleVision pour que cette fonction puisse marcher.

8.1.2 Dossiers

Dans le dialogue de Ressources globales pour Dossiers (*capture d'écran ci-dessous*), vous pouvez ajouter une ressource de dossier tel que décrit ci-dessous.



Icônes de dialogue des Ressources globales

Ajouter une configuration : ouvre le dialogue Ajouter configuration dans lequel vous pouvez saisir le nom de la configuration à ajouter.

Ajouter une configuration en tant que copie : ouvre le dialogue Ajouter configuration dans lequel vous pouvez saisir le nom de la configuration à créer en tant que copie de la configuration sélectionnée.

Supprimer : supprime la configuration sélectionnée.

Ouvrir : chercher le fichier à créer en tant que ressource globale.

Définir l'alias

Définir l'alias (son nom et ses configurations) comme suit :

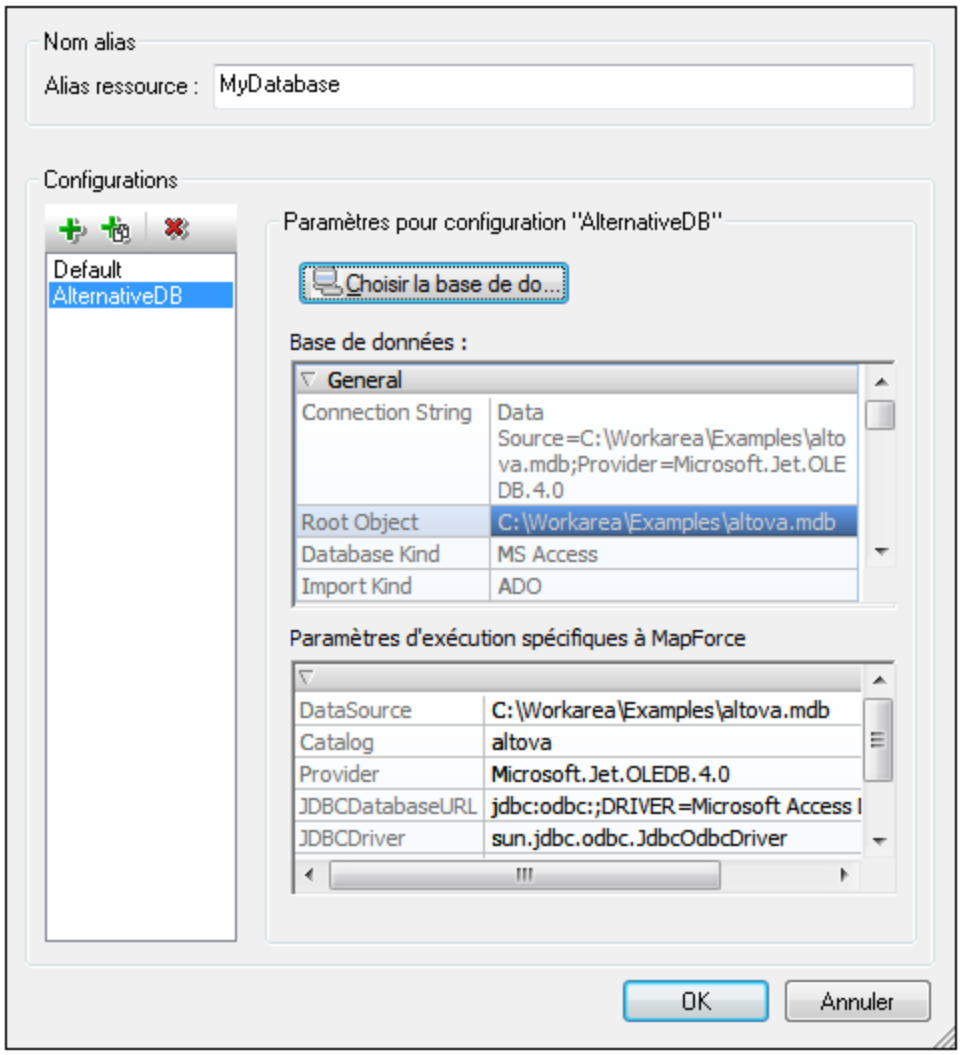
1. *Nommer l'alias :* saisir le nom de l'alias dans la fenêtre de texte *Alias de ressource*.
2. *Ajouter les configurations :* le panneau Configurations contiendra une configuration nommée `Default` (*voir capture d'écran ci-dessus*), qui ne peut pas être supprimée ou renommée. Vous pouvez ajouter autant de configurations supplémentaires pour l'alias sélectionné que vous le souhaitez. Ajouter une configuration en cliquant sur l'icône **Ajouter configuration** ou **Ajouter configuration en tant que copie**. Dans le dialogue qui s'ouvre, saisir le nom de configuration. Cliquer sur **OK**. La nouvelle configuration sera recensée dans le panneau Configurations. Répéter l'opération pour le nombre de configurations que vous souhaitez.
3. *Sélectionner un dossier en tant que ressource d'une configuration :* sélectionner une des configurations dans le panneau Configurations et cherchez le dossier que vous souhaitez créer en tant que ressource globale.

© 2016-2022 Altova GmbH

4. *Définir des configurations multiples le cas échéant :* spécifier une ressource de dossier pour chaque configuration que vous avez créée (répéter l'étape 3 ci-dessus pour chaque configuration que vous avez créée). Vous pouvez ajouter une nouvelle configuration à la définition d'alias à tout moment.
5. *Enregistrer la définition d'alias :* cliquer sur **OK** dans le dialogue de Ressources globales pour enregistrer l'alias et toutes ses configurations en tant que ressource globale. La ressource globale sera recensée sous Dossiers dans le dialogue Gérer les Ressources globales.

8.1.3 Bases de données

Dans le dialogue des Ressources globales pour les bases de données (*voir la capture d'écran ci-dessous*), vous pouvez ajouter une ressource de base de données comme suit :



Icônes du dialogue Ressource globale

Ajouter une configuration : Affiche la boîte de dialogue Ajouter une configuration dans laquelle vous saisissez le nom de la configuration à ajouter.

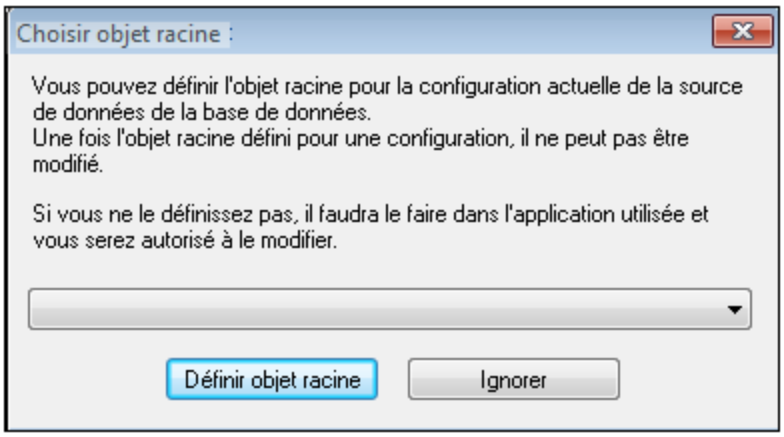
Ajouter une configuration en tant que copie : Affiche la boîte de dialogue Ajouter une configuration dans laquelle vous pouvez saisir le nom de la configuration à créer en tant que une copie de la configuration choisie.

Supprimer : Supprime la configuration sélectionnée.

Définir l'alias

Définir l'alias (son nom et ses configurations) comme suit :

1. *Donner un nom à l'alias :* Saisir le nom de l'alias dans la zone de texte *Alias de ressource*.
2. *Ajouter des configurations :* Le volet de configurations aura une configuration appelée Par défaut (*voir la capture d'écran ci-dessus*). Cette configuration par défaut ne peut pas être supprimée ni changer de nom. Vous pouvez saisir autant de configurations supplémentaires pour l'alias sélectionné que vous voulez. Ajouter une configuration en cliquant sur les icônes **Ajouter Configuration** ou **Ajouter une configuration en tant que copie**. Dans le dialogue qui apparaît, saisissez le nom de configuration. Cliquez sur **OK**. La nouvelle configuration sera recensée dans le volet de configurations. Répéter pour autant de configurations que vous voulez.
3. *Lancer la sélection d'une base de données comme ressource d'une configuration :* Sélectionnez une des configurations dans le volet de configurations et cliquez sur l'icône **Choisir base de données**. Ceci fait apparaître le dialogue de connexion des Ressources globales.
4. *Connexion à la base de données :* Sélectionnez si vous voulez créer une connexion à la base de données en utilisant l'Assistant de connexion, une connexion existante, des connexions ADO, ODBC ou JDBC.
5. *Sélectionner l'objet racine :* Si vous vous connectez à un serveur de base de données où un objet racine peut être sélectionné, vous serez invité dans le dialogue Choisir Objet racine (*capture d'écran ci-dessous*) à choisir un objet racine sur le serveur. Sélectionnez l'objet racine et cliquez sur **Définir l'objet racine**. L'objet racine que vous sélectionnez sera l'objet racine qui est chargé lorsque cette configuration est utilisée.



Si vous décidez de ne pas sélectionner un objet racine (en cliquant sur le bouton **Skip**), alors vous pouvez sélectionner l'objet racine au moment du chargement de la ressource globale.

6. *Définissez de multiples configurations, le cas échéant :* Spécifiez une ressource de base de données pour toute autre configuration que vous avez créé (à savoir, répétez les étapes 3 à 5 ci-dessus pour les

© 2016-2022 Altova GmbH

diverses configurations que vous avez créées). Vous pouvez ajouter une nouvelle configuration à la définition de l'alias à tout moment.

7. *Enregistrer la définition de l'alias :* Cliquez sur **OK** dans le dialogue Ressources globales pour enregistrer l'alias et toutes ses configurations comme ressource globale. La ressource globale sera recensée sous la base de données Gérer les Ressources globales.

8.2 Utiliser les Ressources globales

Il existe plusieurs types de ressources globales (type fichier, type dossier, et type de base de données). Certains scénarios dans lesquels vous pouvez utiliser des ressources globales dans Authentic Desktop sont recensés ici : Fichiers et Dossiers .

Sélections qui déterminent quelle ressource est utilisée

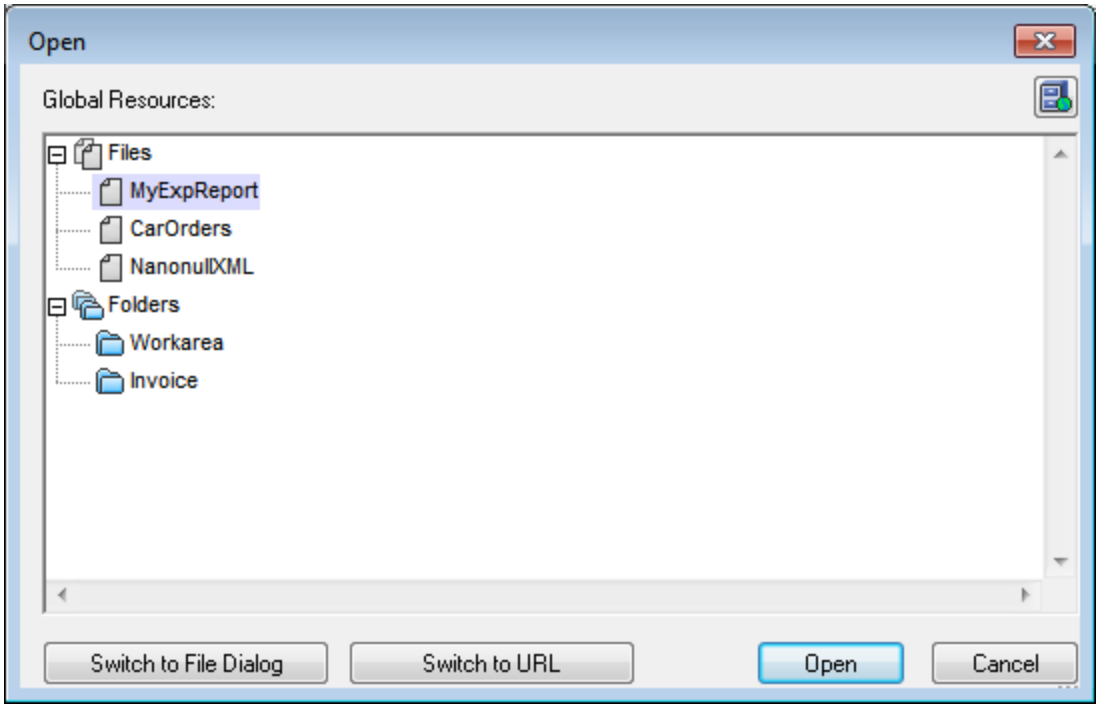
Il y a deux sélections au sein des applications qui déterminent quelles ressources globales peuvent être utilisées et quelles ressources globales peuvent être réellement utilisées à tout moment donné :

- *Le Fichier XML de Ressources globales actif* est sélectionné dans le dialogue Ressources globales. Les définitions de ressources globales qui sont présentes dans le Fichier XML de ressources globales sont disponibles dans tous les fichiers ouverts dans l'application. Seules les définitions dans le Fichier XML de Ressources globales actif sont disponibles. Le Fichier XML de Ressources globales actif peut être changé à tout moment et les définitions de ressources globales contenues dans le nouveau fichier actif remplaceront immédiatement les fichiers actifs précédents. Le Fichier XML de Ressources globales actif détermine donc : (i) quelles ressources globales peuvent être attribuées et (ii) quelles ressources globales sont disponibles pour une consultation (par exemple, si une ressource globale dans un Fichier de Ressources globales est attribuée mais qu'il n'y a pas de ressource globale de ce nom dans le Fichier XML de Ressources globales actif actuellement, alors la ressource globale attribuée (alias) ne pourra pas être consultée).
- *La configuration active est* sélectionnée via l'élément de menu **Outils | Configuration active** ou via la barre outils Ressources globales. Cliquer sur cette commande (ou liste déroulante dans la barre outils) pour ouvrir une liste de configurations à travers tous les alias. La sélection d'une configuration activera cette configuration dans toutes les applications. Cela signifie que quel que soit l'endroit où une ressource globale (ou alias) est utilisée, la ressource correspondant à la configuration active de chaque alias utilisé sera chargée. La configuration active est appliquée à tous les alias utilisés. Si un alias n'a pas de configuration avec le nom de la configuration active, alors la configuration par défaut de cet alias sera utilisée. La configuration active n'est pas pertinente lors de l'attribution de ressources ; elle n'est uniquement déterminante que lorsque les ressources sont réellement utilisées.

8.2.1 Attribuer des Fichiers et des Dossiers

Les ressources de type Fichier et de type Dossier sont attribuées différemment. Dans l'un des scénarios d'utilisation ci-dessous, cliquer sur le bouton **Passer aux ressources globales** pour afficher le dialogue Ouvrir la Ressource globale (*capture d'écran ci-dessous*).

© 2016-2022 Altova GmbH



Gérer les Ressources globales : affiche le dialogue Gérer les Ressources globales.

La sélection d'une *ressource globale de type fichier* permet d'attribuer le fichier. La sélection d'une *ressource globale de type dossier* entraîne l'ouverture du dialogue Ouvrir dans lequel vous pouvez chercher le fichier nécessaire. Le chemin d'accès du fichier sélectionné est saisi de manière relative à la ressource de dossier. Donc si une ressource globale de type dossier doit comporter deux configurations, chacune pointant vers des dossiers différents, les fichiers possédant le même nom mais placés dans des dossiers différents peuvent être ciblés par les deux configurations. Cela peut être utile pour les tests.

Vous pouvez passer au dialogue de fichier ou au dialogue URL en cliquant sur le bouton respectif en bas du dialogue. L'icône **Gérer les Ressources globales** située dans le coin en haut à droite affiche le dialogue Gérer les Ressources globales.

Scénarios d'utilisation

Les ressources de type Fichier et de type Dossier peuvent être utilisées dans les scénarios suivants :

- Ouvrir les ressources globales
- Enregistrer en tant que ressource globale
- Transformations XSLT

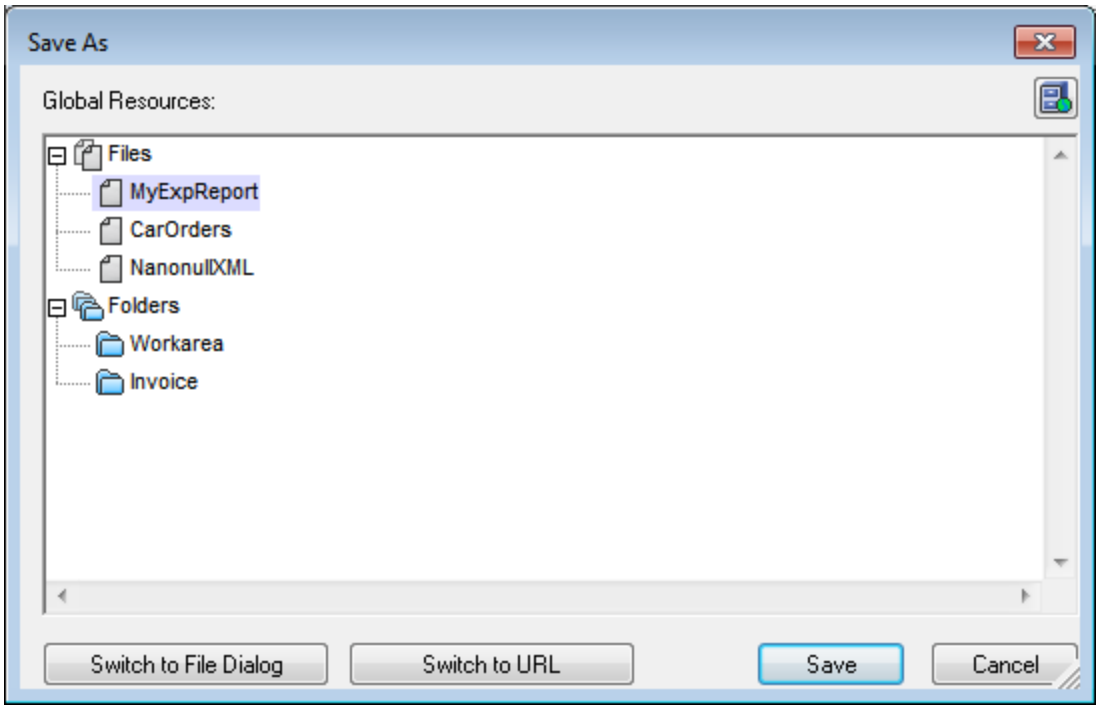
Ouvrir les ressources globales

Une ressource globale peut être ouverte dans Authentic Desktop avec la commande **Fichier | Ouvrir (Passer à la Ressource globale)** et peut être éditée. Dans le cas d'une ressource globale de type fichier, le fichier est ouvert directement. Dans le cas d'une ressource globale de type dossier, un dialogue Ouvrir s'ouvre avec le

dossier associé sélectionné. Vous pouvez ensuite chercher le fichier nécessaire dans les dossiers descendants. Un des avantages lors de l'adressage des fichiers pour les éditer par le biais de ressources globales, est que les fichiers associés peuvent être enregistrés sous les différentes configurations d'une seule ressource globale et qu'ils peuvent être accédés simplement en changeant les configurations. Toutes les modifications d'édition devraient être enregistrées avant de modifier la configuration.

Enregistrer en tant que ressource globale

Un nouveau fichier peut être enregistré en tant que ressource globale. De même, un fichier déjà existant peut être ouvert puis enregistré en tant que ressource globale. Cliquer sur les commandes **Fichier | Enregistrer** ou **Fichier | Enregistrer sous**, pour faire apparaître le dialogue Enregistrer. Cliquer sur le bouton **Passer à la Ressource globale** pour accéder aux ressources globales disponibles (*capture d'écran ci-dessous*), qui sont les alias définis dans le Fichier XML de Ressources globales actuel.



Choisir un alias et cliquer sur **Enregistrer**. Si l'alias est un alias de fichier, le fichier sera enregistré directement. S'il s'agit d'un alias de dossier, un dialogue apparaîtra qui vous invitera à saisir le nom du fichier sous lequel vous souhaitez l'enregistrer. Dans tous les cas, le fichier sera sauvegardé à l'emplacement défini pour la configuration active actuellement.

Note : Chaque configuration pointe vers un emplacement de fichier spécifique, précisé dans la définition de cette configuration. Si le fichier que vous enregistrez en tant que ressource globale ne possède pas le même type d'extension que le fichier à l'emplacement actuel de la configuration, il pourrait se produire des erreurs d'édition et de validation lorsque cette ressource globale est ouverte dans Authentic Desktop. Cela est dû au fait que Authentic Desktop ouvrira le fichier en présumant que le fichier présente le type de fichier spécifié dans la définition de la configuration.

© 2016-2022 Altova GmbH

Transformation XSLT

Cliquer sur la commande **XSL/XQuery | Transformation XSL** ou **XSL/XQuery | Transformation XSL:FO** pour ouvrir un dialogue dans lequel vous pouvez chercher le fichier XSLT ou XML requis. Cliquer sur le bouton **Parcourir** puis sur le bouton **Passer à la Ressource globale** pour ouvrir le dialogue Ouvrir la Ressource globale (*capture d'écran en haut de la section*). Le fichier associé avec la configuration actuellement active de la ressource globale sélectionnée est utilisé pour la transformation.

8.2.2 Changer la Configuration active

Une configuration d'une ressource globale peut être active à tout moment. Cette configuration est appelée la configuration active et elle est active pour toutes les applications. Cela signifie que la configuration active est active pour tous les alias de ressources globales dans tous les fichiers ouverts actuellement et les connexions de source de données. Si un alias n'a pas de configuration avec le nom de la configuration active, la configuration par défaut de cet alias sera utilisée. Prenons en guise d'exemple pour le changement des configurations, le cas suivant : un fichier a été attribué par le biais d'une ressource globale avec plusieurs configurations. Chaque configuration mappe vers un fichier différent. Donc le fichier sélectionné dépend de la configuration sélectionnée en tant que la configuration active de l'application.

Il existe deux manières de passer la configuration active :

- Par la commande de menu **Outils | Configuration active**. Sélectionnez la configuration depuis le sous-menu de la commande.
- Dans la liste de choix de la barre outils de la Ressource globale (*capture d'écran ci-dessous*), sélectionnez la configuration requise.

Ainsi, en changeant la configuration active, vous pouvez changer des fichiers sources qui sont attribués par le biais d'une ressource globale.

9 Contrôle de code source

La prise en charge du contrôle de code source dans Authentic Desktop est disponible avec le Microsoft Source Control Plug-in API (anciennement appelé MSSCCI API), versions 1.1, 1.2 et 1.3. Cela vous permet d'exécuter les commandes de contrôle de source comme par exemple "Archiver" ou "Extraire" directement depuis Authentic Desktop vers n'importe quel système de contrôle de source qui permet à des clients natifs ou tiers de s'y connecter par le biais de la Microsoft Source Control Plug-in API.

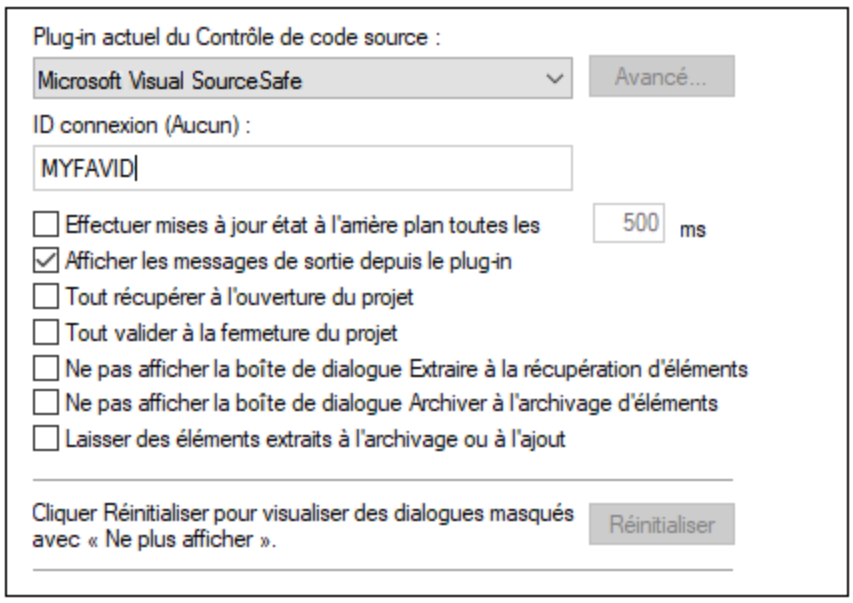
Vous pouvez utiliser, en guise de votre fournisseur de contrôle de source, tout plug-in commercial ou non commercial qui prend en charge la Microsoft Source Control Plug-in API, et qui peut se connecter à un système de contrôle de version compatible. En ce qui concerne la liste des systèmes de contrôle de source et des plug-ins testés par Altova, voir Systèmes de contrôle de la source pris en charge.

Installer et configurer le fournisseur de contrôle de source

Pour visionner les fournisseurs de contrôle de source disponibles sur votre système, procéder comme suit :

1. Dans le menu **Outils**, cliquer sur **Options**.
2. Cliquer dans l'onglet **Contrôle de code source**.

Tout plug-in de contrôle de source compatible avec la Microsoft Source Code Control Plug-in API est affiché dans la liste déroulante **Plug-in actuel du contrôle de code source**.



Si un plug-in compatible ne peut pas être trouvé sur votre système, le message suivant s'affichera :

```
"L'inscription des fournisseurs de contrôle de code source installés n'a pas été trouvée ou est incomplète."
```

Certains systèmes de contrôle de source peuvent ne pas installer automatiquement le plug-in de contrôle de source. Dans ce cas, vous devrez l'installer séparément. Pour plus d'instructions, veuillez vous référer à la documentation du système de contrôle de source respectif. Un (fournisseur) de plug-in compatible avec la

© 2016-2022 Altova GmbH

Microsoft Source Code Control Plug-in API devra être enregistré sous l'entrée de registre suivante sur votre système d'exploitation :

```
HKEY_LOCAL_MACHINE\SOFTWARE\SourceCodeControlProvider\InstalledSCCProviders
```

Une fois correctement installé, le plug-in devient automatiquement disponible dans la liste des plug-ins disponibles pour Authentic Desktop.

Accéder aux commandes de contrôle de source

Les commandes reliées au contrôle de source sont disponibles dans le menu **Projet | Contrôle de code source**.

Ressource / Problèmes de vitesse

Des bases de données de contrôle de source très importantes peuvent introduire une pénalité de vitesse/ressource lors de l'exécution automatique des mises à jour en arrière-plan.

Vous pourrez accélérer votre système en désactivant (ou en augmentant l'intervalle de) l'option **Effectuer des mises à jour d'état en arrière-plan toutes les ... secondes** dans l'onglet **Contrôle de source** accessible depuis **Outils | Options**.

Note : La version **64-bit** de votre application Altova prend automatiquement en charge n'importe lequel des programmes de contrôle de source 32-bit contenu dans cette documentation. Si vous utilisez une application Altova 64-bit avec un programme de contrôle de source 32-bit, l'option **des mises à jour d'état en arrière-plan toutes les ... secondes** est automatiquement grisée et ne peut pas être sélectionnée.

Différencier avec Altova DiffDog

Vous pouvez configurer de nombreux systèmes de contrôle de source (y compris Git et TortoiseSVN) pour qu'ils puissent utiliser Altova DiffDog en tant que leur outils de différenciation. Pour plus d'information concernant DiffDog, voir https://www.altova.com/diffdog.html. Pour consulter une documentation DiffDog, voir https://www.altova.com/documentation.html.

9.1 Configurer le contrôle de source

Le mécanisme permettant de configurer le contrôle de source et de placer les fichiers dans un projet Authentic Desktop se trouvant sous contrôle de source est le suivant :

1. Si cela n'a pas déjà été fait, installer le système de contrôle de source (voir Systèmes de contrôle de source pris en charge) et configurer la base de contrôle de source (archivage) dans lequel vous souhaitez enregistrer votre travail.
2. Créer un dossier de poste de travail local qui contiendra les fichiers de travail que vous souhaitez placer sous contrôle de source. Le dossier qui contient tous vos dossiers et fichiers de poste de travail est appelé le dossier local et le chemin menant au dossier local est appelé le chemin local. Ce dossier local sera lié à un dossier particulier dans l'archivage.
3. Dans votre application Altova, créer un dossier de projet d'application auquel vous devez ajouter les fichiers que vous souhaitez placer sous contrôle de source. Cette organisation de fichiers dans un projet d'application est abstrait. Les fichiers d'un projet référencent les fichiers physiques enregistrés localement, de préférence dans un dossier (avec des sous-dossiers si requis) pour chaque projet.
4. Dans la base de données du contrôle de source du système (aussi appelé contrôle de source ou archivage), un dossier est créé qui est lié au dossier local. Ce dossier (appelé le dossier lié) copiera la structure du dossier local afin que tous les fichiers devant être placés sous contrôle de source se situent correctement en terme d'hiérarchie dans le dossier lié. Le dossier lié est généralement créé lorsque vous ajoutez un fichier ou un projet d'application au contrôle de source pour la première fois. Voir la section, Projet d'application, pour plus d'informations concernant la structure du dossier de l'archivage.
5. Les fichiers de projet sont ajoutés au contrôle de source avec la commande **Projet | Contrôle de source | Ajouter au Contrôle de source**. Lorsque vous ajoutez un projet ou un fichier dans un projet pour la première fois au contrôle de source, les liaisons et la structure de dossier correcte sera créée dans l'archivage.
6. Les actions de contrôle de source, comme l'archivage et l'extraction de fichiers, et la suppression de fichiers depuis le contrôle de source peuvent être effectués depuis la commande dans le sous-menu **Projet | Contrôle de source**. Ces commandes sont décrites dans la section du menu Projet de la Référence de l'utilisateur.

Note : Si vous souhaitez changer le fournisseur de contrôle de source actuel, vous pouvez vous appuyer sur deux méthodes : (i) par le biais des options Contrôle de source (**Outils | Options | Contrôle de source**), ou (ii) dans le dialogue Changer le Contrôle de source (**Projet | Contrôle de source | Changer le Contrôle de source**).

© 2016-2022 Altova GmbH

9.2 Systèmes de Contrôle de source pris en charge

La liste ci-dessous montre les Source Control Servers (SCSs) pris en charge par Authentic Desktop, avec leurs Clients de Contrôle de source (Source Control Clients -SCC) respectifs. La liste est organisée alphabétiquement par SCS. Veuillez noter les éléments suivants :

- Altova a mis en place la Microsoft Source Control Plug-in API (versions 1.1, 1.2 et 1.3) dans Authentic Desktop, et a testé la prise en charge des pilotes recensés et des systèmes de contrôle de révision. Il est prévu que Authentic Desktop poursuivra la prise en charge de ces produits, si et quand ils sont mis à jours.
- Les clients du Contrôle de code source qui ne sont pas recensés ci-dessous, mais qui mettent en place la Microsoft Source Control Plug-in API devraient aussi fonctionner avec Authentic Desktop.

Système de contrôle de source	Clients de Contrôle de source
AccuRev 4.7.0 Windows	AccuBridge pour Microsoft SCC 2008.2
Bazaar 1.9 Windows	Aigenta Unified SCC 1.0.6
Borland StarTeam 2008	Borland StarTeam Cross-Platform Client 2008 R2
Codice Software Plastic SCM Professional 2.7.127.10 (Server)	Codice Software Plastic SCM Professional 2.7.127.10 (SCC Plugin)
Collabnet Subversion 1.5.4	• Aigenta Unified SCC 1.0.6 • PushOK SVN SCC 1.5.1.1 • PushOK SVN SCC x64 version 1.6.3.1 • TamTam SVN SCC 1.2.24
ComponentSoftware CS-RCS (PRO) 5.1	ComponentSoftware CS-RCS (PRO) 5.1
Dynamsoft SourceAnywhere pour VSS 5.3.2 Standard/Professional Server	Dynamsoft SourceAnywhere for VSS 5.3.2 Client
Dynamsoft SourceAnywhere Hosted	Dynamsoft SourceAnywhere Hosted Client (22252)
Dynamsoft SourceAnywhere Standalone 2.2 Server	Dynamsoft SourceAnywhere Standalone 2.2 Client
Git	Plug-in PushOK GIT SCC (voir Source Control with Git)
IBM Rational ClearCase 7.0.1 (LT)	IBM Rational ClearCase 7.0.1 (LT)
March-Hare CVSNT 2.5 (2.5.03.2382)	Aigenta Unified SCC 1.0.6
March-Hare CVS Suite 2008	• Jalindi Igloo 1.0.3 • March-Hare CVS Suite Client 2008 (3321) • PushOK CVS SCC NT 2.1.2.5 • PushOK CVS SCC x64 version 2.2.0.4 • TamTam CVS SCC 1.2.40
Mercurial 1.0.2 pour Windows	Sergey Antonov HgSCC 1.0.1

Système de contrôle de source	Clients de Contrôle de source
Microsoft SourceSafe 2005 avec CTP	Microsoft SourceSafe 2005 avec CTP
Microsoft Visual Studio Team System 2008/2010 Team Foundation Server	Microsoft Team Foundation Server 2008/2010 MSSCCI Provider
Perforce 2008 P4S 2008.1	Perforce P4V 2008.1
PureCM Server 2008/3a	PureCM Client 2008/3a
QSC Team Coherence Server 7.2.1.35	QSC Team Coherence Client 7.2.1.35
Reliable Software Code Co-Op 5.1a	Reliable Software Code Co-Op 5.1a
Seapine Surround SCM Client/Server pour Windows 2009.0.0	Seapine Surround SCM Client 2009.0.0
Serena Dimensions Express/CM 10.1.3 pour Win32 Server	Serena Dimensions 10.1.3 pour Win32 Client
Softimage Alienbrain Server 8.1.0.7300	Softimage Alienbrain Essentials/Advanced Client 8.1.0.7300
SourceGear Fortress 1.1.4 Server	SourceGear Fortress 1.1.4 Client
SourceGear SourceOffsite Server 4.2.0	SourceGear SourceOffsite Client 4.2.0 (Windows)
SourceGear Vault 4.1.4 Server	SourceGear Vault 4.1.4 Client
VisualSVN Server 1.6	• Aigenta Unified SCC 1.0.6 • PushOK SVN SCC 1.5.1.1 • PushOK SVN SCC x64 version 1.6.3.1 • TamTam SVN SCC 1.2.24

© 2016-2022 Altova GmbH

9.3 Dossier Poste de travail local

Les fichiers avec lesquels vous allez travailler doivent être enregistrés dans une hiérarchie à l'intérieur d'un dossier de poste de travail local (*voir diagramme ci-dessous*).

```
Local Workspace Folder
|
|-- MyProject.spp
|-- QuickStart
|   |-- QuickStart.css
|   |-- QuickStart.xml
|   |-- QuickStart.xsd
|-- Grouping
|   |-- Persons
|   |   |-- Persons.xml
```

Le fichier de projet d'application (fichier `.spp`) sera généralement situé directement dans le dossier du poste de travail local (*voir diagramme ci-dessus*).

Si un ou plusieurs fichiers de ce dossier (de poste de travail) sont placés sous contrôle de source, la structure du dossier du poste de travail local sera partiellement ou entièrement reproduite dans l'archivage. Par exemple, si le fichier `Persons.xml` provenant du dossier local affiché ci-dessus est placé sour contrôle de source, le chemin y menant depuis l'archivage sera :

```
[RepositoryFolder]/MyProject/Grouping/Persons/Persons.xml
```

Le dossier `MyProject` dans le dossier d'archivage est lié au dossier local. Généralement, il portera le nom du projet, mais vous pouvez lui donner n'importe quel nom.

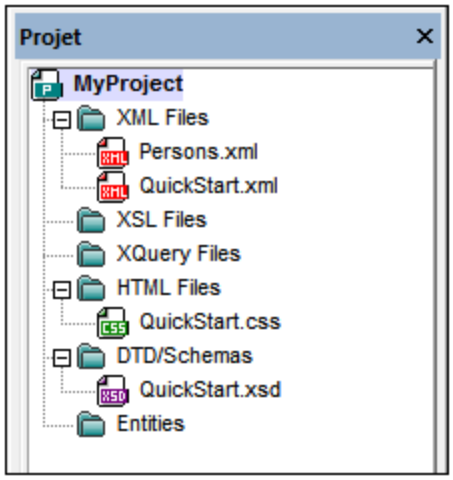
Si tout le projet d'application est placé sous le contrôle de source (en choisissant le nom du projet dans la fenêtre Projets et en le plaçant sous contrôle de source), alors l'ensemble de la structure du dossier local est recréée dans l'archivage.

Note : Les fichiers provenant de l'extérieur du dossier du poste de travail local peuvent être ajoutés au projet d'application. Mais le fait de pouvoir placer ce type de fichier sous contrôle de source dépend du système de contrôle de source que vous utilisez. Certains systèmes de contrôle de source peuvent avoir des difficultés à placer un fichier depuis l'extérieur du dossier local dans l'archivage. C'est pourquoi nous recommandons que tous les fichiers de projet que vous souhaitez placer sous contrôle de source soient placés dans le dossier du poste de travail local.

9.4 Projet d'application

Créer ou charger le projet d'application Altova que vous souhaitez placer sous contrôle de source. Si vous souhaitez placer un seul fichier sous contrôle de source, ce fichier devra être intégré dans un projet, étant donné qu'un contrôle de source peut uniquement être accédé par un projet.

Par exemple, veuillez considérer un projet dans l'application XMLSpy d'Altova. Les propriétés de projet sont enregistrées dans un fichier `.spp`. Dans l'application, le projet est affiché dans la fenêtre Projet de l'application (*voir capture d'écran ci-dessous*). Le projet dans la capture d'écran ci-dessous est nommé `MyProject` et les propriétés du projet sont sauvegardées dans le fichier `MyProject.spp`.



Vous pouvez placer tout le projet (tous les fichiers dans le projet) ou bien seulement certains fichiers de projet sous contrôle de source. **Seuls des fichiers se trouvant dans le projet peuvent être placés sous contrôle de source.** Donc si vous devez ajouter des fichiers au projet avant de pouvoir les placer sous contrôle de source. Le fichier de projet (fichier `.spp`) sera placé automatiquement sous contrôle de source dès qu'un fichier dans le projet est placé sous contrôle de source.

L'ensemble du projet, ou bien un ou plusieurs fichiers du projet, est placé sous contrôle de source par le biais de la commande **Projet | Contrôle de source | Ajouter le Contrôle de source** (*voir section suivante*).

Veuillez noter, néanmoins, que la structure du dossier de l'archivage ne correspond pas à la structure de dossier du projet (*capture d'écran ci-dessus*) mais à la structure du dossier de poste de travail local (*voir diagramme de dossier ci-dessous*). Dans le diagramme ci-dessous, veuillez noter que le dossier `MyProject` dans l'archivage a une structure de dossier correspondant à celle du dossier de poste de travail local. Veuillez noter que le dossier lié se produit dans le dossier d'archivage.

```
Local Workspace Folder          Archivage
|                               |
|-- MyProject.spp               |-- MyProject (lié au Poste de travail local)
|-- QuickStart                  ||-- MyProject.spp
|   |-- QuickStart.css          ||-- QuickStart
|   |-- QuickStart.xml          ||   |-- QuickStart.css
|   |-- QuickStart.xsd          ||   |-- QuickStart.xml
```


© 2016-2022 Altova GmbH

```
|-- Grouping                    ||   |-- QuickStart.xsd
|   |-- Persons                 ||-- Grouping
|   |   |-- Persons.xml         ||   |-- Persons
                                ||   |   |-- Persons.xml
```

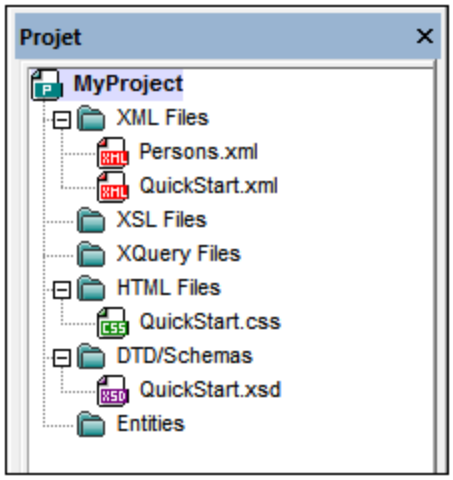
Note : un projet d'application peut contenir des dossiers de projet (vert) et des dossiers externes (jaune). Seuls les fichiers contenus dans les dossiers de projet (vert) peuvent être placés sous contrôle de source. Les fichiers contenus dans les dossiers externes (jaune) ne peuvent pas être placés sous contrôle de source.

Note : les fichiers provenant d'en dehors du dossier de poste de travail local peuvent être ajoutés au projet d'application. Mais que vous placiez ce type de fichier sous contrôle de source dépend du système de contrôle de source que vous utilisez. Certains systèmes de contrôle de source peuvent avoir des difficultés à placer un fichier dans l'archivage depuis l'extérieur du dossier local. C'est pourquoi nous recommandons que tous les fichiers de projet que vous souhaitez placer sous contrôle de source soient situés dans le dossier de poste de travail local.

9.5 Ajouter au contrôle de source

L'ajout du projet au contrôle de source créera automatiquement les liaisons correctes et la structure d'archivage avant d'ajouter le fichier de projet (fichier `.spp`) ou les fichiers individuels au contrôle de source. Ajouter le projet au contrôle de source comme suit.

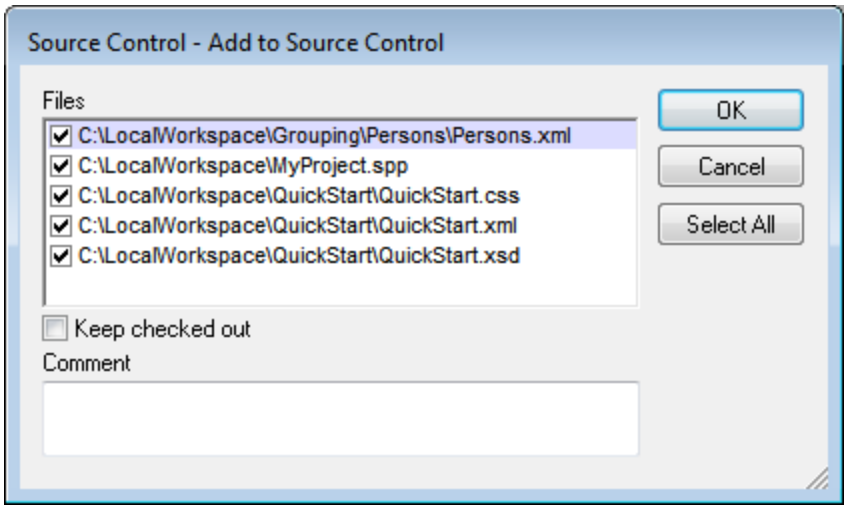
Choisir le projet dans la fenêtre Projet (`MyProject` dans la capture d'écran ci-dessous) de manière à ce qu'il soit marqué (*comme dans la capture d'écran ci-dessous*). En alternative, choisir un seul fichier ou sélectionner plusieurs fichiers en cliquant dessus tout en maintenant la touche **Ctrl** appuyée. L'ajout d'un seul fichier au contrôle de source ajoutera aussi automatiquement le fichier de projet (fichier `.spp`) au contrôle de source.



Ensuite, choisir la commande de menu **Projet | Contrôle de source | Ajouter au Contrôle de source**. Cela entraîne l'ouverture des dialogues de connexion et de configuration du système de contrôle de source actuellement sélectionné. (Vous pouvez changer le système de contrôle de source par le biais du dialogue Changer Contrôle de source (**Projet | Contrôle de source | Changer Contrôle de source**).)

Suivre les instructions du système de contrôle de source pour établir la connexion et la configuration. Une fois cette étape achevée, tous les fichiers sélectionnés pour l'ajout plus le fichier de projet (fichier `.spp`) sont affichés dans un dialogue Ajouter au contrôle de source (*capture d'écran ci-dessous*). Choisir les fichiers que vous souhaitez ajouter et cliquer sur **OK**.

© 2016-2022 Altova GmbH



Les fichiers seront ajoutés à l'archivage et peuvent être archivés ou extraits selon que la case *Garder ces fichiers extraits* a été cochée ou pas.

Notes de configuration

Vous serez éventuellement invité à créer un dossier dans l'archivage pour le projet s'il n'a pas déjà été créé. Si vous êtes invité, créez le dossier. Le dossier de poste de travail local sera lié à ce dossier créé dans l'archivage (*voir les diagrammes ci-dessous*).

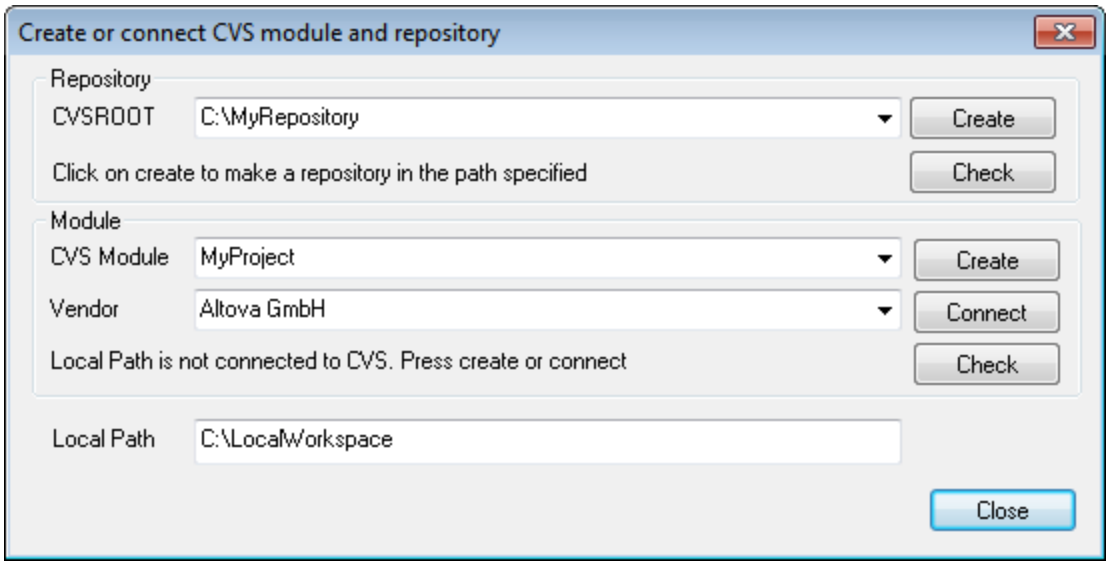
Local Workspace Folder

```
|
|-- MyProject.spp
|-- QuickStart
|   |-- QuickStart.css
|   |-- QuickStart.xml
|   |-- QuickStart.xsd
|-- Grouping
|   |-- Persons
|   |   |-- Persons.xml
```

Archivage

```
|
|-- MyProject (lié au poste de travail local)
| |-- MyProject.spp
| |-- QuickStart
| |   |-- QuickStart.css
| |   |-- QuickStart.xml
| |   |-- QuickStart.xsd
| |-- Grouping
| |   |-- Persons
| |   |   |-- Persons.xml
```

Le dialogue de configuration de Jalindi Igloo est affiché ci-dessous. Le champ CVSROOT est le chemin menant au dossier de l'archivage.

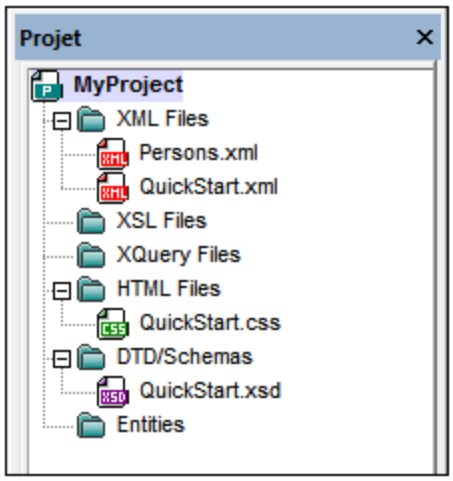


Dans la capture d'écran ci-dessus, le chemin local situe le dossier de poste de travail local qui correspond au module CVS, `MyProject`, et il y est lié.

© 2016-2022 Altova GmbH

9.6 Travailler avec le contrôle de source

Afin de travailler avec le contrôle de source, sélectionner le projet, un dossier de projet, ou un fichier de projet dans la fenêtre Projet (*capture d'écran ci-dessous*) puis choisir la commande que vous souhaitez dans le menu **Projet | Contrôle de source**. Les commandes **Archiver** et **Extraire** sont disponibles en tant que commandes de menu contextuel des items de la fenêtre Projet.



Dans cette section, nous décrivons en détail les fonctions principales du contrôle de source :

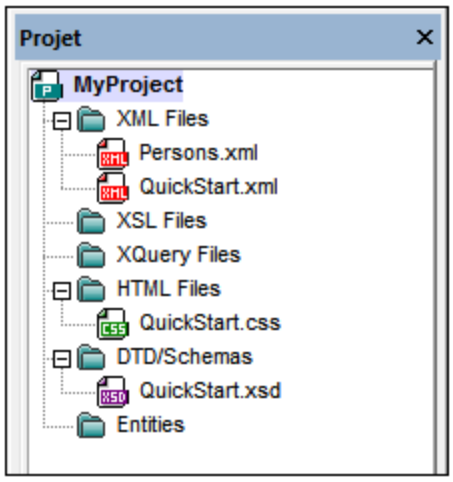
- Ajouter à, Supprimer du contrôle de source
- Extraire, Archiver
- Obtenir les fichiers en lecture seule
- Copier et partager depuis le contrôle de source
- Modifier le contrôle de source

Les commandes supplémentaires dans le menu **Projet | Contrôle de source** sont décrites dans la section Référence menu du manuel. En ce qui concerne les informations spécifiques à un système de contrôle de source particulier, veuillez consulter la documentation d'utilisation de ce système.

9.6.1 Ajouter à, Supprimer du contrôle de source

Ajouter

Une fois qu'un projet a été ajouté au contrôle de source, vous pouvez placer les fichiers sous contrôle de source soit individuellement soit en groupe. On parle aussi d'ajouter les fichiers au contrôle de source. Choisir le fichier dans la fenêtre Projet et cliquer sur la commande **Projet | Contrôle de source | Ajouter au Contrôle de source**. Pour sélectionner des fichiers multiples, maintenir la touche **Ctrl** appuyée tout en cliquant sur les fichiers que vous souhaitez ajouter. Exécuter la commande sur un dossier de projet (vert) (*voir capture d'écran ci-dessous*) pour ajouter tous les fichiers dans le dossier et ses sous-dossiers au contrôle de source.



Lorsque les fichiers sont ajoutés au contrôle de source, la hiérarchie du dossier local est répliquée dans l'archivage (ce n'est pas la hiérarchie du dossier de projet qui est répliquée). Donc, si un fichier se trouve dans un sous-dossiers à x niveaux de profondeurs dans le dossier local, le dossier parent du fichier et tous les anciens dossiers ancêtres sont créés automatiquement dans l'archivage.

Lorsque le premier fichier d'un projet est ajouté au contrôle de source, les liaisons correctes sont créées dans l'archivage et le fichier de projet (fichier `.spp`) est ajouté automatiquement. Pour plus de détails, voir la section Ajouter au contrôle de source.

Symboles du contrôle de source

Les fichiers et le dossier de projet affichent certains symboles, dont la signification est indiquée ci-dessous.

	Archivé. Disponible pour une extraction.
	Extrait par un autre utilisateur. Non-disponible pour une extraction.
	Extrait localement. Peut être édité et archivé.

Supprimer

Pour supprimer un fichier du contrôle de source, choisir le fichier et cliquer sur la commande **Projet | Contrôle de source | Supprimer du contrôle de source**. Vous pouvez aussi supprimer : (i) les fichiers contenus dans un dossier de projet en exécutant la commande dans le dossier, et (ii) le projet complet en exécutant la commande dans le projet.

9.6.2 Extraire, Archiver

Une fois qu'un fichier de projet a été placé sous contrôle de source, il peut être extrait ou archivé en sélectionnant le fichier (dans la fenêtre Projet) et en cliquant sur la commande respective dans le menu **Projet | Contrôle de source** : **Extraire** et **Archiver**.

Lorsqu'un fichier est extrait, une copie de l'archivage est placée dans le dossier local. Un fichier qui est extrait peut être édité. Si un fichier qui se trouve sous contrôle de source n'a pas été extrait, il ne peut pas être édité.

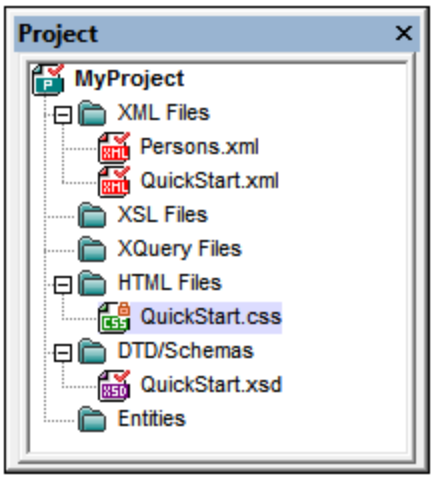
© 2016-2022 Altova GmbH

Une fois que le fichier a été édité, les modifications peuvent être enregistrées dans l'archivage en archivant le fichier. Même si le fichier n'est pas enregistré dans l'application, son archivage permettra d'enregistrer les changements dans l'archivage. Un symbole de coche ou de cadenas dans l'icône de la fenêtre Projet indique si un fichier est extrait ou pas.

Les fichiers et le dossier de projet affichent certains symboles, dont la signification est indiquée ci-dessous.

	Archivé. Disponible pour une extraction.
	Extrait par un autre utilisateur. Non-disponible pour une extraction
	Extrait localement. Peut être édité et archivé.

Le fait de sélectionner le projet ou un dossier dans le projet permet de sélectionner tous les fichiers dans l'objet sélectionné. Afin de sélectionner plusieurs objets (fichiers et dossiers), Appuyer sur la touche **Ctrl** tout en cliquant sur les objets. La capture d'écran ci-dessous montre un projet qui a été extrait. Le fichier `QuickStart.css` a ensuite été archivé.



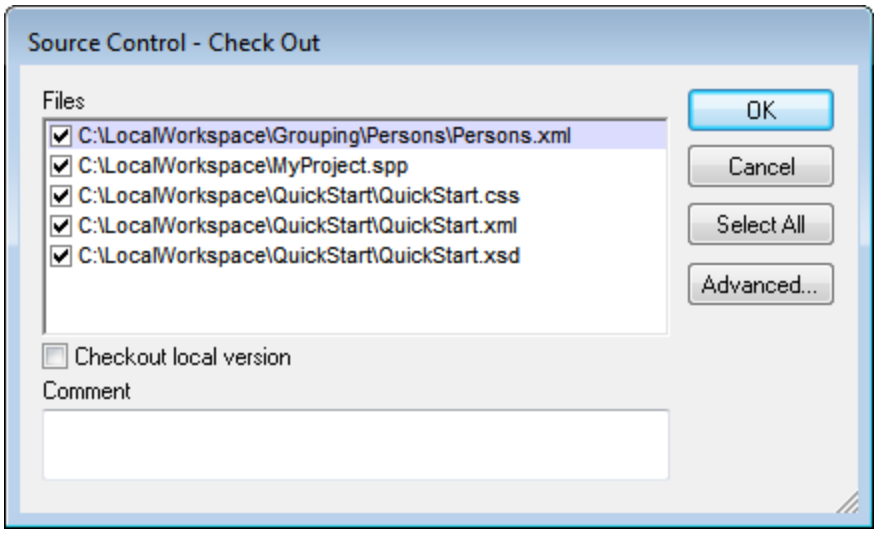
Enregistrer et rejeter les modifications d'édition

Veuillez noter que lorsque vous archivez un fichier, vous pouvez choisir de laisser le fichier extrait. Cela vous permet d'enregistrer les changements d'édition dans l'archivage tout en gardant le fichier extrait, ce qui peut être utile si vous souhaitez enregistrer régulièrement les modifications d'édition dans l'archivage puis continuer l'édition.

Si vous avez extrait un fichier et que vous y avez apporté des modifications, et que vous souhaitez ensuite rejeter ces changements, vous pouvez retourner à la version du document enregistrée dans l'archivage en choisissant la commande **Projet | Contrôle de source | Annuler extraction**.

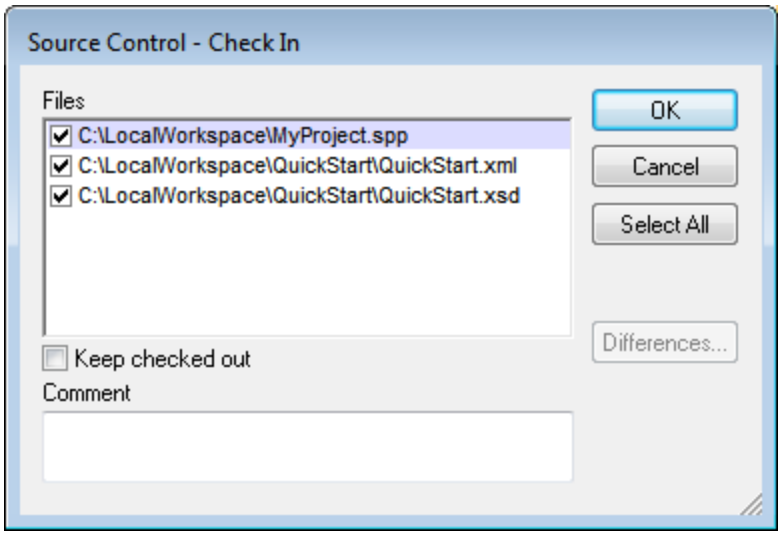
Extraire

Le dialogue Extraire (*capture d'écran ci-dessous*) vous permet de : (i) sélectionner les fichiers à extraire, et (ii) sélectionner si vous souhaitez extraire la version d'archivage ou la version locale.



Archiver

Le dialogue Archiver (*capture d'écran ci-dessous*) vous permet de : (i) sélectionner les fichiers à archiver, et (ii) si vous le souhaitez, de garder le fichier extrait.



Note : Dans les deux dialogues (Extraire et Archiver), des fichiers multiples apparaissent si l'objet sélectionné (projet ou dossier/s de projet) contient plusieurs fichiers.

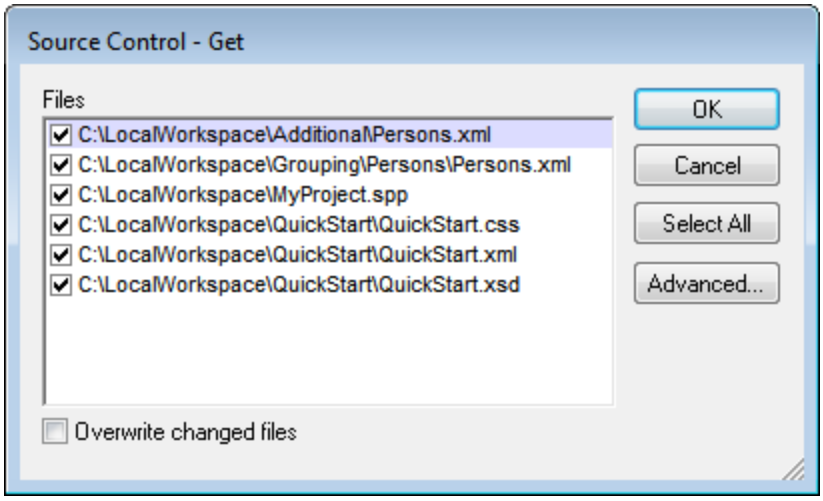
9.6.3 Obtenir les fichiers en lecture seule

La commande **Obtenir** (dans le menu **Projet | Contrôle de source**) extrait les fichier de l'archivage en tant que fichiers en lecture seule. (Pour pouvoir éditer un fichier, vous devez l'extraire.) Le dialogue Obtenir rassemble les fichiers dans l'objet (projet ou dossier) dans lequel la commande **Obtenir** a été exécutée (*voir*

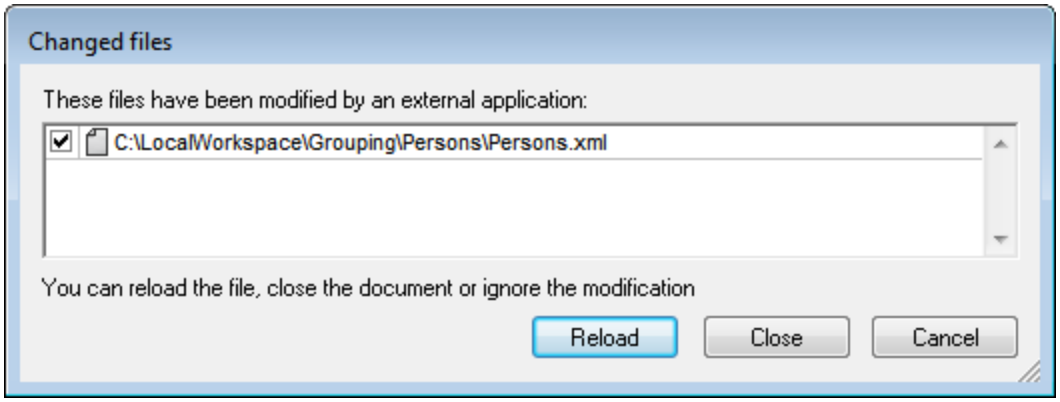
© 2016-2022 Altova GmbH

capture d'écran ci-dessous). Vous pouvez choisir les fichiers à sélectionner en les cochant dans la liste du dialogue Obtenir.

Note : La commande **Obtenir dossiers** vous permet de sélectionner des sous-dossiers individuels dans l'archivage si cela est autorisé par votre système de contrôle de source.

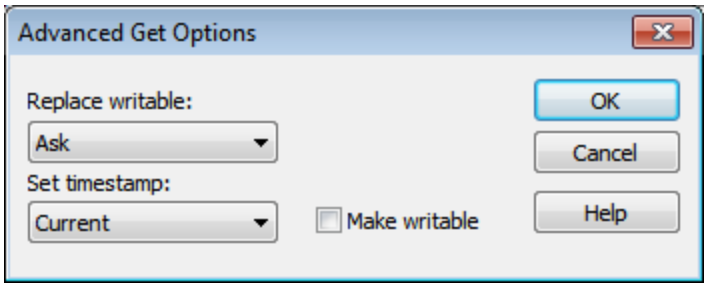


Vous pouvez choisir d'écraser les fichiers extraits modifiés en cochant cette option en bas du dialogue Obtenir. En cliquant sur **OK**, les fichiers seront écrasés. Si un des fichiers écrasé est actuellement ouvert, un dialogue s'affichera (*capture d'écran ci-dessous*) vous demandant si vous souhaitez recharger le fichier (bouton **Recharger**), fermer le fichier (**Fermer**), ou garder la vue actuelle du fichier (**Annuler**).



Options Obtenir avancées

Le dialogue Options | Obtenir avancées (*capture d'écran ci-dessous*) est accessible par le biais de la touche **Avancé** dans le dialogue Obtenir (*voir première capture d'écran dans cette section*).



Ici, vous pouvez définir les options pour (i) remplacer les fichiers accessibles en écriture qui sont extraits, (ii) l'horodatage, et (iii) si la propriété en lecture seule du fichier extrait doit être changé de manière à ce qu'il puisse être accédé en écriture.

Obtenir la dernière version

La commande **Obtenir la dernière version** (dans le menu **Projet | Contrôle de source**) extrait et place la dernière version du contrôle de source du/des fichier/s sélectionné/s dans le répertoire de travail. Les fichiers sont extraits en tant que fichiers en lecture seule et ne sont pas extraits. Cette commande fonctionne comme la commande **Obtenir** (*voir ci-dessus*), mais n'affiche pas le dialogue Obtenir.

Si les fichiers sélectionnés sont actuellement extraits, l'action entreprise dépendra de la manière dont votre système de contrôle de source gère ce genre de situation. Généralement, le système de contrôle de source vous demandera si vous souhaitez remplacer, fusionner ou laisser le fichier extrait tel qu'il est.

Note : Cette commande est récursive lorsqu'elle est exécutée dans un dossier, elle touche tous les fichiers se trouvant en dessous du fichier actuel dans la hiérarchie du dossier.

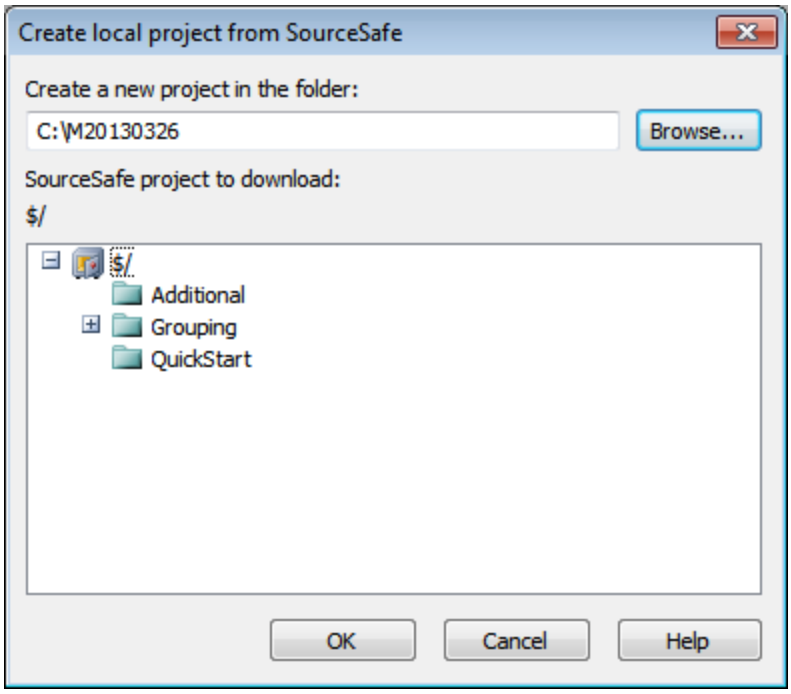
9.6.4 Copier et partager depuis le contrôle de source

La commande **Ouvrir depuis le contrôle de source** crée un nouveau projet d'application depuis un projet se trouvant sous contrôle de source.

Créer le nouveau projet comme suit :

1. Selon le système de contrôle de source utilisé, il peut s'avérer nécessaire, avant de créer un nouveau projet depuis le contrôle de source, de vous assurer qu'aucun autre fichier provenant du projet se trouvant sous contrôle de source n'est extrait.
2. Il n'est pas nécessaire d'ouvrir un projet dans l'application, mais vous pouvez le faire.
3. Choisir la commande **Projet | Contrôle de source | Ouvrir depuis le contrôle de source**.
4. Le système de contrôle de source actuellement configuré affichera ses dialogues de vérification et de connexion. Effectuez la connexion au dossier lié dans l'archivage que vous souhaitez copier.
5. Dans le dialogue qui s'affiche (*capture d'écran ci-dessous*), chercher le dossier local vers lequel les contenus du dossier lié dans l'archivage (auquel vous venez de vous connecter) doit être copié. Dans la capture d'écran ci-dessous, le dossier lié est appelé `MyProject` et est représenté par le signe `$` ; le dossier local est `C:\M20130326`.

© 2016-2022 Altova GmbH

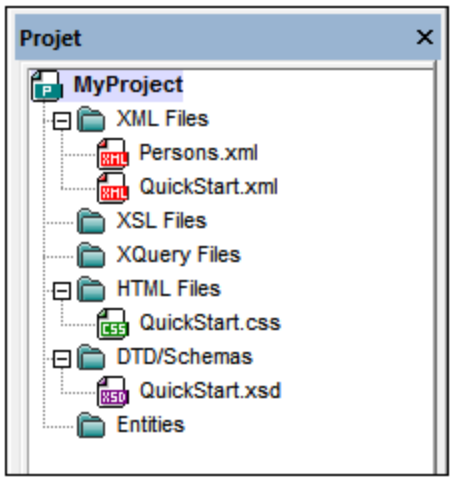


6. Cliquer sur **OK**. Les contenus du dossier lié (`MyProject`) seront copiés dans le dossier local `C:\M20130326`., et un dialogue s'ouvrira vous demandant de choisir le fichier de projet (fichier `.spp`) qui doit être créé en tant que le nouveau projet.
7. Choisir le fichier `.spp` qui aura été copié vers le dossier local. Dans notre exemple, il s'agira de `MyProject.spp` situé dans le dossier `C:\M20130326`. Un nouveau projet nommé `MyProject` sera créé dans l'application et sera affiché dans la fenêtre Projet. Les fichiers du projet se trouvent dans le dossier `C:\M20130326`.

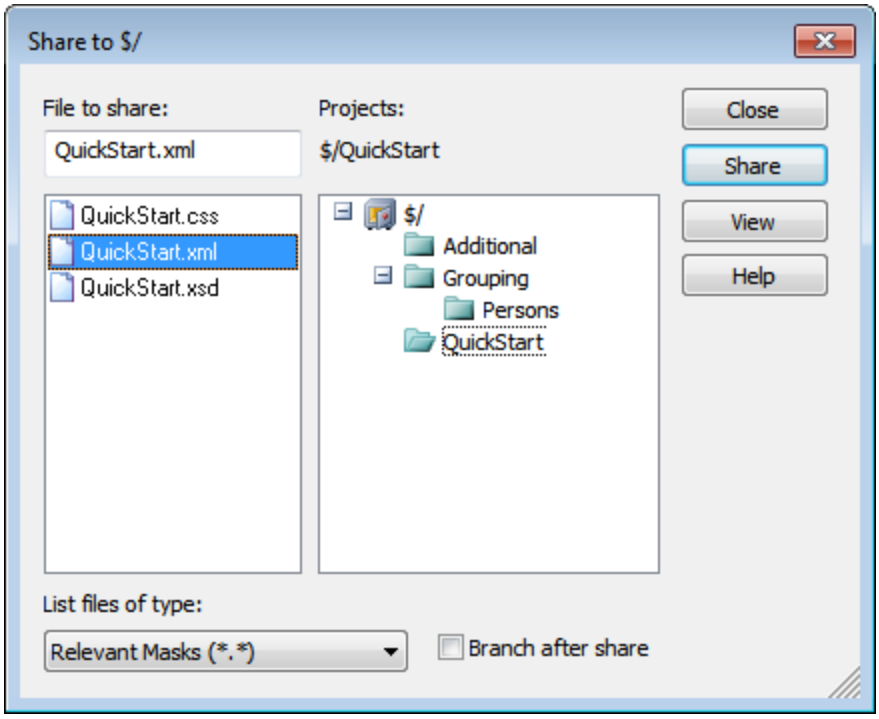
Partager depuis le contrôle de source

La commande **Partager depuis le Contrôle de source** est prise en charge lorsque le système de contrôle de source utilisé prend en charge des partages. Vous pouvez partager un fichier de manière à ce qu'il soit disponible dans plusieurs emplacements locaux. Un changement effectué dans un de ces fichiers locaux se répercutera dans toutes les autres versions "partagées".

Dans la fenêtre Projet de l'application, veuillez tout d'abord sélectionner le projet (*marqué dans la capture d'écran ci-dessous*). Puis cliquer sur **Partager depuis le Contrôle de source**.



Le dialogue Partager avec [Dossier] (*capture d'écran ci-dessous*) s'ouvre.

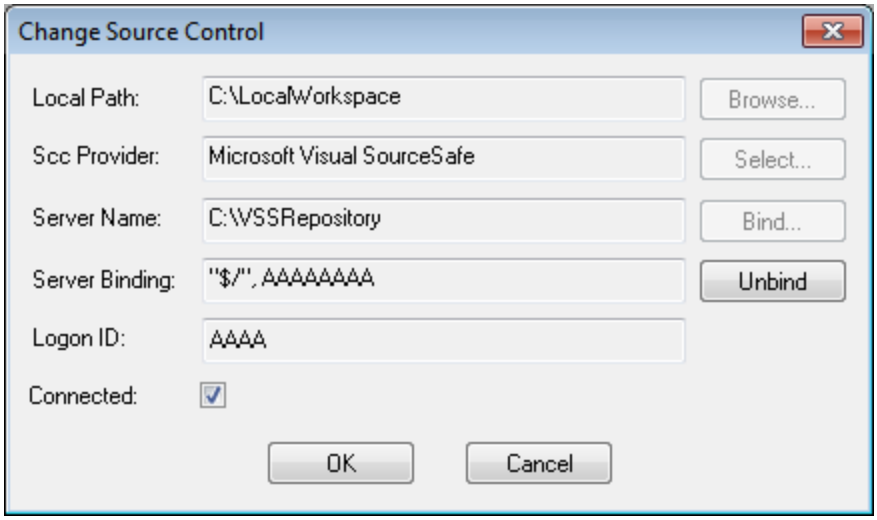


Afin de sélectionner les fichiers à partager, choisissez tout d'abord, dans le volet de droite de l'arborescence de projet (*voir capture d'écran ci-dessus*), le dossier dans lequel les fichiers se trouvent. Les fichiers contenus dans le dossier choisi sont affichés dans le volet de gauche. Sélectionner le fichier que vous souhaitez partager (plusieurs fichiers en appuyant sur la touche **Ctrl** et en cliquant sur les fichiers que vous souhaitez partager). Les fichiers sélectionnés seront affichés dans le champ de saisie *Fichiers à partager* (en haut à gauche). Les fichiers disparaissent du volet de gauche. Cliquer sur **Partager** et puis **Fermer** pour copier les fichiers sélectionnés dans le dossier de partage local. Lorsque vous cliquez sur **Fermer**, les fichiers à partager seront copiés dans l'emplacement local sélectionné.

Le dossier de partage est noté dans le nom du dialogue Partager avec [Dossier]. Dans la capture d'écran ci-dessus, il s'agit du dossier local (puisque le signe `$` est le dossier dans l'archivage auquel le dossier local est

 © 2016-2022 Altova GmbH

lié). Vous pouvez consulter et configurer le dossier de partage dans le dialogue Modifier le Contrôle de source (*capture d'écran ci-dessous*, **Modifier le Contrôle de source**) en changeant le chemin local et la liaison de serveur.



Pour plus de détails concernant le partage avec votre système de contrôle de source, voir la documentation de l'utilisateur du système de contrôle de source.

9.6.5 Modifier le contrôle de source

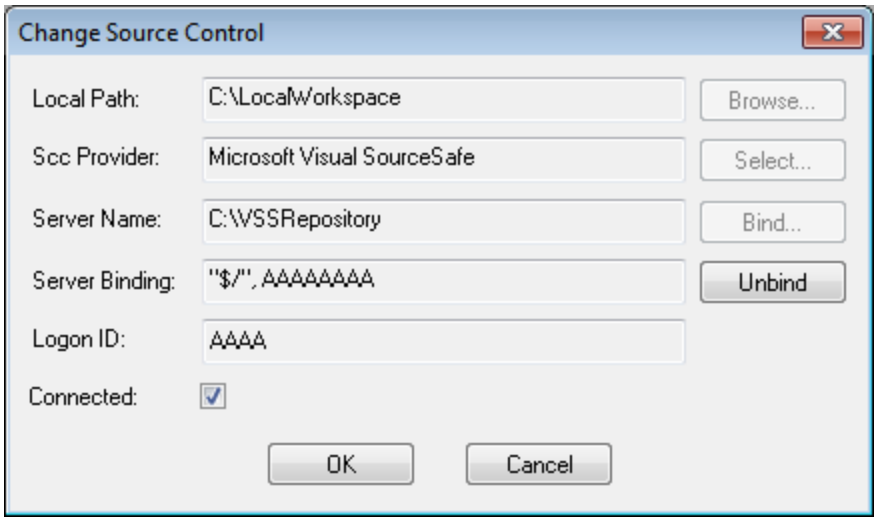
Les paramètres du contrôle de source peuvent être modifiés avec deux commandes dans le menu **Projet | Contrôle de source** :

- **Gestionnaire de Contrôle de source**, qui ouvre l'application du système de contrôle de source et vous permet de configurer les bases de données et les liaisons.
- **Modifier Contrôle de source**, qui permet d'ouvrir le dialogue Modifier le contrôle de source, dans lequel vous pouvez modifier le système de contrôle de source utilisé par l'application Altova et la liaison actuelle. Ce dialogue est décrit ci-dessous.

La liaison actuelle est ce que le projet d'application actif utilisera pour se connecter à la base de données du contrôle de source. La liaison actuelle est correcte lorsque le fichier de projet d'application (fichier `.spp`) se trouve dans le dossier local et que le dossier lié dans l'archivage se trouve à l'endroit où les fichiers de ce projet sont stockés. Généralement, le dossier lié et sa sous--structure correspondra au dossier du poste de travail local et à sa sous-structure.

Dans le dialogue Modifier le Contrôle de source (*capture d'écran ci-dessous*), vous pouvez modifier le système de contrôle de source (*Fournisseur SCC*), le dossier local (*Chemin local*), et la liaison d'archivage (*Nom de serveur* et *Liaison de serveur*).

Ce n'est qu'après avoir défait la liaison actuelle que les paramètres peuvent être changés. Défaire la liaison actuelle avec la touche **Annuler**. Tous les paramètres sont maintenant éditables.



Modifier les paramètres de contrôle de source comme suit :

1. Utiliser la touche **Chercher** pour parcourir le dossier local et la touche **Sélectionner** pour choisir parmi les systèmes de contrôle de source installés.
2. Ensuite, vous pourrez lier le dossier local à une base de donnée d'archivage. Cliquer sur la touche **Lier**. Le dialogue de connexion de votre système de contrôle de source s'affiche.
3. Si vous avez saisi une *ID de connexion*, cela sera transféré au système de contrôle de source ; sinon, vous devrez saisir vos détails d'enregistrement dans le dialogue de connexion.
4. Choisir la base de données dans l'archivage que vous souhaitez lier à ce dossier local. Ce paramètre peut être réparti sur plusieurs dialogues.
5. Une fois que le paramètre a été créé, cliquer sur **OK** dans le dialogue Modification du Contrôle de source.

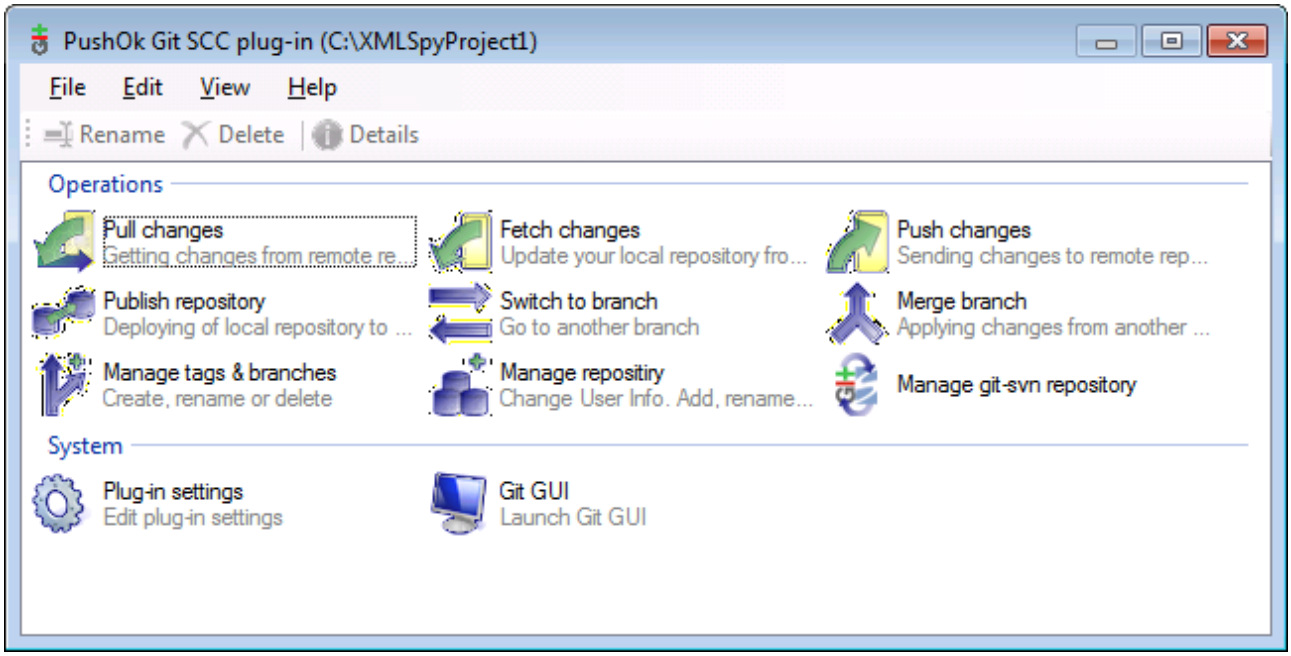
© 2016-2022 Altova GmbH

9.7 Contrôle de source avec Git

La prise en charge de Git en tant que système de contrôle de source dans Authentic Desktop est disponible par le biais d'un plugin tiers appelé **plugin GIT SCC** (http://www.pushok.com/software/git.html).

> Au moment de la rédaction de cette documentation, le **plugin GIT SCC** est disponible pour une utilisation expérimentale. L'enregistrement avec le éditeur du plugin est exigée pour pouvoir utiliser le plugin.

Le plugin GIT SCC vous permet de travailler avec un archivage Git avec les commandes disponibles dans le menu **Projet | Contrôle de source** de Authentic Desktop. Veuillez noter que les commandes dans le menu **Projet | Contrôle de source** de Authentic Desktop sont fournies par Microsoft Source Control Plug-in API (MSSCCI API), qui aborde une philosophie de design différente de Git. C'est ainsi que le plugin fait office d'intermédiaire entre des fonctions de type "Visual Source Safe" et des fonctions Git. D'un côté, cela signifie qu'une commande comme **Obtenir la dernière version** peut ne pas être applicable avec Git. D'un autre côté, il existe des nouvelles actions spécifiques à Git, qui sont disponibles dans le dialogue "Gestionnaire de Contrôle de source" fourni par le plugin (sous le menu **Projet | Contrôle de source | Gestionnaire du Contrôle de source** de Authentic Desktop).



Le dialogue Gestionnaire de Contrôle de source

Les autres commandes que vous utiliserez fréquemment sont disponibles directement dans le menu **Projet | Contrôle de source**.

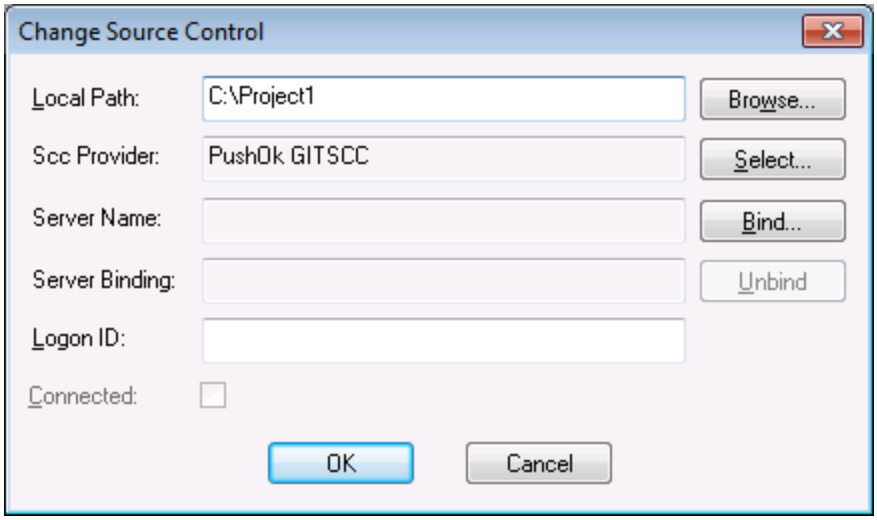
Les sections suivantes décrivent la configuration initiale du plugin, et le flux de travail de base :

- Activer le Contrôle de source avec le plugin Git SCC
- Ajouter un projet au Contrôle de source avec Git
- Cloner un projet depuis le Contrôle de source avec Git

9.7.1 Activer le Contrôle de source avec le plugin Git SCC

Pour activer le contrôle de source Git avec Authentic Desktop, le **plug-in PushOK GIT SCC** tiers doit être installé, enregistré et sélectionné en tant que fournisseur de contrôle de source, comme suit :

1. Télécharger le fichier d'installation de plugin depuis le site web de l'éditeur (http://www.pushok.com), l'exécuter et suivre les instructions d'installation.
2. Dans le menu **Projet** de Authentic Desktop, cliquer sur **Changer contrôle de source**, et vérifier que **PushOk GITSCC** a été sélectionné en tant que fournisseur de contrôle de source. Si vous ne voyez pas **Push Ok GITSCC** dans la liste des fournisseurs, il est probable que l'installation du plugin n'a pas réussi. Dans ce cas, consultez la documentation de l'éditeur pour une trouver une solution.



3. Lorsqu'un dialogue vous invite à enregistrer le plugin, cliquer sur **Enregistrement** et suivre les étapes de l'assistant pour terminer le processus d'enregistrement.

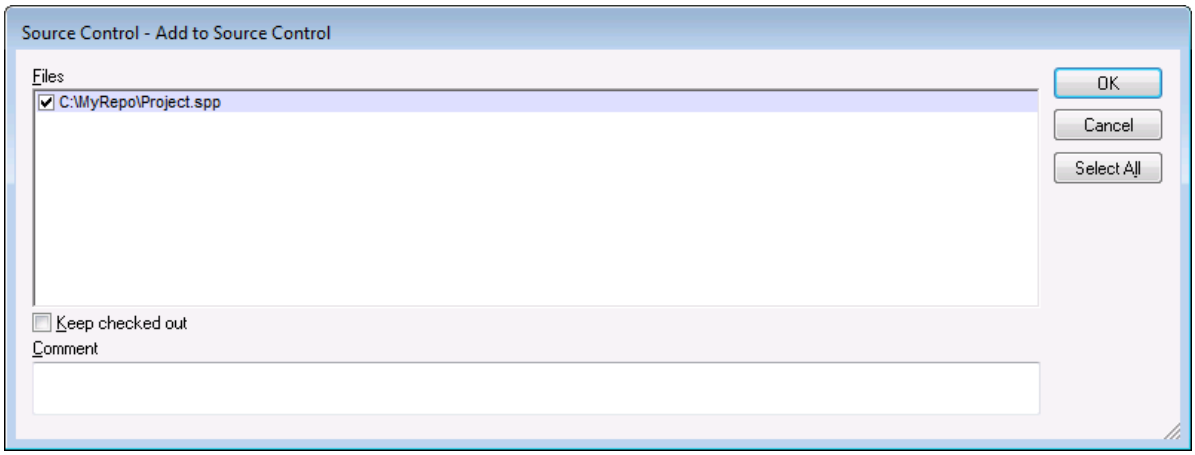
9.7.2 Ajouter un projet au Contrôle de source avec Git

Vous pouvez enregistrer des projets Authentic Desktop en tant qu'archivages Git. La structure des fichiers ou des dossiers que vous ajoutez au projet correspondraient à la structure de l'archivage Git.

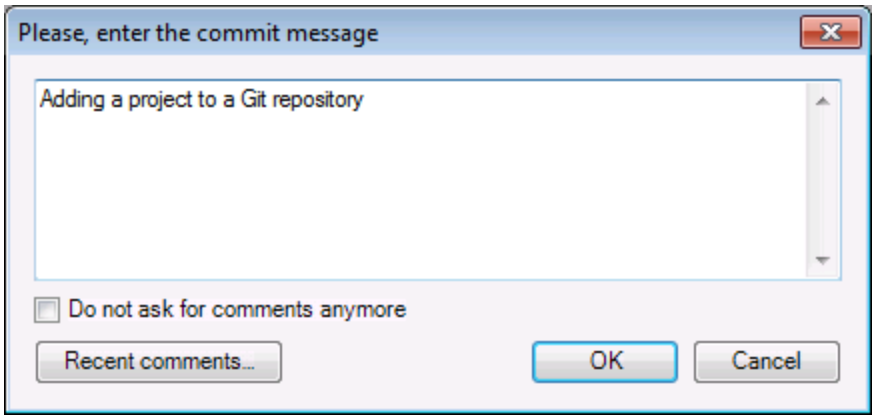
Pour ajouter un projet au contrôle de source Git :

1. Veuillez vous assurer que le **Plug-in PushOK GIT SCC** est configuré en tant que le fournisseur de contrôle de source (voir Activer le contrôle de source Git avec le Plugin GIT SCC).
2. Créer un nouveau projet en utilisant la commande de menu **Projet | Créer Projet**.
3. Enregistrer le projet dans un dossier local, par exemple `C:\MyRepo\Project.spp`
4. Dans le menu **Projet**, sous **Contrôle de source**, cliquer **Ajouter au Contrôle de source**.

© 2016-2022 Altova GmbH

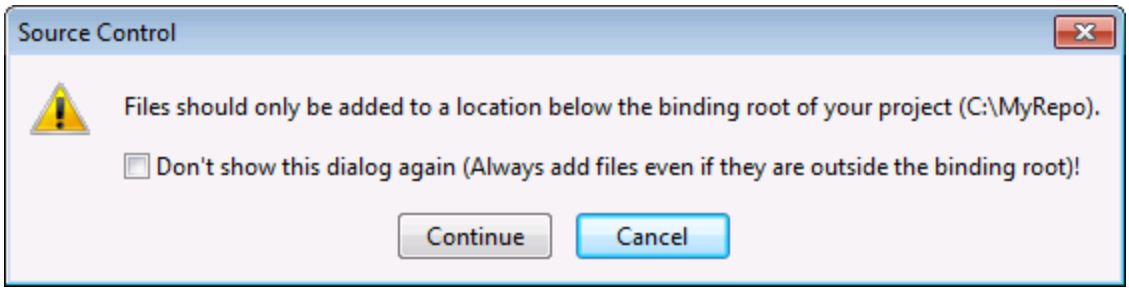


5. Cliquer sur **OK**.



6. Saisir le texte de votre message de validation et cliquer sur **OK**.

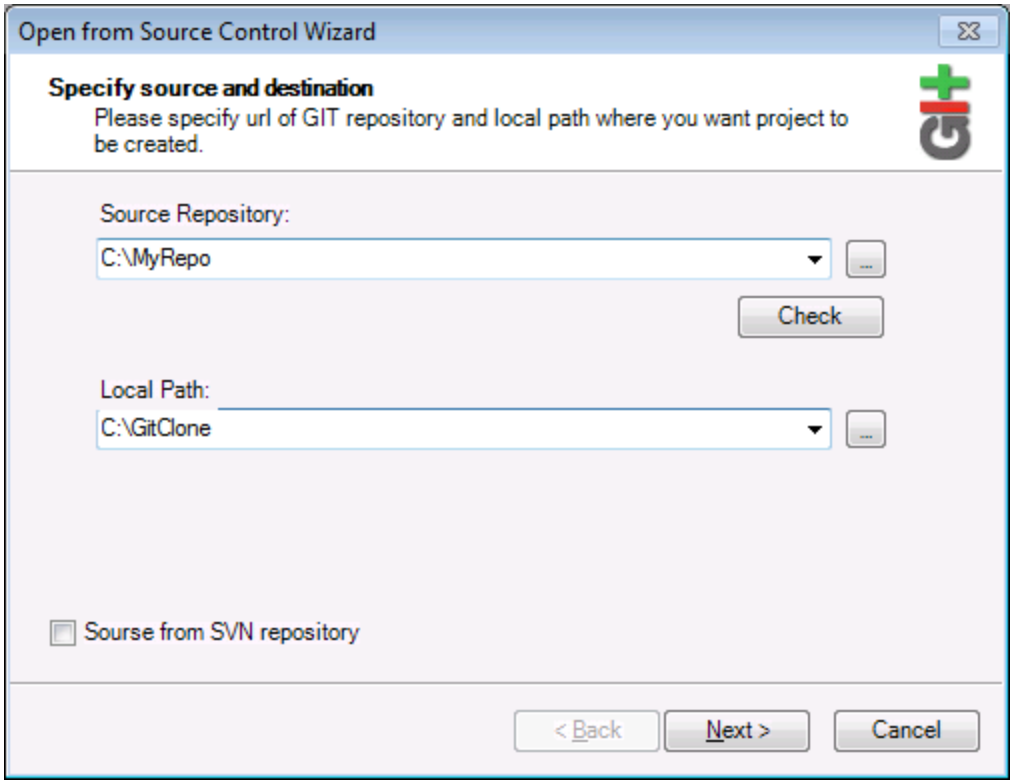
Vous pouvez maintenant commencer à ajouter des fichiers et des dossiers à votre projet. Veuillez noter que tous les fichiers de projet et les dossiers doivent se trouver sous un dossier racine du projet. Par exemple, si le projet a été créé dans le dossier `C:\MyRepo`, alors seuls les fichiers se trouvant sous `C:\MyRepo` devraient être ajoutés au projet. Sinon, si vous essayez d'ajouter à votre projet des fichiers qui se trouvent en dehors du dossier racine de projet, un message d'avertissement s'affichera :



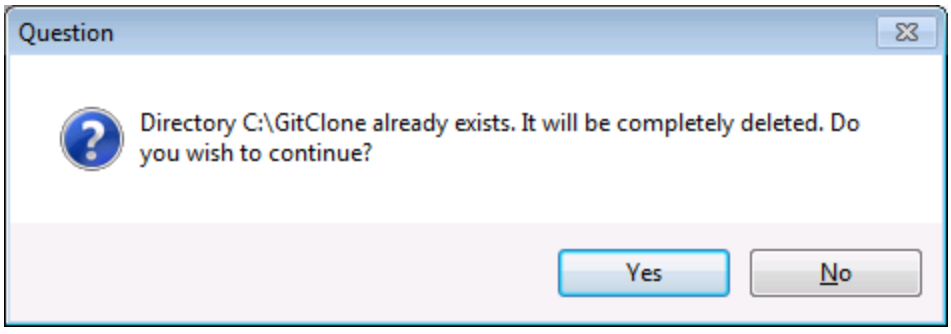
9.7.3 Cloner un projet depuis le Contrôle de source avec Git

Les projets qui ont précédemment été ajoutés au Contrôle de source Git (voir Ajouter un projet au contrôle de source Git) peuvent être ouverts depuis l'archivage Git comme suit :

1. Veuillez vous assurer que le **Plugin PushOK GIT SCC** est configuré en tant que le fournisseur de contrôle de source (voir Activer le contrôle de source Git avec le Plugin GIT SCC).
2. Dans le menu **Projet**, cliquer sur **Contrôle de source | Ouvrir depuis le Contrôle de source**.
3. Saisir le chemin ou l'URL de l'archivage de la source. Cliquer sur **Contrôler** pour vérifier la validité du chemin ou de l'URL.

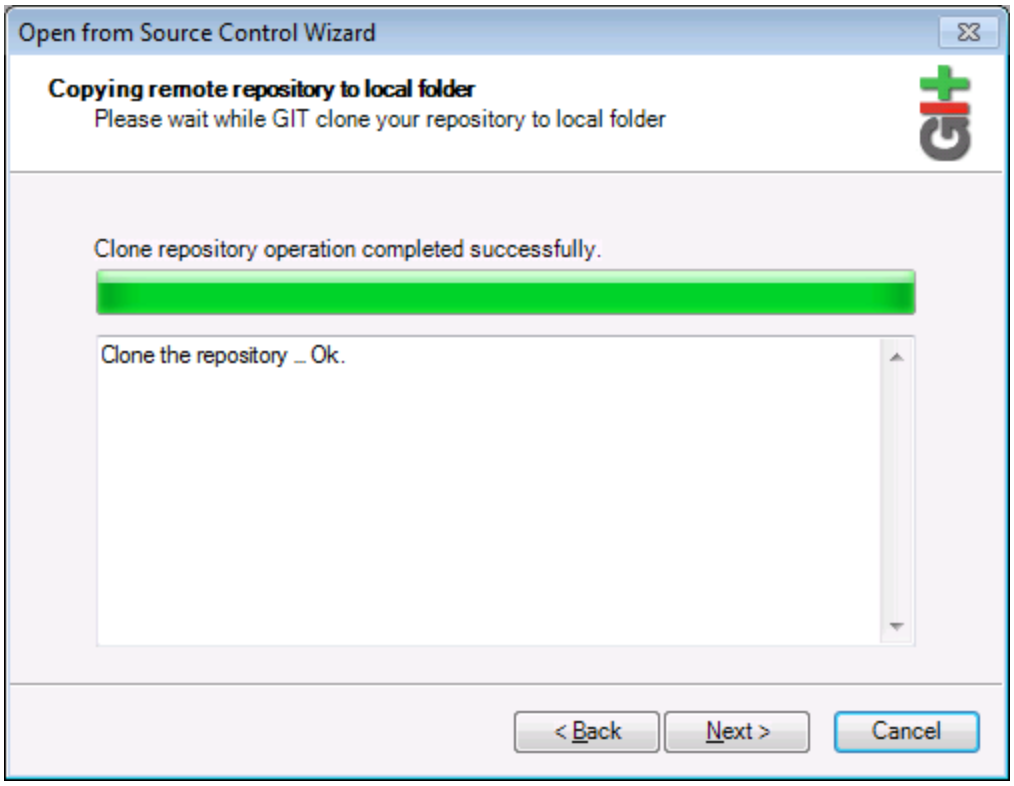


4. Sous **Chemin local**, saisir le chemin vers le dossier local où vous souhaitez que le projet soit créé et cliquer sur **Suivant**. Si le dossier local existe (même s'il est vide), le dialogue suivant s'ouvre :



5. Cliquer **Oui** pour confirmer, puis cliquer sur **Suivant**.

© 2016-2022 Altova GmbH



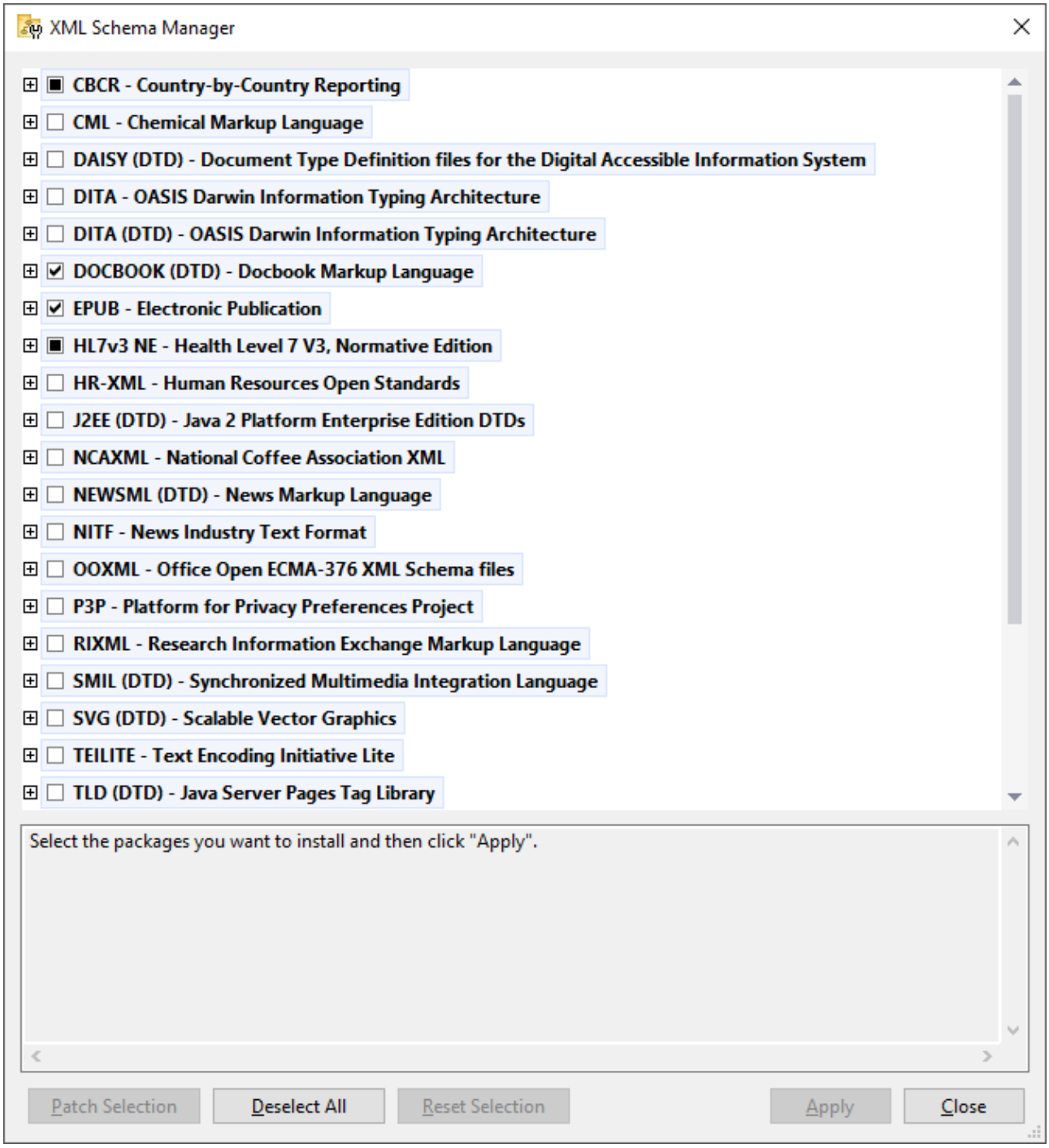
6. Suivez les étapes restantes nécessaires pour votre cas de figure particulier.
7. Une fois l'assistant terminé, un dialogue Chercher apparaît, vous demandant d'ouvrir le fichier de projet Authentic Desktop (*.spp). Choisir le fichier de projet pour charger les contenus du projet dans Authentic Desktop.

10 Gestionnaire de schéma

XML Gestionnaire de schéma est un outil qui propose un moyen centralisé d'installer et de gérer des schémas XML (DTD pour XML et Schémas XML) pour une utilisation sur toutes les applications XBRL Altova, y compris Authentic Desktop

- Sur Windows, Gestionnaire de schéma a une interface utilisateur graphique *(voir la capture d'écran ci-dessous*) et est aussi disponible dans la ligne de commande. (Les applications desktop d'Altova sont disponibles sur Windows uniquement ; *voir la liste ci-dessous*.)
- Sur Linux et Mac Gestionnaire de schéma, l'outil est disponible uniquement dans la ligne de commande. (Les applications serveur d'Altova sont disponibles sur Windows, Linux et macOS ; *voir la liste ci-dessous*.)

© 2016-2022 Altova GmbH



Application d'Altova qui fonctionnent avec Schema Manager

Applications desktop (Windows uniquement)	Applications de serveur (Windows, Linux, macOS)
XMLSpy (toutes éditions)	RaptorXML Server, RaptorXML+XBRL Server

MapForce (toutes éditions)	StyleVision Server
StyleVision (toutes éditions)	
Authentic Desktop Enterprise Edition	

Installation et désinstallation de Gestionnaire de schéma

Gestionnaire de schéma est installé automatiquement quand vous installez d'abord une nouvelle version de l'Altova Mission Kit Enterprise Edition ou toute application XML-schema-aware d'Altova (*voir la table ci-dessus*).

De même, il est supprimé automatiquement lorsque vous désinstallez la dernière application XML-schema-aware d'Altova depuis votre ordinateur.

Fonctions <% SCHEMA-MANAGER%>

Gestionnaire de schéma propose les fonctions suivantes :

- Affiche les schémas XML installés sur votre ordinateur et contrôle si de nouvelles versions sont disponibles pour le téléchargement.
- Télécharge des versions plus récentes des schémas XML indépendamment du cycle de release des produits Altova. (Altova stocke des schémas en ligne et vous pouvez les télécharger via Gestionnaire de schéma.)
- Installer ou désinstaller une des multiples versions d'un schéma donné (ou toutes les versions, si nécessaire).
- Un schéma XML peut avoir des dépendances sur d'autres schémas. Lorsque vous installez ou désinstallez un schéma particulier, Gestionnaire de schéma vous informe sur d'autres schémas dépendants et les installera ou désinstallera également automatiquement.
- Gestionnaire de schéma utilise le mécanisme du catalogue XML pour mapper les références de schéma aux fichiers locaux. Dans le cas de larges schémas XML, le traitement sera plus rapide que si les schémas étaient à un emplacement à distance.
- Tous les schémas majeurs sont disponibles via Gestionnaire de schéma et sont régulièrement mis à jour pour les dernières versions. Ceci vous fournit une ressource unique pour gérer tous vos schémas et les mettre à disposition de toutes les applications XML-schema-aware d'Altova.
- Les changements réalisés dans Gestionnaire de schéma prennent effet pour tous les produits d'Altova sur cet appareil.

Comment cela fonctionne ?

Altova stocke tous les schémas XML utilisés dans les produits Altova en ligne. Ce référentiel est mis à jour lorsque de nouvelles versions de schémas sont publiées. Gestionnaire de schéma affiche des informations sur les derniers schémas disponibles lorsqu'ils sont appelés dans son formulaire GUI de même que sur CLI. Vous pouvez ensuite installer, mettre à jour ou désinstaller les schémas via Gestionnaire de schéma.

Gestionnaire de schéma installe également les schémas d'une autre manière. Sur le site web d'Altova (https://www.altova.com/schema-manager), vous pouvez sélectionner un schéma et ses Schémas dépendants que vous souhaitez installer. Le site web préparera un fichier de type `.altova_xmlschemas` pour le téléchargement qui contient des informations sur la sélection de schéma. Lorsque vous double-cliquez sur ce fichier ou le passez à Gestionnaire de schéma via CLI comme argument de la commande `installer`, Gestionnaire de schéma installera les schémas que vous avez sélectionnés.

© 2016-2022 Altova GmbH

Cache local : suivre vos schémas
Toutes les informations sur les schémas installés sont suivies dans un répertoire cache centralisé sur votre ordinateur, situé ici :

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

Ce répertoire cache est mis à jour régulièrement avec le dernier statut des schémas dans l'emplacement de stockage en ligne d'Altova. Ces mises à jour sont réalisées aux moments suivants :

- À chaque fois que vous lancez Gestionnaire de schéma.
- Lorsque vous exécutez Authentic Desktop pour la première fois dans un jour donné du calendrier.
- Si Authentic Desktop est ouvert plus de 24 heures, le cache est mis à jour toutes les 24 heures.
- Vous pouvez aussi mettre à jour le cache en exécutant la commande de `mise à jour` dans l'interface de ligne de commande.

Pour cette raison, le cache permet à Gestionnaire de schéma de suivre continuellement vos schémas installés par rapport aux schémas disponibles en ligne sur le site web d'Altova.

Ne modifiez pas le cache manuellement !

Le répertoire de cache local est entretenu automatiquement sur la base des schémas que vous installez ou désinstallez. Il ne devrait pas être altéré ou supprimé manuellement. Si vous êtes amené à réinitialiser Gestionnaire de schéma à son état original "intact", alors, sur l'interface de la ligne de commande (CLI) : (i) exécutez la commande `reset`, et (ii) exécutez la commande `initialize`. (En alternative, exécutez la commande **`reset`** avec l'option **`--i`**.)

10.1 Exécuter Schema Manager

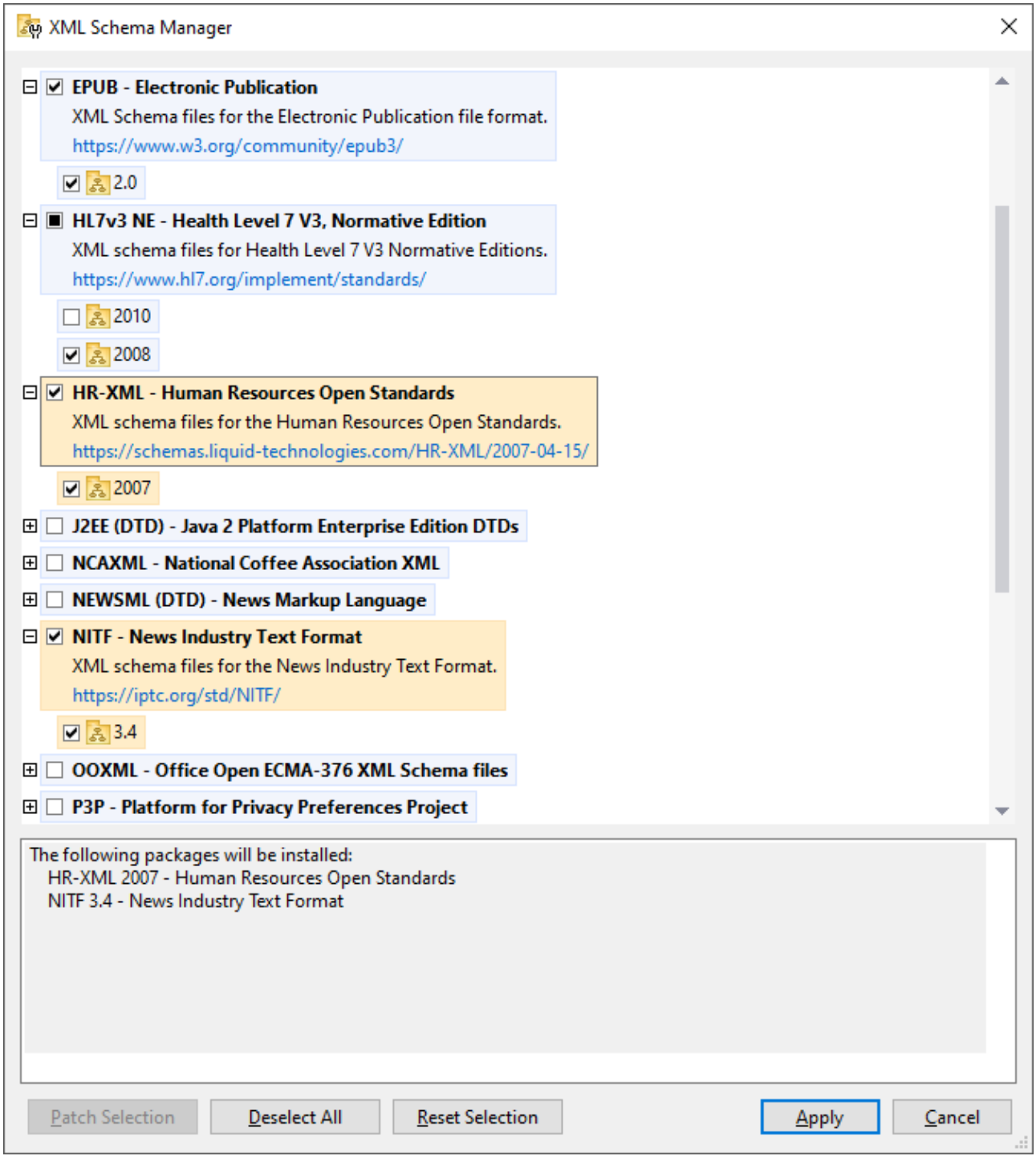
Interface utilisateur graphique

Vous pouvez accéder à la GUI de Gestionnaire de schéma des manières suivantes :

- *Durant l'installation de Authentic Desktop:* Vers la fin de la procédure d'installation, sélectionnez la case à cocher *Invoke Altova XML-Schema Manager* pour accéder directement à la GUI de Gestionnaire de schéma. Ceci vous permettra d'installer les schémas au cours de la procédure d'installation de votre application Altova.
- *Après l'installation de Authentic Desktop:* Une fois que votre installation a été installée, vous pouvez accéder à la GUI de Gestionnaire de schéma à tout moment, via la commande de menu **Tools | XML Schema Manager**.
- Via le fichier `.altova_taxonomies` téléchargé de Altova website: Double-cliquez sur le fichier téléchargé pour exécuter la GUI de Gestionnaire de schéma, qui sera configurée pour installer les schémas que vous avez sélectionnés (le site web) pour installation.

Une fois que la GUI de Gestionnaire de schéma (*capture d'écran ci-dessous*) a été ouverte, les schémas déjà installés seront affichés tels sélectionnés. Si vous voulez installer un schéma additionnel, sélectionnez-le. Si vous voulez désinstaller un schéma déjà installé, désélectionnez-le. Une fois que vous avez faits vos sélections et/ou désélections, vous êtes prêts pour appliquer vos changements. Les schémas qui seront installés ou désinstallés seront mis en surbrillance et un message sur les modifications à venir sera posté dans le volet Messages au niveau inférieur de la fenêtre Gestionnaire de schéma (*voir la capture d'écran*).

© 2016-2022 Altova GmbH



Interface de ligne de commande

Vous pouvez exécuter Gestionnaire de schéma depuis une interface de ligne de commande en sélectionnant son fichier exécutable, **`xmlschemamanager.exe`**.

Le fichier **xmlschemamanager.exe** est situé dans le dossier suivant :

- *Sur Windows :* `C:\ProgramData\Altova\SharedBetweenVersions`
- *Sur Linux ou macOS (application serveur uniquement) :* `%INSTALLDIR%/bin`, où `%INSTALLDIR%` est le répertoire d'installation du programme.

Vous pouvez alors utiliser toute commande dans la section de référence de la commande CLI.

Pour afficher l'aide pour la commande, exécutez l'étape suivante :

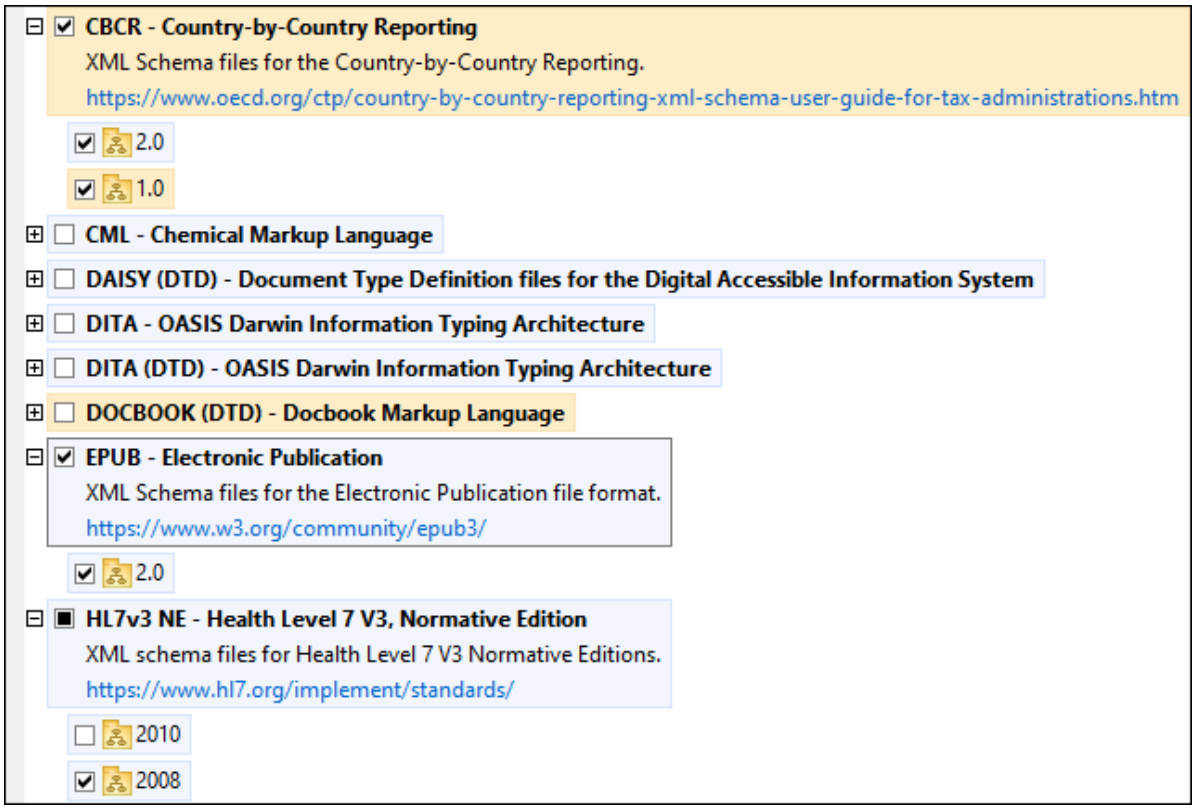
- *Sur Windows :* `xmlschemamanager.exe --help`
- *Sur Linux ou macOS (application serveur uniquement) :* `sudo ./xmlschemamanager --help`

© 2016-2022 Altova GmbH

10.2 Catégories de statut

Gestionnaire de schéma catégorise les schémas sous sa gestion comme suit :

- *Schémas installés.* Ceux-ci sont affichés dans la GUI avec leurs cases à cocher sélectionnées (*dans la capture d'écran ci-dessous, les versions cochées et bleues des schémas EPUB et HL7v3 NE sont des schémas installés*). Si toutes les versions de schéma sont sélectionnées, alors la marque de sélection est une coche. Si au moins une version de schéma est décochée, alors la coche de sélection est un carré coloré plein. Vous pouvez décocher un schéma installé pour le **désinstaller** ; (*dans la capture d'écran ci-dessous, le DocBook DTD est installé et a été désélectionné, le préparant ainsi pour la désinstallation*).
- *Schémas désinstallés disponibles.* Ils sont affichés dans la GUI avec leurs cases à cocher non sélectionnées. Vous pouvez sélectionner les schémas que vous souhaitez **installer**.



- Les *Schémas pouvant être mises à niveau* sont ceux qui ont été revus par leurs émetteurs depuis qu'ils ont été installés. Ils sont indiqués dans la GUI par une icône . Vous pouvez **retoucher** le schéma installé avec la révision disponible.

Points à noter

- Dans la capture d'écran ci-dessus, les deux schémas CBCR sont cochés. Celui avec un arrière-plan bleu est déjà installé. Celui avec un arrière-plan jaune est désinstallé et a été sélectionné pour l'installation. Notez que le schéma HL7v3 NE 2010 n'est pas installé et n'a pas été sélectionné pour l'installation.
- Un arrière-plan jaune signifie que le schéma sera modifié d'une manière ou d'une autre quand le bouton **Appliquer** est cliqué. Si un schéma est décoché et a un arrière-plan jaune, cela signifie qu'il sera

désinstallé quand le bouton **Appliquer** est cliqué. Dans la capture d'écran ci-dessus, le DocBook DTD a un tel statut.

- Lorsque vous exécutez Gestionnaire de schéma depuis la ligne de commande, la commande `list` est utilisée avec différentes options pour pour recenser les différentes catégories de schémas :

`xmlschemamanager.exe list`	Recense tous les schémas installés et disponibles ; ceux pouvant être mis à niveau sont également indiqués.
`xmlschemamanager.exe list -i`	Recense les schémas installés uniquement ; ceux pouvant être mis à niveau sont également indiqués
`xmlschemamanager.exe list -u`	Recense les schémas pouvant être mis à niveau

Note : Sur Linux et macOS, **`use sudo ./ xmlschemamanager list`**

© 2016-2022 Altova GmbH

10.3 Retoucher ou Installer un schéma

Retoucher un schéma installé

Occasionnellement, des schémas XML peuvent recevoir des patchs (mises à niveau ou révisions) de leurs émetteurs. Lorsque Gestionnaire de schéma détecte que des patchs sont disponibles, ceux-ci sont indiqués dans les listes de schéma de Gestionnaire de schéma et vous pouvez installer les patchs rapidement.

Dans la GUI

Les patchs sont indiqués par l'icône  . (*Voir aussi la rubrique précédente sur les catégories de statut.*) Si les patchs sont disponibles, le bouton **Patch Selection** sera activé. Cliquez dessus pour sélectionner et préparer tous les patchs pour installation. Dans la GUI, l'icône de chaque schéma sera patchée de  à  , et le volet des Messages en bas du dialogue recense les patchs qui doivent être appliqués. Lorsque vous êtes prêt pour installer des patchs sélectionnés, cliquez sur **Appliquer**. Tous les correctifs seront appliqués ensemble. Notez que si vous décochez un schéma marqué pour une correction, vous désinstallerez de fait ce schéma.

Sur le CLI

Pour appliquer un patch dans l'interface de ligne de commande :

1. Exécuter la commande `list -u` . Cela liste tout schéma lorsque des mises à niveau sont disponibles.
2. Exécutez la commande `upgrade` pour installer les patchs.

Installer un schéma disponible

Vous pouvez installer des schémas en utilisant soit la GUI Gestionnaire de schéma ou en envoyant à Gestionnaire de schéma les instructions d'install via la ligne de commande.

Note : Si le schéma actuel référence d'autres schémas, les schémas référencées sont aussi installés.

Dans la GUI

Pour installer des schémas utilisant la GUI Gestionnaire de schéma GUI, sélectionnez les schémas que vous voulez installer et cliquez sur **Appliquer**.

Vous pouvez aussi sélectionner les schémas que vous voulez installer sur le site web d'Altova et générer un fichier téléchargeable `.altova_xmlschemas`. Lorsque vous double-cliquez sur ce fichier, il ouvrira Gestionnaire de schéma avec les schémas que vous vouliez présélectionner. La seule chose qui vous reste à faire, c'est cliquer sur **Appliquer**.

Sur le CLI

Pour installer des schémas via la ligne de commande, exécutez la commande `install` :

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

où `Schéma` est le schéma (ou les schémas) que vous voulez installer ou un fichier `.altova_xmlschemas`. Un schéma est référencé par un identifiant de format `<name>-<version>`. (Les identifiants de schémas sont affichés quand vous exécutez la commande `list`.) Vous pouvez saisir autant de schémas que vous le souhaitez. Pour plus de détails, voir la description de la commande `install`.

Note : sur Linux ou macOS, utilisez la commande **`sudo ./xmlschemamanager`**.

Installer un schéma requis

Lorsque vous exécutez une commande activée par XML dans Authentic Desktop, et que Authentic Desktop découvre qu'un schéma dont il a besoin pour exécuter la commande n'est pas présente ou est incomplète, Gestionnaire de schéma affichera l'information sur les schémas manquants. Vous pouvez ensuite installer directement tout schéma manquant via Gestionnaire de schéma.

Dans la GUI de Schema Manager, vous pouvez consulter tous les schémas précédemment installés à tout moment en exécutant Gestionnaire de schéma depuis **Outils | Gestionnaire de schéma**.

© 2016-2022 Altova GmbH

10.4 Désinstaller un schéma, Réinitialiser

Désinstaller un schéma

Vous pouvez désinstaller des schémas en utilisant soit la GUI Gestionnaire de schéma ou en envoyant à Gestionnaire de schéma les instructions d'installation via la ligne de commande.

Note : si la Police que vous voulez désinstaller référence d'autres Schémas, alors les Schémas référencées sont également désinstallées.

Dans la GUI

Pour désinstaller les schémas utilisant la GUI Gestionnaire de schéma, effacez leurs cases à cocher et cliquez sur **Appliquer**. Les schémas sélectionnés et leurs schémas référencées seront désinstallés.

Pour désinstaller tous les schémas, cliquez sur **Désélectionner tout** et cliquez sur **Appliquer**.

Sur le CLI

Pour désinstaller des schémas via la ligne de commande, exécutez la commande `désinstaller` :

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

où chaque argument **Schéma** est le schéma que vous voulez désinstaller ou un fichier **.altova_xmlschemas**. Un schéma est spécifié par un identifiant qui a un format **<name>-<version>**. (Les identifiants de schémas sont affichés quand vous exécutez la commande `list`.) Vous pouvez saisir autant de schémas que vous le souhaitez. Pour plus de détails, voir la description de la commande `désinstaller`.

Note : sur Linux ou macOS, utilisez la commande **sudo ./xmlschemamanager**.

Réinitialiser Gestionnaire de schéma

Vous pouvez réinitialiser Gestionnaire de schéma. Ceci supprime toutes les schémas installés et le répertoire de mise sous cache.

- Dans la GUI, cliquez sur **Reset Selection**.
- Dans la CLI, exécutez la commande `reset`.

Une fois avoir exécuté cette commande, vous devrez exécuter la commande `initialize`, pour pouvoir recréer le répertoire de mise sous cache. En alternative, exécutez la commande `reset` avec l'option `-i`.

Notez que `reset -i` restaure l'installation originale du produit, il est recommandé d'exécuter la commande `update` après avoir réalisé la réinitialisation. En alternative, exécutez la commande `reset` avec les options `-i` and `-u`.

10.5 Interface de ligne de commande (CLI)

Pour appeler Gestionnaire de schéma dans la ligne de commande, vous devez connaître le chemin de l'exécutable. Par défaut, l'exécutable Gestionnaire de schéma est installé dans le chemin suivant :

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

Note : sur les systèmes Linux et macOS, une fois que vous avez changé le répertoire à celui contenant l'exécutable, vous pouvez appeler l'exécutable avec **`sudo ./xmlschemamanager`**. Le préfixe **`./`** indique que l'exécutable est le répertoire actuel. Le préfixe **`sudo`** indique que la commande que la commande doit être exécutée avec des privilèges root.

Syntaxe de ligne de commande

La syntaxe générale pour utiliser la ligne de commande est la suivante :

```
<exec> -h | --help | --version | <command> [options] [arguments]
```

Dans l'extrait ci-dessus, la barre verticale **`|`** sépare un ensemble d'items mutuellement exclusifs. Les crochets **`[]`** indiquent des items optionnels. De manière générale, vous pouvez saisir le chemin d'exécutable suivi soit par les options `-h`, `--help`, ou `--version` ou par une commande. Chaque commande peut contenir des options et des arguments. La liste des commandes est décrite dans les sections suivantes.

10.5.1 help

Cette commande propose une aide contextuelle pour les commandes liées à l'exécutable Gestionnaire de schéma.

Syntaxe

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

Où `[command]` est un argument optionnel qui spécifie un nom de commande valide.

Veuillez noter les points suivants :

- Vous pouvez invoquer de l'aide en saisissant la commande suivie par **`-h`** ou **`--help`**, par exemple : **`<exec> list-h`**
- Si vous tapez **`-h`** or **`--help`** directement après la commande exécutable et avant une commande, vous recevrez une aide générale (pas d'aide pour la commande), par exemple : **`<exec> -h list`**

Exemple

La commande suivante affiche une aide concernant la commande `list` :

```
xmlschemamanager help list
```


© 2016-2022 Altova GmbH

10.5.2 info

Cette commande affiche des informations détaillées pour chacun des schémas fournis en tant qu'argument de `Schéma`. Cette information inclut le titre, la version, description, l'éditeur et tout schéma soumis et tout schéma référencé, et informe si le schéma a été installé ou non.

Syntaxe

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

- L'argument **`Schéma`** est le nom d'un schéma ou une partie du nom de schéma. (Pour afficher une ID de pack de schéma et des informations détaillées sur son statut d'installation, vous devriez utiliser la commande `list`.)
- Utiliser **`<exec> info -h`** pour afficher l'aide de la commande.

Exemple

La commande suivante affiche l'information sur les derniers schémas **`DocBook-DTD`** et **`NITF`** :

```
xmlschemamanager info doc nitf
```

10.5.3 initialize

Cette commande initialise l'environnement Gestionnaire de schéma. Elle crée un répertoire de cache où les informations concernant tous les schémas sont stockées localement. L'initialisation est réalisée automatiquement la première fois qu'une application schema-cognizant d'Altova est installée. Vous n'aurez pas besoin d'exécuter cette commande dans des circonstances normales, mais vous devrez l'exécuter généralement après la commande **`reset`**.

Syntaxe

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

Options

La commande **`initialize`** accepte les options suivantes :

`--silent, --s`	Afficher uniquement des messages d'erreur. Le réglage par défaut est **`faux`**.
`--verbose, --v`	Afficher des informations détaillées lors de l'exécution. Le réglage par défaut est **`faux`**.
`--help, --h`	Afficher l'aide pour la commande.

Exemple

La commande suivante initialise Gestionnaire de schéma:

```
xmlschemamanager initialize
```

10.5.4 install

Cette commande installe un ou plusieurs schémas.

Syntaxe

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

Pour installer de multiples schémas, ajoutez l'argument **Schéma** de nombreuses fois.

L'argument **Schéma** est l'un des suivants :

- Un identifiant de schéma (avoir un format de **<name>-<version>**, par exemple : **cbcr-2.0**). Pour trouver les identifiants de schémas que vous voulez, exécutez la commande `list`. Vous pouvez aussi utiliser des identifiants abrégés s'ils sont uniques, par exemple **docbook**. Si vous utilisez un identifiant abrégé, alors la dernière version de ce schéma sera installée.
- Le chemin vers un fichier **.altova_xmlschemas** téléchargé depuis le site web d'Altova. Pour information sur ces fichiers, voir *Introduction à SchemaManager : Comment cela fonctionne-t-il ?*.

Options
La commande **install** accepte les options suivantes :

`--silent, --s`	Afficher uniquement des messages d'erreur. Le réglage par défaut est **faux**.
`--verbose, --v`	Afficher des informations détaillées lors de l'exécution. Le réglage par défaut est **faux**.
`--help, --h`	Afficher l'aide pour la commande.

Exemple

La commande suivante installe le schéma CBCR 2.0 (Country-By-Country Reporting) et le dernier DocBook DTD:

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

10.5.5 list

Cette commande recense les schémas sous la gestion de Gestionnaire de schéma. La liste affiche comme suit

- Tous les schémas disponibles
- Les schémas contenant le string dans leur nom soumis comme argument de **Schéma**
- Seuls les schémas installés
- Seuls les schémas qui peuvent être mis à niveau

© 2016-2022 Altova GmbH

Syntaxe

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

Si aucun argument de **Schéma** n'est soumis, alors toutes les schémas disponibles sont recensés. Autrement, les schémas sont recensés par des options soumises (*voir l'exemple ci-dessous*). Notez que vous pouvez soumettre l'argument de **Schéma** de nombreuses fois.

Options
La commande **list** accepte les options suivantes :

`--installed, --i`	Recenser uniquement la liste des schémas installés. Le réglage par défaut est **faux**.
`--upgradeable, --u`	Recenser uniquement les schémas lorsque des mises à niveau (patchs) sont disponibles. Le réglage par défaut est **faux**.
`--help, --h`	Afficher l'aide pour la commande.

Exemples

- Pour exécuter tous les schémas disponibles, exécutez : **xmlschemamanager list**
- Pour recenser les schémas installés, exécutez : **xmlschemamanager list -i**
- Pour recenser tous les schémas qui contiennent soit "doc", soit "nitf" dans leur nom, exécutez **xmlschemamanager list doc** :

10.5.6 reset

Cette commande supprime tous les schémas installés et le répertoire de mise sous cache. Vous réinitialiserez complètement votre environnement de schéma. Une fois avoir exécuté cette commande, vous devrez exécuter la commande initialize, pour pouvoir recréer le répertoire de mise sous cache. En alternative, exécuter la commande **reset** avec l'option **-i**. Pusique **reset -i** restaure l'installation originale du produit, nous vous recommandons que vous exécutiez la commande update après avoir réalisé la réinitialisation et l'initialisation. En alternative, exécutez la commande **reset** avec les options **-i** and **-u**

Syntaxe

```
<exec> reset [options]
```

Options
La commande **reset** accepte les options suivantes :

`--init, --i`	Initialiser Gestionnaire de schéma après le reset. Le réglage par défaut est **faux**.
`--update, --u`	Mettre à jour la liste de schémas disponibles dans le cache. Le réglage par défaut est **faux**.

`--silent, --s`	Afficher uniquement des messages d'erreur. Le réglage par défaut est **faux**.
`--verbose, --v`	Afficher des informations détaillées lors de l'exécution. Le réglage par défaut est **faux**.
`--help, --h`	Afficher l'aide pour la commande.

Exemples

- Pour réinitialiser Gestionnaire de schéma, exécuter : **xmlschemamanager reset**
- Pour réinitialiser Gestionnaire de schéma et l'initialiser, exécutez : **xmlschemamanager reset -i**
- Pour réinitialiser Gestionnaire de schéma, initialiser-le et mettez à jour sa liste de schéma, exécutez : **xmlschemamanager reset -i-u**

10.5.7 uninstall

Cette commande désinstalle un ou plusieurs schémas. Par défaut, tout schéma référencé par la taxonomie actuelle sera également désinstallé. Pour désinstaller uniquement le schéma actuel et garder les schémas référencés, définir l'option `--k`.

Syntaxe

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

Pour désinstaller de multiples schémas, ajoutez l'argument **Schéma** de nombreuses fois.

L'argument **Schéma** est l'un des suivants :

- Un identifiant de schéma (avoir un format de **<name>-<version>**, par exemple : **cbcr-2.0**). Pour trouver les identifiants de schéma qui sont installés, exécutez la commande `list -i` . Vous pouvez aussi utiliser un nom de schéma abrégé s'il est unique, par exemple **docbook**. Si vous utilisez un nom abrégé, alors tous les schémas qui contiennent une abréviation dans leur nom seront désinstallés.
- Le chemin vers un fichier **.altova_xmlschemas** téléchargé depuis le site web d'Altova. Pour information sur ces fichiers, voir *Introduction à SchemaManager : Comment cela fonctionne-t-il ?*.

Options
La commande **désinstaller** accepte les options suivantes :

`--keep-references, --k`	Définir cette option pour garder les schémas référencés. Le réglage par défaut est **faux**.
`--silent, --s`	Afficher uniquement des messages d'erreur. Le réglage par défaut est **faux**.
`--verbose, --v`	Afficher des informations détaillées lors de l'exécution. Le réglage par défaut est **faux**.
`--help, --h`	Afficher l'aide pour la commande.

© 2016-2022 Altova GmbH

Exemple

La commande suivante désinstalle les schémas CBCR 2.0 et EPUB 2.0 et leurs dépendances :

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

La commande suivante désinstalle la taxonomie **eba-2.10** mais pas les schémas qu'elle référence :

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

10.5.8 update

Cette commande requête la liste des schémas disponibles depuis l'emplacement de stockage en ligne et met à jour le répertoire de mise sous cache local. Vous devriez exécuter cette commande sauf si vous avez réalisé un reset et initialize.

Syntaxe

```
<exec> update [options]
```

Options

La commande **mise à jour** accepte les options suivantes :

`--silent, --s`	Afficher uniquement des messages d'erreur. Le réglage par défaut est **faux**.
`--verbose, --v`	Afficher des informations détaillées lors de l'exécution. Le réglage par défaut est **faux**.
`--help, --h`	Afficher l'aide pour la commande.

Exemple

La commande suivante met à jour le cache local avec la liste des derniers schémas :

```
xmlschemamanager update
```

10.5.9 upgrade

Cette commande met à niveau toutes les schémas installés qui peuvent être mis à niveau à la dernière version *patchée* disponible. Vous pouvez identifier des schémas à mettre à niveau en exécutant la commande list -u.

Note : La commande **mettre à niveau** supprime une Police dépréciée si aucune version plus récente n'est disponible.

Syntaxe

```
<exec> upgrade [options]
```

Options
La commande **mise à niveau** accepte les options suivantes :

`--silent, --s`	Afficher uniquement des messages d'erreur. Le réglage par défaut est **faux**.
`--verbose, --v`	Afficher des informations détaillées lors de l'exécution. Le réglage par défaut est **faux**.
`--help, --h`	Afficher l'aide pour la commande.

© 2016-2022 Altova GmbH

11 Authentic Desktop dans Visual Studio

Authentic Desktop peut être intégré dans les versions 2012/2013/2015/2017/2019/2022 de Microsoft Visual Studio IDE. Cela permet d'allier le meilleur des deux mondes, à savoir, l'intégration des capacités d'édition XML avec l'environnement de développement avancé de Visual Studio.

Dans cette section, nous décrivons :

- Le processus d'installation général et l'intégration du plugin Authentic Desktop dans Visual Studio.
- Les différences entre la version Visual Studio et la version autonome.

11.1 Installer le plugin Authentic Desktop pour Visual Studio

Pour installer le Plug-in Authentic Desktop pour Visual Studio, suivez les étapes suivantes :

1. Installer Microsoft Visual Studio 2012/2013/2015/2017/2019/2022. Veuillez noter qu'à partir de Visual Studio 2022, Visual Studio est mis à disposition uniquement comme application 64-bit.
2. Installez Authentic Desktop. Si vous avez installé Visual Studio 2022+, alors vous devez installer la version 64-bit de Authentic Desktop.
3. Téléchargez et exécutez le pack d'intégration Authentic Desktop pour Microsoft Visual Studio. Ce pack est disponible sur la page de téléchargement de Authentic Desktop sur www.altova.com

Une fois que le pack d'intégration a été installé, vous pourrez utiliser Authentic Desktop dans l'environnement de Visual Studio.

Important

Vous devez utiliser le pack d'intégration correspondant à votre version de Authentic Desktop (la version actuelle est 2023).

© 2016-2022 Altova GmbH

11.2 Différences avec la Version standalone

Cette section recense les différences entre les versions de Visual Studio et les versions differ de Authentic Desktop.

Assistants à la saisie (fenêtres Outils dans Visual Studio)

Les assistants à la saisie de <%DESK%> sont disponibles en tant que fenêtres Outils dans Visual Studio. Veuillez noter les points suivants. (Pour une description des assistants à la saisie et de la GUI <% AUTH-DESK%>, voir la section, GUI et environnement.)

- Vous pouvez glisser des fenêtres d'assistants à la saisie à n'importe quel endroit de l'environnement de développement.
- Cliquer avec la touche de droite sur un onglet d'assistant à la saisie pour personnaliser encore plus votre interface. Les options de configuration de l'assistant à la saisie sont les suivantes : ancrable, dissimuler, flottant, et auto-dissimuler.

Commandes Authentic Desktop en tant que commandes Visual Studio

Certaines commandes Authentic Desktop sont présentes en tant que commandes Visual Studio dans la GUI Visual Studio. Il s'agit de :

- ***Annuler, Rétablir :*** Ces commandes Visual Studio touchent toutes les actions dans l'environnement de développement Visual Studio.
- ***Projets :*** Les projets Authentic Desktop sont gérés en tant que projets Visual Studio.
- ***Personnaliser les barres d'outils, Personnaliser les commandes :*** Les onglets Barres d'outils et Commandes contenues dans le dialogue Personnaliser (**Outils | Personnaliser**) contiennent les commandes Visual Studio et les commandes Authentic Desktop.
- ***Modes :*** Dans le menu **Mode**, le sous-menu **Authentic Tool Windows**contient des options pour activer les fenêtres de l'assistant de saisie et les autres barres latérales, et pour activer et désactiver certains guides d'édition
- ***Aide Authentic :*** Ce menu Authentic Desktop apparaît en tant que sous-menu dans le menu **Help** de Visual Studio.

Note : dans Visual Studio 2019 et plus élevé, la fonctionnalité de Authentic Desktop peut être accédée dans le menu **Extensions** de Visual Studio. Dans des versions antérieures de Visual Studio, les fonctions de Authentic Desktop sont disponibles dans des menus de niveau supérieur de Visual Studio.

Note : les commandes de la barre d'outils me sont pas prises en charge. Si vous avez défini une commande de barre d'outils dans Authentic Desktop qui exécute une commande ou un script, alors cette commande de barre d'outils ne sera pas disponible dans le plug-in.

Informations complémentaires

Veuillez trouver ci-dessous quelques notes et astuces supplémentaires :

- Pour éditer un fichier XML avec le plugin Authentic, choisir la commande **Fichier | Ouvrir**. Ensuite, dans le dialogue Open file, choisir si vous souhaitez ouvrir une ressource globale Authentic ou un fichier Authentic via une URL.

12 Authentic Desktop dans Eclipse

Eclipse est un framework open source qui intègre différents types d'applications fournies sous la forme de plugins. Le Package d'intégration Authentic Desktop pour Eclipse vous permet d'intégrer et accéder la fonctionnalité de Authentic Desktop dans la Plateforme Eclipse pour Windows. Les versions prises en charge d'Eclipse sont : 2022-09, 2022-06, 2022-03, 2021-12.

Dans cette section, nous allons décrire les points suivants :

- Comment installer le Package d'intégration pour Eclipse et intégrer Authentic Desktop dans Eclipse
- Perspective Authentic Desktop dans Eclipse
- Autres Points d'entrée Authentic Desktop dans Eclipse

© 2016-2022 Altova GmbH

12.1 Installer le Package d'intégration pour Eclipse

Prérequis

- Eclipse 2022-09, 2022-06, 2022-03, 2021-12 (http://www.eclipse.org) exploitation 64-bit.
- Java Runtime Environment (JRE) ou Java Development Kit (JDK) pour la plateforme 64-bit.
- Authentic Desktop 64-bit.

Note : tous les prérequis cités ci-dessus doivent être dotés de la plateforme 64-bit. L'intégration avec d'autres plateformes 32-bit n'est plus prise en charge, bien qu'elle puisse encore fonctionner.

Une fois que les pérequis ci-dessus sont en place, vous pouvez installer le Package d'intégration (64-bit) Authentic Desktop pour intégrer Authentic Desktop dans Eclipse. L'intégration peut être effectuée au cours de l'installation du Package d'intégration ou manuellement depuis Eclipse une fois que le Package d'intégration a été installé. Le Package d'intégration Authentic Desktop est disponible pour être téléchargé à l'adresse https://www.altova.com/components/download.

Note : Eclipse doit être fermé pendant que vous installez ou désinstallez le Package d'intégrationAuthentic Desktop.

Intégrez Authentic Desktop lors de l'installation du Package d'intégration

Vous pouvez intégrez Authentic Desktop dans Eclipse lors de l'installation du Package d'intégration Authentic Desktop. Pour ce faire, suivez les étapes suivantes :

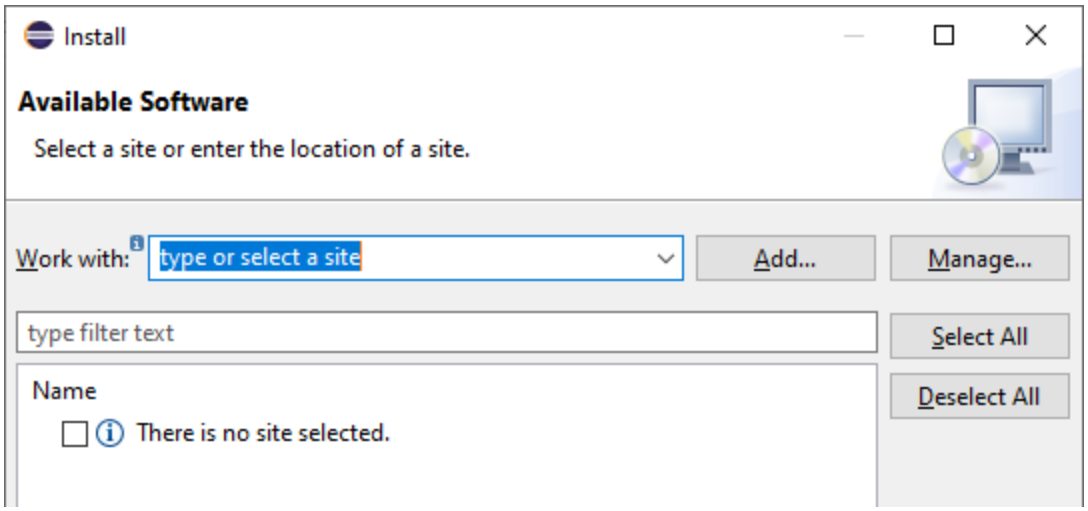
1. Exécutez le Package d'intégration Authentic Desktop pour lancer l'assistant d'installation.
2. Suivez les étapes initiales de l'installation avec l'assistant d'installation.
3. Dans l'étape d'intégration, sélectionnez *Permettre à l'assistant d'intégrer Altova Authentic Desktop plug-in dans Eclipse*, et cherchez le répertoire dans lequel le fichier exécutable pour Eclipse (`eclipse.exe`) se trouve.
4. Cliquez sur **Suivant** et terminez l'installation.

La perspective Authentic Desktop et les menus seront disponibles dans Eclipse la prochaine fois que vous le démarrez.

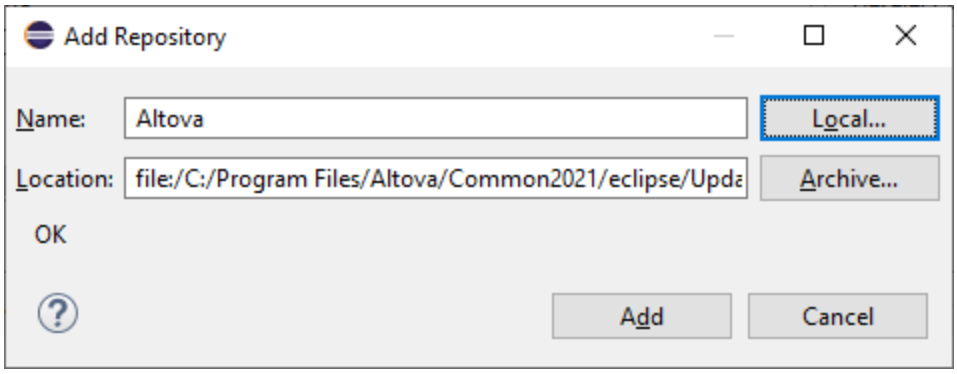
Intégrer Authentic Desktop dans Eclipse manuellement

Une fois que vous avez installé le Package d'intégration Authentic Desktop, vous pouvez intégrer manuellement Authentic Desktop dans Eclipse comme suit:

1. Dans Eclipse, sélectionnez la commande de menu **Aide | Installer nouveau logiciel**.
2. Dans la boîte de dialogue Installer, cliquez sur **Ajouter**.



3. Dans la boîte de dialogue Ajouter Référentiel, cliquez sur **Local**. Recherchez le dossier `C:\Program Files\Altova\Common2023\eclipse\UpdateSite` et sélectionnez-le. Donner un nom au site (comme « Altova »).



4. Répétez les étapes 2 à 3 ci-dessus, en sélectionnant cette fois-ci le dossier `C:\Program Files\Altova\<% APPNAMESHORT%>\eclipse\UpdateSite` et en donnant un nom comme « Altova » Authentic Desktop".
5. Dans la boîte de dialogue Installer, sélectionnez *Uniquement sites locaux*. Puis, sélectionnez le dossier « catégorie Altova » et cliquez sur **Suivant**.
6. Revérifiez les éléments à installer puis cliquez sur **Suivant** pour procéder.
7. Pour accepter le contrat de licence, sélectionnez la case à cocher respective.
8. Cliquez sur **Terminer** et terminez l'installation.

Note : Si vous avez des problèmes avec le plug-in (icônes manquantes, par exemple), lancez Eclipse depuis la ligne de commande avec l'indicateur `-clean`.

© 2016-2022 Altova GmbH

12.2 Perspective Authentic Desktop dans Eclipse

Dans Eclipse, une perspective est un affichage GUI qui est configuré avec la fonctionnalité d'une application spécifique. Lorsque Authentic Desktop est intégré dans Eclipse, une perspective Authentic Desktop par défaut est créée automatiquement. Cette perspective est une GUI qui inclut les éléments de GUI de Authentic Desktop : ses modes d'édition, ses menus, ses assistants à la saisie, et d'autres barres latérales.

Lorsqu'un fichier dont le type de fichier est associé avec Authentic Desktop est ouvert (`.xml`) (`.xml`, par exemple), ce fichier peut être édité dans la perspective Authentic Desktop. De même, un fichier d'un autre type de fichier peut être ouvert dans une autre perspective dans Eclipse. En outre, pour tout fichier actif, vous pouvez sauter la perspective, vous permettant donc d'éditer ou de traiter ce fichier dans un autre environnement.

Deux avantages principaux se présentent donc pour les perspectives :

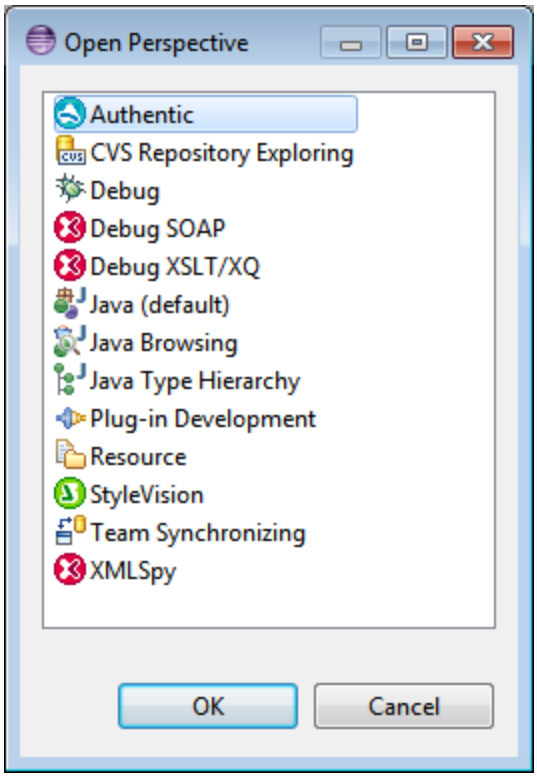
1. la possibilité de changer rapidement l'environnement de travail du fichier actif et
2. La possibilité de passer entre les fichiers sans devoir ouvrir un nouvel environnement de développement (l'environnement associé est disponible dans une perspective)

Le travail avec la perspective Authentic Desktop implique les éléments suivants :

- Passer à la perspective Authentic Desktop.
- Définir les préférences pour la perspective Authentic Desktop.
- Personnaliser la perspective Authentic Desktop.

Passer à la perspective Authentic Desktop

Dans Eclipse, sélectionnez la commande **Window | Perspective | Open Perspective | Other**. Dans le dialogue qui s'ouvre (*capture d'écran ci-dessous*), sélectionnez **Authentic Desktop** et cliquez sur **Ouvrir**.



La fenêtre vide ou le document actif aura maintenant la perspective Authentic Desktop. Voici comment l'utilisateur change de perspective par le menu. Pour accéder à une perspective plus rapidement depuis une autre perspective, la perspective requise peut être recensée dans le sous-menu **Open Perspective** au-dessus de l'item **Other**. Ce paramètre est dans le dialogue de personnalisation (*voir plus bas ci-dessous*).

Les perspectives peuvent aussi être changées lorsqu'un fichier est ouvert ou est rendu actif. La perspective de l'application associée avec un type de fichier d'un fichier sera ouvert automatiquement lorsque ce fichier est ouvert pour la première fois. Avant de changer de perspective, un dialogue apparaît vous demandant si vous souhaitez associer automatiquement la perspective par défaut avec ce type de fichier. Vérifiez l'option *Do Not Ask Again* si vous souhaitez associer la perspective avec un filetype sans devoir y être invité chaque fois qu'un fichier de ce filetype est ouvert, puis cliquez sur **OK**.

Les préférences pour la perspective Authentic Desktop

Les préférences d'une perspective comprennent : (i) un paramètre pour changer automatiquement la perspective lorsqu'un fichier d'un type de fichier associé est ouvert (*voir ci-dessus*), et (ii) des options pour inclure ou exclure des barres d'outils Authentic Desktop individuelles et (iii) accéder aux options Authentic Desktop. Pour accéder au dialogue Préférences, sélectionnez la commande **Window | Preferences**. Dans la liste des perspectives du panneau de gauche, sélectionnez Authentic Desktop, puis sélectionnez les préférences requises. Terminez en cliquant sur **OK**.

Personnalisez la perspective Authentic Desktop

Les options de personnalisation vous permettent de déterminer quels raccourcis et commandes sont inclus dans la perspective. Pour accéder au dialogue Personnaliser la Perspective, transformez la perspective en perspective active et sélectionnez la commande **Window | Perspective | Customize Perspective**.

- Dans les onglets *Toolbar Visibility* et *Menu Visibility*, vous pourrez spécifier quelles barres d'outils et

 © 2016-2022 Altova GmbH

menu doivent être affichés.
- Dans l'onglet *Action Set Availability*, vous pouvez ajouter des groupes de commandes à leurs menu parents et à la barre d'outils. Si vous souhaitez activer un groupe de commande, cochez sa case.
- Dans l'onglet *Shortcuts* du dialogue Customize Perspective, vous pouvez configurer des raccourcis pour les sous-menus. Sélectionnez le sous-menu dans la liste déroulante des sous-menus. Puis sélectionnez une catégorie de raccourcis, et cochez les raccourcis que vous souhaitez inclure à la perspective.

Cliquer sur **Appliquer et Fermer** pour terminer la personnalisation et pour que les modifications prennent effet.

12.3 Autres Points d'entrée Authentic Desktop dans Eclipse

Outre la perspective Authentic Desktop, deux autres points d'entrée dans Eclipse peuvent être utilisés pour accéder la fonctionnalité Authentic Desktop :

- menu Authentic Desktop
- barre d'outils Authentic Desktop

Menu Authentic Desktop dans Eclipse

Le menu **Authentic Desktop** d'Eclipse contient les commandes Authentic Desktop qui fournissent la fonctionnalité Authentic Desktop. Ces commandes apparaissent dans de nombreux menus de la version autonome de Authentic Desktop.

Barre d'outils Authentic Desktop dans Eclipse

La barre d'outils Authentic Desktop dans Eclipse (*voir la capture d'écran ci-dessous*) contient deux boutons.

Ces boutons font la chose suivante :

- Ouvrir l'Aide Authentic Desktop
- Fournit un accès aux commandes Authentic Desktop (comme alternative pour y accéder depuis le menu **Authentic Desktop**, *voir ci-dessus*).

Note : les commandes de la barre d'outils me sont pas prises en charge. Si vous avez défini une commande de barre d'outils dans Authentic Desktop qui exécute une commande ou un script, alors cette commande de barre d'outils ne sera pas disponible dans le plug-in.

© 2016-2022 Altova GmbH

13 Commandes de menu

Cette section contient une description de toutes les commandes de menu de Authentic Desktop. Vous trouverez les commandes standard de Windows comme ((**Ouvrir**, **Enregistrer**, **Couper**, **Copier** et **Coller**) dans les menus Fichier et Éditer.

Ci-dessous, vous trouverez une liste des menus de Authentic Desktop

- Menu Fichier
- Menu Édition
- Menu Projet
- Menu XML
- Menu XSL/XQuery
- Menu Authentic
- Menu Affichage
- Menu Navigateur
- Menu Outils
- Menu Fenêtre
- Menu Aide

Cette section contient également une description des commandes qui peuvent être utilisées via la ligne de commande.

13.1 Menu Fichier

Le menu **Fichier** contient les commandes pour les opérations de fichier, classées comme dans la plupart des applications Windows. Il s'agit de :

- Nouveau
- Open
- Reload
- Encodage
- Fermer, Tout fermer, Tout fermer sauf actifs
- Enregistrer, Enregistrer sous, Enregistrer tout
- Envoyer par e-mail
- Imprimer
- Aperçu d'impression
- Paramètres d'impression
- Fichiers récents
- Quitter

13.1.1 Nouveau

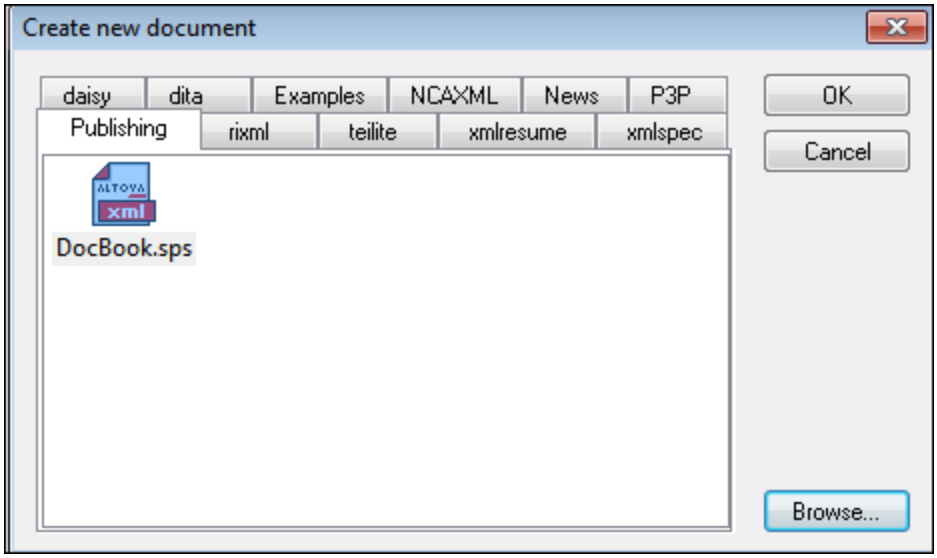
Icône et raccourci

Icône :	
Raccourci :	**Ctrl+N**

Description

Cette commande vous permet d'ouvrir un nouveau modèle de document XML dans Authentic View. Le modèle de document XML est basé sur une StyleVision Power Stylesheet (fichier .sps), et peut être ouvert en choisissant le StyleVision Power Stylesheet (fichier SPS) dans le dialogue Créer nouveau document (*capture d'écran ci-dessous*). En sélectionnant une SPS et en cliquant sur **OK**, le modèle de document XML défini pour ce fichier SPS est ouvert dans Authentic View.

© 2016-2022 Altova GmbH



Le dialogue Créer nouveau document propose une variété de modèles de documents XML qui sont basés sur des DTD ou des schémas populaires. En alternative, vous pouvez aussi chercher un fichier SPS taillé sur mesure auquel un fichier XML modèle a été attribué. Les fichiers SPS sont créés en utilisant Altova StyleVision, une application qui permet de concevoir des modèles de document XML basés sur un schéma DTD ou XML. Une fois avoir conçu la SPS exigée dans StyleVision, un fichier XML est attribué (dans StyleVision) en tant que fichier XML modèle dans la SPS. Les données dans ce fichier XML fournissent les données de démarrage du nouveau modèle de contenu qui est ouvert dans le Authentic View de Authentic Desktop.

Le nouveau modèle de document XML aura donc des propriétés de présentation de document définies dans la SPS et les données du fichier XML qui ont été choisies en tant que fichier XML de modèle. L'utilisateur Authentic View peut maintenant éditer les modèles de document XML dans une interface graphique WYSIWYG, et l'enregistrer en tant que modèle XML.

13.1.2 Ouvrir

Icône et raccourci

Icône :	
Raccourci :	**Ctrl+O**

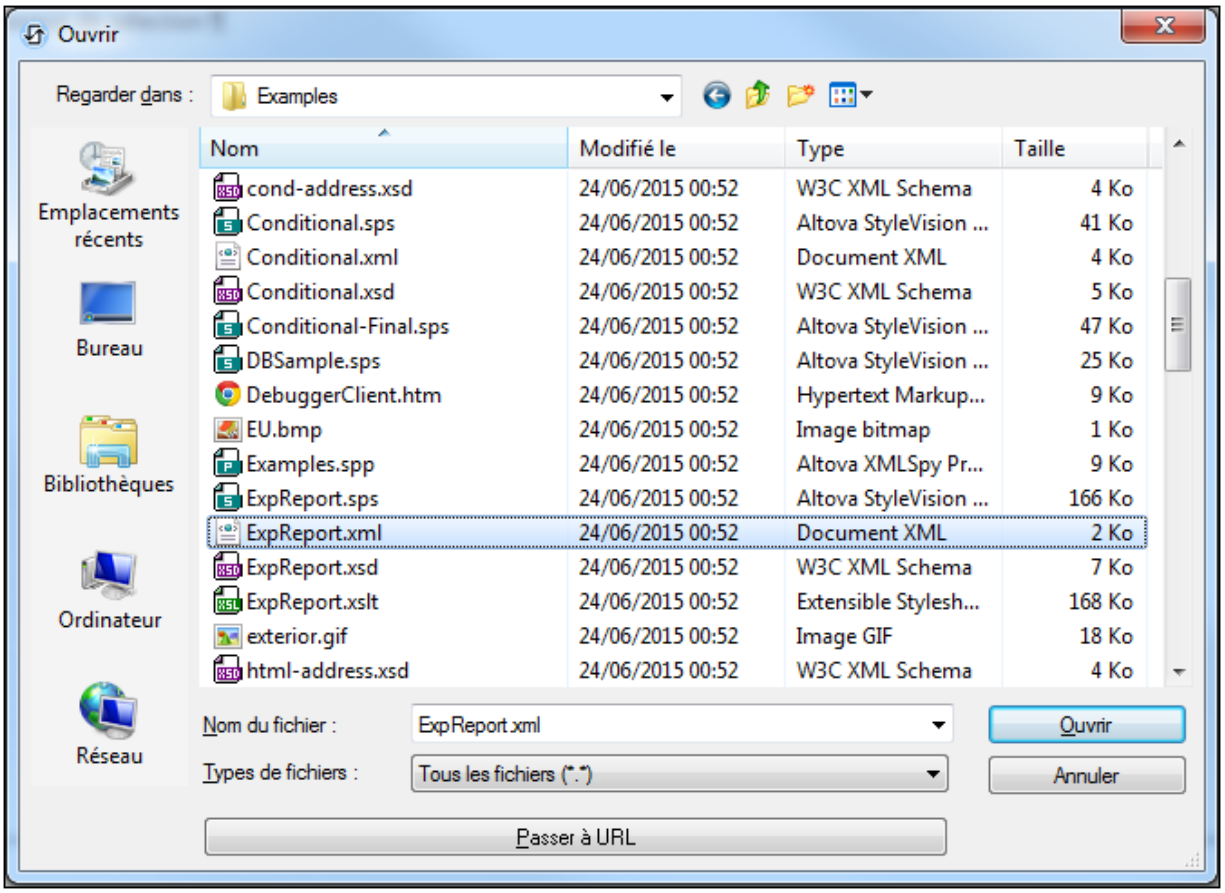
Description

La commande **Ouvrir** déclenche l'ouverture du dialogue Windows Ouvrir que vous connaissez bien et vous permet d'ouvrir tout document lié à XML ou tout document texte. Dans le dialogue Ouvrir, vous pouvez sélectionner plus d'un fichier à ouvrir. Utiliser la liste de choix Types de fichier pour limiter le type de fichiers affichés dans le dialogue. (La liste des types de fichiers disponibles peut être configurée dans la section Types de fichier du dialogue Options (**Outils | Options**).) Lorsqu'un fichier XML est ouvert, sa bonne formation est

vérifiée. Si le fichier n'est pas bien formé, vous obtiendrez une erreur "fichier n'est pas bien formé". Réparez l'erreur et choisir la commande de menu **XML | Vérifier la bonne formation (F7)** pour revérifier. Si vous avez choisi une validation sur ouverture automatique et que le fichier est invalide, vous obtiendrez un message d'erreur. Réparer l'erreur et choisir la commande de menu **XML | Valider XML (F8)** pour revalider.

▼ Sélectionner et enregistrer les fichiers via des URL et des Ressources globales

Dans plusieurs dialogues Fichiers Ouvrir et Fichier Enregistrer, vous pouvez choisir de sélectionner le fichier nécessaire ou d'enregistrer un fichier via une URL ou une Ressource globale (*voir capture d'écran ci-dessous*). Cliquer sur **Passer à URL** ou sur **Passer à Ressources globales** pour vous rendre sur un de ces processus de sélection.

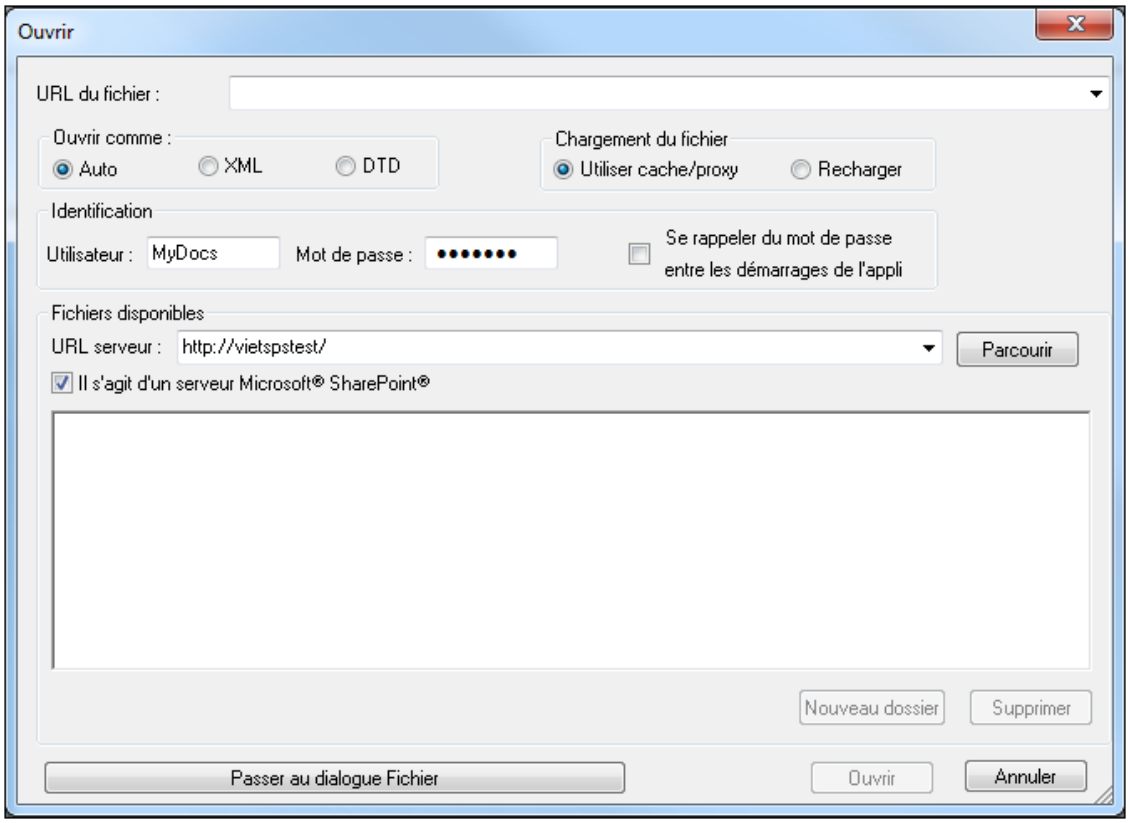


Sélectionner les fichiers via des URL

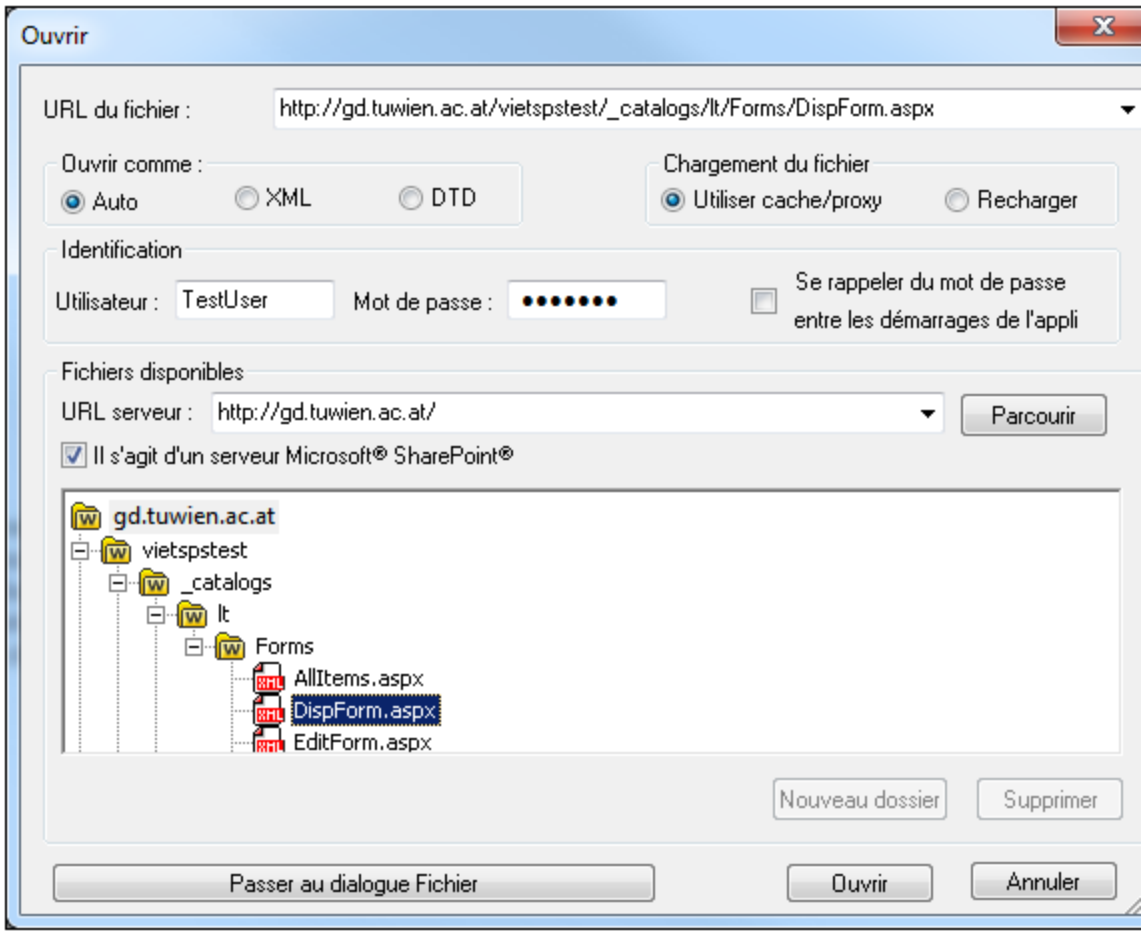
Pour sélectionner un fichier via une URL (pour l'ouvrir ou l'enregistrer), procédez comme suit :

1. Cliquer sur la commande **Passer à URL**. Cela vous permet de passer au mode URL du dialogue Ouvrir ou Enregistrer (*la capture d'écran ci-dessous montre le dialogue Ouvrir*).

© 2016-2022 Altova GmbH



2. Saisir l'URL à laquelle vous souhaitez accéder dans le champ *URL de serveur* (*capture d'écran ci-dessus*). Si le serveur est un Microsoft® SharePoint® Server, cochez la case correspondante *Microsoft® SharePoint® Server*. Voir les Notes Microsoft® SharePoint® Server ci-dessous pour plus d'informations concernant le travail avec des fichiers sur ce type de serveur.
3. Si le serveur est protégé par un mot de passe, veuillez saisir votre ID d'utilisateur et votre mot de passe dans les champs *Utilisateur* et *Mot de passe*.
4. Cliquer sur **Parcourir** pour consulter et naviguer dans la structure du répertoire du serveur.
5. Dans l'arborescence du dossier, recherchez le fichier que vous souhaitez charger puis cliquer dessus.



L'URL du fichier apparaît dans le champ URL de fichier (*voir capture d'écran ci-dessus*). Ce n'est qu'à ce moment que le bouton **Ouvrir** ou **Enregistrer** devient actif.

6. Cliquer sur **Ouvrir** pour charger le fichier ou **Enregistrer** pour l'enregistrer.

Notez les points suivants :

- La fonction Parcourir est uniquement disponible sur les serveurs qui prennent en charge WebDAV et sur les Microsoft SharePoint Servers. Les protocoles pris en charge sont FTP, HTTP et HTTPS.
- Pour vous donner un plus grand contrôle du processus de chargement lors de l'ouverture d'un fichier, vous pouvez choisir de charger le fichier par le biais du cache local ou d'un serveur proxy (ce qui accélère considérablement le processus si le fichier a été chargé avant). En alternative, vous pouvez choisir de recharger le fichier si vous travaillez par exemple avec un système de publication électronique ou de base de données ; dans ce cas, choisir l'option **Recharger**.

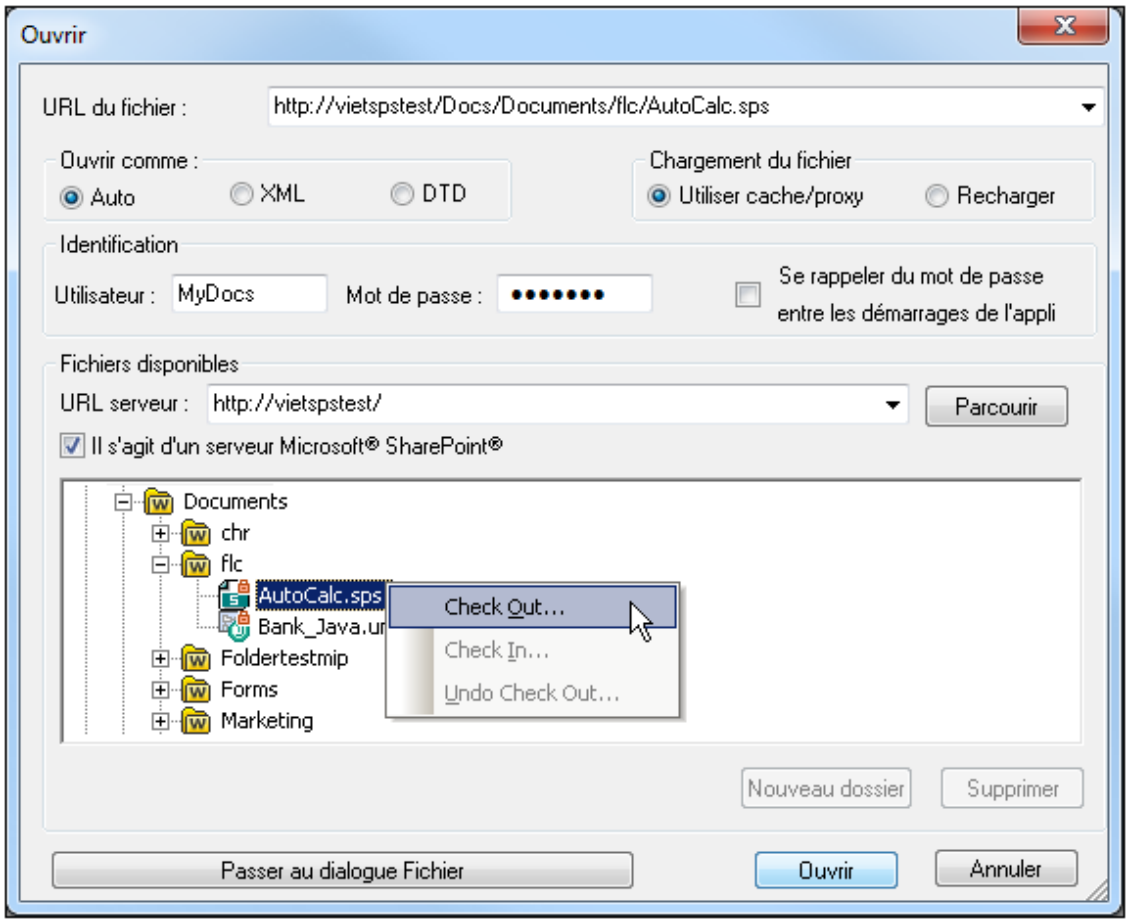
.

▼ Notes Microsoft® SharePoint® Server

© 2016-2022 Altova GmbH

Noter les points suivants concernant les fichiers sur les Microsoft® SharePoint® Servers :

- Dans la structure du répertoire qui apparaît dans le panneau des Fichiers disponibles (*capture d'écran ci-dessous*), les icônes de fichier ont des symboles qui indiquent le statut de d'archivage/récupération des fichiers.



Cliquer avec la touche de droite sur un fichier, pour ouvrir un menu contextuel contenant les commandes disponibles pour ce fichier (*capture d'écran ci-dessus*).

- Les icônes de fichier sont présentées ci-dessous :

	Archivé. Disponible pour une récupération.
	Récupéré par un autre utilisateur. N'est pas disponible pour la récupération.
	Récupéré localement. Peut être édité et archivé.

- Après avoir effectué une récupération de fichier, vous pouvez l'éditer dans votre application Altova et l'enregistrer en utilisant **Fichier | Enregistrer (Ctrl+S)**.
- Vous pouvez faire un archivage du fichier édité par le biais du menu contextuel dans le dialogue Ouvrir URL (*voir capture d'écran ci-dessus*) ou via le menu contextuel qui s'ouvre sur un clic avec

le bouton de droite de la souris sur l'onglet de fichier dans la fenêtre principale de votre application (*capture d'écran ci-dessous*).

- Lorsqu'un fichier est récupéré par un autre utilisateur, il n'est plus disponible pour une récupération.
- Lorsqu'un fichier est récupéré localement par vous, vous pouvez annuler la récupération avec la commande Undo Check-Out (Annuler récupération) dans le menu contextuel. En conséquence, le fichier sera retourné sans aucun changement sur le serveur.
- Si vous procédez à une récupération d'un fichier dans une application Altova, vous ne pourrez pas faire de récupération dans une autre application Altova. Le fichier sera considéré comme s'il avait déjà été récupéré par vous. Les commandes disponibles à ce moment dans toute application Altova prenant en charge Microsoft® SharePoint® Server sera : **Check In (archiver)** et **Undo Check Out (Annuler récupération)**.

▼ Ouvrir et enregistrer des fichiers via Ressources globales

Pour ouvrir ou enregistrer un fichier via des ressources globales, cliquer sur **Passer à la Ressource globale**. Un dialogue s'ouvrira dans lequel vous pouvez sélectionner la ressource globale. Ces dialogues sont décrits dans la section Utiliser les Ressources globales. Pour une description générale des Ressources globales, consulter la section Ressources globales dans cette documentation.

13.1.3 Recharger

Icône

Icône :	

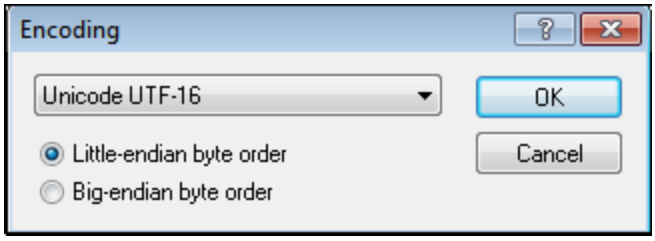
Description

Recharge tous les documents ouverts modifiés en dehors de Authentic Desktop. Si un ou plusieurs documents ont été modifiés en-dehors de Authentic Desktop, une invitation apparaît vous demandant si vous souhaitez recharger le/s document/s modifié/s. Si vous choisissez de recharger, toutes les modifications effectuées sur le fichier depuis le dernier enregistrement seront perdues.

13.1.4 Encodage

La commande **Encodage** vous permet de : (i) consulter l'encodage actuel du document actif (XML ou non XML), et (ii) sélectionner un encodage différent avec lequel le document actif sera enregistré la prochaine fois.

© 2016-2022 Altova GmbH



Dans les documents XML, si vous sélectionnez un encodage différent que celui utilisé actuellement, l'attribut d'encodage dans la déclaration XML sera modifié en conséquence. Pour les encodages de caractère two-byte et four-byte (UTF-16, UCS-2 et UCS-4) , vous pouvez également spécifier l'ordre des octets à utiliser pour le fichier. Un autre moyen de changer l'encodage d'un document XML est d'éditer l'attribut d'encodage de la déclaration XML du document directement. Les encodages par défaut pour des documents XML et non XML nouveaux et existants peuvent être définis dans la section d'encodage du dialogue des Options.

Note : Lors de l'enregistrement d'un document, Authentic Desktop vérifie automatiquement les spécifications d'encodage et vous permet de choisir l'encodage approprié par le biais du dialogue Encodage. Si votre document contient des caractères qui ne peuvent pas être représentés dans l'encodage sélectionné et que vous essayez d'enregistrer le fichier, vous recevrez un avertissement à cet effet.

13.1.5 Fermer, Tout fermer, Tout fermer sauf actifs

Fermer

La commande **Fermer** ferme la fenêtre de document active. Si le fichier a été modifié (indiqué par un astérisque * placé après le nom du fichier dans la barre de titre), vous serez invité à enregistrer le fichier d'abord.

Tout fermer

La commande **Tout fermer** ferme toutes les fenêtres de document ouvertes. Si un document a été modifié (indiqué par un astérisque * placé après le nom du fichier dans la barre de titre), vous serez invité à enregistrer le fichier d'abord.

Tout fermer sauf actifs

La commande **Tout fermer sauf actifs** ferme toutes les fenêtres de document ouvertes sauf la fenêtre de document active. Si un document a été modifié (indiqué par un astérisque * placé après le nom du fichier dans la barre de titre), vous serez invité à enregistrer le fichier d'abord.

13.1.6 Enregistrer, enregistrer sous, Enregistrer tout

Icônes et raccourcis

Commande	*Icône*	*Raccourci*

Enregistrer		**Ctrl+S**
Tout enregistrer		

Enregistrer

La **commande Enregistrer (Ctrl+S)** enregistre le contenu du document actif sous le fichier à partir duquel il a été ouvert. Lors de l'enregistrement d'un document, la bonne formation du fichier est vérifiée automatiquement. Le fichier sera également validé automatiquement si cette option a été configurée dans la section Fichier du dialogue Options (**Outils | Options**). La déclaration XML est également vérifiée en termes de spécifications d'encodage, et cet encodage est appliqué au document lorsque le fichier est enregistré.

Enregistrer sous

La commande **Enregistrer sous** déclenche l'ouverture du dialogue Windows "Enregistrer sous" que vous connaissez bien, dans lequel vous pouvez saisir le nom et l'emplacement du fichier actif que vous souhaitez enregistrer sous un nom. Les mêmes vérifications que pour la commande **Enregistrer** sont effectuées.

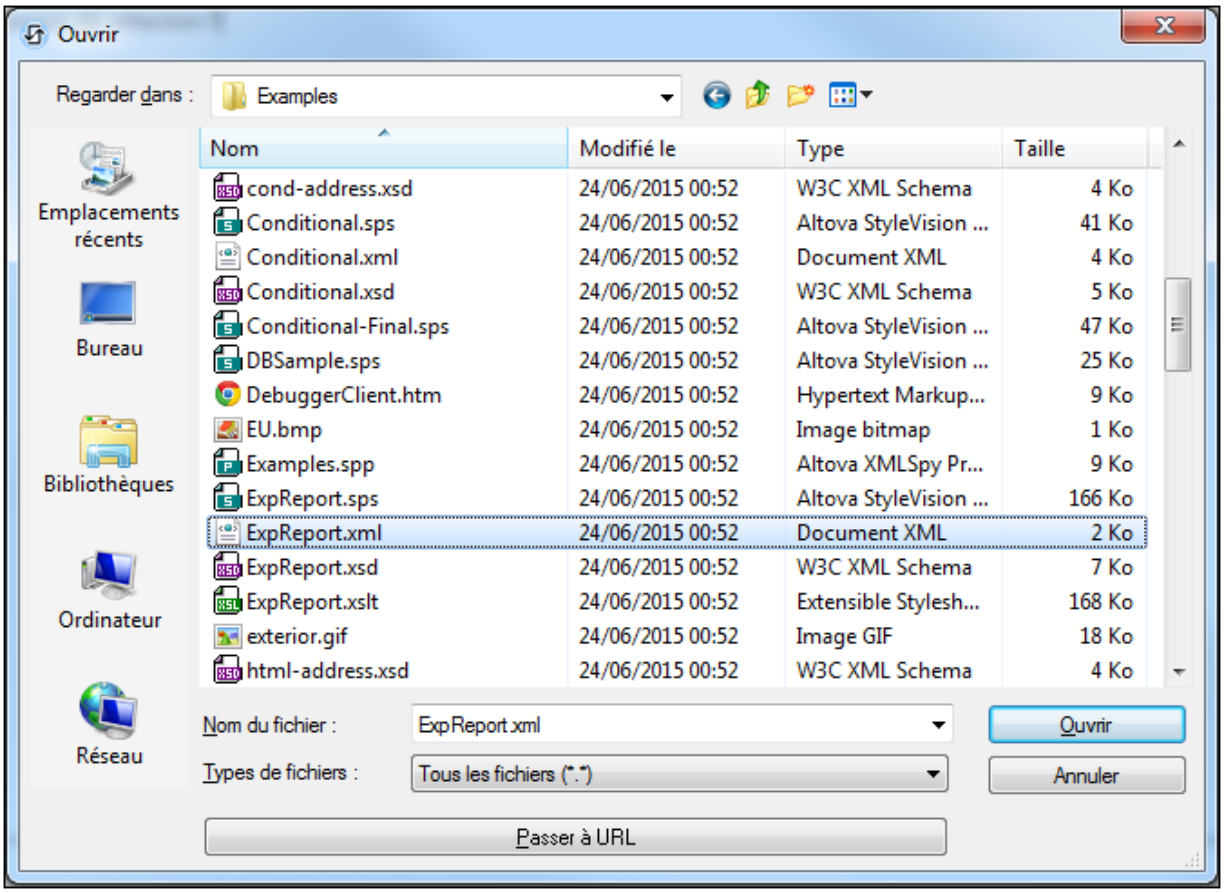
Tout enregistrer

La commande **Tout enregistrer** enregistre toutes les modifications qui ont été réalisées sur tout document ouvert. La commande est utile si vous éditez plusieurs documents simultanément. Si un document n'a pas été enregistré avant (par exemple, après avoir été créé récemment), le dialogue Enregistrer sous est présenté pour ce document.

▼ Sélectionner et enregistrer les fichiers via des URL et des Ressources globales

Dans plusieurs dialogues Fichiers Ouvrir et Fichier Enregistrer, vous pouvez choisir de sélectionner le fichier nécessaire ou d'enregistrer un fichier via une URL ou une Ressource globale (*voir capture d'écran ci-dessous*). Cliquer sur **Passer à URL** ou sur **Passer à Ressources globales** pour vous rendre sur un de ces processus de sélection.

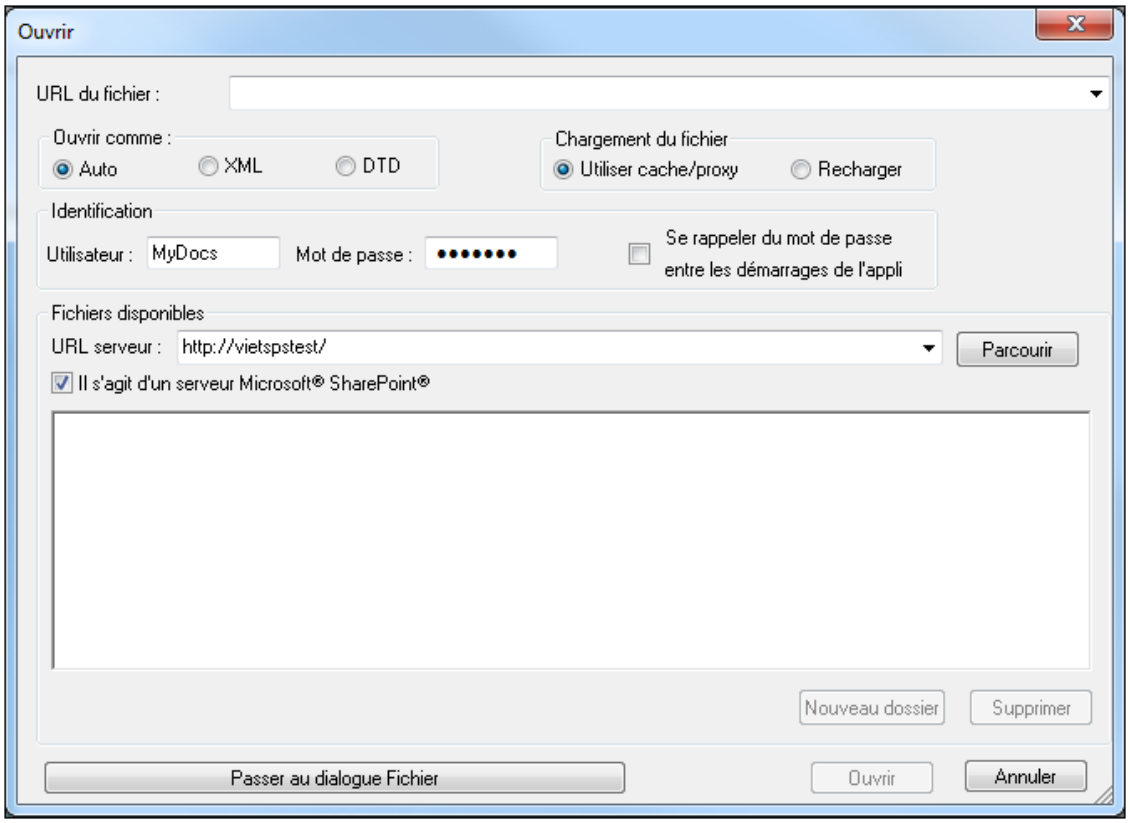
© 2016-2022 Altova GmbH



Sélectionner les fichiers via des URL

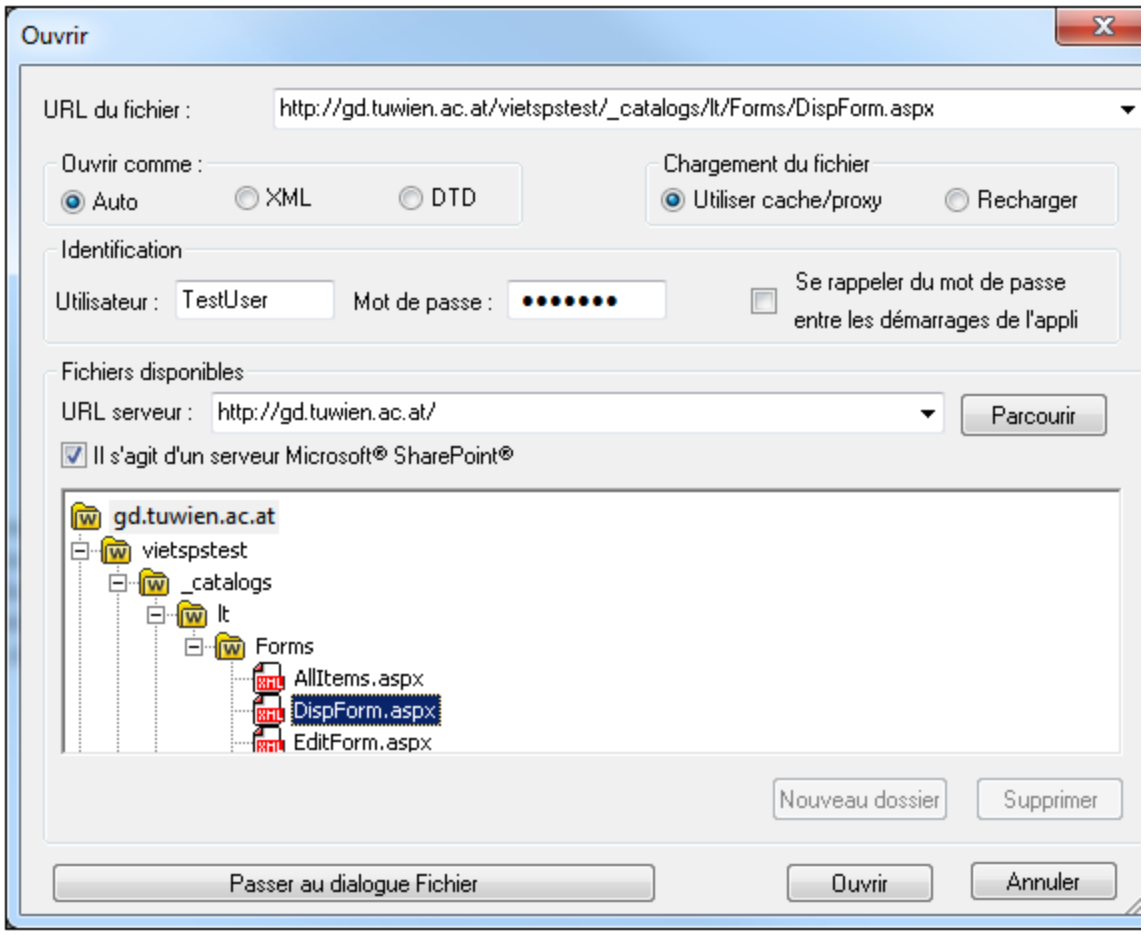
Pour sélectionner un fichier via une URL (pour l'ouvrir ou l'enregistrer), procédez comme suit :

1. Cliquer sur la commande **Passer à URL**. Cela vous permet de passer au mode URL du dialogue Ouvrir ou Enregistrer (*la capture d'écran ci-dessous montre le dialogue Ouvrir*).



2. Saisir l'URL à laquelle vous souhaitez accéder dans le champ *URL de serveur* (*capture d'écran ci-dessus*). Si le serveur est un Microsoft® SharePoint® Server, cochez la case correspondante *Microsoft® SharePoint® Server*. Voir les Notes Microsoft® SharePoint® Server ci-dessous pour plus d'informations concernant le travail avec des fichiers sur ce type de serveur.
3. Si le serveur est protégé par un mot de passe, veuillez saisir votre ID d'utilisateur et votre mot de passe dans les champs *Utilisateur* et *Mot de passe*.
4. Cliquer sur **Parcourir** pour consulter et naviguer dans la structure du répertoire du serveur.
5. Dans l'arborescence du dossier, recherchez le fichier que vous souhaitez charger puis cliquer dessus.

© 2016-2022 Altova GmbH



L'URL du fichier apparaît dans le champ URL de fichier (*voir capture d'écran ci-dessus*). Ce n'est qu'à ce moment que le bouton **Ouvrir** ou **Enregistrer** devient actif.

6. Cliquer sur **Ouvrir** pour charger le fichier ou **Enregistrer** pour l'enregistrer.

Notez les points suivants :

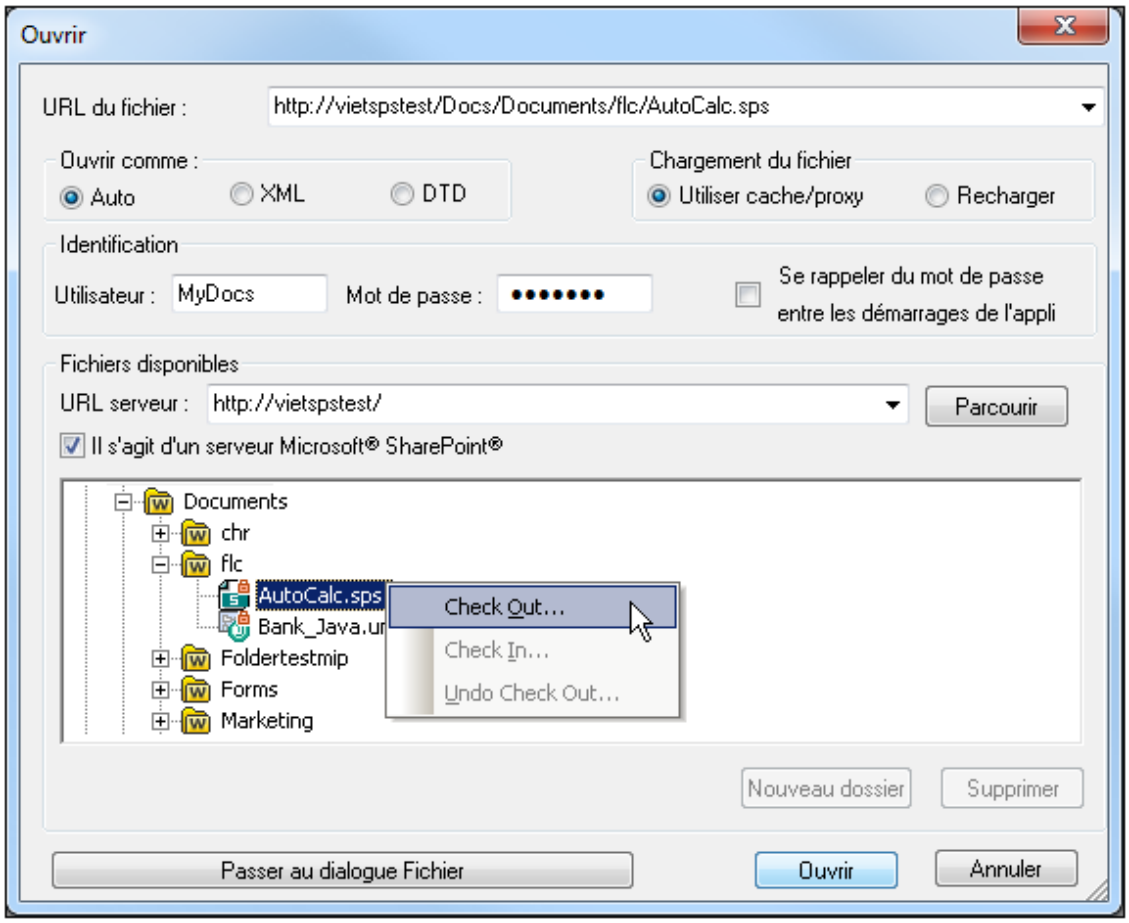
- La fonction Parcourir est uniquement disponible sur les serveurs qui prennent en charge WebDAV et sur les Microsoft SharePoint Servers. Les protocoles pris en charge sont FTP, HTTP et HTTPS.
- Pour vous donner un plus grand contrôle du processus de chargement lors de l'ouverture d'un fichier, vous pouvez choisir de charger le fichier par le biais du cache local ou d'un serveur proxy (ce qui accélère considérablement le processus si le fichier a été chargé avant). En alternative, vous pouvez choisir de recharger le fichier si vous travaillez par exemple avec un système de publication électronique ou de base de données ; dans ce cas, choisir l'option **Recharger**.

.

▼ Notes Microsoft® SharePoint® Server

Noter les points suivants concernant les fichiers sur les Microsoft® SharePoint® Servers :

- Dans la structure du répertoire qui apparaît dans le panneau des Fichiers disponibles (*capture d'écran ci-dessous*), les icônes de fichier ont des symboles qui indiquent le statut de d'archivage/récupération des fichiers.



Cliquer avec la touche de droite sur un fichier, pour ouvrir un menu contextuel contenant les commandes disponibles pour ce fichier (*capture d'écran ci-dessus*).

- Les icônes de fichier sont présentées ci-dessous :

	Archivé. Disponible pour une récupération.
	Récupéré par un autre utilisateur. N'est pas disponible pour la récupération.
	Récupéré localement. Peut être édité et archivé.

- Après avoir effectué une récupération de fichier, vous pouvez l'éditer dans votre application Altova et l'enregistrer en utilisant **Fichier | Enregistrer (Ctrl+S)**.
- Vous pouvez faire un archivage du fichier édité par le biais du menu contextuel dans le dialogue Ouvrir URL (*voir capture d'écran ci-dessus*) ou via le menu contextuel qui s'ouvre sur un clic avec

© 2016-2022 Altova GmbH

le bouton de droite de la souris sur l'onglet de fichier dans la fenêtre principale de votre application (*capture d'écran ci-dessous*).

- Lorsqu'un fichier est récupéré par un autre utilisateur, il n'est plus disponible pour une récupération.
- Lorsqu'un fichier est récupéré localement par vous, vous pouvez annuler la récupération avec la commande Undo Check-Out (Annuler récupération) dans le menu contextuel. En conséquence, le fichier sera retourné sans aucun changement sur le serveur.
- Si vous procédez à une récupération d'un fichier dans une application Altova, vous ne pourrez pas faire de récupération dans une autre application Altova. Le fichier sera considéré comme s'il avait déjà été récupéré par vous. Les commandes disponibles à ce moment dans toute application Altova prenant en charge Microsoft® SharePoint® Server sera : **Check In (archiver)** et **Undo Check Out (Annuler récupération)**.

▼ Ouvrir et enregistrer des fichiers via Ressources globales

Pour ouvrir ou enregistrer un fichier via des ressources globales, cliquer sur **Passer à la Ressource globale**. Un dialogue s'ouvrira dans lequel vous pouvez sélectionner la ressource globale. Ces dialogues sont décrits dans la section Utiliser les Ressources globales. Pour une description générale des Ressources globales, consulter la section Ressources globales dans cette documentation.

13.1.7 Envoyer par e-mail

Icône

Icône :	

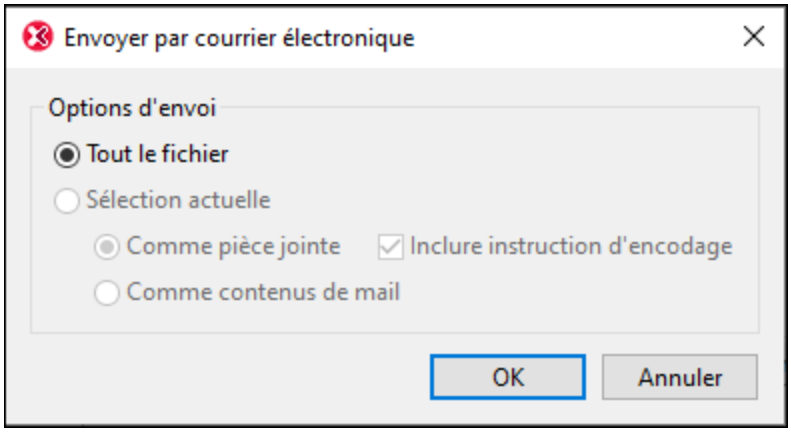
Description

La commande **Envoyer par e-mail** vous permet d'envoyer un document XML actif ou un document PXF en pièce jointe. Vous pouvez également sélectionner de multiples fichiers dans la fenêtre Projet pour les envoyer comme pièces jointes dans l'e-mail. Dépendant du type de document, un document ou une sélection peut être envoyé en pièce jointe, comme contenu ou comme lien. Voir la table ci-dessous pour les détails.

Ce que vous pouvez envoyer	Comment vous pouvez l'envoyer
Document XML actif ou PXF	En tant que pièce jointe d'e-mail
Un ou plusieurs fichiers dans la fenêtre Projet	En tant que pièce jointe d'e-mail

Une ou plusieurs URL dans la fenêtre Projet	En tant que pièce jointe ou lien

- Lorsque la commande **Envoyer par mail** est invoquée dans la sélection du document XML actif, le dialogue Envoyer par e-mail (*capture d'écran ci-dessous*) s'ouvre avec *Whole File* étant la seule option activée ; les autres options sont désactivées. Cliquer sur **OK** pour ouvrir un e-mail avec le fichier sélectionné en pièce jointe.



- Lorsque vous envoyez des fichiers depuis la fenêtre de Projet, un e-mail est ouvert avec les fichiers sélectionnés ajoutés comme pièces jointes.
- Les URL contenues dans la fenêtre Projet peuvent être envoyées en tant que pièce jointe ou en tant que lien. Choisissez l'option que vous souhaitez avoir et cliquez sur **OK**.

13.1.8 Imprimer

Icône et raccourci

Icône :	
Raccourci :	**Ctrl+P**

Description

La commande **Imprimer** ouvre la boîte de dialogue d'impression, dans laquelle vous pouvez sélectionner les options d'impression pour imprimer le document actif actuellement.

13.1.9 Aperçu d'impression, paramètres d'impression

Aperçu d'impression

La commande de l'**Aperçu d'impression** est disponible dans Authentic View. Elle ouvre un aperçu d'impression du document actuellement actif.

© 2016-2022 Altova GmbH

Dans le mode Aperçu d'impression, la barre d'outils Aperçu d'impression en haut à gauche de la fenêtre d'aperçu propose des options d'impression et d'aperçu. Les touches de navigation sont en bas de la fenêtre d'aperçu.

Note : afin de permettre les couleurs et les images d'arrière-plan dans l'Aperçu d'impression, suivez les étapes suivantes : (i) dans le menu **Outils** d'Internet Explorer, cliquez sur **Options Internet**, puis sur l'onglet Avancé ; (ii) dans Paramètres, sous Impression, cochez l'option *Imprimer les couleurs et images d'arrière-plan*, puis (iii) cliquez sur **OK**.

Paramètres d'impression

La commande **Configuration de l'impression** affiche le dialogue Paramètres d'impression spécifique à l'imprimante dans laquelle vous pouvez spécifier les paramètres d'impression concernant le format papier et l'orientation de page. Ces paramètres s'appliqueront à toutes les tâches d'impression ultérieures.

13.1.10 Fichiers récents, Quitter

Fichiers récents

En bas du menu **Fichier**, vous trouverez une liste des neufs fichiers utilisés récemment, le fichier ouvert le plus récemment se trouvant tout en haut de la liste. Vous pouvez ouvrir ces fichiers en cliquant sur leur nom. Pour ouvrir un fichier de la liste à l'aide du clavier, appuyer **Alt+F** pour ouvrir le menu **Fichier**, puis appuyer sur le numéro du fichier que vous souhaitez ouvrir.

Quitter

Quitte Authentic Desktop. Si d'autres fichiers contenant des changements non enregistrés sont ouverts, vous serez invité à enregistrer ces changements. Authentic Desktop enregistre aussi les modifications des paramètres de programme et les informations concernant les fichiers utilisés récemment.

13.2 Menu Édition

Le menu **Éditer** contient des commandes pour éditer des documents dans Authentic Desktop. Ceux-ci contiennent les commandes connues Undo, Redo, Cut, Copy, Paste, Delete, Select All, Find, Find Next et Replace.

13.2.1 Annuler, Rétablir

Icônes et raccourcis

Commande	*Icône*	*Raccourci*
Annuler		**Ctrl+Z**
Rétablir		**Ctrl+Y**

Annuler

La commande **Annuler** prend en charge un nombre illimité de niveaux d'annulation. Chaque action peut être annulée et il est possible d'annuler une commande après l'autre. L'historique d'annulation est conservé après l'utilisation de la commande **Enregistrer**, vous permettant de retourner à l'état original du document avant l'enregistrement de vos changements. Les commandes **Annuler** et **Rétablir** vous permettent de vous déplacer à votre guise dans tout l'historique (*voir la commande **Rétablir** ci-dessous*).

Rétablir

La commande **Rétablir** vous permet de rétablir des commandes annulées précédemment, vous disposez donc d'un historique complet du travail effectué. Les commandes **Annuler** et **Rétablir** vous permettent de vous déplacer à votre guise sur tout l'historique.

13.2.2 Couper, Copier, Coller, Supprimer

Icônes et raccourcis

Commande	*Icône*	*Raccourci*
Couper		**Ctrl+X** ou **Shift+Del**
Copier		**Ctrl+C**
Coller		**Ctrl+V**
Supprimer		**Suppr**

© 2016-2022 Altova GmbH

Couper

La commande **Couper** copie le texte ou les éléments sélectionnés sur le presse-papier, supprimant la sélection de son emplacement actuel.

Copier

La commande **Copier** copie le texte ou les éléments sélectionnés sur le presse-papier. Cela peut être utilisé pour dupliquer des données dans Authentic Desktop ou pour déplacer des données dans une autre application.

Coller

La commande **Copier** insère le contenu du presse-papier dans le positionnement actuel du curseur..

Supprimer

La commande **Supprimer** supprime le texte ou les éléments sélectionnés actuellement sans les placer dans le presse-papier.

13.2.3 Sélectionner tout

La commande **Sélectionner tout** (**Ctrl+A**) sélectionne le contenu du document entier.

13.2.4 Recherche, Trouver suivant

Icônes et raccourcis

Commande	*Icône*	*Raccourci*
Recherche		**Ctrl+F**
Trouver suivant		**F3**

Recherche

La commande **Recherche** affiche le dialogue Recherche dans lequel vous pouvez spécifier le string que vous souhaitez trouver et les autres options pour la recherche. Pour trouver le texte, le saisir dans le champ de saisie ou utiliser la liste de choix pour sélectionner un terme parmi les 10 derniers termes recherchés, puis spécifier les options de recherche.

Note : Les commandes **Recherche** et **Trouver suivant** peuvent aussi être utilisées pour trouver des noms de fichier et de dossier lorsqu'un projet est sélectionné pour la fenêtre Projet.

Trouver suivant

La commande **Trouver suivant** répète la dernière commande Chercher. Elle cherche l'occurrence suivante du texte saisi.

13.2.5 Remplacer

Icônes et raccourcis

Commande	*Icône*	*Raccourci*
Remplacer		**Ctrl+H**

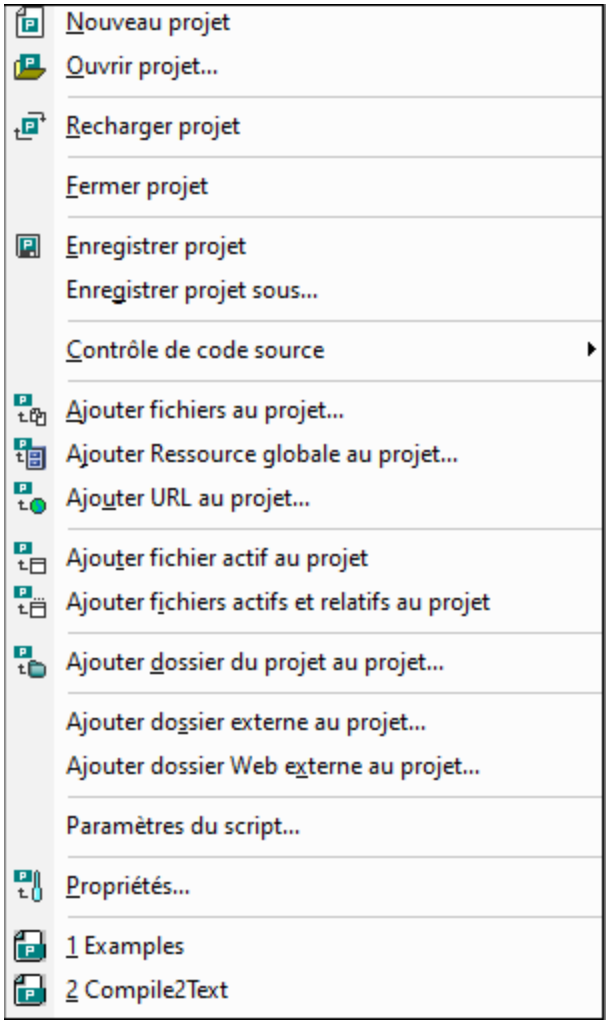
Description

La commande **Remplacer** vous permet de trouver et de remplacer un string de texte après l'autre. Elle présente les mêmes options que la commande **Recherche**. Vous pouvez remplacer les items individuellement ou vous pouvez utiliser la touche **Remplacer tout** pour effectuer une opération globale chercher-et-remplacer.

© 2016-2022 Altova GmbH

13.3 Menu Projet

Authentic Desktop utilise le mode bien connu d'arborescence pour gérer des fichiers ou les URL multiples dans les projets XML. Les fichiers et URL peuvent être réunis dans des dossiers regroupés par une extension en commun ou tout autre critère arbitraire, permettant une structuration et une manipulation batch en toute simplicité.



Veuillez noter : La plupart des commandes liées au projet sont aussi disponibles dans le menu contextuel, qui apparaît lorsque vous cliquez avec la touche de droite sur un item quelqconque dans la fenêtre de projet.

Chemins absolus et relatifs

Chaque projet est enregistré en tant que fichier de projet et a l'extension `.spp`. Ces fichiers sont en fait des documents XML que vous pouvez éditer comme tout fichier XML habituel. Dans le fichier de projet, par exemple, les chemins absolus sont utilisés pour des fichiers/dossiers de même niveau ou plus haut, et les chemins relatifs pour les fichiers/dossiers dans le dossier actuel ou dans les sous-fichiers. Par exemple, si votre structure de répertoire ressemble à cela :

```
|-Folder1
|       |
|       |-Folder2
|               |
|               |-Folder3
|                       |
|                       |-Folder4
```

Si votre fichier `.spp` est situé dans le `Folder3`, alors les références vers les fichiers dans `Folder1` et `Folder2` auront environ l'aspect suivant :

```
c:\Folder1\NameOfFile.ext
c:\Folder1\Folder2\NameOfFile.ext
```

Les références aux fichiers dans le `Folder3` et `Folder4` auront environ l'aspect suivant :

```
.\NameOfFile.ext
.\Folder4\NameOfFile.ext
```

Si vous souhaitez vous assurer que tous les chemins seront relatifs, enregistrer les fichiers `.spp` dans le répertoire racine de votre disque de travail.

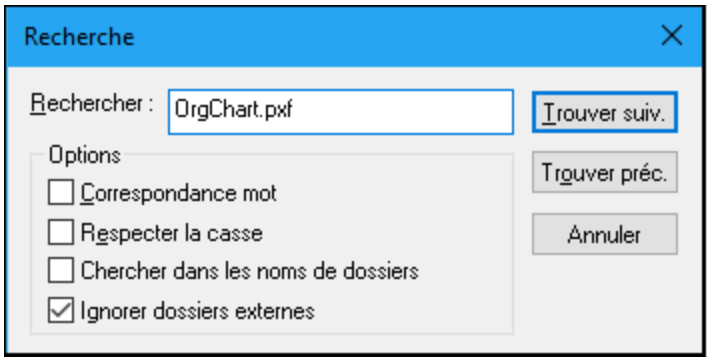
Glisser/déposer

Dans la fenêtre Projet, un dossier peut être glissé dans un autre dossier ou dans un autre emplacement dans le même dossier. Un fichier peut être glissé dans un autre dossier, mais ne peut pas être déplacé dans le sein du même dossier (dans lequel les fichiers sont classés par ordre alphabétique). De plus, les fichiers et dossiers peuvent être glissés depuis Windows File Explorer vers la fenêtre de projet.

Rechercher dans le projet

Vous pouvez chercher les fichiers de projet et les dossiers en utilisant leurs noms ou une partie de leur nom. Si la recherche aboutit, les fichiers ou les dossiers trouvés seront marqués un par un.

Pour lancer une recherche, Choisir la fenêtre du Projet en cliquant dessus (ou dedans), puis sélectionner la commande **Édition | Recherche** (ou le raccourcis **Ctrl+F**). Dans le dialogue Recherche qui apparaît (*capture d'écran ci-dessous*), saisir le texte que vous souhaitez chercher et sélectionner ou désélectionner les options de recherche (*explications ci-dessous*) conformément à vos exigences.



Les options de recherche suivantes sont disponibles :

© 2016-2022 Altova GmbH

- La correspondance mot entier est plus limitée puisque le texte entier doit correspondre à un mot entier dans le nom du fichier ou du dossier. Dans les noms du fichier, les parties avant et après le point (sans le point) sont toutes traitées en tant qu'un mot.
- Vous pouvez préciser que la casse du mot recherché doit correspondre exactement au string contenu dans le nom du fichier ou du dossier.
- Les noms de dossier peuvent être inclus à la recherche. Dans le cas contraire, seuls les noms de fichiers seront recherchés.
- Les dossiers externes peuvent être inclus ou exclus de la recherche. Les dossiers externes sont des dossiers réels dans le système ou le réseau, contrairement aux dossiers de projet qui sont créés dans le cadre du projet et non dans le système.

Si la recherche aboutit, le premier item correspondant est marqué dans la barre latérale du Projet. Vous pouvez ensuite parcourir tous les items correspondants retournés en cliquant sur les boutons **Trouver suivant** et **Trouver précédant** dans le dialogue Recherche.

Réinitialiser les projets

Si vous effectuez une modification dans un dossier externe, cette modification ne sera pas réfléchie dans la Fenêtre du Projet tant que le projet est réinitialisé.

Ressources globales dans le menu contextuel

Lorsque vous cliquez avec la touche de droite dans un dossier contenu dans la fenêtre Projet, dans le menu contextuel qui apparaît, vous pouvez sélectionner l'item de menu **Ajouter Ressource globale** pour ajouter une ressource globale. La commande de menu elle-même apparaît dans le dialogue « Choose Global Resource », qui liste les ressources globales file-type et folder-type dans le fichier XML des Ressources globales actives actuelles. Choisir la ressource globale exigée et elle sera ajoutée au dossier de projet sélectionné.

Fournisseurs de projets et de contrôle de source

Si vous prévoyez d'ajouter un projet Authentic Desktop à un archivage de contrôle de source, veuillez vous assurer que le positionnement des fichiers de projet dans la structure de système de fichier hiérarchique est une structure qui vous permet d'ajouter des fichiers uniquement depuis les niveaux inférieurs (le répertoire de racine constituant le sommet de l'arborescence du répertoire).

En d'autres termes, le répertoire où le **fichier de projet** est situé, représente essentiellement le **répertoire racine** du projet dans l'archivage de contrôle de source. Les fichiers ajoutés depuis les niveaux supérieurs (le répertoire de racine du projet) seront ajoutés au projet Authentic Desktop, mais leur emplacement dans l'archivage peut être inattendu, s'ils peuvent y être placés.

Par exemple, selon la structure de répertoire affichée ci-dessus, si un fichier de projet est enregistré dans le `Folder3` puis placé sous contrôle de source :

- Fichiers ajoutés au Folder1 ne peut pas être placé sous le contrôle source,
- Les fichiers ajoutés au Folder2 sont ajoutés au répertoire racine de l'archivage au lieu du dossier de projet mais se trouvent encore sous contrôle de source,
- Les fichiers ajoutés au Folder3 et au Folder4 fonctionnent comme prévu et sont placés sous contrôle de source.

13.3.1 Nouveau Projet

La commande **Nouveau projet** crée un nouveau projet dans Authentic Desktop. Si vous travaillez actuellement avec un autre projet, une invite s'affichera vous demandant si vous souhaitez fermer tous les documents faisant partie du projet actuel. Le nom du projet est assigné quand vous enregistrez le projet en tant que `.spp` file.

13.3.2 Ouvrir Projet

La commande **Ouvrir le projet...** ouvre un projet existant dans Authentic Desktop. Si vous travaillez actuellement avec un autre projet, le projet précédent est tout d'abord fermé.

13.3.3 Recharger Projet

La commande **Recharger le projet** recharge le projet actuel depuis le disque. Si vous travaillez dans un environnement à utilisateurs multiples, il peut parfois devenir nécessaire de recharger le projet depuis le disque parce que d'autres utilisateurs peuvent avoir effectué des changements au projet.

Note : Les fichiers de projet (fichiers `.spp`) sont en fait des documents XML que vous pouvez éditer comme tout fichier XML habituel.

13.3.4 Fermer Projet

La commande **Fermer projet** ferme le projet actuel. Si le projet a été modifié, vous serez invité à enregistrer le projet d'abord. Lorsqu'un projet est modifié d'une manière ou d'une autre, un astérisque sera ajouté au nom du projet dans la fenêtre Projet.

© 2016-2022 Altova GmbH

13.3.5 Enregistrer projet, Enregistrer projet sous

La commande **Enregistrer projet** enregistre le projet actuel. Vous pouvez aussi enregistrer un projet en activant la fenêtre de projet et en cliquant sur l'icône  .

La commande **Enregistrer projet sous** enregistre le projet actuel avec un nouveau nom que vous pouvez saisir lorsque vous y êtes invité.

13.3.6 Contrôle de code source

Votre application Altova prend en charge Microsoft SourceSafe et d'autres référentiels compatibles. Une liste des systèmes pris en charge est indiquée dans la section Systèmes de contrôle de source pris en charge. Cette section décrit les commandes dans le sous-menu **Projet | Contrôle de source** qui sont utilisés pour fonctionner avec le système de contrôle de source depuis votre application Altova.

Aperçu de la fonction Contrôle de source

Le mécanisme pour placer les fichiers dans un projet d'application sous contrôle de source est le suivant :

1. Dans Authentic Desktop, un dossier de projet d'application contenant les fichiers à placer sous contrôle de source est créé. Généralement, le dossier de projet d'application correspondra à un dossier local dans lequel se trouvent les fichiers de projet. Le chemin menant vers le dossier local est référé en tant que le chemin local.
2. Dans la base de données du système de contrôle de source (également appelé contrôle de source ou référentiel), un dossier est créé qui contiendra les fichiers à placer sous contrôle de source.
3. Les fichiers de projet d'application sont ajoutés au contrôle de source avec la commande **Projet | Contrôle de source | Ajouter au Contrôle de source**.
4. Les actions de contrôle de source, comme l'archivage, l'extraction et la suppression de fichiers du contrôle de source, peuvent être effectuées en utilisant les commandes dans le sous-menu **Projet | Contrôle de source**. Les commandes dans ce sous-menu sont recensées dans les sous-sections de cette section.

Note : si vous souhaitez modifier le fournisseur actuel du contrôle de source, vous avez deux possibilités : (i) par le biais des options du contrôle de source (**Outils | Options | Contrôle de source**), ou (ii) dans le dialogue Changer Contrôle de source (**Projet | Contrôle de source | Changer Contrôle de source**).

Note : veuillez noter qu'un projet de contrôle de source n'est pas identique à un projet d'application. Les projets de contrôle de source sont dépendants des répertoires alors que les projets Authentic Desktop sont des constructions logiques sans dépendance directe à un répertoire.

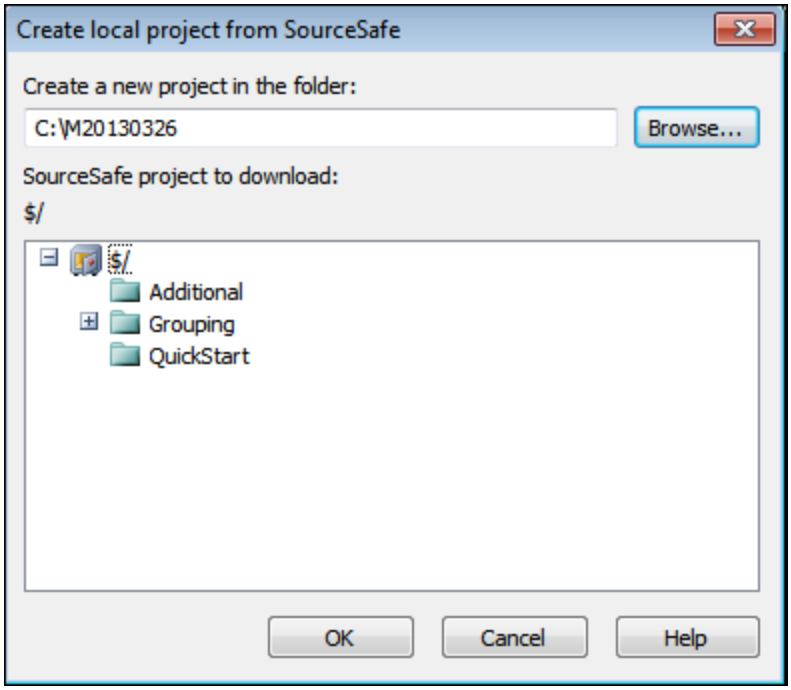
Pour plus d'informations, voir la section Contrôle de source.

13.3.6.1 Ouvrir depuis le Contrôle de source

La commande **Ouvrir depuis le Contrôle de source** crée un nouveau projet d'application depuis un projet se trouvant sous contrôle de source.

Créer le nouveau projet comme suit :

1. Selon le système de contrôle de source utilisé, il peut être nécessaire, avant de créer un nouveau projet depuis le contrôle de source, de vous assurer qu'aucun fichier n'est extrait provenant du projet.
2. Il n'est pas nécessaire d'ouvrir un projet dans l'application, mais il est possible de le faire.
3. Choisir la commande Projet | Contrôle de source | Ouvrir depuis le Contrôle de source.
4. Le système de contrôle de source actuellement configuré ouvrira sa vérification et les dialogues de connexion. Établissez la connexion au référentiel que vous souhaitez, c'est-à-dire au dossier lié dans l'archivage qui correspond au dossier local.
5. Dans le dialogue qui s'ouvre (*capture d'écran ci-dessous*), recherchez le dossier local dans lequel les contenus du dossier lié dans le référentiel (auquel vous venez de vous connecter) doivent être copiés. Dans la capture d'écran ci-dessous, le dossier lié est appelé `MyProject` et est représenté par le signe `$` ; le dossier local est `C:\M20130326`.



6. Cliquez sur **OK**. Les contenus du dossier lié (`MyProject`) seront copiés dans le dossier local `C:\M20130326`., et un dialogue s'ouvre pour vous inviter à choisir le fichier de projet (fichier `.spp`) qui doit être créé en tant que le nouveau projet.
7. Choisir le fichier `.spp` qui aura été copié dans le dossier local. Dans notre exemple, tceci sera situé dans `MyProject.spp` dans le dossier `C:\M20130326`. Un nouveau projet appelé `MyProject` sera créé dans l'application et sera affiché dans la fenêtre Projet. Les fichiers de projet se trouveront dans le dossier `C:\M20130326`.

© 2016-2022 Altova GmbH

Symboles de contrôle de source

Les fichiers et dossier de projet affichent certains symboles dont la signification est expliquée ci-dessous.

	Archivé. Disponible pour extraction.
	Extrait par un autre utilisateur. Non disponible pour une extraction.
	Extrait localement. Peut être édité et archivé.

13.3.6.2 Activer Contrôle de source

La commande **Activer le contrôle de source** vous permet d'activer ou de désactiver le contrôle de source pour un projet d'application. Choisir cette option sur n'importe quel fichier ou dossier pour activer/désactiver le contrôle de source pour l'ensemble du projet. Une fois que le contrôle de source est activé, les statuts d'archivage/d'extraction des différents fichiers sont extraits et affichés dans la fenêtre Projet.

	Archivé. Disponible pour extraction.
	Extrait par un autre utilisateur. Non disponible pour une extraction.
	Extrait localement. Peut être édité et archivé.

13.3.6.3 Obtenir la dernière version

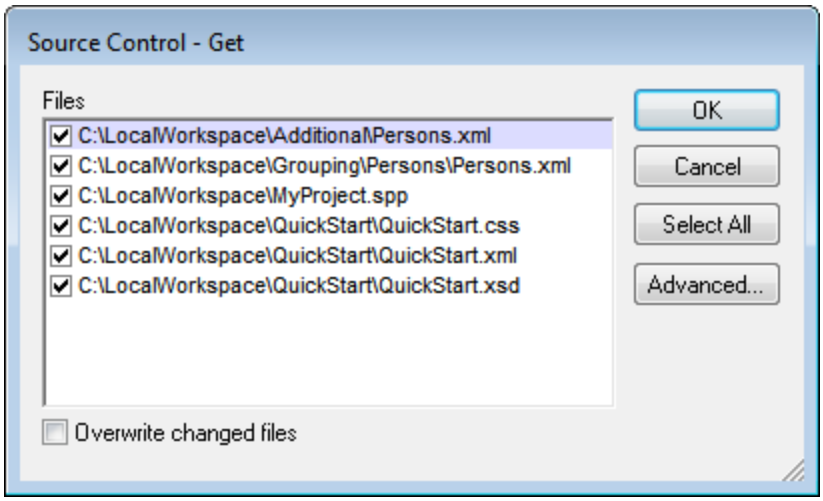
La commande **Obtenir la dernière version** (dans le menu **Projet | Contrôle de source**) extrait et place la dernière version du contrôle de source du/des fichier/s sélectionné/s dans le répertoire de travail. Les fichiers sont extraits en lecture seule et ne sont pas extraits. Cette commande fonctionne comme la commande **Obtenir**, mais n'est pas affichée dans le dialogue Obtenir.
Si les fichiers sélectionnés sont actuellement extraits, l'action choisie dépendra de la manière dont votre système de contrôle de source gère ce type de situation. Généralement, le système de contrôle de source vous demandera si vous souhaitez remplacer, fusionner ou conserver le fichier extrait tel quel.

Note : Cette commande est récursive lorsqu'elle est effectuée sur un dossier, c'est-à-dire qu'elle touche tous les fichiers se trouvant en-dessous du fichier actuel dans la hiérarchie du dossier.

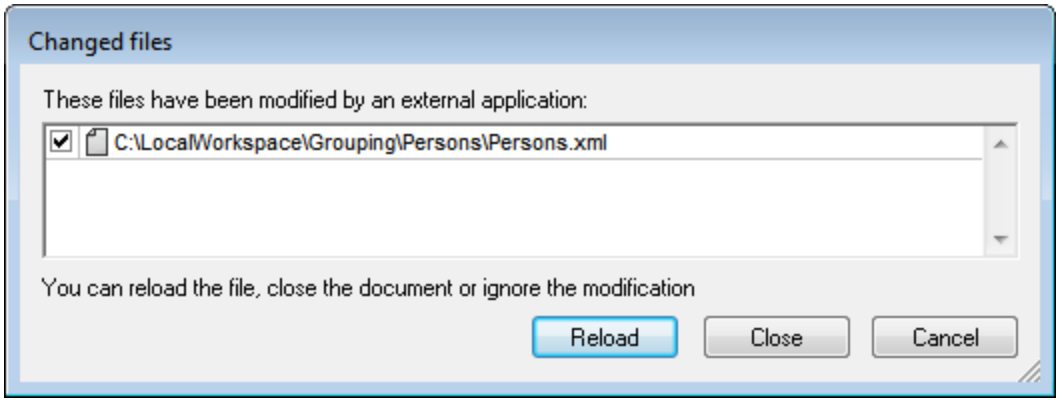
13.3.6.4 Obtenir, Obtenir les dossiers

La commande **Get** (dans le menu **Projet | Contrôle de source**) extrait les fichiers du référentiel comme fichiers en lecture seule. (Pour éditer un fichier, il faut l'extraire.) **Le dialogue Obtenir recense les fichiers dans l'objet (projet ou dossier) dans lequel la commande** Obtenir a été exécutée (*voir capture d'écran ci-dessous*). Vous pouvez choisir les fichiers à extraire en les archivant.

Note : La commande **Obtenir dossier** vous permet de choisir des sous-dossiers individuels dans l'archivage si cela est permis par votre système de contrôle de source.



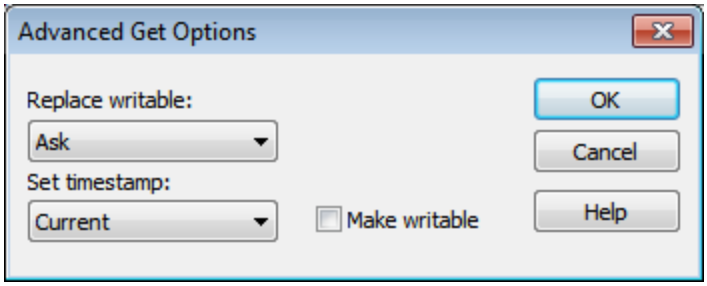
Vous pouvez décider d'écraser des fichiers extraits ayant été modifiés en cochant cette option située en bas du dialogue Obtenir. Cliquez sur **OK** pour écraser les fichiers. Si l'un des fichiers écrasés est actuellement ouvert, un dialogue apparaît (*capture d'écran ci-dessous*) demandant si vous souhaitez recharger le/s fichier/s (touche **Recharger**), fermer le/s fichier/s (**Fermer**) ou conserver le mode actuel du fichier (**Annuler**).



Options Obtenir avancées

Le dialogue Options Obtenir avancées (*capture d'écran ci-dessous*) est accessible depuis la touche **Avancé** dans le dialogue Obtenir (*voir première capture d'écran dans cette section*).

© 2016-2022 Altova GmbH



Ici, vous pouvez définir les options pour (i) remplacer les fichiers accessibles en écriture qui sont extraits, (ii) l'horodatage, et (iii) si vous souhaitez changer la propriété lecture seule du fichier extrait pour le rendre accessible en écriture.

13.3.6.5 Extraire, Archiver

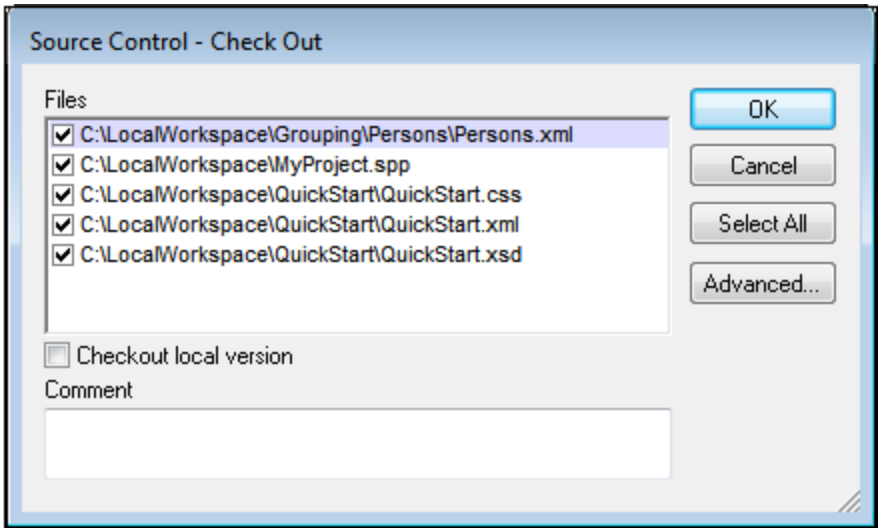
Une fois qu'un fichier de projet a été placé sous la source de contrôle, il peut être extrait ou archivé en sélectionnant le fichier (dans la fenêtre de projet) et en cliquant la commande respective dans le menu **Projet | Contrôle de source** : **Extraire** et **Archiver**.

Lorsqu'un fichier est extrait, une copie provenant de l'archivage est placée dans le dossier local. Un fichier qui est extrait peut être édité. Si un fichier se trouvant sous contrôle de source n'est pas extrait, il ne pourra pas être édité. Une fois qu'un fichier a été édité, les changements peuvent être enregistrés dans l'archivage en validant le fichier. Même si vous n'enregistrez pas le fichier, en le validant, vous pourrez enregistrer les changements dans l'archivage. Un coche ou un cadenas dans l'icône vous indique que le fichier est extrait.

Les fichiers et dossier de projet affichent certains symboles dont la signification est expliquée ci-dessous.

	Archivé. Disponible pour extraction.
	Extrait par un autre utilisateur. Non disponible pour une extraction.
	Extrait localement. Peut être édité et archivé.

En choisissant le projet ou un dossier dans le projet, vous sélectionnez tous les fichiers se trouvant dans l'objet sélectionné. Pour sélectionner de multiples objets (fichiers et dossiers), appuyez sur la touche Ctrl tout en cliquant sur les objets. La capture d'écran ci-dessous montre un projet qui a été extrait. Le fichier `QuickStart.css` a été archivé par la suite.



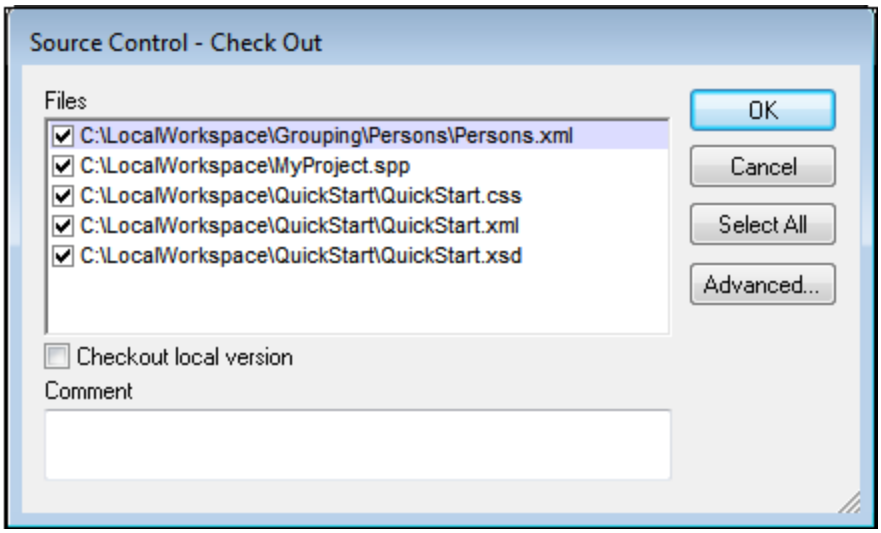
Enregistrer et rejeter des modifications d'édition

Veuillez noter que, lorsque vous archivez un fichier, vous pouvez choisir de laisser ce fichier extrait. De fait, les modifications d'édition sont enregistrées dans l'archivage alors que le fichier reste extrait, ce qui peut être utile lorsque vous souhaitez enregistrer des modifications d'édition de temps en temps puis poursuivre l'édition.

Si vous avez extrait un fichier puis que vous l'avez édité, mais que vous souhaitez annuler ces changements, vous pouvez retourner à la version de document enregistrée dans l'archivage en choisissant la commande **Projet | Contrôle de source | Annuler extraction**.

Extraction

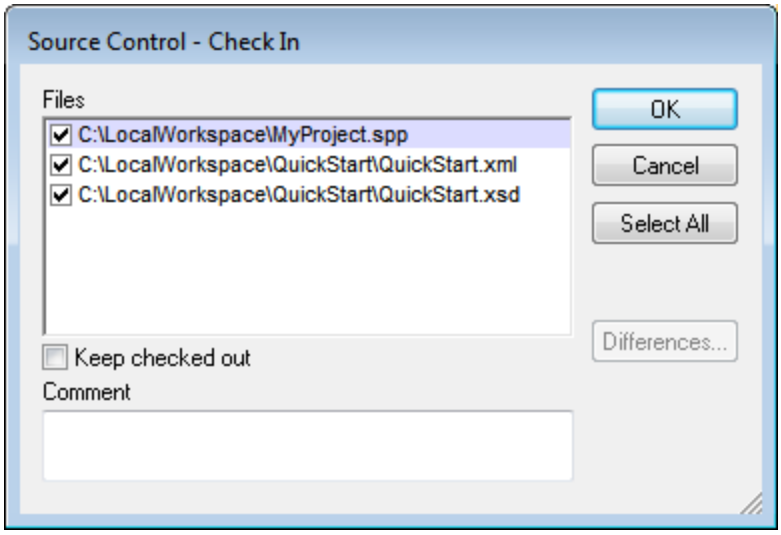
Le dialogue Extraction (*capture d'écran ci-dessous*) vous permet de : (i) sélectionner les fichiers à extraire, et (ii) de choisir si vous souhaitez extraire la version de l'archivage ou la version locale.



© 2016-2022 Altova GmbH

Archivage

Le dialogue Archiver (*capture d'écran ci-dessous*) vous permet de : (i) sélectionner les fichiers à archiver, et (ii) si vous le souhaitez, de garder le fichier extrait.



Note : dans les deux dialogues (Extraire et Archiver), plusieurs fichiers apparaissent si l'objet sélectionné (projet ou dossiers de projet) contient plusieurs fichiers.

13.3.6.6 Annuler Extraction

Si vous avez extrait un fichier puis que vous l'avez édité, mais que vous souhaitez annuler ces changements, vous pouvez retourner à la version de document enregistrée dans l'archivage en choisissant la commande **Projet | Contrôle de source | Annuler extraction**.

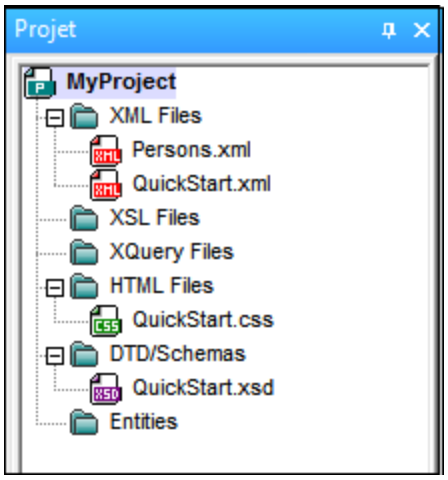
Les fichiers et dossier de projet affichent certains symboles dont la signification est expliquée ci-dessous.

	Archivé. Disponible pour extraction.
	Extrait par un autre utilisateur. Non disponible pour une extraction.
	Extrait localement. Peut être édité et archivé.

13.3.6.7 Ajouter au Contrôle de source

Une fois qu'un projet a été ajouté au contrôle de source, vous pouvez ajouter des fichiers individuellement ou en groupe dans le contrôle de source. Choisir les fichiers dans la fenêtre Projet et cliquer sur la commande **Projet | Contrôle de source | Ajouter au Contrôle de source**. Pour sélectionner des fichiers multiples, maintenez appuyée la touche **Ctrl** en cliquant sur les fichiers que vous souhaitez ajouter. Exécuter la commande dans un

dossier de projet (vert) (*voir la capture d'écran ci-dessous*) ajoute tous les fichiers au dossier et ses sous-dossiers au contrôle de source.



Lorsque des fichiers sont ajoutés au contrôle de source, la hiérarchie de dossier local est répliquée dans le référentiel (pas la hiérarchie du dossier du projet). Donc, si un fichier est dans un sous-fichier X dans les abysses du dossier local, alors le dossier parent du fichier et tous les autres dossiers ancêtre sont automatiquement créés dans le référentiel.

Lorsque le premier fichier d'un projet est ajouté au contrôle de source, les liaisons correctes sont créées dans le référentiel et le fichier de projet (fichier `.spp`) est ajouté automatiquement. Pour plus de détails, voir la section Ajouter au Contrôle de source.

Symboles de contrôle de source

Les fichiers et dossier de projet affichent certains symboles dont la signification est expliquée ci-dessous.

	Archivé. Disponible pour extraction.
	Extrait par un autre utilisateur. Non disponible pour une extraction.
	Extrait localement. Peut être édité et archivé.

13.3.6.8 Supprimer du Contrôle de source

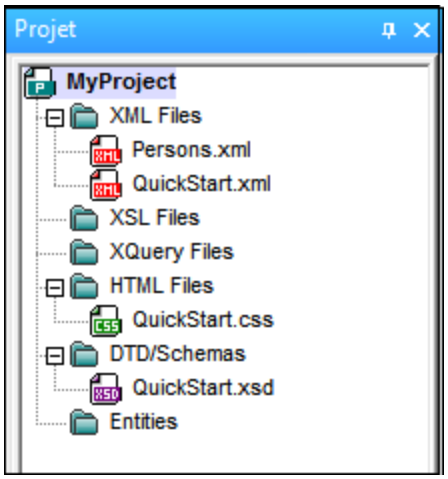
Pour supprimer un fichier du contrôle de source, choisir le fichier et cliquer sur la commande **Projet | Contrôle de source | Supprimer du Contrôle de source**. Vous pouvez aussi supprimer : (i) les fichiers dans un dossier de projet en exécutant la commande sur le dossier, (ii) de multiples fichiers que vous sélectionnez en maintenant la touche **Ctrl** appuyée et (iii) le projet entier en exécutant la commande sur le projet.

© 2016-2022 Altova GmbH

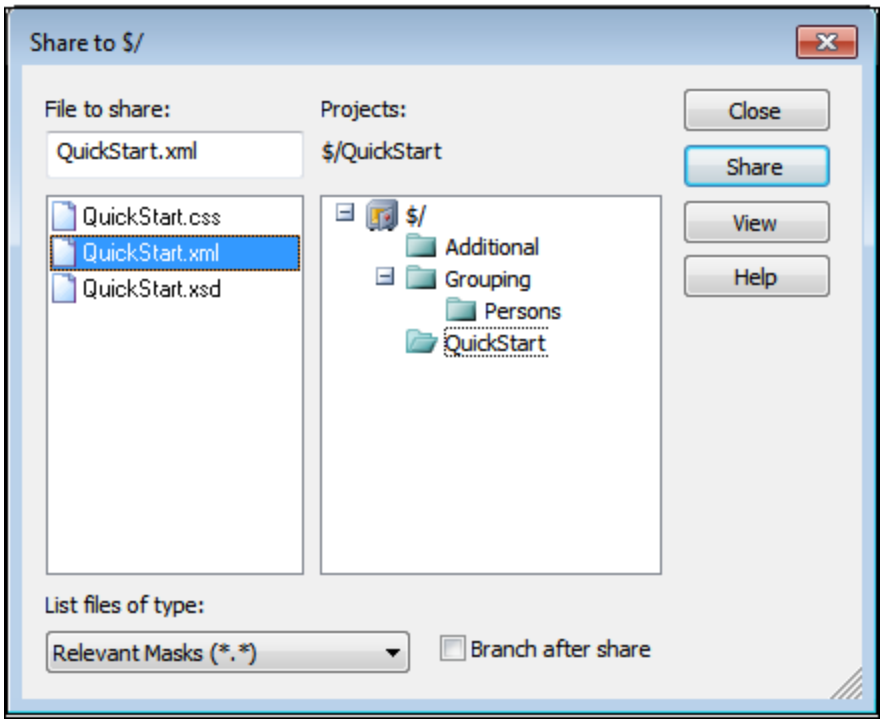
13.3.6.9 Partager depuis le Contrôle de source

La commande **Partager depuis le Contrôle de source** est prise en charge lorsque le système de contrôle de source utilisé prend en charge les partages. Vous pouvez partager un fichier de manière à ce qu'il soit disponible dans plusieurs endroits locaux. Une modification dans un de ces fichiers locaux sera reflétée dans toutes les autres versions « partagées ».

Dans la fenêtre d'application du projet, sélectionnez d'abord le projet (*en surbrillance dans la capture d'écran ci-dessous*). Ensuite, cliquez sur **Partager depuis le Contrôle de source**.



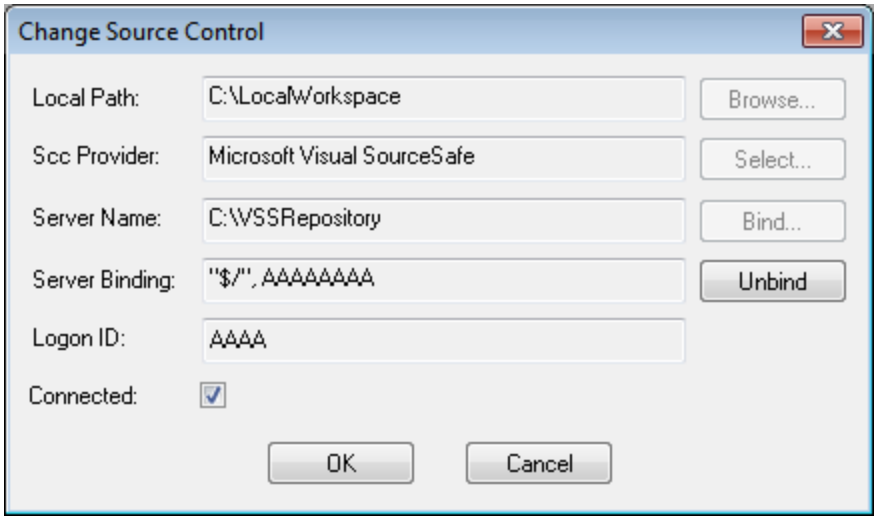
Le dialogue Partager avec [Dossier] (*capture d'écran ci-dessous*) s'ouvre.



Pour sélectionner les fichiers à partager, choisissez tout d'abord dans l'arborescence de projet située dans le panneau de droite le dossier dans lequel se trouvent les fichiers. Ces fichiers sont affichés dans le volet de gauche. Choisir le fichier que vous souhaitez partager (plusieurs fichiers en appuyant sur la touche **Ctrl** et en cliquant sur les fichiers que vous souhaitez partager). Les fichiers sélectionnés seront affichés dans le champ de saisie *Fichiers à partager* (*en haut à gauche*). Cliquez sur **Partager**, puis sur **Fermer** pour copier les fichiers sélectionnés dans le dossier de partage local.

Le dossier de partage est noté dans le nom du dialogue Partager avec [Dossier]. Dans la capture d'écran ci-dessus, il s'agit du dossier local (puisque le signe $ est le dossier dans le référentiel dans lequel le dossier local est lié). Vous pouvez voir et configurer le dossier de partage dans le dialogue Changer Contrôle de source (*capture d'écran ci-dessous*, **Changer Contrôle de source**) en modifiant le chemin local et la liaison au serveur.

© 2016-2022 Altova GmbH

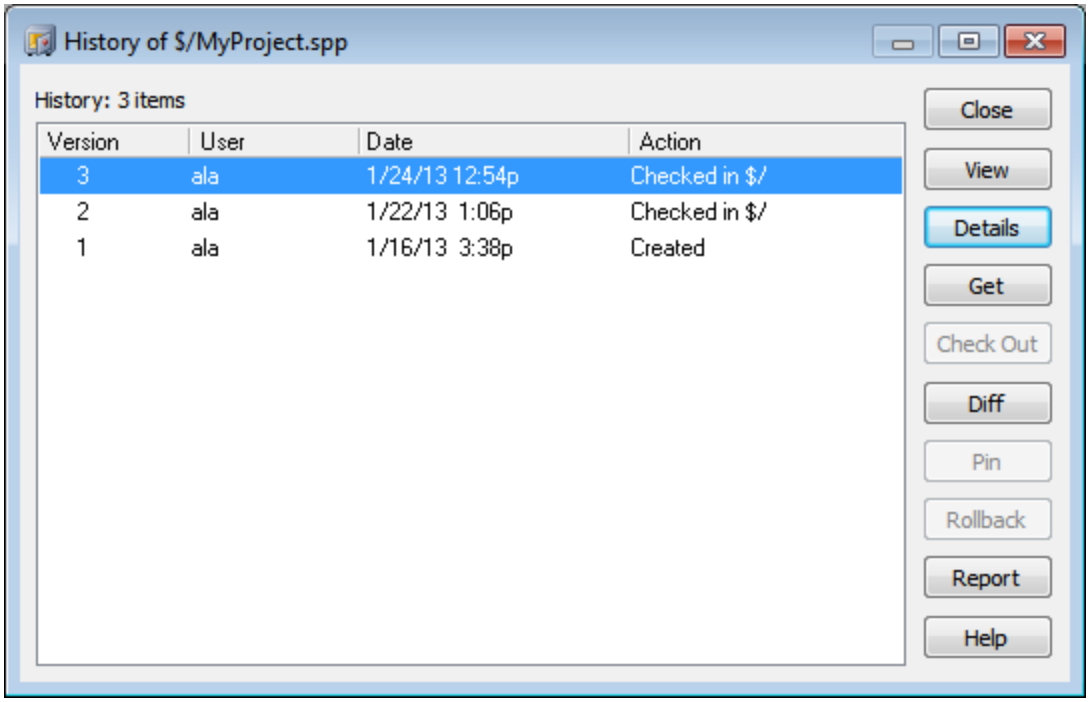


Pour plus de détails concernant le partage de votre système de contrôle de source, voir la documentation de l'utilisateur du système de contrôle de source.

13.3.6.10 Afficher Historique

La commande **Afficher historique** active la fonction Afficher historique du système de contrôle de source. Elle affiche l'historique du fichier sélectionné dans la fenêtre de projet. Sélectionnez le titre de projet dont vous voulez afficher l'historique (fichier `.spp`). Vous pouvez consulter des informations concernant les versions précédentes d'un fichier et les différences ou d'extraire les versions précédentes d'un fichier.

La capture d'écran ci-dessous montre le dialogue de l'historique du système de contrôle de source Visual SourceSafe. Elle recense les différentes versions du fichier `MyProject.spp`.



Ce dialogue Historique fournit différentes manières de comparer et obtenir des versions spécifiques du fichier en question. Double-cliquez sur une entrée dans la liste ouvre la boîte de dialogue Détails de l'Historique pour ce fichier. Les touches du dialogue ont les fonctions suivantes :

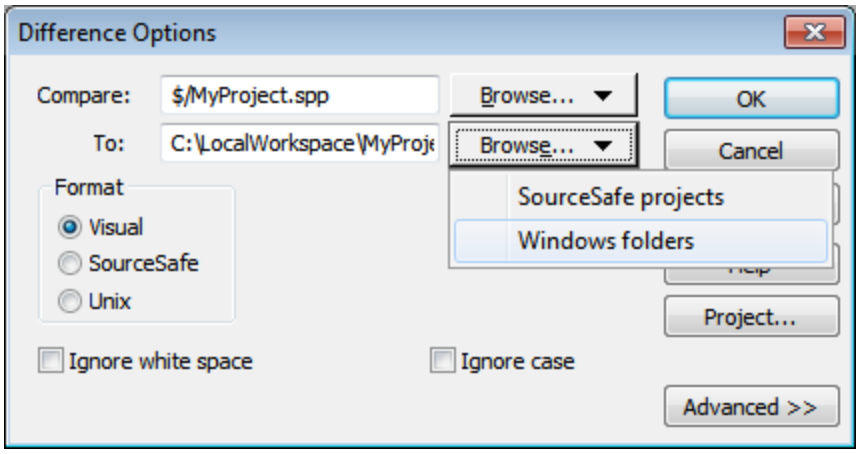
- *Fermer :* Ferme ce dialogue.
- *Mode :* Ouvre un autre dialogue dans lequel vous pouvez sélectionner le type d'affichage avec lequel vous souhaitez voir le fichier.
- *Détails :* Ouvre un dialogue dans lequel vous pouvez voir les propriétés du fichier actif actuellement.
- *Get :* Extrait une version de fichier précédente et la place dans le répertoire de travail.
- *Extraire :* Vous permet d'extraire une version précédente du fichier.
- *Diff :* Ouvre le dialogue options de différenciation pour une différenciation entre deux versions de fichier. Utiliser **CTRL+Clic** pour marquer les deux versions de fichier dans cette fenêtre, puis cliquez sur Diff pour consulter les différences entre elles.
- *Pin :* Épingle et détache une version du fichier, vous permettant de définir la version du fichier spécifique à utiliser lors de la différenciation de deux fichiers.
- *Annuler :* Annule et retourne à la version sélectionnée du fichier.
- *Rapport :* Génère un rapport d'historique que vous pouvez envoyer à l'imprimante, au fichier ou au presse-papiers.
- *Aide :* Ouvre l'aide en ligne du plugin du fournisseur du contrôle de source.

13.3.6.11 Afficher les différences

La commande **Afficher les différences** est activée lorsqu'un fichier est sélectionné dans la fenêtre Projet. Pour sélectionner le fichier de projet (fichier `.spp`), choisissez le titre de projet dans la fenêtre Projet. La commande **Afficher les différences** lance l'outil de différenciation du système de contrôle de source de manière à ce que les différences entre les fichiers peuvent être vérifiées directement depuis l'application Altova.

© 2016-2022 Altova GmbH

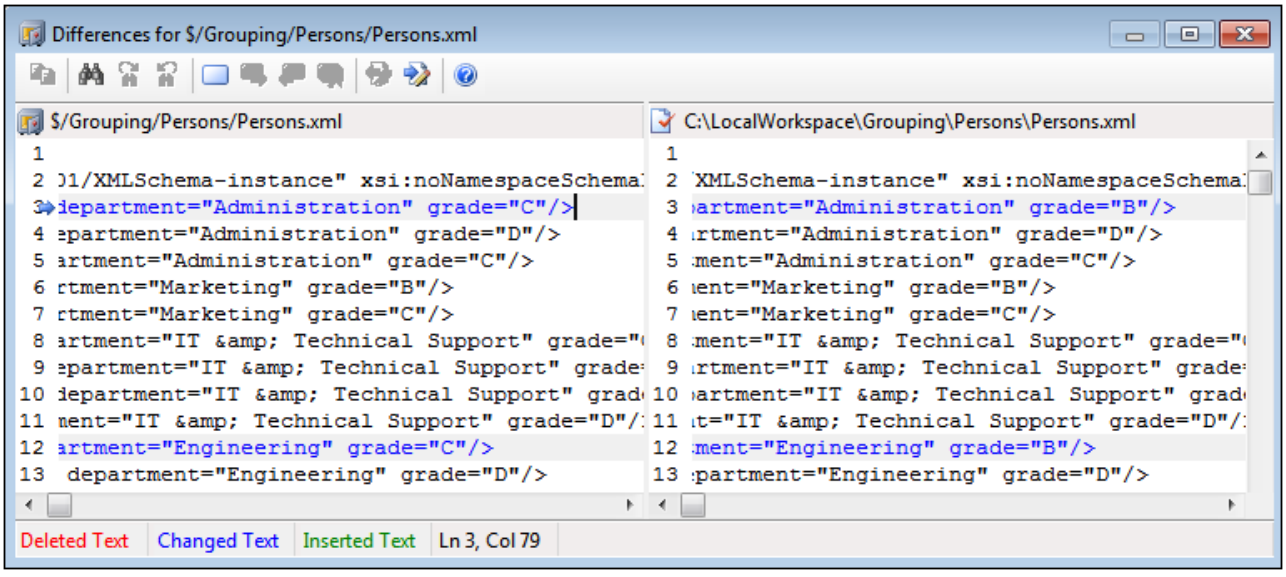
La capture d'écran ci-dessous montre les outils de différenciation du système de contrôle de source Visual SourceSafe.



Le référentiel et les versions locales sont affichés par défaut dans les champs de saisie *Comparer* et *vers* respectivement. Vous pouvez chercher d'autres fichiers de la manière suivante :

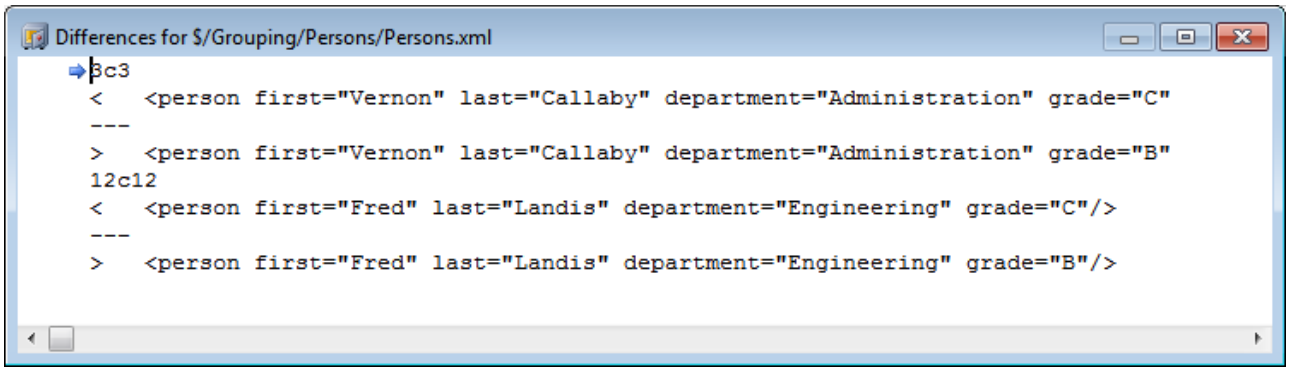
1. À partir de la liste déroulante dans le bouton **Chercher**, choisir des projets SourceSafe (pour chercher des fichiers d'archivage) ou des dossiers Windows (pour chercher des dossiers locaux).
2. Cherchez les fichiers que vous souhaitez, puis sélectionnez-les.

Sélectionnez les options que vous souhaitez.et cliquez sur **OK** pour exécuter la vérification. Les résultats de différenciation sont affichés dans une fenêtre séparée. Les captures d'écran ci-dessous montrent les résultats d'une vérification en deux formats.



La capture d'écran ci-dessus montre les résultats de différenciation Visual SourceSafe dans un format Visual (*voir dialogue d'Options ci-dessus*), alors que la capture d'écran ci-dessous montre les résultats dans un format

Unix. Les deux présentent deux différences, chacune d'entre elle est une modification du degré (grade) de C à B.



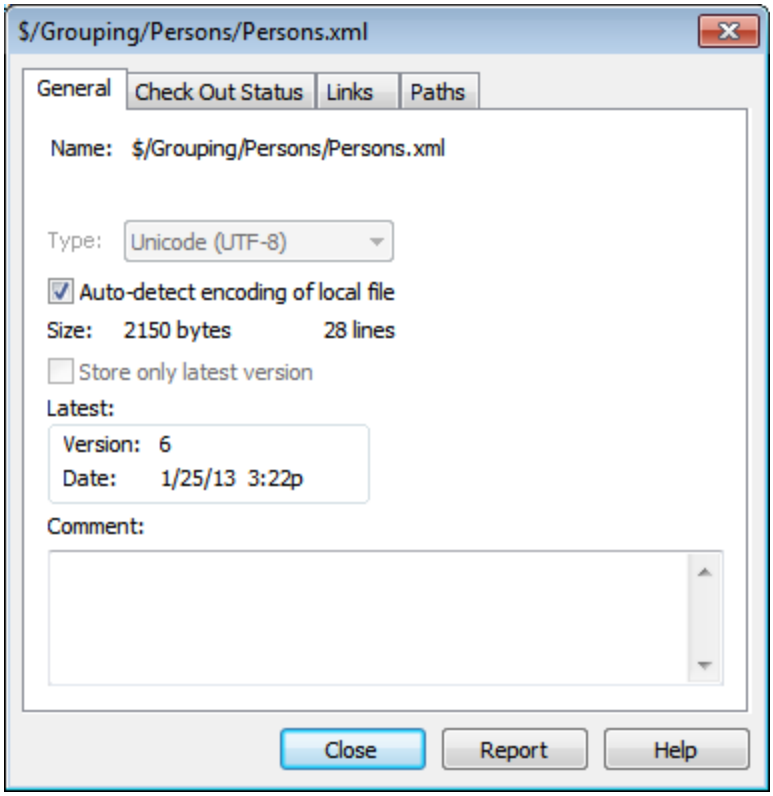
Pour une description détaillée de la manière dont votre système de contrôle de source gère la différenciation, voir la documentation de l'utilisateur du système de contrôle de source.

13.3.6.12 Afficher Propriétés

La commande **Afficher Propriétés** affiche les propriétés du fichier sélectionné actuellement (*voir la capture d'écran ci-dessous*). Les propriétés affichées dépendent du système de contrôle de source que vous utilisez. La capture d'écran ci-dessous montre les propriétés de Visual SourceSafe lorsqu'il est le système de contrôle de source actif.

Veuillez noter que cette commande peut uniquement être utilisée sur des fichiers individuels.

© 2016-2022 Altova GmbH



Pour plus de détails, voir la documentation de l'utilisateur du système de contrôle de source.

13.3.6.13 Réinitialiser statut

La commande **Réinitialiser statut** réinitialise le statut de tous les fichiers de projet, indépendamment de leur statut actuel.

13.3.6.14 Gestionnaire de Contrôle de source

La commande **Gestionnaire de Contrôle de source** lance votre logiciel de contrôle de source avec son interface d'utilisateur native.

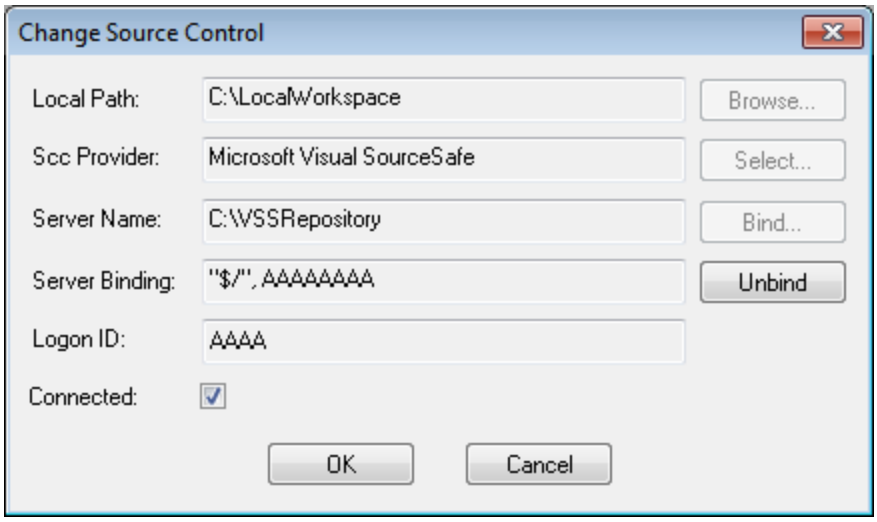
13.3.6.15 Modifier Contrôle de source

La liaison actuelle est celle que le projet d'application actif utilisera pour se connecter à la base de donnée de contrôle de source, c'est pourquoi la liaison actuelle doit être correcte. Cela signifie que le fichier de projet d'application (fichier.spp) doit se trouver dans le chemin de dossier local et le dossier lié dans l'archivage doit

être la base de données où ces fichiers du projet sont stockés. Généralement, le dossier lié et sa sous-structure correspondront au dossier de l'espace de travail local et à sa sous-structure.

Dans le dialogue Changer le Contrôle de source (*voir la capture d'écran ci-dessous*), vous pouvez changer le système de contrôle de source (*Fournisseur SCC*), le dossier local (*Chemin local*) et la liaison du référentiel (*Nom de serveur* et *Liaison de serveur*).

Ce n'est qu'après avoir délié la liaison actuelle que les configurations peuvent être modifiées. Délier la liaison actuelle avec la touche **Délier**. Toutes les configurations sont maintenant éditables.



Modifier les paramètres du contrôle de source comme suit :

1. Utiliser la touche **Chercher** pour chercher dans le dossier local et sur la touche **Choisir** pour choisir parmi les systèmes de contrôle de source installés.
2. Ensuite, vous pouvez lier le dossier local à une base de données d'archivage. Cliquer sur la touche **Lier** pour ce faire. Le dialogue de connexion de votre système de contrôle de source s'ouvre.
3. Si vous avez saisi une *ID de login*, elle sera transmise au système de contrôle de source ; sinon, vous devrez saisir vos détails de connexion dans le dialogue de connexion.
4. Choisir la base de données dans le référentiel que vous souhaitez lier à ce dossier local. Cette configuration peut être étalée sur plusieurs dialogues.
5. Une fois que la configuration a été créée, cliquez sur **OK** dans le dialogue Changer Contrôle de source.

13.3.7 Ajouter des fichiers au projet

La commande **Projet** | **Ajouter fichier au projet** ajoute le fichier actif au projet actuel. Utiliser cette commande pour ajouter des fichiers à n'importe quel dossier dans votre projet. Vous pouvez soit sélectionner un seul fichier soit un groupe de fichiers (avec **Ctrl+ clic**) dans le dialogue Ouvrir. Si vous ajoutez des fichiers au projet, ils seront distribués parmi les dossiers respectifs basés sur les Extensions de type de fichier définies dans le dialogue Propriétés de projet.

© 2016-2022 Altova GmbH

13.3.8 Ajouter une Ressource globale au Projet

La commande **Projet** | **Ajouter Ressource globale au projet** permet d'ouvrir le dialogue Choisir Ressource globale, dans lequel vous pouvez choisir une ressource globale d'un fichier ou d'un type de dossier à ajouter au projet. Une fois qu'une ressource globale de type fichier a été sélectionnée, le fichier est ajouté au dossier approprié basé sur l'extension de type de fichier définie dans le dialogue Propriétés de projet. Si une ressource globale de type fichier a été sélectionnée, ce dossier sera ouvert dans un dialogue à fichiers ouverts et vous serez invité à choisir un fichier ; le fichier sélectionné sera ajouté au dossier approprié basé sur l'extension de type de fichier définie dans le dialogue Propriétés de projet. Pour une description de ressources globales, voir la section Ressources globales de cette documentation.

13.3.9 Ajouter une URL au projet

La commande **Projet** | **Ajouter URL au projet** ajoute une URL au projet actuel. Les URL contenues dans un projet entraînent l'inclusion de l'objet cible de l'URL dans le projet. Quelle que soit l'opération batch effectuée sur une URL ou dans un dossier qui contient un objet à URL, Authentic Desktop extrait le document depuis l'URL et exécute l'opération nécessaire.

13.3.10 Ajouter fichier actif au projet

La commande **Projet** | **Ajouter fichier actif au projet** ajoute le fichier actif au projet actuel. Si vous venez d'ouvrir un fichier depuis votre disque dur ou par le biais d'une URL, vous pouvez ajouter le fichier au projet actuel avec cette commande.

13.3.11 Ajouter des fichiers actifs et liés au projet

La commande **Projet** | **Ajouter fichiers actifs et liés à un projet** ajoute le document XML actuellement actif et tous les fichiers liés au projet. Lorsque vous travaillez sur un document XML basé sur un DTD ou un Schéma, cette commande ajoute non seulement le document XML mais aussi tous les fichiers liés (par exemple, le DTD et toutes les entités parsées externes auxquelles le DTD se réfère) au projet actuel.

Veuillez noter : Les fichiers référencés par les instructions de traitement (tels que les fichiers XSLT), ne sont pas considérés être des fichiers liés.

13.3.12 Ajouter un dossier de projet au projet

La commande **Projet** | **Ajouter Dossier de projet au projet** ajoute un nouveau dossier au projet actuel. Utiliser cette commande pour ajouter un nouveau dossier au projet actuel ou un sous-dossier à un dossier de projet. Vous pouvez aussi accéder à cette commande depuis le menu contextuel en cliquant avec la touche de droite sur un dossier dans la fenêtre de projet.

Note : Un dossier de projet peut être glissé et déposé dans un autre dossier de projet ou dans n'importe quel endroit dans le projet. Un dossier peut aussi être glissé depuis Windows (File) Explorer et déposé dans tout autre dossier de projet.

Note : Les dossiers de projet sont verts, alors que les dossiers externes sont jaunes.

13.3.13 Ajouter un dossier externe au projet

La commande **Projet** | **Ajouter dossier externe au projet** ajoute un nouveau dossier externe au projet actuel. Utiliser cette commande pour ajouter un dossier local ou de réseau au projet actuel. Vous pouvez aussi accéder à cette commande depuis le menu contextuel si vous cliquez avec la touche de droite un dossier dans la fenêtre de projet.

Note : les dossiers de externes sont jaunes, alors que les dossiers de projet sont verts.

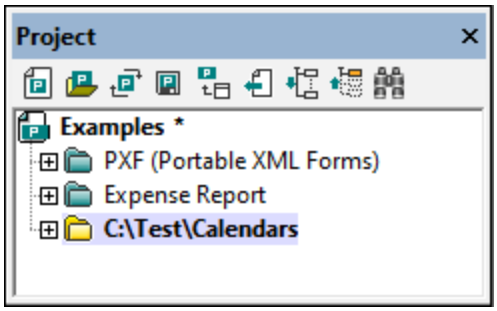
Note : les fichiers contenus dans des dossiers externes ne peuvent pas être placés sous le contrôle de source.

Ajouter des dossiers externes aux projets

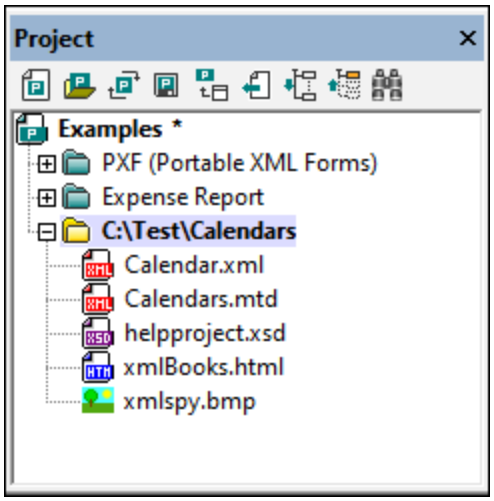
Pour ajouter un dossier externe au projet :

1. Choisir l'option de menu **Projet | Ajouter dossier externe au projet**.
2. Choisir le dossier que vous souhaitez inclure et cliquez sur **OK** pour confirmer. Le dossier sélectionné apparaîtra alors dans la fenêtre Projet.

© 2016-2022 Altova GmbH



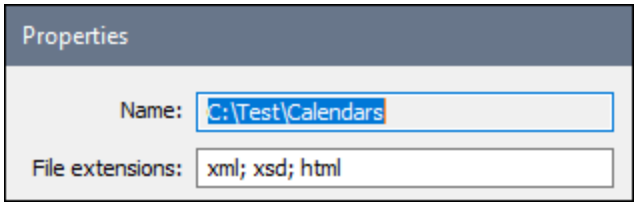
3. Cliquer sur l'icône plus pour consulter le contenu du dossier.



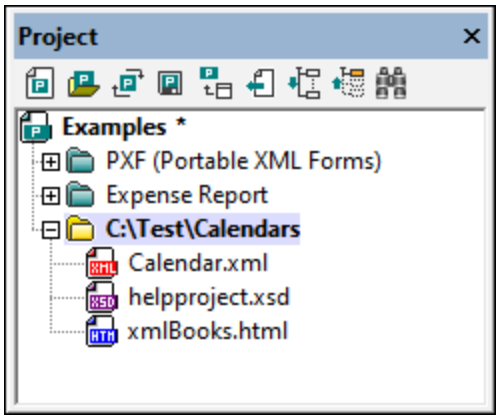
Filtrer les contenus des dossiers

Pour filtrer les contenus du dossier :

1. Cliquez avec la touche de droite sur le dossier externe que vous avez ajouté et sélectionnez les **Propriétés**. Cela permet d'ouvrir le dialogue Propriétés.



2. Cliquez dans le champ *extensions de fichier* et saisissez les extensions de fichier des types de fichier que vous voulez voir, séparant les types de fichiers par un point virgule (*voir la capture d'écran ci-dessus*).
3. Cliquer sur **OK** pour confirmer.



La fenêtre de projet ne montre plus que les types de fichiers sélectionnés.

Valider des dossiers externes

Pour valider et vérifier la bonne-formation d'un dossier externe :

1. Choisir les types de fichier que vous souhaitez voir ou vérifier depuis le dossier externe.
2. Cliquer sur le dossier et cliquer sur la commande de menu **XML | Vérifier la bonne formation** ou **Valider XML**(raccourcis **F7** or **F8**, respectivement). Tous les fichiers visibles sous le dossier sont vérifiés. Si un fichier est mal formé ou invalide, ce fichier sera ouvert dans la fenêtre principale et vous pourrez l'éditer.
3. Corriger l'erreur et exécuter le processus de validation une fois de plus et revérifier.

Mettre à jour un dossier de projet

Vous pouvez ajouter ou supprimer des fichiers dans le répertoire local ou de réseau à tout moment. Pour mettre à jour l'aperçu du dossier, cliquez avec la touche de droite sur le dossier externe et choisissez l'option de menu popup **Réinitialiser**.

Supprimer les dossiers externes et les fichiers contenus

Choisir un dossier externe et appuyer sur la touche **Supprimer** pour supprimer le dossier depuis la fenêtre Projet. Il est également possible de cliquer avec la touche de droite sur le dossier externe et choisir la commande **Supprimer**. Chacune de ces actions ne supprime que le dossier externe de la fenêtre de projet. Le dossier externe n'est pas supprimé du disque dur ou du réseau.

Pour supprimer un fichier dans un dossier externe, vous devez le supprimer physiquement du disque dur ou du réseau. Pour voir le changement dans le projet, réinitialiser les contenus du dossier externe (cliquer avec la touche de droite sur le dossier externe et choisir **Réinitialiser**).

Note : un dossier externe peut être glissé et déposé dans un dossier de projet ou dans tout autre emplacement dans le projet (mais pas dans un autre dossier externe). Aussi, un dossier externe peut être glissé de Windows (File) Explorer et déposé dans tout emplacement de la fenêtre de projet excepté dans un autre dossier externe.

© 2016-2022 Altova GmbH

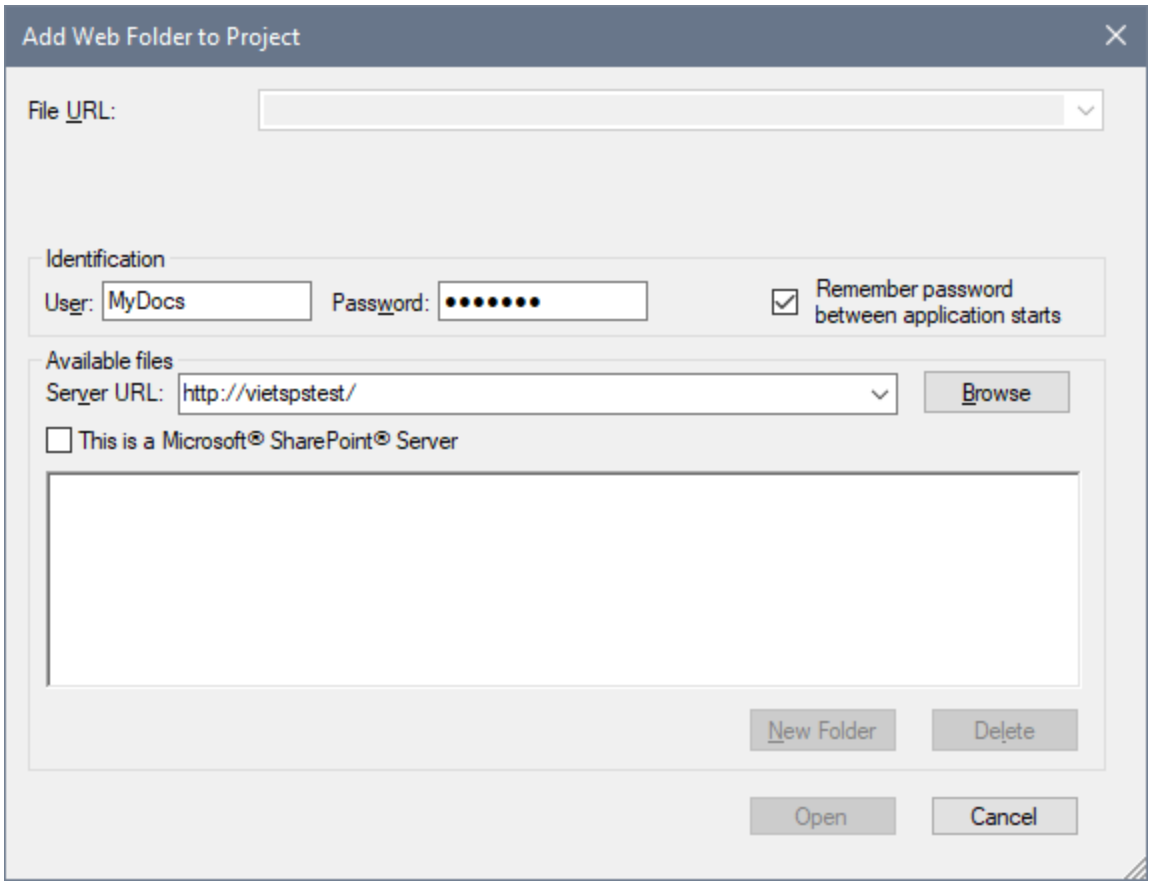
13.3.14 Ajouter un dossier web externe au projet

Cette commande ajoute un nouveau dossier Web externe au projet actuel. Vous pouvez aussi accéder à cette commande depuis le menu contextuel si vous cliquez avec la touche de droite un dossier dans la fenêtre de projet. Notez que les fichiers contenus dans des dossiers externes ne peuvent pas être placés sous le contrôle de source.

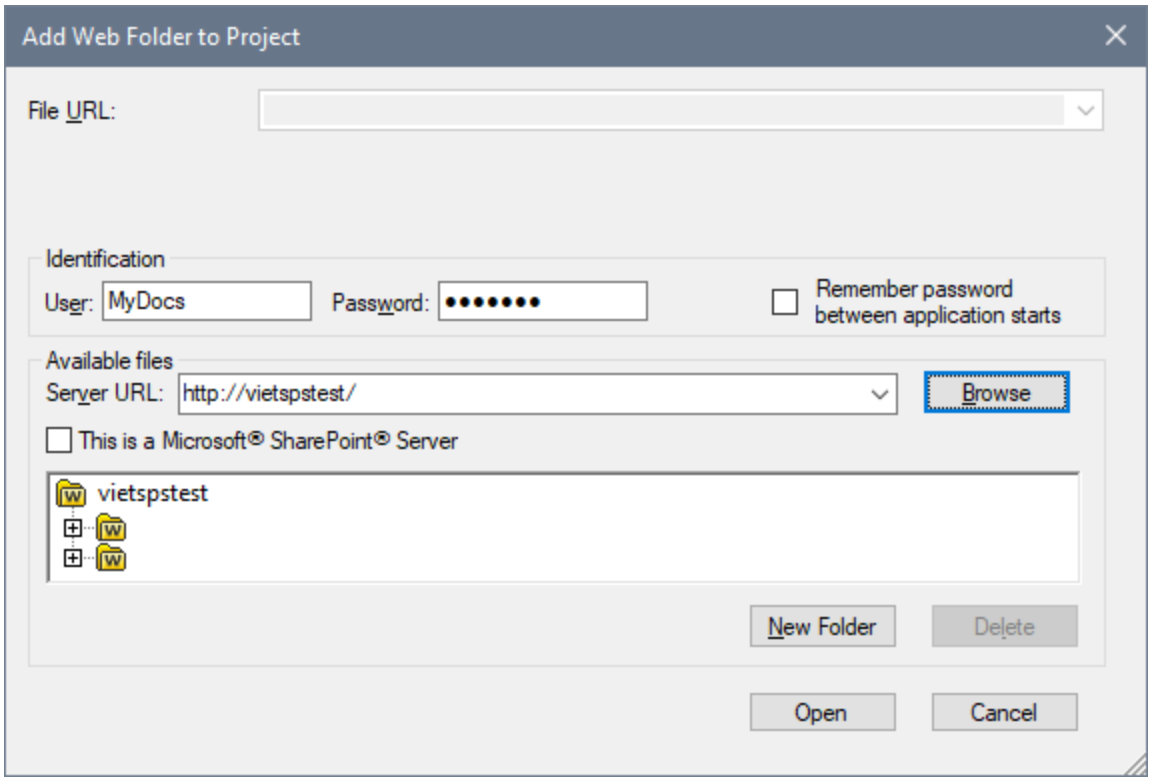
Ajouter un dossier Web externe au projet

Pour ajouter un dossier Web externe au projet, procédez comme suit :

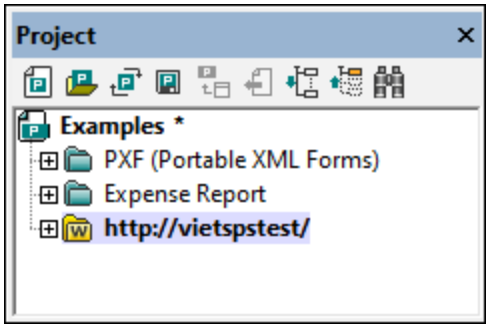
1. Choisir l'option de menu option **Projet | Ajouter dossier Web externe au projet**. Cela ouvre le dialogue Paramètres de requête SOAP (*capture d'écran ci-dessous*).



2. Cliquer dans le champ URL de Serveur et saisir l'URL de l'URL de serveur. Si le serveur est un Microsoft® SharePoint® Server, cochez cette option. Voir la section *Dossier sur un Microsoft® SharePoint® Server* ci-dessous pour plus d'informations concernant le travail avec des fichiers sur ce type de serveur.
3. Si le serveur est protégé par un mot de passe, veuillez saisir votre ID d'utilisateur et votre mot de passe dans les champs *Utilisateur* et *Mot de passe*.
4. Cliquez sur **Parcourir** pour vous connecter au serveur et consultez les dossiers disponibles.

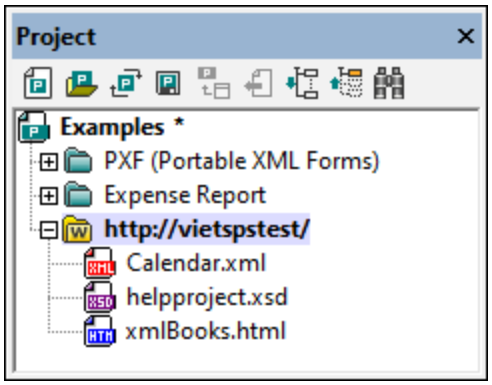


5. Cliquez sur le dossier que vous souhaitez ajouter au mode de projet. Le bouton **Ouvrir** ne s'activera qu'après avoir effectué cette étape. L'URL du dossier apparaît maintenant dans le champ URL de fichier.
6. Cliquez sur **Ouvrir** pour ajouter le dossier au projet.



7. Cliquez sur l'icône plus pour consulter le contenu du dossier.

© 2016-2022 Altova GmbH



Filtrer les contenus de dossier

Pour filtrer les contenus d'un dossier, cliquez avec la touche de droite sur le dossier et sélectionnez **Propriétés** depuis le menu contextuel. Dans le dialogue Propriétés qui s'ouvre, cliquez dans le champ *Extensions de fichier* et saisir les extensions de fichier des types de fichier que vous souhaitez voir (par exemple, des fichiers XML et XSD). Séparez chaque type de fichier avec un point-virgule (par exemple : `xml; xsd; sps`). La fenêtre de projet n'affichera que ce dossier avec des fichiers ayant une extension spécifiée.

Valider et vérifier la bonne formation d'un dossier

Pour vérifier les fichiers dans un dossier pour la bonne formation ou pour les valider, sélectionnez le dossier, puis cliquez sur l'icône de la commande de menu **XML | Vérifier la bonne formation** ou **XML | Valider XML** (raccourci **F7** ou **F8**, respectivement). Tous les fichiers visibles dans le dossier sont vérifiés. Si un fichier est mal formé ou invalide, ce fichier sera ouvert dans la fenêtre Principale et vous pourrez l'éditer. Corrigez l'erreur et redémarrez le processus pour revérifier le reste du dossier. Veuillez noter que vous pouvez sélectionner des fichiers discontinus dans le dossier en maintenant appuyée la touche **Ctrl** et en cliquant individuellement sur les fichiers. Seuls ces fichiers seront contrôlés lorsque vous appuierez sur **F7** ou **F8**.

Mettre à jour les contenus du dossier de projet

Des fichiers peuvent être ajoutés ou supprimés du dossier Web à tout moment. Pour mettre à jour l'aperçu du dossier, cliquez avec la touche de droite sur le dossier externe et choisissez l'option du menu contextuel **Réinitialiser**.

Supprimer les dossiers et les fichiers

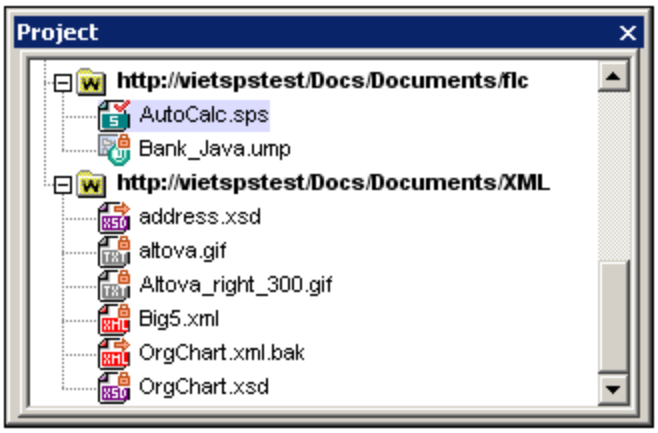
Puisque c'est le dossier Web qui a été ajouté au projet, seul le dossier Web peut être supprimé du projet (et non pas les fichiers s'y trouvant). Vous pouvez supprimer un dossier Web d'un projet, soit (i) en cliquant avec la touche de droite sur le dossier et en choisissant **Supprimer**, ou (ii) en choisissant le dossier et en appuyant sur la touche **Supprimer**. Cela supprime le dossier uniquement de l'aperçu Projet ; mais ne supprime rien du tout dans le serveur web.

Note : Double-cliquer sur un seul fichier et appuyer sur la touche **Supprimer** ne supprime pas un fichier depuis la fenêtre de projet. Vous devez le supprimer physiquement sur le serveur puis rafraîchir les contenus du dossier externe.

Les dossiers sur un Microsoft® SharePoint® Server

Lorsqu'un dossier dans un Microsoft® SharePoint® Server a été ajouté à un projet, les fichiers contenus dans le dossier peuvent être extraits et archivés par le biais de commandes dans le menu contextuel de la liste des fichiers dans la fenêtre Projet (*voir la capture d'écran ci-dessous*). Pour accéder ces commandes, cliquez avec la touche de droite sur le fichier sur lequel vous voulez travailler et sélectionnez la commande souhaitée (**Check Out**, **Check In**, **Undo Check Out**).

L'ID de l'utilisateur et le mot de passe peuvent être enregistrés dans les propriétés des dossiers individuels dans le projet, vous permettant ainsi de sauter le processus de vérification à chaque fois que le serveur est accédé.



Dans la fenêtre de projet (*capture d'écran ci-dessous*), les icônes de fichier ont des symboles qui indiquent le statut d'archivage/de récupération des fichiers. Les différents icônes de fichier sont affichés ci-dessous :

	Archivé. Disponible pour extraction.
	Extrait par un autre utilisateur. Non disponible pour une extraction.
	Extrait localement. Peut être édité et archivé.

Veuillez noter les points suivants :

- Une fois avoir extrait un fichier, vous pouvez l'éditer dans votre application Altova et l'enregistrer avec **Fichier | Enregistrer (Ctrl+S)**.
- Vous pouvez archiver le fichier édité par le biais du menu contextuel dans la fenêtre Projet (*voir la capture d'écran ci-dessus*), ou par le biais du menu contextuel qui s'ouvre lorsque vous cliquez avec la touche de droite dans l'onglet Fichier de la fenêtre principale de votre application (voir la *capture d'écran ci-dessous*).

- Lorsqu'un fichier est extrait par un autre utilisateur, il ne peut pas être extrait.
- Lorsqu'un fichier est extrait localement par vous, vous pouvez annuler l'extraction avec la commande Annuler l'extraction dans le menu contextuel. Cela a pour effet de retourner le fichier dans le serveur sans les modifications.

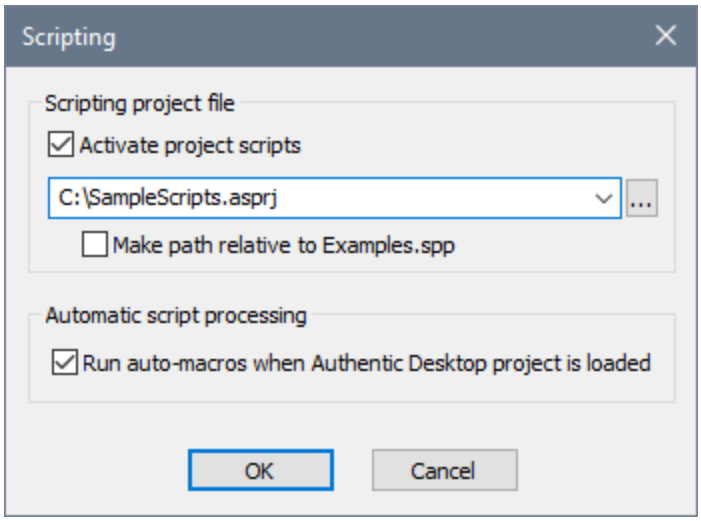
© 2016-2022 Altova GmbH

- Si vous extrayez un fichier dans une des applications Altova, vous ne pouvez pas l'extraire dans une autre application Altova. Le fichier est considéré être déjà extrait par vous. Les commandes disponibles dans ce cas pour les applications Altova prenant en charge Microsoft® SharePoint® Server seront : **Archiver** et **Annuler l'extraction**.

13.3.15 Paramètres du script

Un projet de script est attribué à un projet de Authentic Desktop comme suit :

1. Dans la GUI de Authentic Desktop, ouvrir le projet d'application exigé.
2. Choisir la commande de menu **Projet | Paramètres de script**. Le dialogue Script (*capture d'écran ci-dessous*) s'ouvre.



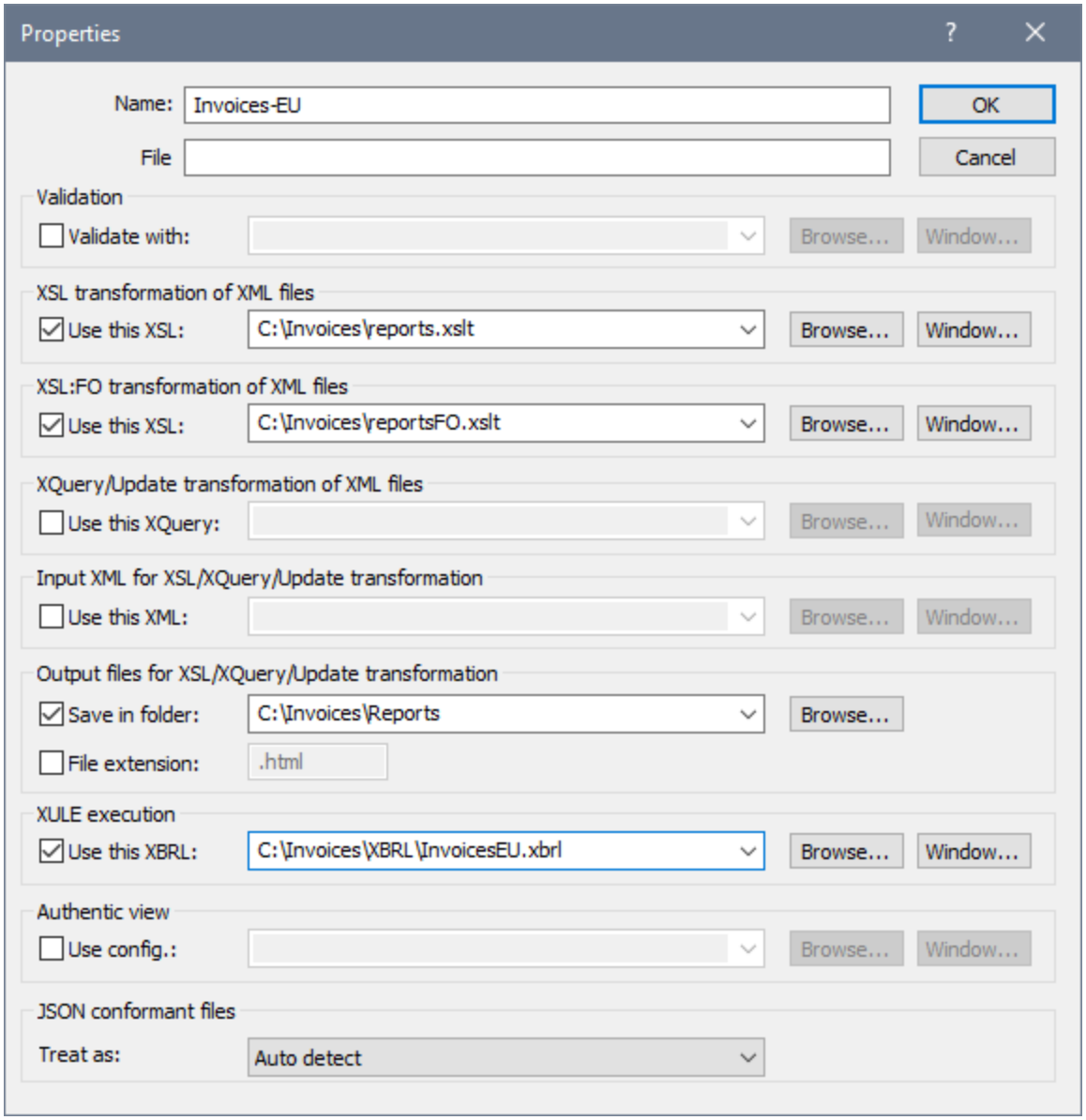
3. Cochez la case *Activer les scripts de projet* et sélectionnez le projet de script requis (fichier `.asprj`). Si vous souhaitez exécuter des auto-macros lorsque le projet de Authentic Desktop est chargé, cocher la case *Exécuter auto-macros*.
4. Cliquez sur **OK** pour finir.

Note : Pour désactiver (donc, désattribuer) le projet de script d'un projet Authentic Desktop, décochez la case *Activer scripts de projet*.

13.3.16 Propriétés

La commande **Projet** | **Propriétés de projet** ouvre le dialogue Propriétés (*capture d'écran ci-dessous*) du projet actif. Si vous cliquez avec la touche de droite sur un dossier dans la fenêtre Projet (contrairement au dossier de projet) et que vous sélectionnez **Propriétés**, le dialogue Propriétés de ce dossier est ouvert. Les paramètres de dialogue sont décrits ci-dessous.

Note : Si votre fichier de projet se trouve sous contrôle de source, une invite apparaîtra vous demandant si vous souhaitez extraire le fichier de projet (`.spp`). Cliquez sur **OK** si vous voulez éditer les paramètres et pouvoir les enregistrer.



Paramètres

Extensions de fichier
Le paramètre *Extensions de fichier* est activé pour les dossiers individuels, et non pas pour le dossier de projet. Lorsqu'un fichier est ajouté à un projet, il sera ajouté au dossier pour lequel son extension de fichier a été définie. Par exemple, si un fichier nommé `MyReport.xml` est ajouté au projet. Si les extensions de fichier `.xml` ont été définies dans le dossier `Invoices-EU` (tel qu'affiché dans la capture d'écran ci-dessus), alors

© 2016-2022 Altova GmbH

`MyReport.xml` sera ajouté au dossier `Invoices-EU`. S'il y a plus d'un dossier auquel vous souhaitez ajouter des fichiers XML, vous devriez ajouter des fichiers XML individuels directement au dossier (au lieu du projet).

ID d'utilisateur et mot de passe pour les dossiers externes
Dans les dossiers externes (y compris les dossiers Web externes), vous pouvez enregistrer l'ID d'utilisateur et le mot de passe qui peuvent être nécessaires pour accéder au serveur.

Validation
Le DTD, Schéma XML, ou schéma JSON qui doit être utilisé pour valider les fichiers dans le dossier actuel (ou tout le projet si les propriétés sont celles du projet).

Transformation XSL des fichiers XML
La feuille de style XSLT à utiliser pour la transformation XSLT des fichiers XML dans le dossier.

Transformation XSL-FO des fichiers XML
La feuille de style XSLT pour transformer des fichiers XML dans le dossier pour XSL-FO.

Transformation XQuery/Update des fichiers XML
Le fichier XQuery ou XQuery Update à utiliser pour des exécutions XQuery ou des exécutions XQuery Update de fichiers XML dans le dossier.

XML d'entrée pour la transformation XSL/XQuery/Update de fichiers XML
Le fichier XML à utiliser en tant qu'entrée pour les transformations XSLT ou les exécutions XQuery/XQuery Update avec les fichiers XSLT, XQuery ou XQuery Update respectifs dans le dossier.

Fichiers de sortie pour la transformation XSL/XQuery/Update
Le répertoire de destination des transformations, et, en option, l'extension de fichier du document de résultat.

Exécution XULE
Le fichier d'instance XBRL à traiter avec le document XULE qui est actif dans la fenêtre d'application XMLSpy.

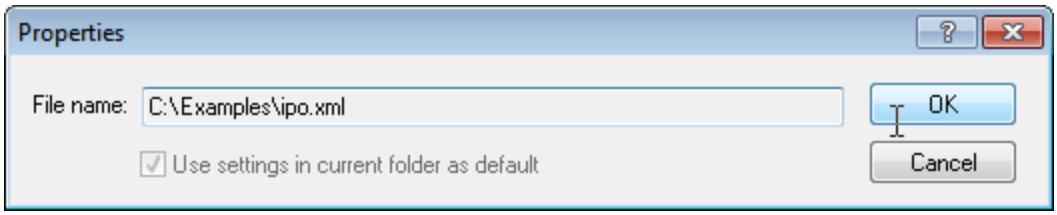
Mode Authentic
Utiliser config spécifie le StyleVision Power Stylesheet (fichier SPS) à utiliser pour l'affichage Authentic View des fichiers XML dans le dossier. Veuillez noter que le fichier XML doit être valide par rapport au même schéma utilisé pour le SPS.

Fichiers conformes à JSON
Cette propriété spécifie si un dossier de projet contient des fichiers de schéma JSON ou des fichiers d'instance JSON. Il peut être très utile d'aider à identifier les fichiers de schéma JSON si les fichiers ne sont pas clairement identifiés en tant que de fichier de schéma JSON par le mot-clé **`$schema`** et les fichiers se référencent l'un l'autre. Vous pouvez les définir comme *Instance JSON, Schéma JSON,* ou *Auto detect*. Le paramètre par défaut de *Auto-detect* amènerait XMLSpy à vérifier la structure et le contenu des fichiers JSON pour déterminer son type.

Notes à propos des propriétés du projet

Notez les points suivants sur la précédence quand, au sein d'une hiérarchie, différents fichiers sont spécifiés pour être utilisés pour des procédures telles que la validation ou la transformation :

- Lorsque des validations ou des transformations XSLT/XQuery sont effectuées par le biais des menus contextuels du dossier de projet, la validation ou les fichiers de transformation spécifiés dans ce dialogue prévalent sur toute attribution dans le fichier XML.
- De même, les paramètres spécifiés pour les dossiers de projet individuels prévalent sur les paramètres définis pour les dossiers ancêtre.
- Si un fichier est présent dans plusieurs dossiers du projet et que plusieurs fichiers de validation ou de transformation lui ont été attribués dans des dossiers différents, vous pouvez définir quelle attribution utiliser lorsque le fichier est traité en dehors du projet. Spécifiez cela comme suit : Trouver dans le dossier du projet le fichier dont vous souhaitez utiliser l'attribution. Cliquer avec la touche de droite dans ce dossier de projet et choisir **Propriétés**. Dans le dialogue qui apparaît (*capture d'écran ci-dessous*), choisir *Utiliser les paramètres dans le dossier actuel en défaut*. (Le dossier actuel est le dossier de projet dans lequel se trouve le fichier.) Si l'option est désactivée, cela signifie que les paramètres du dossier actuel sont déjà sélectionnés en tant que les paramètres par défaut à utiliser. Si vous choisissez une instance de fichier qui se trouve dans un dossier de projet qui n'est pas l'instance par défaut, l'option sera activée et vous pouvez faire passer les paramètres par défaut pour en faire les paramètres de ce dossier. Veuillez noter que si le fichier a une attribution locale (c'est-à-dire une attribution dans le fichier lui-même), alors l'attribution locale sera utilisée et les paramètres de dossier par défaut seront ignorés.



13.3.17 Projets les plus récents

Cette commande affiche le nom de fichier et le chemin pour les neuf derniers projets utilisés, permettant un accès rapide à ces fichiers.

Notez aussi que Authentic Desktop peut automatiquement ouvrir le dernier projet que vous avez utilisé dès que vous lancez Authentic Desktop. (Section **Outils | Options | Fichier**, Projet | Ouvre le dernier projet au démarrage du programme).

© 2016-2022 Altova GmbH

13.4 Menu XML

Le menu **XML** contient des commandes qui sont communément requises lorsqu'on travaille avec des documents XML. Parmi les tâches XML les plus fréquemment utilisées font partie les vérifications pour la bonne formation des documents et la validité des documents XML. Les commandes pour ces tâches sont dans le menu.

13.4.1 Check Well-Formedness

F7

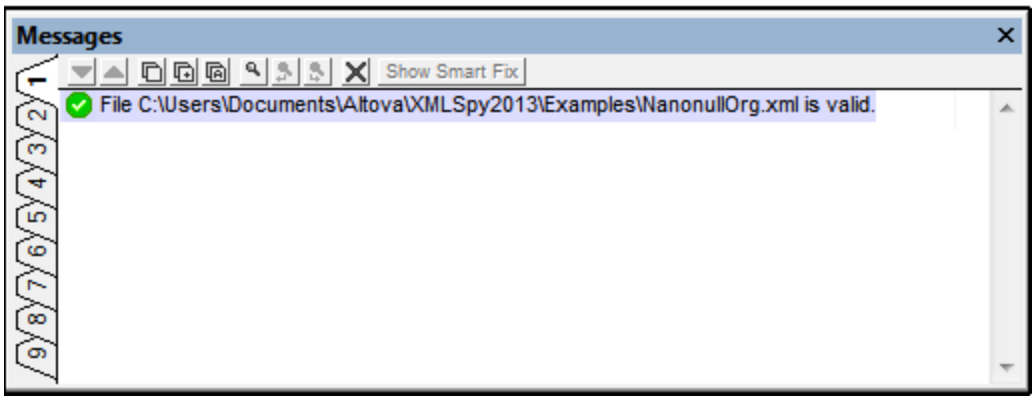
La commande **XML | Vérification de la bonne formation (F7)** vérifie le document actif pour la bonne formation avec les définitions de la spécification XML 1.0. Chaque document XML **doit** être bien formé. Authentic Desktop vérifie la bonne formation dès qu'un document est ouvert, rechargé ou enregistré. Si un document n'est pas bien formé, ceci sera automatiquement rapporté dans la fenêtre des Messages et le document XML sera affiché dans le mode Texte. Vous pouvez corriger l'erreur dans le mode View, puis retourner à Authentic View.

13.4.2 Valider XML

F8

La commande **XML | Valider (F8)** vous permet de valider des documents XML par rapport aux DTD, Schémas XML et autres schémas. Vous pouvez spécifier qu'un document soit automatiquement validé lorsqu'un fichier est ouvert ou enregistré (**Outils | Options | Fichier**). Notez que vous pouvez aussi valider quand vous éditez si vous activez à la commande **Valider sur Édition**.

Si le document est valide, un message apparaîtra dans la fenêtre Messages (*voir la capture d'écran ci-dessous*). Sinon, un message qui décrit l'erreur est affiché. Vous pouvez cliquer sur les liens dans le message d'erreur pour passer au nœud dans le document XML où l'erreur a été trouvée Si vous corrigez une erreur, vous devrez exécuter la commande **Valider (F8)** à nouveau pour vous assurer que l'erreur a bien été réparée.



Note : la fenêtre Messages a neuf onglets. Le résultat de validation est toujours affiché dans l'onglet actif. Donc vous pouvez valider un document XML dans l'onglet 1 et conserver le résultat dans l'onglet. Pour valider un second document, passer à l'onglet 2 (ou l'onglet 3 si vous préférez) avant d'effectuer le contrôle. Si vous ne souhaitez pas changer d'onglets, l'onglet 1 (ou l'onglet actif) sera écrasé avec les résultats de la dernière validation.

Valider depuis la fenêtre Projet

La commande **Valider** peut aussi s'appliquer à un fichier, dossier ou un groupe de fichiers dans le projet actif. Choisir le fichier ou le dossier requis dans la fenêtre Projet (en cliquant dessus). Puis, cliquez sur **XML | Valider** ou **F8**. Les fichiers invalides dans un projet seront ouverts et rendus actifs dans la Fenêtre principale et le message d'erreur *Le fichier n'est pas valide* sera affiché.

Validation automatique avec Raptor XML 2023

RaptorXML est l'application autonome d'Altova pour une validation XML, une transformation XSLT et une transformation XQuery. Elle peut être utilisée depuis la ligne de commande, par le biais d'une interface COM, dans des programmes Java et dans les applications .NET. Les tâches de validation peuvent donc être automatisées avec l'utilisation de RaptorXML. Par exemple, vous pouvez créer un fichier batch qui appelle RaptorXML pour effectuer une validation sur un ensemble de documents et envoie la sortie dans un fichier texte. Voir la documentation RaptorXML pour tout détail.

13.4.3 Valider sur Édition

La commande **Valider sur Édition** active/désactive le mode*Valider sur Édition*, qui active la validation dès que vous saisissez Authentic View. Le mode peut également être activé/désactivé par le biais de la touche de la barre d'outils de la commande ou de l'option *Validation* > *Sur Édition* de la Section fichier du dialogue des options.

© 2016-2022 Altova GmbH

13.5 Menu XSL/XQuery

Le langage de transformation XSL vous permet de spécifier comment un document XML devrait être converti en d'autres documents XML ou fichiers de texte. Un type de document XML qui est généré avec un document XSLT est un document FO, qui peut ensuite être traité pour générer une sortie PDF. Authentic Desktop est doté de processeurs XSLT intégrés (pour XSLT 1.0, XSLT 2.0, et XSLT 3.0) et peut être lié à un processeur FO dans votre système pour transformer des fichiers XML et générer toute une variété de sorties. L'emplacement du processeur FO doit être spécifié dans la section XSL du dialogue Options (Outils | Options) afin de pouvoir l'utiliser directement depuis l'interface Authentic Desktop.

Les caommandes suivantes sont disponibles dans le menu XSL/XQuery :

- Transformation XSL
- Transformations XSL-FO
- Paramètres XSL / Variables XQuery

13.5.1 Transformation XSL

F10

La commande **XSL/XQuery | Transformation XSL** transforme un document XML utilisant une feuille de style XSLT attribuée. La transformation peut être exécutée en utilisant le built-in approprié Altova XSLT Engine (Altova XSLT 1.0 Engine pour les feuilles de style XSLT 1.0 ; Altova XSLT 2.0 Engine pour les feuilles de style XSLT 2.0 ; Altova XSLT 3.0 Engine pour les feuilles de style XSLT 3.0), le module MSXML fourni par Microsoft, ou un processeur XSLT externe. Le processeur utilisé pour la transformation est spécifié dans la section XSL du dialogue Options (**Outils | Options**).

Si votre document XML contient une référence vers une feuille de style XSLT, alors cette feuille de style est utilisée pour la transformation. (Si le document XML fait partie d'un projet, une feuille de style XSLT peut être spécifiée sur une base par-dossier dans le dialogue Propriétés de projet. Cliquez avec la touche de droite dans le/s dossier/s ou fichier/s du projet que vous souhaitez transformer et sélectionnez une Transformation XSL.) Si une feuille de style XSLT n'a pas été assignée vers un fichier XML, vous êtes invité à utiliser la feuille de style XSLT. Vous pouvez aussi sélectionner un fichier par le biais de la ressource globale ou d'une URL (cliquer sur la touche **Chercher**) ou sur un fichier dans une des fenêtres ouvertes dans XMLSpy (cliquer sur la touche **Fenêtre**).

Transformations automatisées avec RaptorXML

RaptorXML est l'application autonome d'Altova pour une validation XML, une transformation XSLT et une transformation XQuery. Elle peut être utilisée depuis la ligne de commande, par le biais d'une interface COM, dans des programmes Java et dans les applications .NET. Les tâches d'exécution XQuery peuvent donc être automatisées avec l'aide de RaptorXML. Par exemple, vous pouvez créer un fichier batch qui invite RaptorXML à exécuter des transformations sur un ensemble de documents et envoie la sortie vers un fichier de texte. Voir la documentation RaptorXML pour tout détail.

Transformations en fichiers ZIP

Afin d'appliquer la sortie dans un fichier ZIP, y compris des fichiers Open Office XML (OOXML) tels `.docx`, un doit spécifier un protocole ZIP dans le chemin de fichier du fichier de sortie. Par exemple :

```
filename.zip|zip/filename.xxx

filename.zip|zip/filename.xxx
```

Note : La structure de répertoire pourrait devoir être créée avant d'exécuter la transformation. Si vous générez des fichiers pour une archive Open Office XML, vous devrez zipper les fichiers d'archive afin de créer un fichier OOXML de niveau supérieur (par exemple, `.docx`).

13.5.2 Transformation XSL-FO

Ctrl+F10

FO est un format XML qui décrit les documents paginés. Un processeur FO, tel que le FOP du projet Apache XML, prend un fichier FO comme entrée et génère un PDF comme sortie. La production d'un document PDF depuis un document XML est, pour cette raison, un processus à deux étapes.

1. Le document XML est transformé en un document FO utilisant une feuille de style XSLT.
2. Le document FO est traité par un processeur FO pour générer le PDF (ou une sortie alternative).

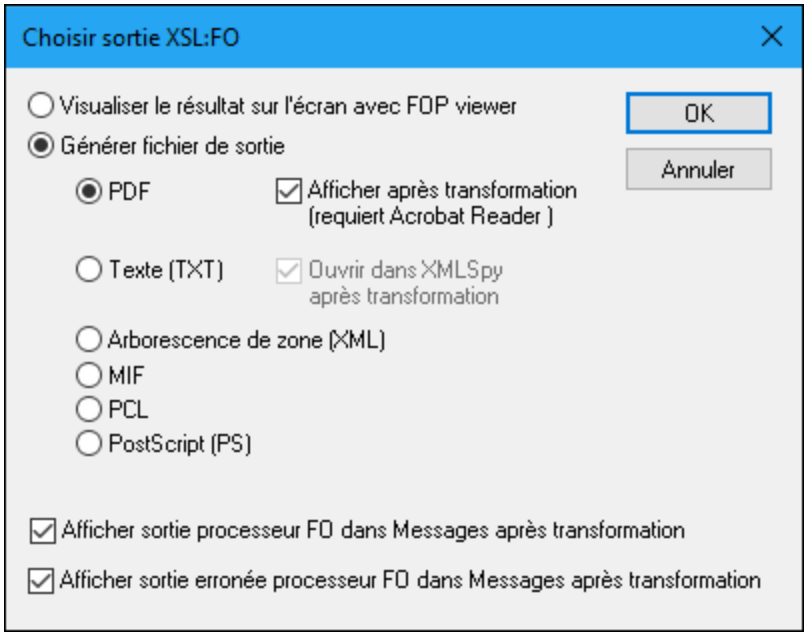
La commande **XSL/XQuery | Transformation XSL:FO** transforme un document XML ou un document FO en un PDF.

- Si la commande **Transformation XSL:FO** est exécutée dans un document XML source, alors les deux étapes énumérées ci-dessus sont exécutées, en séquence, l'une après l'autre. Si la feuille de style XSLT requise pour transformer en FO n'est pas référencée dans le document XML, vous êtes invité à en attribuer une pour la transformation. Notez que vous pouvez aussi sélectionner un fichier par le biais de la ressource globale ou d'une URL (cliquer sur la touche **Chercher**) ou sur un fichier dans une des fenêtres ouvertes dans Authentic Desktop (cliquer sur la touche **Fenêtre**). La transformation de XML en XSL-FO est exécutée par le processeur XSLT spécifié dans la section XSL du dialogue des Options (**Outils | Options**). Par défaut, le processeur XSLT sélectionné est le processeur built-in XSLT de Authentic Desktop. Le document FO résultant est directement traité avec le processeur FO spécifié dans la section XSL du dialogue des Options (**Outils | Options**).
- Si la commande **Transformation XSL:FO** est exécutée dans un document FO, alors le document est traité avec le processeur FO spécifié dans la section XSL du dialogue des Options (**Outils | Options**).

Sortie de transformation XSL:FO

La commande **Transformation XSL:FO** ouvre le dialogue Choisir Sortie XSL:FO (*capture d'écran ci-dessous*). (Si le document actif est un document XML sans attribution XSLT, un fichier XSLT vous sera d'abord demandé.)

© 2016-2022 Altova GmbH



Vous pouvez afficher la sortie du processeur FO directement à l'écran utilisant le mode FOP ou vous pouvez générer un fichier de sortie dans un des formats suivants : PDF, texte, une arborescence de zone XML, MIF PCL, ou PostScript. Vous pouvez aussi activer des messages du processeur FO pour afficher (i) le message de sortie standard du processeur dans la fenêtre Messages ; et (ii) les messages d'erreur du processeur dans la fenêtre Messages. Pour activer soit une de ces deux options, cochez la case à cocher appropriée en bas du dialogue.

Veuillez prendre note des points suivants :

- À moins d'avoir décoché l'option pour installer le moteur FOP du Projet Apache XML, celui-ci sera installé dans le dossier `C:\ProgramData\Altova\SharedBetweenVersions`. Si installé, le chemin y menant aura été saisi automatiquement dans la section XSL du dialogue des Options (**Outils | Options**) comme processeur FO à utiliser. Vous pouvez déterminer le chemin vers n'importe lequel des moteurs FO que vous souhaitez utiliser.
- La commande de transformation XSL:FO ne peut pas uniquement être utilisée sur le fichier actif dans la fenêtre principale mais aussi dans tout fichier ou dossier que vous sélectionnez dans un projet actif. À cette fin, cliquez avec la touche de droite et sélectionnez **Transformation XSL:FO**. La feuille de style XSLT attribuée au dossier de projet sélectionné est utilisée.

13.5.3 Paramètres XSL / Variables XQuery

La commande **XSL/XQuery | Paramètres XSL / Variables XQuery** ouvre le dialogue Paramètres d'entrée XSLT / Variables externes XQuery (*voir la capture d'écran*). Vous pouvez saisir le nom d'un ou de plusieurs paramètres que vous souhaitez passer à la feuille de style XSLT, ou une ou plusieurs variables externes XQuery que vous souhaitez passer au document XQuery et leurs valeurs respectives. Ces paramètres sont utilisés comme suit dans Authentic Desktop:

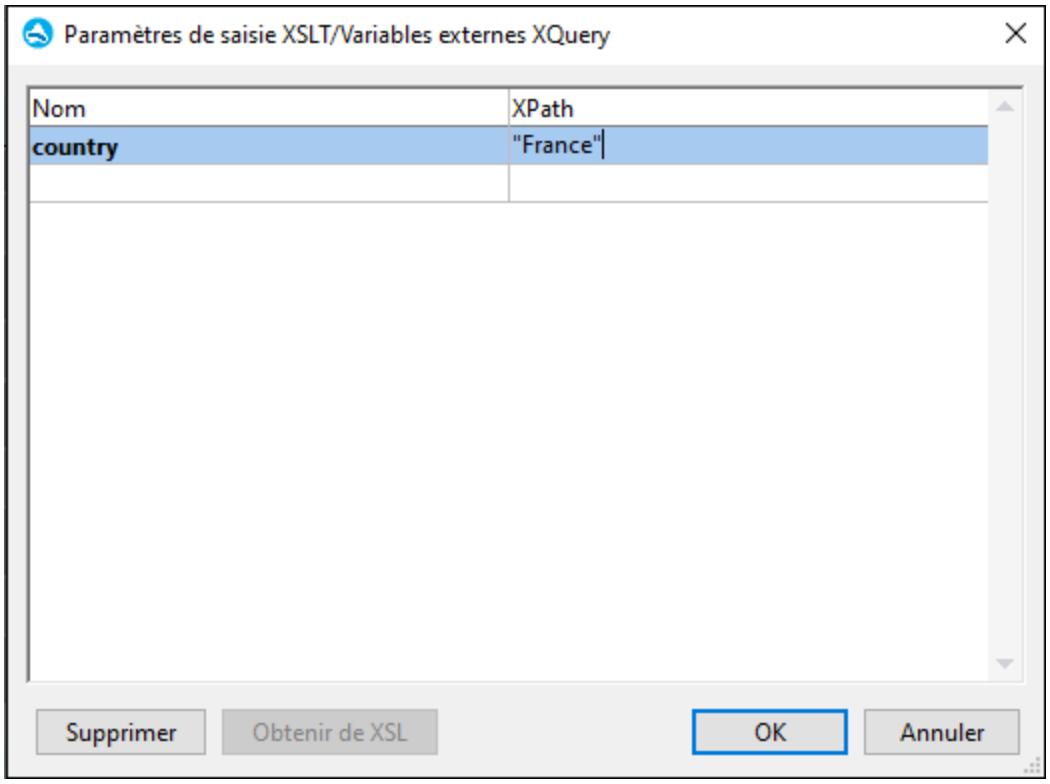
- Lorsque la commande **Transformation XSL** dans le menu XSL/XQuery est utilisée pour transformer un document XML, les valeurs de paramètre actuellement enregistrées dans le dialogue sont passées dans le document XSLT sélectionné et sont utilisées pour la transformation.
- Lorsque la commande **Exécution XQuery** dans le menu XSL/XQuery est utilisée pour traiter un document XQuery, les valeurs de variables externes XQuery actuellement enregistrées dans le dialogue sont passées dans le document XQuery pour l'exécution.

Note : les paramètres ou les variables que vous saisissez dans le dialogue Paramètres de saisie XSLT /Variables externes XQuery sont uniquement transférés sur le moteur XSLT intégré Altova. C'est pourquoi, si vous utilisez MSXML ou un autre moteur externe que vous avez configuré, ces paramètres ne sont pas transférés à ce moteur.

Note : Il ne s'agit pas d'une erreur si un paramètre XSLT ou une variable XQuery externe est définie dans le dialogue Paramètres de saisie XSLT /Variables externes XQuery mais il n'est pas utilisé dans le document XSLT/XQuery ou dans la transformation.

Utiliser les paramètres XSLT

La valeur que vous saisissez pour le paramètre est une expression XPath. Notez que les strings de texte dans XPath sont délimités par des guillemets.



Une fois qu'un ensemble de valeurs de paramètres est saisi dans le dialogue, il est utilisé pour toutes les transformations ultérieures jusqu'à ce qu'il soit supprimé explicitement ou que l'application soit redémarrée. Les paramètres saisis dans le dialogue sont spécifiés au niveau de l'application, et seront passés dans le document XSLT respectif pour chaque transformation effectuée par le biais de l'IDE à partir de ce point. Cela signifie que :

© 2016-2022 Altova GmbH

- Les paramètres ne sont pas associés à un document particulier
- Chaque paramètre saisi dans le dialogue est supprimé une fois que Authentic Desktop a été fermé.

Note : La touche **Obtenir de XSL** est activée dans Authentic View lorsqu'un document XSLT fait partie du document actif. Elle insère des paramètres déclarés dans le document XSLT actif à l'intérieur du dialogue ensemble avec les valeurs par défaut de ce paramètres.

Exemple d'utilisation pour les paramètres XSLT

Nous avons un document XML qui contient les noms de pays et leurs capitales respectives :

```
<document>
   <countries>
      <country name="USA" capital="Washington DC"/>
      <country name="UK" capital="London"/>
      <country name="France" capital="Paris"/>
      <country name="Russia" capital="Moscow"/>
      <country name="China" capital="Beijing"/>
   </countries>
</document>
```

Le document XSLT suivant générera un document XML qui affiche un pays depuis le fichier XML ensemble avec la capitale du pays. Le pays est sélectionné en saisissant le nom en tant que valeur du paramètre appelé **`country`** (*affiché en surbrillance jaune ci-dessous*).

```
<xsl:stylesheet version="2.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
   <xsl:param name="country" select="'USA'"/>
   <xsl:template match="countries">
      <xsl:for-each select="country[@name=$country]">
         <country>
            <name><xsl:value-of select="$country"/></name>
            <capital><xsl:value-of select="@capital"/></capital>
         </country>
      </xsl:for-each>
   </xsl:template>
</xsl:stylesheet>
```

Lorsque ce document XSLT est exécuté dans le document XML recensé ci-dessus, le résultat sera :

```
<country><name>USA</name><capital>Washington DC</capital></country>
```

Maintenant, si vous créez un paramètre dans le dialogue Paramètre d'entrée XSLT / Variables externes XQuery appelé **`country`** et lui donner une valeur (*voir la capture d'écran ci-dessus*), alors cette valeur sera passée au paramètre **`country`** dans la feuille de style XSLT pour la transformation. De cette manière, vous pouvez passer différentes valeurs à différents paramètres lors de l'exécution.

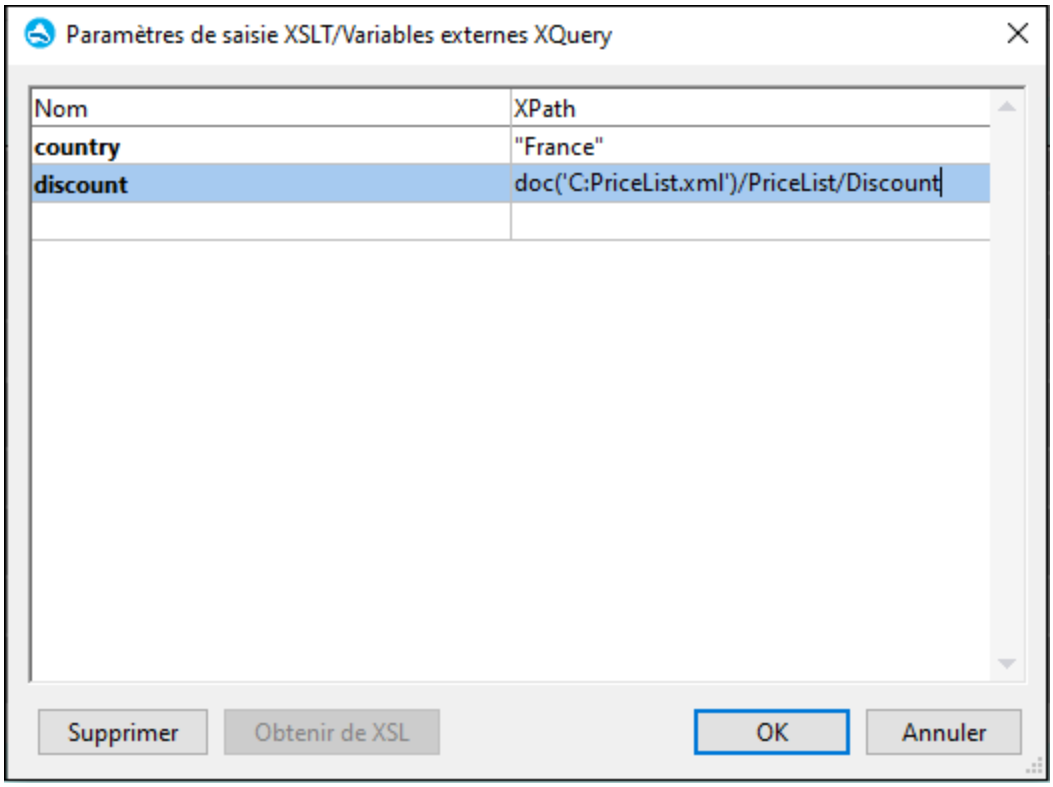
Veuillez noter les points suivants :

- Si vous utilisez la commande **Transformation XSL:FO** (**XSL/XQuery | XSL:FO Transformation), alors les paramètres saisis dans le dialogue Paramètres d'entrée XSLT / Variables externes XQuery** ne sont pas passés à la feuille de style. Si ces paramètres doivent être utilisés dans la sortie PDF, transformez d'abord de XML à FO utilisant la commande de transformation XSLT (**XSL/XQuery | XSL Transformation**), puis transformez FO en PDF utilisant la commande **XSL:FO Transformation** (**XSL/XQuery | XSL:FO Transformation**).

- Si vous utilisez un processeur XSLT autre que que les moteurs Altova XSLT intégrés, les paramètres que vous saisissez dans le dialogue Paramètres d'entrée ne seront pas passés au processeur externe.

Utiliser les variables XQuery externes

La valeur que vous saisissez pour des variables XQuery externes pourrait être une expression XPath sans guillemets ou un string de texte délimité par des guillemets. Le type de données de la variable externe est spécifié dans la déclaration variable dans le document XQuery.



Une fois qu'un ensemble de variables XQuery externe est saisi dans le dialogue, celles-ci sont utilisées pour toutes les exécutions ultérieures jusqu'à ce qu'elles soient explicitement supprimées ou que l'application est redémarrée. Les variables saisies dans le dialogue sont spécifiées au niveau de l'application, et seront passées dans le document XQuery respectif pour chaque exécution effectuée par le biais de l'IDE à partir de ce point. Cela signifie que :

- Les variables ne sont pas associées à un document particulier
- Chaque variable saisie dans le dialogue est supprimée une fois que Authentic Desktop a été fermé.

Exemple d'utilisation pour des variables XQuery externes

Dans l'exemple suivant, une variable `$first` est déclarée dans le document XQuery et est ensuite utilisée dans la clause de retour de l'expression FLWOR :

```
xquery version "1.0";
déclarer variable $first comme xs:string externe ;
let $last := "Jones"
```


© 2016-2022 Altova GmbH

```
return concat($first, " ", $last )
```

Cet XQuery renvoie `Peter Jones` si la valeur de la variable externe (saisie dans le dialogue Paramètres d'entrée XSLT / Variables externes XQuery) est `Peter`. Veuillez noter les points suivants :

- Le mot-clé `externe` dans la déclaration de variable dans le document XQuery indique que cette variable est une variable externe.
- Définir le type statique de la variable est optionnel. Si le type de données pour la variable n'est pas spécifiée dans la déclaration de variable, alors la valeur de variable obtient le type `xs:untypedAtomic`.
- Si une variable externe est déclarée dans le document XQuery, mais qu'aucune variable de ce nom n'est passée au document XQuery, alors une erreur est rapportée.
- Si une variable externe est déclarée et est saisie dans le dialogue Paramètres d'entrée XSLT / Variables externes XQuery, alors elle est considérée avoir l'étendue pour que le document XQuery soit exécuté. Si une nouvelle variable avec le nom est déclarée à l'intérieur du document XQuery, la nouvelle variable écrase temporairement la variable externe in-scope. Par exemple, le document XQuery ci-dessous renvoie `Paul Jones` même si la variable externe in-scope `$first` a une valeur `Peter`.

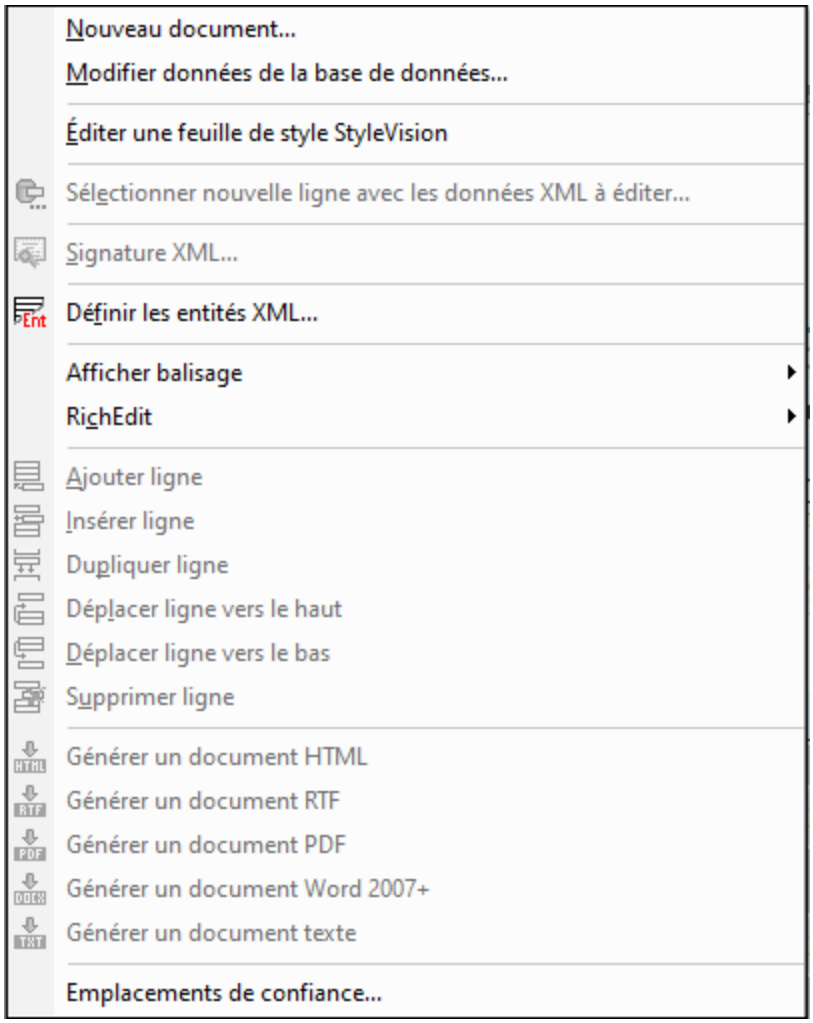
```
xquery version "1.0";
déclarer variable $first comme xs:string externe ;
let $first := "Paul"
let $last := "Jones"
return concat($first, " ", $last )
```

13.6 Menu Authentic

Authentic View vous permet d'éditer des documents XML **basés sur des StyleVision Power Stylesheet (fichiers .sps) créés dans le produitStyleVision d'Altova !** Ces feuilles de style contiennent des informations qui permettent à un fichier XML d'être affichés graphiquement dans Authentic View. Outre le fait de contenir des informations d'affichage, les StyleVision Power Stylesheet permettent aussi d'écrire des données dans le fichier XML. Ces données sont traitées dynamiquement à l'aide des capacités disponibles dans les feuilles de style XSLT et la sortie est instantanément produite dans Authentic View.

De plus, les StyleVision Power Stylesheet peuvent être créées pour afficher un mode XML éditable d'une base de données. La StyleVision Power Stylesheet contient des informations pour se connecter à la base de données, affichant des données provenant de la base de données dans Authentic View, et enregistre des données dans la base de données.

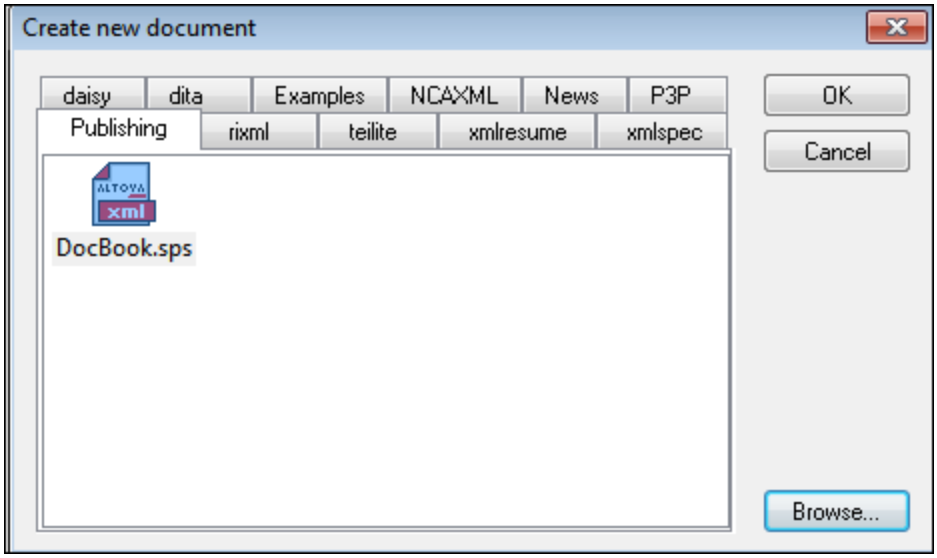
Le menu **Authentic** contient des commandes pertinentes à l'édition des documents XML dans Authentic View. Pour voir un tutoriel de Authentic View, consulter la section Tutoriels Mode Authentic .



 © 2016-2022 Altova GmbH

13.6.1 Nouveau Document

Cette commande vous permet d'ouvrir un nouveau modèle de document XML dans Authentic View. Le modèle de document XML est basé sur une StyleVision Power Stylesheet (fichier .sps), et peut être ouvert en choisissant le StyleVision Power Stylesheet (fichier SPS) dans le dialogue Créer nouveau document (*capture d'écran ci-dessous*). En sélectionnant une SPS et en cliquant sur **OK**, le modèle de document XML défini pour ce fichier SPS est ouvert dans Authentic View.



Le dialogue Créer nouveau document propose une variété de modèles de documents XML qui sont basés sur des DTD ou des schémas populaires. En alternative, vous pouvez aussi chercher un fichier SPS taillé sur mesure auquel un fichier XML modèle a été attribué. Les fichiers SPS sont créés en utilisant Altova StyleVision, une application qui permet de concevoir des modèles de document XML basés sur un schéma DTD ou XML. Une fois avoir conçu la SPS exigée dans StyleVision, un fichier XML est attribué (dans StyleVision) en tant que fichier XML modèle dans la SPS. Les données dans ce fichier XML fournissent les données de démarrage du nouveau modèle de contenu qui est ouvert dans le Authentic View de Authentic Desktop.

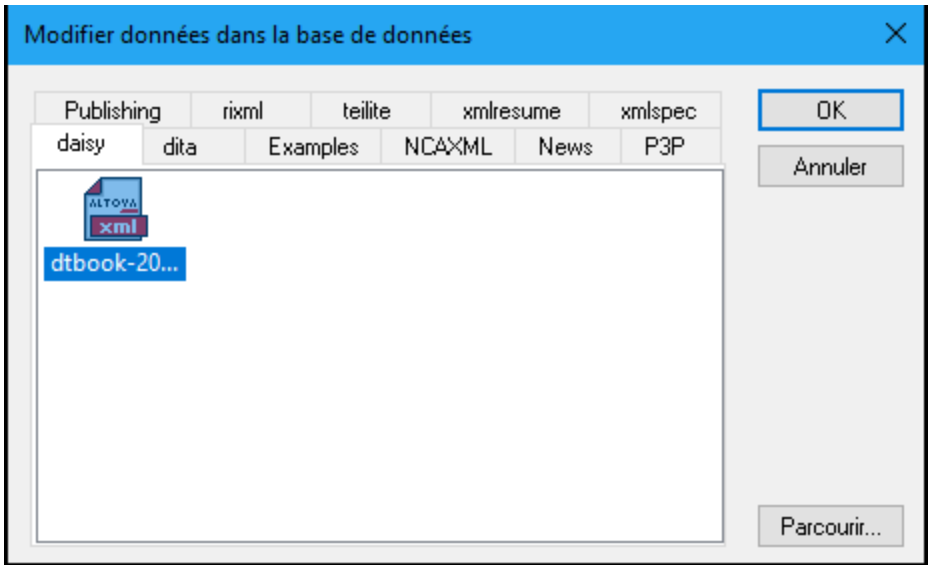
Le nouveau modèle de document XML aura donc des propriétés de présentation de document définies dans la SPS et les données du fichier XML qui ont été choisies en tant que fichier XML de modèle. L'utilisateur Authentic View peut maintenant éditer les modèles de document XML dans une interface graphique WYSIWYG, et l'enregistrer en tant que modèle XML.

13.6.2 Éditer les données de la base de données

La commande **Authentic** | **Éditer les données de la base de données...** vous permet d'ouvrir un affichage éditable d'une base de données (BD) dans Authentic View. Toutes les informations concernant la connexion à la BD et comment afficher la BD et accepter les changements de la BD dans Authentic View sont contenues dans une StyleVision Power Stylesheet. La StyleVision Power Stylesheet est basée sur BD et vous pouvez

l'ouvrir avec la commande **Éditer les données de la base de données...**. Cela vous permet d'établir une connexion à la BD et d'afficher les données BD (par une lentille XML) dans Authentic View.

Cliquer sur la commande **Éditer les données de la base de données** ouvre un dialogue de données de la base de données Éditer (*voir la capture d'écran ci-dessous*). Chercher le fichier SPS requis et le sélectionner. Il se connecte à la BD et ouvre un affichage éditable de la BD dans Authentic View. Le design du mode BD affiché dans Authentic View est contenu dans la StyleVision Power Stylesheet.



Note : si, avec la commande **Éditer les données de base de données**, vous tentez d'ouvrir une StyleVision Power Stylesheet qui n'est pas basée sur une BD ou pour ouvrir une StyleVision Power Stylesheet basée sur BD qui a été créée dans une version de StyleVision avant la release StyleVision 2005, vous recevrez une erreur.

Note : les StyleVision Power Stylesheet sont créés utilisant Altova StyleVision.

13.6.3 Éditer Feuilles de style de StyleVision

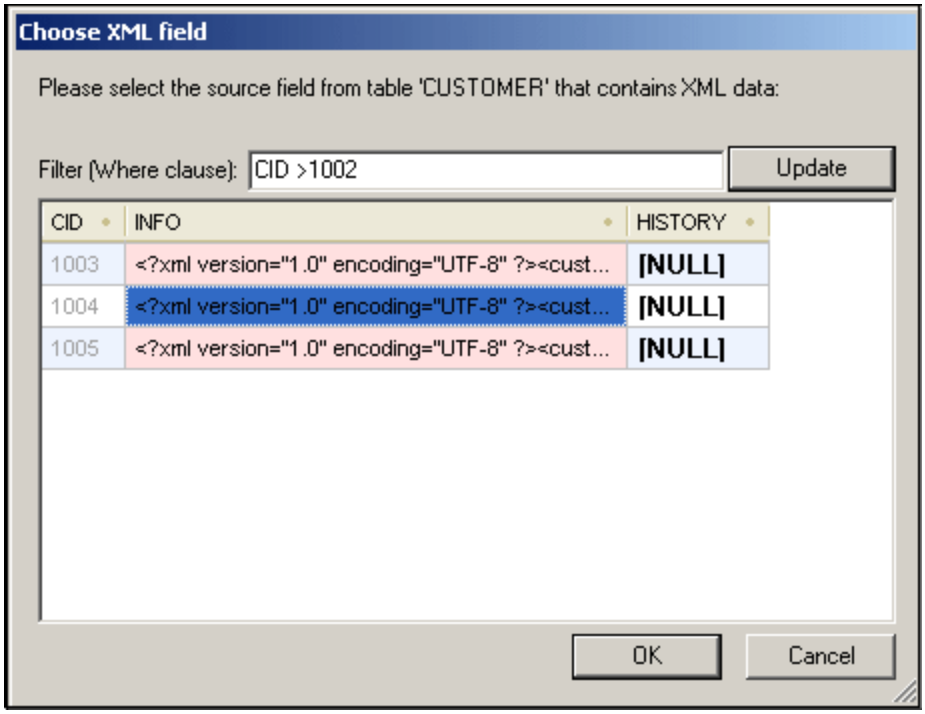
La commande **Authentic | Éditer une feuille de style StyleVision** est uniquement disponible dans le mode Authentic, mais seulement si une StyleVision Power Stylesheet a été assignée au document XML. Elle lance StyleVision et vous permet d'éditer la StyleVision Power Stylesheet immédiatement dans StyleVision.

13.6.4 Sélectionner nouvelle ligne avec données XML à éditer

La commande **Sélectionner nouvelle ligne avec données XML à éditer** vous permet de sélectionner une nouvelle ligne depuis la table pertinente dans une BD XML, comme IBM DB2. Cette ligne apparaît dans Authentic View, peut y être éditée et puis enregistrée dans la BD.

© 2016-2022 Altova GmbH

Lorsqu'une BD XML est utilisée en tant que la source de données XML, les données XML qui sont affichées dans Authentic View sont celles du document XML contenu dans une des cellules de la colonne de données XML. La commande **Sélectionner nouvelle ligne avec données XML à éditer** vous permet de choisir un document depuis une autre cellule (ou ligne) de cette colonne XML. La sélection de la commande **Sélectionner nouvelle ligne** permet d'afficher le dialogue Choisir le champ XML (*capture d'écran ci-dessous*), qui affiche la table contenant la colonne XML.



Vous pouvez saisir un filtre pour cette table. Le filtre devrait être une clause SQL `WHERE` (uniquement la condition, sans le mot-clé `WHERE`, par exemple : `CID>1002`). Cliquer sur **Mettre à jour** pour réinitialiser le dialogue. Dans la capture d'écran ci-dessus, vous pouvez voir le résultat d'un mode filtré. Ensuite, sélectionner la cellule contenant le document XML requis et cliquer sur **OK**. Le document XML dans la cellule (ligne) sélectionnée est chargé dans Authentic View.

13.6.5 Signature XML

La commande **Signature XML** est disponible dans Authentic View quand les SPS associées ont des Signatures XML activées. La commande **Signature XML** est aussi disponible en tant que l'icône de barre d'outils Signature XML  dans la barre d'outils Authentic.

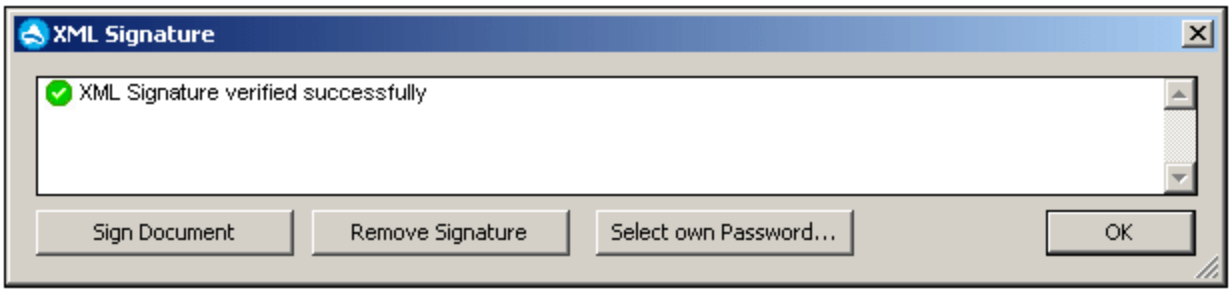
Vérification et certificat/mot de passe

Cliquer sur la commande **Signature XML** pour lancer le processus de vérification de la signature. Si aucune signature n'est présente dans le document, un message à cet effet est affiché dans le dialogue de Signature XML (*voir la capture d'écran ci-dessous*), et le dialogue aura un bouton qui permet à l'utilisateur Authentic View de signer le document.

Si le bouton **Sélectionner Propre Certificat** ou **Sélectionner Propre Mot de passe** est présent dans le dialogue, ceci signifie que Authentic View a obtenu l'option de sélectionner un propre certificat/mot de passe. (Le fait de savoir si un certificat ou un mot de passe doit être choisi aura été décidé par le designer de la SPS au moment de la configuration de la signature. La signature sera soit basée sur un certificat ou sur un mot de passe.) Cliquer sur un de ces boutons, s'ils sont présents dans le dialogue, activer l'utilisateur Authentic View pour charger un certificat ou saisir un mot de passe. La sélection de l'utilisateur Authentic View est stockée dans la mémoire et est valide pour la session actuelle uniquement. Si, après avoir choisi un certificat ou un mot de passe, le document ou l'application est fermée, les paramètres du certificat/mot de passe sont rétablis sur les paramètres enregistrés à l'origine avec la SPS.

Informations de vérification et d'authentification

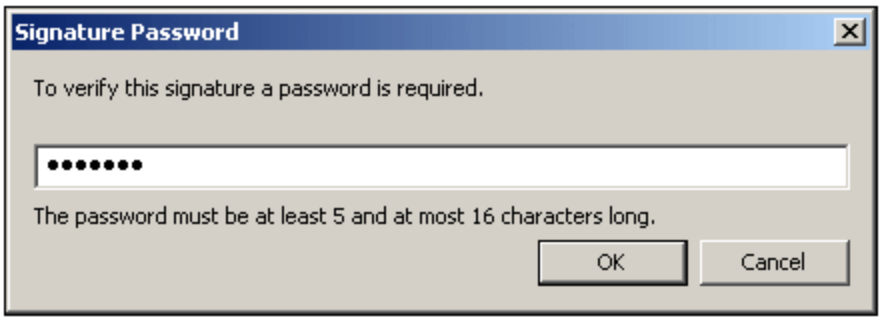
Si le processus de vérification est exécuté sur un document signé, deux situations générales sont possibles. First: Si l'information d'authentification est disponible (dans la signature ou la SPS), alors le processus de vérification est exécuté directement et le résultat est affiché (*capture d'écran ci-dessous*).



L'information d'authentification est soit l'information de la clé de signature ou le mot de passe de signature. Le designer de la SPS aura spécifié si l'information de la clé du certificat est enregistrée dans la signature lorsque le document XML est signé ou, dans le cas d'une signature à base de mot de passe, si le mot de passe est enregistré dans la SPS. Dans les deux cas, l'authentification est disponible. Par conséquence, le processus de vérification sera exécuté directement, sans qu'il soit nécessaire d'utiliser une entrée de la part de l'utilisateur Authentic View.

La deuxième situation possible se produit lorsque l'information d'authentification n'est pas disponible dans la signature (information de la clé du certificat) ou le fichier SPS (mot de passe). Dans cette situation, l'utilisateur Authentic View sera invité à fournir l'information d'authentification : un mot de passe (*voir capture d'écran ci-dessous*) ou l'emplacement d'un certificat.

© 2016-2022 Altova GmbH



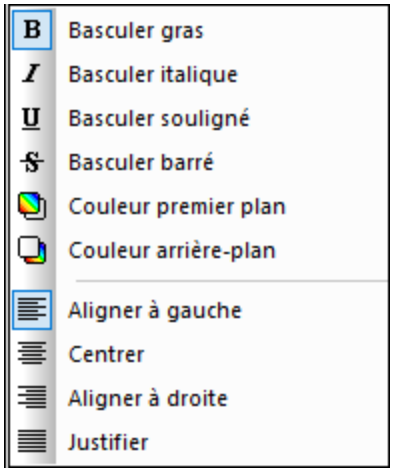
13.6.6 Afficher balisage

La commande **Afficher balisage** a un sous-menu avec des options pour contrôler l'affichage du balisage dans Authentic View. Ces options sont décrites ci-dessous.

	**Dissimuler balisage** dissimule tous les symboles de balisage.
	**Afficher Petit balisage** affiche le balisage en tant que petits symboles.
	**Afficher Grand balisage** affiche le balisage en tant que grands symboles.
	**Afficher balisage mixte :** La personne qui conçoit StyleVision Power Stylesheet peut soit spécifier un grand balisage, petit balisage, ou pas de balisage pour des éléments/attributs individuels du document. Le balisage mixte affiche le balisage personnalisé.

13.6.7 RichEdit

Passer avec la souris sur la commande **RichEdit** pour ouvrir un sous-menu contenant les commandes de marquage RichEdit (*capture d'écran ci-dessous*). Les commandes de menu de ce sous-menu sont uniquement activées dans Authentic View et lorsque le curseur est placé dans un élément qui a été créé en tant qu'un composant RichEdit dans le design SPS.



Les propriétés de style du texte du menu RichEdit seront appliquées au texte sélectionné lorsqu'une commande de marquage RichEdit est cliquée. L'utilisateur Authentic View peut, outre la police et la taille de la police spécifiée dans la barre d'outils Authentic, préciser en plus l'épaisseur de police, le style de police, la décoration de police, la couleur, la couleur d'arrière-plan et l'alignement du texte sélectionné.

13.6.8 Ajouter/Insérer/Dupliquer/Supprimer ligne

Les commandes **Table Row** énumérées ci-dessous vous permettent de structurer des tables dans Authentic View.

	**Ajouter Ligne** ajoute une ligne à la table actuelle.
	**Insérer Ligne** insère une ligne à la table actuelle.
	**Dupliquer Ligne** duplique la ligne de la table actuelle en dessous de la ligne actuelle.
	**Supprimer Ligne** supprime la ligne de table actuelle.

13.6.9 Déplacer Ligne Haut/Bas

Les prochaines commandes de ligne suivantes sont activées dans les tables du mode Authentic :

- **Déplacer Ligne vers le haut** déplace la ligne de table présente d'une ligne vers le haut dans Authentic View.
- **Déplacer Ligne vers le haut** déplace la ligne de table actuelle d'une ligne vers le haut dans Authentic View.

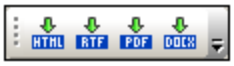
© 2016-2022 Altova GmbH

13.6.10 Générer HTML, RTF, PDF, Word 2007+ , document texte

Ces cinq documents génère les documents de sortie depuis le document XML de Authentic View stocké dans le fichier PXF :

- **Générer un document HTML**
- **Générer un document RTF**
- **Générer un document PDF**
- **Générer un document Word 2007+**
- **Générer un document Texte**

Ils sont également disponibles dans la barre d'outils du formulaire portable XML (PXF) (*voir la capture d'écran ci-dessous*).

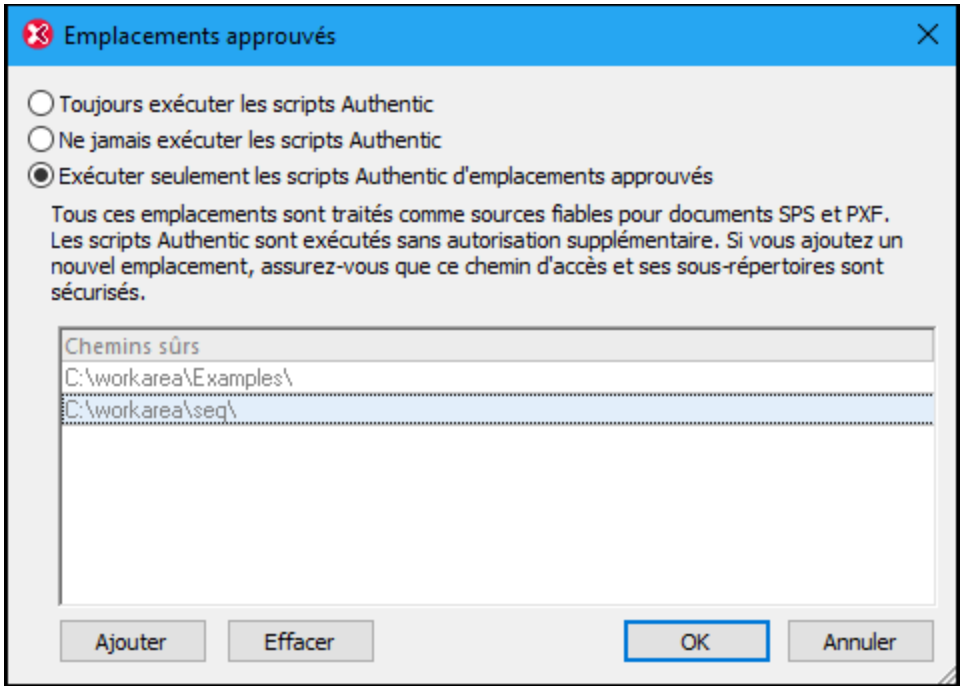


Cliquer sur les commandes individuelles ou boutons génère respectivement la sortie de HTML, RTF, PDF ou DocX.

Ces boutons sont activés quand un fichier PXF est ouvert dans Authentic View. Des commandes et boutons individuels sont activés si le fichier PXF a été configuré pour contenir la feuille de style pour ce format de sortie spécifique. Par exemple, si le fichier PXF a été configuré pour contenir les feuilles de style XSLT pour HTML et RTF, alors uniquement les commandes et boutons de la barre d'outils seront activés tandis que ceux pour la sortie texte, PDF et DocX (Word 2007+) seront désactivés.

13.6.11 Emplacements approuvés

La commande Emplacements approuvés ouvre le dialogue Emplacements approuvés (*capture d'écran ci-dessous*) dans lequel vous pouvez spécifier les paramètres de sécurité pour les scripts dans une SPS. Quand un fichier XML est basé sur une SPS contenant du script est basculé vers Authentic View, le script sera autorisé d'être exécuté ou non dépendant des paramètres que vous faites dans le dialogue.



Les trois options disponibles sont :

- Les scripts Authentic sont toujours exécutés lorsqu'un fichier est ouvert dans le Mode Authentic.
- Les scripts Authentic ne sont jamais exécutés lorsqu'un fichier est ouvert dans le Mode Authentic.
- Seuls les scripts Authentic dans des emplacements approuvés sont exécutés. La liste des emplacements (dossiers) approuvés est affichée dans le volet du bas. Utiliser la touche **Ajouter** pour chercher un dossier et l'ajouter à la liste. Pour supprimer une entrée de la liste, choisir une entrée dans la liste des Emplacements approuvés et cliquer sur **Supprimer**.

© 2016-2022 Altova GmbH

13.7 Menu Affichage

Le menu **Mode** (*capture d'écran ci-dessous*) contrôle l'affichage de la fenêtre Principale active et vous permet de changer la manière dont est affiché le document. Cette section fournit une description des commandes dans le menu **Mode**.

13.7.1 Mode Authentic

Cette commande fait passer le document actuel à Authentic View.

Authentic View vous permet d'éditer des documents XML basés sur des modèles StyleVision Power Stylesheet crées dans l'application StyleVision d'Altova. Ces modèles (feuilles de style StyleVision ou fichiers SPS) affichent des documents XML dans un format graphique qui permet une édition plus simple du document XML (plus simple que de l'éditer dans un format de texte avec des balises).

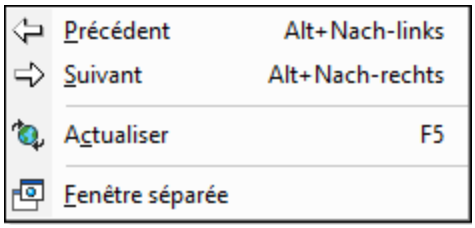
13.7.2 Mode Navigateur

Cette commande fait passer le document actuel au Mode Navigateur. Un navigateur activé par XML présente le document XML utilisant l'information depuis des CSS et/ou de feuilles de style XSL disponibles.

En passant au Mode Navigateur, le document est d'abord vérifié pour sa validité si l'option *Valider sur enregistrement* dans la section de fichier du dialogue des Options (**Outils | Options**) est vérifiée. Pour plus d'informations, voir la section Mode navigateur de cette documentation.

13.8 Menu Navigateur

Les commandes de menu **Navigateur** sont activées dans le mode Navigateur uniquement.



Précédent, Suivant

La commande **Suivant** (*raccourci :* **Alt + flèche gauche**) affiche la page consultée précédemment. La touche **Retour arrière** obtient le même effet. La commande est utile lorsque vous cliquez sur un lien dans votre document XML puis souhaitez retourner à votre document XML.

La commande **Suivant** (*raccourci :* **Alt + flèche droite**) vous fait avancer à travers les pages consultées précédemment dans le mode Navigateur.

Arrêter

La commande **Arrêter** est activée dans le Mode Navigateur et indique au navigateur d'arrêter le chargement de votre document. Cela est utile si des fichiers externes volumineux ou des graphiques sont téléchargés par le biais d'une connexion Internet lente et que vous souhaitez interrompre le processus.

Actualiser

La commande **Actualiser (F5)** est activée dans le Mode Navigateur et met à jour le Mode Navigateur en rechargeant le document actuel et les documents liés au document actuel (comme des feuilles de style CSS et XSL, et des DTD).

Polices

La commande **Polices** déroule un sous-menu à partir duquel vous pouvez sélectionner la taille de police par défaut pour rendre le texte de votre document XML.

Fenêtre séparée

La commande **Fenêtre séparée** est activée dans le Mode Navigateur et décroche le Mode Navigateur du document des autres modes. En tant que fenêtre séparée, le Mode Navigateur peut être affiché côte-à-côte avec un mode d'édition du document.

Pour actualiser le Mode Navigateur séparé après avoir effectué une modification dans un mode d'édition, appuyer **F5** dans le mode d'édition. Pour réancrer une fenêtre en mode Navigateur séparée dans la fenêtre d'application, activez la fenêtre du mode Navigateur et cliquez sur la commande **Fenêtre séparée**.

© 2016-2022 Altova GmbH

13.9 Menu Outils

Le menu Outils vous permet de :

- Contrôler l'orthographe de vos documents XML
- Accéder à l'environnement des scripts de Authentic Desktop. Vous pouvez créer, gérer et stocker vos propres formulaires, macros et gestionnaires d'événements
- Consulter les macros attribués actuellement
- Accéder aux commandes personnalisées qui utilisent des applications externes. Ces commandes peuvent être créées dans l'onglet Outils du dialogue Personnaliser.
- Définir et utiliser les ressources globales
- Personnaliser votre version de Authentic Desktop : définir vos propres barres d'outils, raccourcis de clavier, menus et macros
- Définir des paramètres globaux de Authentic Desktop

13.9.1 Orthographe

Le vérificateur orthographique de Authentic Desktop contenant des dictionnaires de langue intégré (*voir la note ci-dessous*) est activé dans Authentic View.

Note : Les dictionnaires intégrés qui sont inclus avec le logiciel Altova ne proposent pas de préférences linguistique par Altova. La sélection de dictionnaires dépend de la disponibilité de dictionnaires qui permet la redistribution avec des logiciels commerciaux, tels les licences pour MPL, LGPL, ou BSD. De nombreux autres dictionnaires open-source existent, mais sont distribués sous des licences plus restrictives, telle que la licence GPL. De nombreux autres dictionnaires sont disponibles comme faisant partie d'un programme d'installation séparé http://www.altova.com/dictionaries. À vous de choisir si vous souhaitez utiliser des dictionnaires en se basant sur les licences applicables au dictionnaire et si le dictionnaire est approprié pour votre utilisation avec le logiciel sur votre ordinateur.

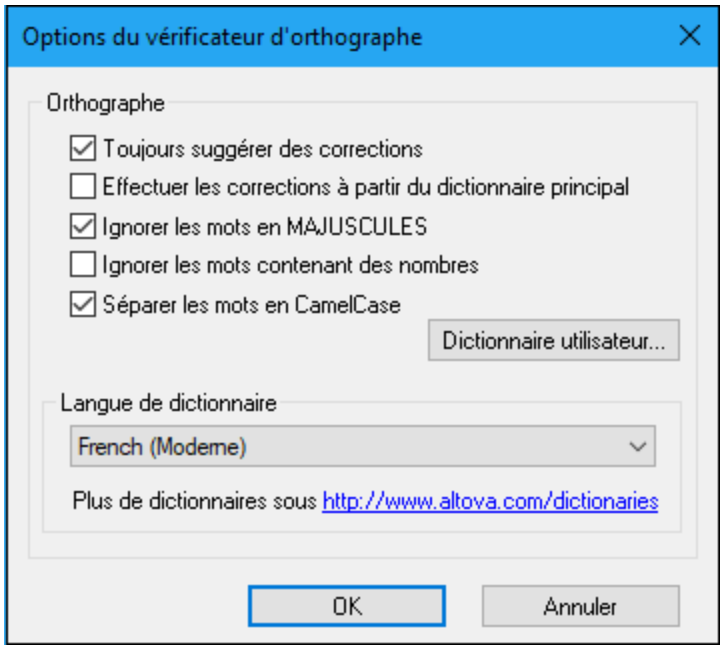
Cette section décrit comment utiliser le vérificateur orthographique. Elle est organisée en les sous-sections suivantes :

- Choisir la langue du vérificateur automatique
- Exécuter la vérification orthographique

Choisir la langue du vérificateur automatique

La langue du vérificateur orthographique peut être définie comme suit :

1. Cliquez sur la commande de menu **Outils | Options d'orthographe**.
2. Dans le dialogue Options du vérificateur orthographique qui s'ouvre maintenant (*capture d'écran ci-dessous*), choisissez un des dictionnaires installés à partir de la liste déroulante de la liste de choix de la Langue du dictionnaire.



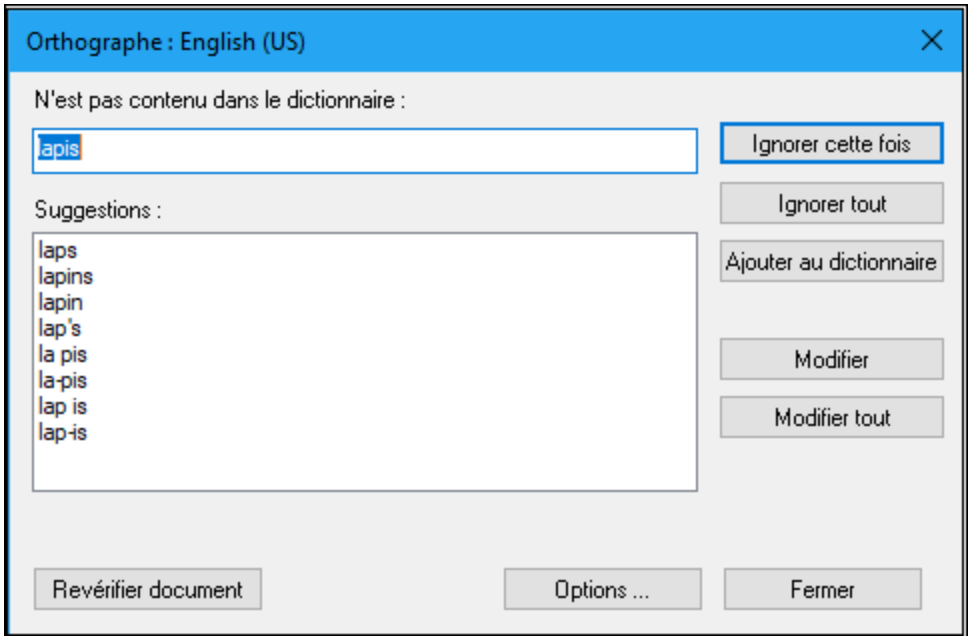
3. Cliquez sur **OK** pour finir.

La Langue du dictionnaire que vous avez choisie sera utilisée par la vérification orthographique pour les vérifications. Si la langue que vous souhaitez n'est pas encore installée, vous pouvez télécharger des dictionnaires de langues supplémentaires. Vous trouverez une description dans la section Ajouter des dictionnaires pour la vérification orthographique.

Exécuter la vérification orthographique

La commande **Outils | Orthographe (Shift+F7)** commence automatiquement à vérifier le document XML actuellement actif conformément à l'étendue définie. Si un mot inconnu est rencontré, le dialogue *Orthographe: N'est pas contenu dans le dictionnaire* s'ouvre (voir la *capture d'écran ci-dessous*). Sinon, la vérification orthographique se poursuit jusqu'au bout du document.

© 2016-2022 Altova GmbH



Les différentes parties de l'*orthographe : N'est pas contenu dans le dictionnaire* et les options disponibles sont décrites ci-dessous :

N'est pas contenu dans le dictionnaire
Cette fenêtre contient le mot qui ne peut pas être trouvé dans le dictionnaire de langue ou le dictionnaire utilisateur sélectionné. Les options suivantes sont disponibles :

- Vous pouvez éditer le mot dans la fenêtre manuellement ou sélectionner une suggestion depuis le volet *Suggestions*. Ensuite, cliquez sur **Modifier** pour remplacer le mot dans le document XML avec le mot éditer. (Double-cliquez sur une suggestion pour l'insérer directement dans le document XML.) Lorsqu'un mot est affiché dans la case de texte*N'est pas contenu dans le dictionnaire*, il est également marqué dans le document XML, vous pouvez donc éditer le mot directement dans le document si vous le souhaitez. Cliquer sur **Modifier tout** pour remplacer toutes les occurrences du mot dans le document XML avec le mot édité.
- Vous pouvez choisir de ne pas effectuer de modifications et d'ignorer les avertissements du vérificateur — orthographique, soit uniquement pour l'occurrence actuelle du mot soit pour toutes les occurrences.
- Vous pouvez ajouter le mot au dictionnaire d'utilisateur, le mot sera donc considéré comme étant correct pour toutes les vérifications depuis la vérification actuelle jusqu'à la fin du document.

Suggestions
Cette liste affiche des mots qui ressemblent au mot inconnu (fourni par les dictionnaires de langue et d'utilisateur). Double-cliquer sur un mot dans la liste l'insère automatiquement dans le document et continue le processus de vérification orthographique.

Ignorer cette fois
Cette commande vous permet de continuer à vérifier le document tout en ignorant la première occurrence du mot inconnu. Le même mot sera marqué à nouveau s'il réapparaît dans le document.

Ignorer tout
Cette commande ignore toutes les instances du mot inconnu dans le document entier.

Ajouter au dictionnaire
Cette commande ajoute le mot inconnu au **dictionnaire utilisateur**. Vous pouvez accéder au dictionnaire utilisateur (afin de l'éditer) par le biais du dialogue Options du vérificateur orthographique.

Changer
Cette commande remplace le mot actuellement marqué dans le document XML avec le mot (édité) dans la fenêtre *N'est pas contenu dans le dictionnaire*.

Modifier tout
Cette commande remplace toutes les occurrences du mot actuellement marqué dans le document XML avec le mot (édité) dans la fenêtre *N'est pas contenu dans le dictionnaire*.

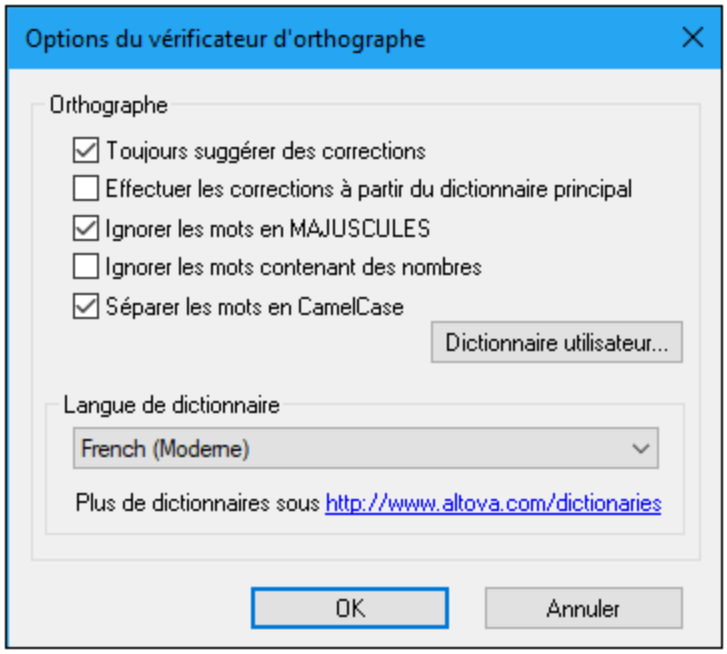
Revérifier le document
La touche **Revérifier document** redémarre la vérification à partir du début du document.

Options
Cliquer sur la touche **Options** ouvre la boîte de dialogue Options d'orthographe.

Fermer
Cette commande ferme la fenêtre du dialogue Orthographe.

13.9.2 Options du vérificateur orthographique

La commande **Outils | Options du vérificateur orthographique** ouvre le dialogue des Options du vérificateur orthographique (*voir la capture d'écran ci-dessous*), qui est utilisé pour définir les options globales de vérification orthographique.



© 2016-2022 Altova GmbH

Toujours suggérer des corrections :
En activant cette option, des suggestions (provenant du dictionnaire de langue et du dictionnaire d'utilisateur) s'affichent dans la liste de choix Suggestions. En désactivant cette option, aucune suggestion ne sera affichée.

Effectuer les corrections à partir du dictionnaire principal :
En activant cette option, seul le dictionnaire de langue (dictionnaire principal) sera utilisé. Le dictionnaire d'utilisateur n'est pas utilisé pour proposer des suggestions. Il désactive aussi le bouton **Dictionnaire utilisateur**, empêchant toute édition par le dictionnaire d'utilisateur.

Ignorer les mots en MAJUSCULES :
En activant cette option, les mots en majuscule seront ignorés.

Ignorer les mots contenant des numéros :
En activant cette option, les mots contenant des nombres seront ignorés.

Séparer les mots en CamelCase
Les mots en CamelCase sont des mots qui contiennent une majuscule à l'intérieur du mot. Par exemple, dans le mot "CamelCase", le "C" de "Case" est en majuscule et peut donc être considéré comme étant une CamelCase. Étant donné que les mots en CamelCase sont rarement trouvés dans les dictionnaires, le correcteur orthographique les marquerait comme des erreurs. Pour éviter ce problème, l'option *Séparer les mots en CamelCase* sépare les mots en CamelCase dans leurs composants à majuscules et contrôle chaque composant individuellement. Cette option est contrôlée par défaut.

Langue de dictionnaire
Utiliser cette liste de choix pour sélectionner la langue de dictionnaire que vous souhaitez utiliser pour la vérification orthographique. La sélection par défaut est US English. Les autres dictionnaires en d'autres langues peuvent être téléchargés gratuitement depuis le site web Altova.

Ajouter des dictionnaires au vérificateur orthographique

Pour chaque langue de dictionnaire, il existe deux fichiers de dictionnaire Hunspell qui fonctionnent ensemble : un fichier `.aff` et un fichier `.dic`. Tous les dictionnaires de langue sont installés dans un dossier `Lexicons` sous : `C:\ProgramData\Altova\SharedBetweenVersions\SpellChecker\Lexicons`.

Dans le dossier `Lexicons`, plusieurs dictionnaires de langue différents sont stockés dans un dossier différent : `<language name>\<dictionary files>`. Par exemple, les fichiers pour les deux dictionnaires de langue anglaise (`English (British)` et `English (US)`) seront stockés comme dans l'exemple ci-dessous :

```
C:\ProgramData\Altova\SharedBetweenVersions\SpellChecker\Lexicons\English (British)
\en_GB.aff
C:\ProgramData\Altova\SharedBetweenVersions\SpellChecker\Lexicons\English (British)
\en_GB.dic
C:\ProgramData\Altova\SharedBetweenVersions\SpellChecker\Lexicons\English (US)\en_US.aff
C:\ProgramData\Altova\SharedBetweenVersions\SpellChecker\Lexicons\English (US)\en_US.dic
```

Dans le dialogue Options du vérificateur d'orthographe, la liste déroulante de la liste de choix *Langue de dictionnaire* affiche les dictionnaires de langue. Ces dictionnaires sont ceux disponibles dans le dossier `Lexicons` et ont les mêmes noms que les sous-dossiers de langue dans le dossier `Lexicons`. Par exemple, dans le cas des dictionnaires de langue anglaise affichés ci-dessus, les dictionnaires apparaîtraient dans la liste de choix Langue de dictionnaire en tant que : *English (British)* et *English (US)*.

Tous les dictionnaires installés sont partagés par les différents utilisateurs de l'appareil et les versions principales différentes des produits Altova (32-bit ou 64-bit).

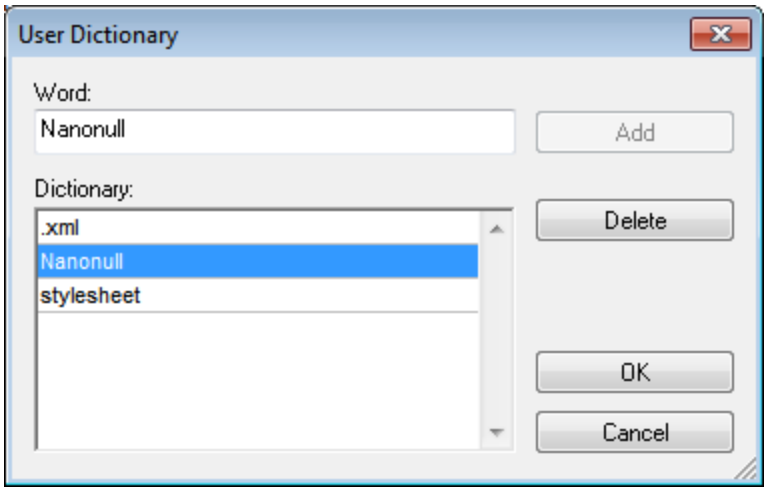
Vous pouvez ajouter des dictionnaires pour le vérificateur orthographique de deux manières, aucune des deux ne nécessite que les fichiers soient enregistrés avec le système :

- En ajoutant les dictionnaires Hunspell dans un nouveau sous-dossier du dossier `Lexicons`. Les dictionnaires Hunspell peuvent être téléchargés, par exemple depuis http://wiki.services.openoffice.org/wiki/Dictionaries ou http://extensions.services.openoffice.org/en/dictionaries. (Veuillez noter qu'OpenOffice utilise le format zippé `OXT`. Il faut donc changer l'extension en `.zip` et dézipper le fichier `.aff` et `.dic` dans les dossiers de langue dans le dossier `Lexicons`. Veuillez aussi noter que les dictionnaires Hunspell sont basés sur les dictionnaires Myspell. Les dictionnaires Myspell peuvent donc aussi être utilisés.)
- En utilisant l'installateur de dictionnaire Altova, qui installe un pack de dictionnaires en plusieurs langues par défaut dans l'emplacement correct de votre appareil. L'installateur peut être téléchargé par le biais du lien dans le volet Langue de dictionnaire du dialogue Options du vérificateur orthographique (*voir capture d'écran ci-dessous*). L'installation des dictionnaires doit être effectuée avec des droits d'administrateur, sinon l'installation échouera avec une erreur.

Note : À vous de choisir si vous acceptez les termes de la licence applicables au dictionnaire et si le dictionnaire est approprié pour votre utilisation avec le logiciel sur votre ordinateur.

Travailler avec le dictionnaire d'utilisateur

Chaque utilisateur a un dictionnaire d'utilisateur dans lequel les mots autorisés par l'utilisateur peuvent être stockés. Pendant une vérification orthographique, l'orthographe est vérifiée par rapport à une liste de mots contenant les mots dans le dictionnaire de langue et le dictionnaire d'utilisateur. Vous pouvez ajouter et supprimer des mots du dictionnaire d'utilisateur par le biais du Dialogue Dictionnaire utilisateur (*capture d'écran ci-dessous*). Ce dialogue est accédé en cliquant sur la touche Dictionnaire utilisateur dans le dialogue Options du vérificateur d'orthographe (*voir seconde capture d'écran dans cette section*).



Pour ajouter un mot au dictionnaire utilisateur, saisir le mot dans le champ de saisie Mot et cliquer sur

© 2016-2022 Altova GmbH

Ajouter. Le mot sera ajouté à la liste alphabétique dans le volet Dictionnaire. Pour supprimer un mot du dictionnaire, choisir le mot dans le volet Dictionnaire et cliquer sur **Supprimer**. Le mot sera supprimé du volet Dictionnaire. Lorsque vous avez terminé d'éditer le dialogue Dictionnaire utilisateur, cliquer sur **OK** pour que les changements soient enregistrés dans le dictionnaire utilisateur.

Les mots peuvent être ajoutés au Dictionnaire utilisateur pendant une vérification orthographique. Si un mot inconnu est rencontré pendant une vérification orthographique, le dialogue Orthographe s'ouvre et vous invite à choisir une action. Si vous cliquez sur la touche **Ajouter au dictionnaire**, le mot inconnu sera ajouté au dictionnaire utilisateur.

Le Dictionnaire utilisateur se trouve sous : `C:\Users\<user>\Documents\Altova\SpellChecker\Lexicons\user.dic`

13.9.3 Éditeur de script

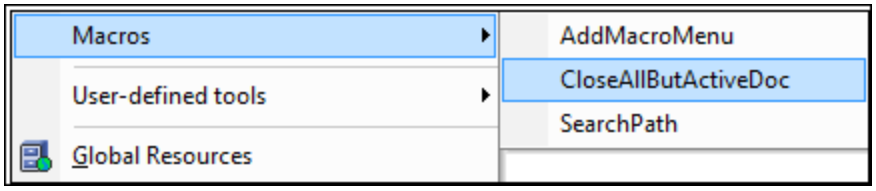
La commande **Éditeur de script** ouvre la fenêtre Éditeur de script. Le fonctionnement de l'Éditeur de script est décrit dans la section Script de cette documentation.

Note : Pour que l'Éditeur de script soit exécuté, .NET Framework version 2.0 ou ultérieur doit être installé sur votre appareil.

13.9.4 Macros

Exécuter un macro comme suit :

1. Placer votre curseur au-dessus de la commande **Macros**
2. Le sous-menu qui se déploie contient une liste de macros dans le Projet de Scripting qui est actuellement active dans Authentic Desktop (*voir la capture d'écran ci-dessous*). Le Projet de Scripting actif est spécifié dans les Paramètres de Scripting du dialogue des Options.



3. Cliquez sur un macro pour l'exécuter.

13.9.5 Outils définis par l'utilisateur

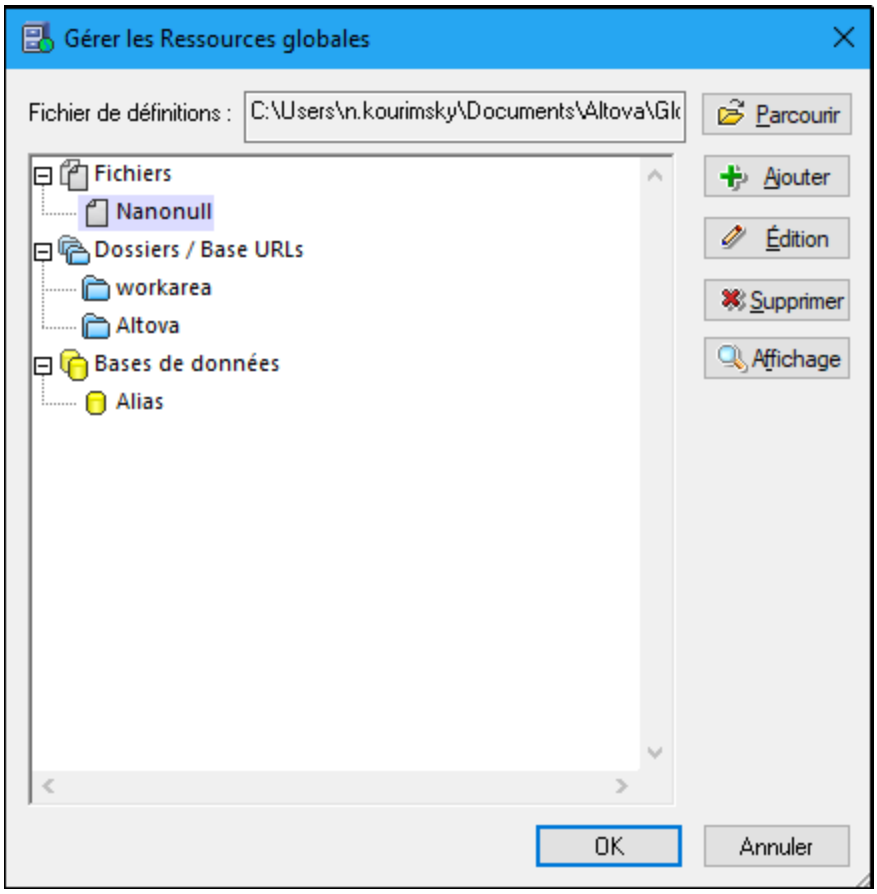
Placer le curseur sur la commande **Outils définis par l'utilisateur** pour faire défiler un sous-menu contenant des commandes personnalisées qui utilisent des applications externes. Vous pouvez créer ces commandes dans l'onglet Outils du dialogue Personnaliser. Cliquer sur une de ces commandes personnalisées pour exécuter l'action associée avec cette commande.

La commande **Outils définis par l'utilisateur | Personnaliser** ouvre l'onglet Outils du dialogue Personnaliser (dans lequel vous pouvez créer les commandes personnalisées qui apparaissent dans le menu de la commande **Outils définis par l'utilisateur**.)

13.9.6 Ressources globales

La commande **Ressources globales** ouvre le dialogue Ressources Globales (*capture d'écran ci-dessous*), dans lequel vous pouvez :

- Spécifier le fichier XML de Ressources globales à utiliser pour les ressources globales.
- Ajouter des fichiers, des dossiers et des ressources globales de base de données (ou des alias)
- Spécifier des configurations variées pour chaque ressource globale (alias). Chaque configuration mappe vers une ressource spécifique.



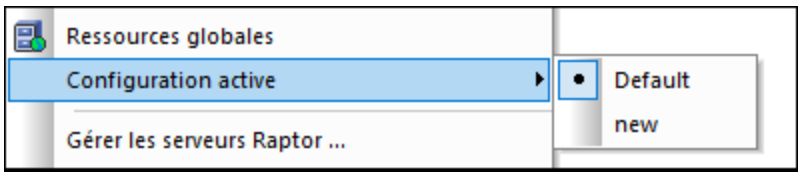
Vous trouverez plus d'informations concernant la définition des ressources globales dans la section Définir des Ressources globales.

Note: le dialogue Ressources globales d'Altova peut aussi être accédé par le biais de la barre outils Ressources globales (**Outils | Personnaliser | Barres outils | Ressources globales**).

© 2016-2022 Altova GmbH

13.9.7 Configuration active

Passer avec la souris sur l'item de menu Configuration Activepour dérouler un sous-menu contenant toutes les configurations définies dans le fichier XML Ressources Globales actuellement actif (capture d'écran ci-dessous).



La configuration active est indiquée avec une puce. Dans la capture d'écran ci-dessus, la configuration active actuellement est `Défaut`. Pour changer la configuration active, sélectionner la configuration que vous souhaitez rendre active.

Note: La configuration active peut aussi être sélectionnée par le biais de la barre d'outils Ressources globales (**Outils | Personnaliser | Barres d'outils | Ressources globales**).

13.9.8 Personnaliser

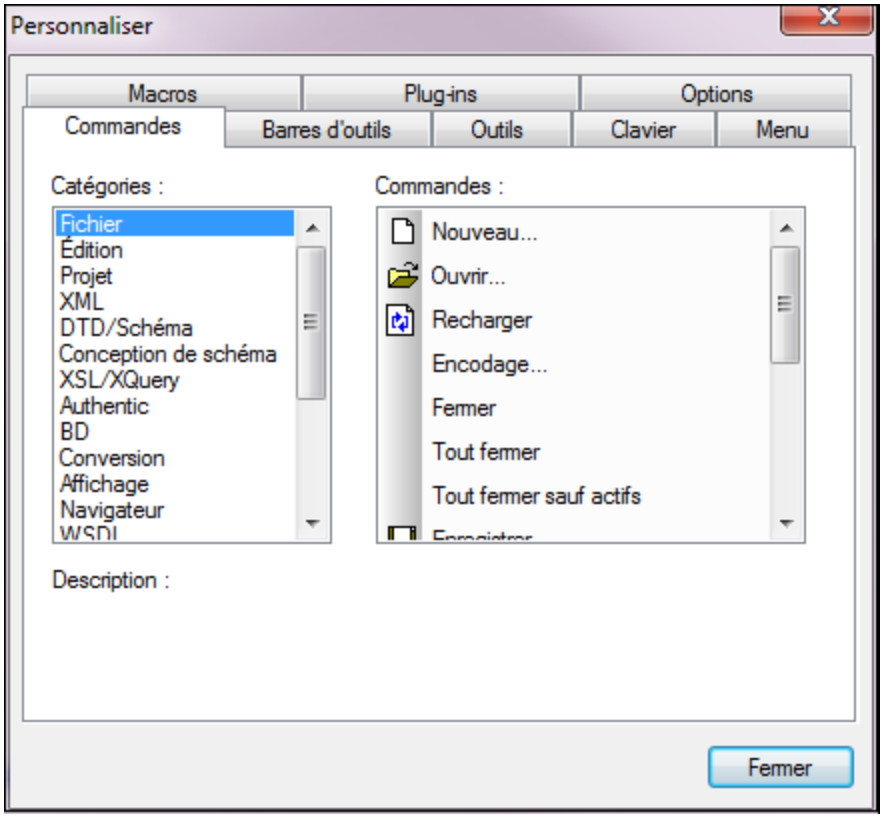
La commande **Personnaliser** vous permet de personnaliser les menus d'application et les barres d'outils selon vos besoins. Cliquer sur la commande pour ouvrir le dialogue Personnaliser qui comporte les onglets suivants :

- Commandes : Toutes les commandes macro et d'application peuvent êtres glissées de cet onglet dans les barres de menu, les menus et les barres d'outils.
- Barres d'outils : Les barres d'outils peuvent être activées, désactivées et réinitialisées individuellement.
- Outils : Les commandes qui ouvrent des programmes externes depuis l'intérieur de l'interface peuvent être ajoutées à l'interface.
- Clavier : Les raccourcis de clavier peuvent être créés pour une application individuelle et des commandes macro.
- Menu : Les barres de menu et menus contextuels à personnaliser sont sélectionnés et rendus actifs dans cet onglet. Travaille en collaboration avec l'onglet Commandes.
- Macros : Les macros peuvent avoir de nouvelles commandes associées.
- Plug-ins : Les plug-ins peuvent être activés et intégrés dans l'interface.
- Options : Les options d'affichage pour les barres d'outils sont définies dans cet onglet.

Cette section décrit également le menu contextuel qui apparaît lorsque le dialogue Personnaliser est ouvert et la barre de menu, le menu, et les items de barre d'outils sont cliqués avec la touche droite de la souris.

13.9.8.1 Commandes

L'onglet **Commandes** vous permet de personnaliser vos menus et vos barres d'outils. Vous pouvez ajouter des commandes d'application aux menus et barres d'outils selon vos préférences. Veuillez noter, toutefois, que vous ne pouvez pas créer vous-même de nouvelles commandes d'application ou des menus.



Pour ajouter une commande à une barre d'outils ou à un menu :

1. Choisir l'item de menu **Outils | Personnaliser**. Le dialogue Personnaliser apparaît.
2. Sélectionner la catégorie **Toutes les commandes** dans la liste *Catégories*. Les commandes disponibles apparaissent dans la liste *Commandes*.
3. Cliquez sur une commande dans la liste *Commandes* et glissez-la vers un menu ou une barre d'outils existant. Une barre **I** apparaît lorsque vous placez le curseur sur une position valide pour déposer la commande.
4. Relâcher le bouton de la souris à la position où vous souhaitez insérer la commande.

Veuillez noter les points suivants.

- Lorsque vous glissez une commande, un petit bouton apparaît au bout du pointeur de la souris : cela indique que la commande est en train d'être déplacée.
- Un "**x**" en-dessous du pointeur indique que la commande ne peut pas être déposée dans la position actuelle du curseur.
- Si le curseur est déplacé sur une position où il est possible de déposer une commande (une barre outils ou un menu), le "**x**" disparaît et une barre **I** indique la position valide.

© 2016-2022 Altova GmbH

- Les commandes peuvent être placées dans des menus ou des barres outils. Si vous avez créé votre propre barre d'outils, vous pouvez utiliser ce mécanisme de personnalisation pour la remplir.
- Déplacer le curseur sur un menu fermé pour ouvrir ce menu, vous permettant d'insérer la commande à n'importe quel endroit de ce menu.

Ajouter les commandes aux menus contextuels

Vous pouvez aussi ajouter les commandes aux menus contextuels en glissant les commandes depuis la liste *Commandes* dans le menu contextuel. La procédure est la suivante :

1. Dans le dialogue Personnaliser, cliquez sur **Menu** tab.
2. Dans le volet Menu contextuel, sélectionner un menu contextuel depuis la liste de choix. Le menu contextuel sélectionné s'ouvre.
3. Dans le dialogue Personnaliser, retournez à l'onglet Commandes.
4. Glissez la commande que vous souhaitez créer depuis la liste *Commandes* et la déposer dans l'emplacement souhaité dans le menu contextuel.

Supprimer une commande ou un menu

Pour supprimer une commande d'un menu, d'un menu contextuel (voir ci-dessus pour plus de détails concernant l'accès aux menus contextuels), ou d'une barre d'outils ou pour supprimer un menu entier, procéder comme suit :

1. Ouvrir le dialogue Personnaliser (**Outils | Personnaliser**). Le dialogue Personnaliser apparaît.
2. Avec le dialogue Personnaliser ouvert (et un onglet sélectionné), cliquez avec la touche de droite sur un menu ou une commande de menu, puis sélectionnez **Supprimer** depuis le menu contextuel qui s'ouvre. **En alternative, glissez le menu ou la commande de menu jusqu'à ce qu'une icône "**x" apparaisse en-dessous du pointeur de la souris, puis déposer le menu ou la commande de menu. Le menu ou la commande de menu sera supprimée.

Pour recompiler les commandes de menu supprimées, utilisez les mécanismes décrits dans cette section. Pour rétablir un menu supprimé, aller sur **Outils | Personnaliser | Menu**, et cliquer sur le bouton **Réinitialiser** du volet *Menus cadre d'application*. En alternative, aller sur **Outils | Personnaliser | Barre outils**, sélectionner la barre Menu et cliquer sur le bouton **Réinitialiser**.

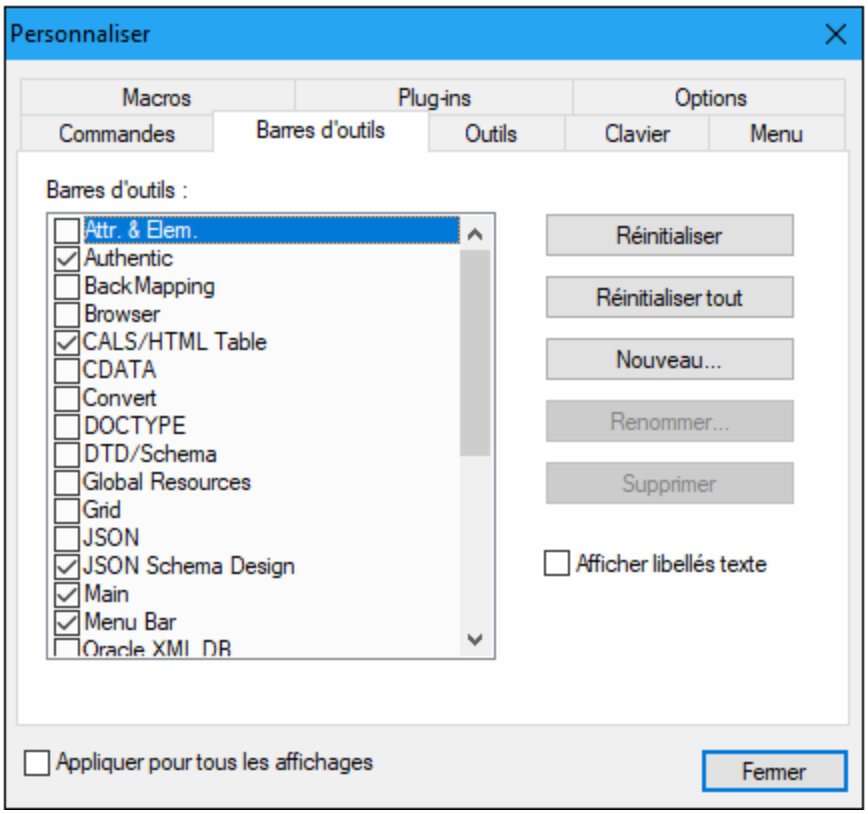
13.9.8.2 Barres d'outils

L'onglet **Barres d'outils** vous permet de : (i) activer ou désactiver les barres d'outils spécifiques (à savoir, de décider lesquelles sont à afficher dans l'interface); (ii) définir quelles icônes sont affichées dans chaque barre d'outils ; et (iii) créer vos propres barres d'outils spécialisées.

Les barres d'outils contiennent des icônes pour les commandes de menu les plus fréquemment utilisées. L'information concernant chaque icône est affichée dans une infobulle et dans la barre de statut lorsque le curseur est placé au-dessus de l'icône. Vous pouvez glisser une barre outils n'importe où sur l'écran où elle apparaîtra en tant que fenêtre flottante.

Note : Pour ajouter une commande à une barre d'outils, glisser la commande que vous souhaitez ajouter depuis la liste *Commandes* dans l'onglet **Commandes** à la barre outils. Pour supprimer une commande d'une barre d'outils, ouvrir le dialogue personnaliser et, après avoir sélectionné un onglet, glisser la commande hors de la barre outils (voir **Commandes** pour plus de détails).

Note : les paramètres de barre d'outils définis dans un mode particulier sont, par défaut, valides pour ce mode uniquement. Pour appliquer les paramètres à tous les modes, cliquer sur la case à cocher en bas du dialogue.



Les fonctions suivantes sont disponibles :

- *Pour activer ou désactiver une barre d'outils :* cliquer sur la case à cocher dans la liste *Barres d'outils*.
- *Appliquer les changements à tous les modes :* Cliquer la case à cocher en bas du dialogue. Sinon, les changements seront appliqués uniquement au mode actif. Veuillez noter que seuls les changements effectués **après** avoir cliqué sur la case *Tous les modes* s'appliqueront à tous les modes.
- *Pour ajouter une nouvelle barre outils* : cliquer sur le bouton **Nouveau** et donner un nom à la barre d'outils dans le dialogue Nom de la barre d'outils qui s'ouvre. À partir de l'onglet **Commandes**, glisser les commandes dans la nouvelle barre d'outils.
- *Pour changer le nom d'une barre d'outils ajoutée :* sélectionner la barre d'outils ajoutée dans le volet Barres d'outils, cliquer sur le bouton **Renommer** et changer le nom dans le dialogue Barre d'outils qui s'ouvre.
- *Pour réinitialiser la barre Menu :* : Choisir l'item de la *barre de menu* dans le volet des barres d'outils, puis cliquez sur **Réinitialiser**. Cela permet de réinitialiser la barre Menu à l'état d'origine au moment de l'installation de l'application.

© 2016-2022 Altova GmbH

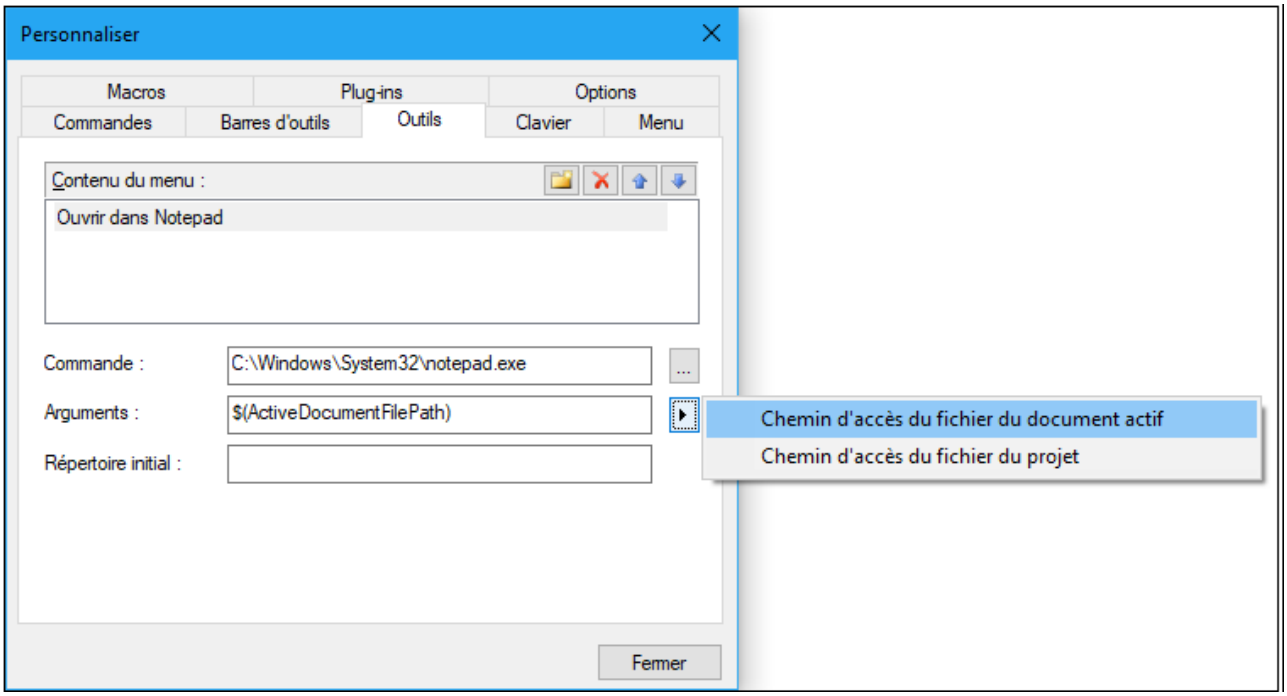
- *Pour réinitialiser toutes les commandes de barre d'outils et des commandes de menu :* Cliquer sur la touche **Tout réinitialiser**. Cela permet de réinitialiser toutes les barres d'outils et les menus à leur état d'origine au moment de l'installation de l'application.

- *Pour supprimer une barre d'outils :* choisir la barre d'outils que vous souhaitez supprimer dans le volet Barres d'outils et cliquer sur **Supprimer**.

- *Pour afficher les libellés de texte de commandes dans une barre d'outils particulière* : choisir cette barre d'outils et cliquer la case à cocher *Afficher les libellés de texte* . Veuillez noter que les libellés doivent être activés séparément pour chaque barre d'outils.

13.9.8.3 Outils

L'onglet **Outils** vous permet de définir les commandes à utiliser pour les applications externes depuis l'intérieur de Authentic Desktop. Ces commandes seront ajoutées au menu **Outils | Outils définis par l'utilisateur**. Par exemple, le fichier actif dans la fenêtre principale de Authentic Desktop peut être ouverte dans une application externe, comme Notepad, en cliquant sur une commande dans le menu Outils | Outils définis par l'utilisateur que vous avez créé.



Pour configurer une commande à utiliser dans une application externe, procéder comme suit :

1. Dans le volet *Contenus de menu* (*voir capture d'écran ci-dessus*), cliquer sur l'icône **Nouveau** dans la barre de titre du volet et, dans la ligne d'item qui est créée, saisir le nom de la commande de menu que vous souhaitez. Dans la capture d'écran ci-dessus, nous avons saisi une seule commande de menu, **Ouvrir dans le Notepad**. Nous souhaitons utiliser cette commande pour ouvrir le document actif dans l'application de bloc-notes externe. D'autres commandes peuvent être ajoutées à la liste de commande en cliquant sur l'icône **Nouveau**. Une commande peut être déplacée vers le haut ou le bas

dans la liste par rapport aux autres commandes en utilisant les icônes **Déplacer vers le haut** et **Déplacer vers le bas**. Pour supprimer une commande, la choisir et cliquer sur l'icône **Supprimer**.

2. Afin d'associer une application externe avec une commande, sélectionner la commande dans le volet *Contenus de menu*. Puis, dans le champ *Commande*, saisir ou chercher le chemin menant vers le fichier exécutable de l'application externe. Dans la capture d'écran ci-dessus, le chemin vers l'application Notepad a été saisie dans le champ *Commande*.
3. Les actions disponibles pour être réalisées avec l'application externe sont affichées lorsque vous cliquez sur la touche fléchée du champ *Arguments* (*voir capture d'écran ci-dessus*). Ces actions sont décrites dans la liste ci-dessous. Lorsque vous sélectionnez un action, un string de code est saisi dans le champ *Arguments*.
4. Si vous souhaitez spécifier un répertoire de travail actuel, il faut le saisir dans le champ *Répertoire initial*.
5. Cliquer sur **Fermer** pour terminer.

La/les commande/s que vous avez créée/s apparaîtra/-ont dans le menu **Outils | Outils définis par l'utilisateur**, et dans le menu contextuel des fichiers et des dossiers — de la fenêtre Projet, dans le sous-menu **Outils définis par l'utilisateur**.

Lorsque vous cliquez sur la commande (dans le menu **Outils | Outils définis par l'utilisateur**) que vous avez créé, l'action que vous avez associée avec la commande sera exécutée. L'exemple de commande affiché dans la capture d'écran ci-dessus effectue l'action suivante : Elle ouvre, dans le bloc-notes, le document qui est actif dans la fenêtre principale de Authentic Desktop. La commande d'application externe est aussi disponible dans le menu contextuel des fichiers dans la fenêtre Projet (cliquer avec la touche de droite sur un fichier dans la fenêtre Projet pour afficher le menu contextuel de ce fichier). Via la fenêtre Projet, vous pouvez aussi ouvrir de multiples fichiers (pour des applications qui le permettent) en faisant une multi-sélection, puis en sélectionnant la commande depuis le menu contextuel.

Arguments

Le champ *Arguments* spécifie l'action à exécuter par la commande d'application externe. Les arguments suivants sont disponibles.

- *Chemin de fichier de document actif :* La commande dans le menu **Outils | Outils définis par l'utilisateur** ouvre le document actif Authentic Desktop dans l'application externe. La commande dans le menu contextuel d'un fichier dans la fenêtre Projet ouvre le fichier sélectionné dans l'application externe.
- *Chemin d'accès de fichier de la fenêtre du Projet :* Ouvre le fichier de projet Authentic Desktop (le fichier `.spp`) dans l'application externe.

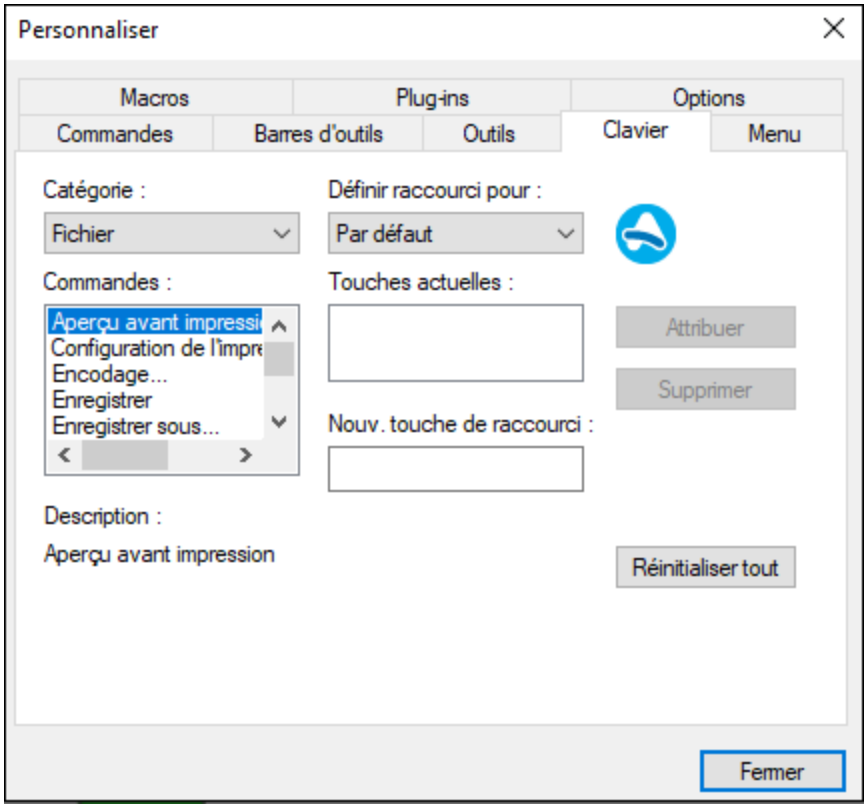
Répertoire initial

L'entrée du *Répertoire initial* est optionnelle et est un chemin qui sera utilisé en tant que le répertoire actuel.

13.9.8.4 Clavier

L'onglet **Clavier** vous permet de créer de nouveaux raccourcis de clavier ou de changer des raccourcis existants, pour toute commande d'application.

© 2016-2022 Altova GmbH



Pour attribuer un nouveau raccourci à une commande, ou pour changer un raccourci existant, procéder comme suit.

1. Sélectionner la catégorie *All Commands* dans la zone de liste modifiable *Category*. Notez que si un macro a été sélectionné comme commande associée , alors les macros sont également disponibles pour la sélection dans la zone de liste modifiable *Category* et un raccourci pour le macro peut être défini.
2. Dans la liste *Commandes*, sélectionnez la commande à laquelle vous souhaitez attribuer un nouveau raccourci ou sélectionnez la commande dont vous souhaitez changer le raccourci.
3. Cliquer sur le bouton dans le champ de saisie *Nouvelle touche de raccourci*, et appuyer sur les touches que vous souhaitez attribuer à cette commande. Le raccourci apparaît dans le champ *Appuyer sur Nouvelle clé de raccourci*. Si le raccourci n'a pas encore été attribué à une autre commande, le bouton **Attribuer** est activé. Si le raccourci a déjà été attribué à une commande, celle-ci s'affichera sous le champ de saisie et le bouton**Attribuer** sera désactivé. (Pour effacer l'entrée dans la zone de texte *Appuyer sur Nouvelle clé de raccourci*, appuyez sur les clés de commandes **Ctrl**, **Alt** ou **Shift**.
4. Cliquez sur le bouton **Attribuer** pour attribuer le raccourci. Le raccourci apparaît maintenant dans la liste *Touches actuelles*. Vous pouvez attribuer plusieurs raccourcis à une seule commande.
5. Cliquez sur la bouton **Fermer**pour confirmer.

Supprimer un raccourci

Un raccourci ne peut pas être attribué pour plusieurs commandes. Si vous souhaitez supprimer un raccourci, cliquer dessus dans la liste Touches actuelles puis cliquer sur le bouton **Supprimer**.

Configurer raccourci pour

Actuellement, les accélérateurs peuvent uniquement être définis en tant que défaut. Aucun autre mode n'est disponible.

Raccourcis de clavier par défaut

Les raccourcis par défaut des commandes utilisées communément sont listés ci-dessous. Un aperçu de toutes les commandes du menu de l'application est disponible dans le modèle de clavier (**Aide | Modèle de clavier**).

- *Function-key shortcuts (y compris pour la validation et la transformation)*

F1	Menu d'aide
F1 + Alt	Ouvrir le dernier fichier
F3	Trouver suivant
F4 + CTRL	Fermer la fenêtre active
F4 + Alt	Fermer Authentic Desktop
F5	Réinitialiser
F6 + CTRL	Faire défiler les fenêtres ouvertes
F7	Vérifier la bonne formation
F8	Valider
F10	Transformation XSL
F10 + CTRL	Transformation XSL:FO

- *Commandes de fichier et d'applications*

Alt + F1	Ouvrir le dernier fichier
Ctrl + O	Ouvrir le fichier
Ctrl + N	Nouveau fichier
Ctrl + P	Imprimer fichier
Ctrl + S	Enregistrer fichier
CTRL + F4	Fermer la fenêtre active
CTRL + F6	Faire défiler les fenêtres ouvertes
CTRL + TAB	Sauter entre des documents ouverts
Alt + F4	Fermer Authentic Desktop

- *Touches diverses*

Touches fléchées haut/bas	Déplacer le curseur ou la barre de sélection
Échap	Abandonner les éditions ou fermer le dialogue
Retour	Confirmer la sélection
Suppr	Supprimer le caractère ou la sélection

© 2016-2022 Altova GmbH

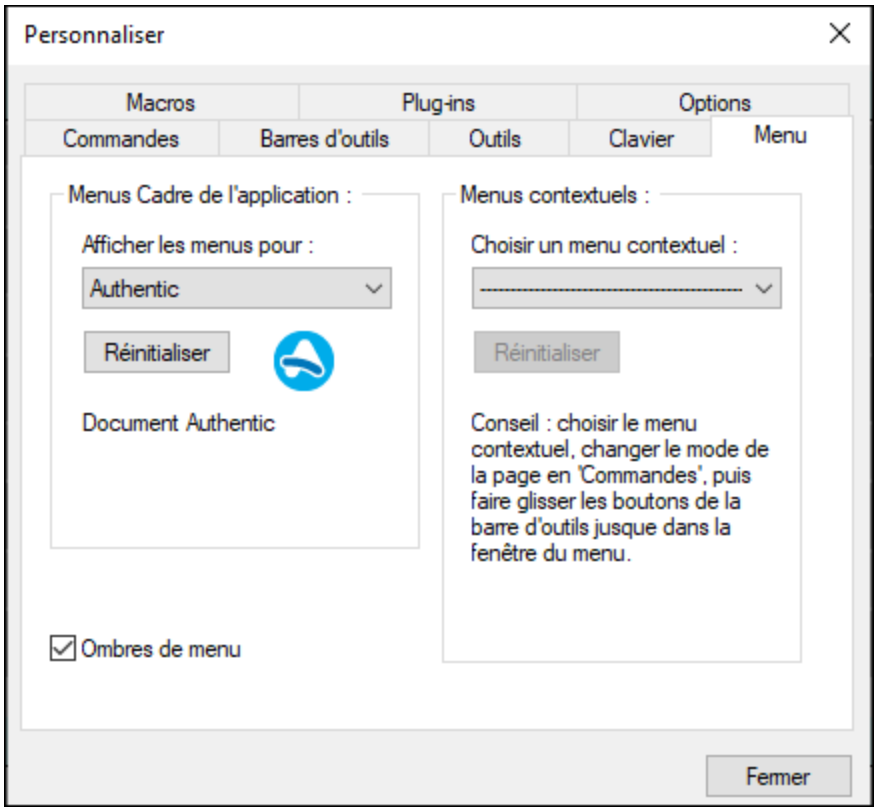
Shift + Del	Couper

- *Commandes d'édition*

CTRL + A	Sélectionner tout
Ctrl + F	Recherche
Ctrl + G	Aller à la ligne/caractère
Ctrl + H	Remplacer
Ctrl + V	Coller
Ctrl + X	Couper
CTRL + Y	Rétablir
Ctrl + Z	Annuler

13.9.8.5 Menu

L'onglet **Menu** vous permet de personnaliser les deux barres de menu principales (barres de menu défaut et application) et les menus de l'application contextuels.



Personnaliser la barre de menu par défaut et la barre de menu d'application

La barre de menu d'application est la barre de menu qui est affichée lorsqu'un ou plusieurs documents sont ouverts dans la fenêtre principale. La barre de menu d'application est la barre de menu qui est affichée lorsqu'un ou plusieurs documents sont ouverts dans la fenêtre principale. Chaque barre de menu peut être personnalisée séparément et les changements de personnalisation effectués sur l'une d'entre elles n'influent pas sur l'autre.

Pour personnaliser une barre de menu, la sélectionner dans la liste de choix *Afficher les menus pour* (*voir capture d'écran ci-dessus*). Ensuite passer à l'onglet Commandes du dialogue Personnalisation et glisser les commentaires depuis la liste Commandes dans la barre de menu ou dans n'importe lequel des menus.

Supprimer les commandes des menus et réinitialiser les barres de menu

Pour **supprimer** un menu complet ou une commande à l'intérieur d'un menu, procéder comme suit :

1. Dans le volet Application Frame Menus, sélectionner soit *Défaut* (qui montre les menus disponibles lorsqu'aucun document n'est ouvert) ou *XMLSpy* Authentic (qui montre les menus disponibles lorsqu'un ou plusieurs documents sont ouverts).
2. Lorsque le dialogue Personnaliser est ouvert, choisir (i) le menu que vous souhaitez supprimer depuis la barres de menu de l'application ou (ii) la commande que vous souhaitez supprimer depuis un de ces menus.
3. Soit (i) glisser le menu depuis la barre de menu ou la commande de menu depuis le menu, soit (ii) cliquer avec la touche de droite sur le menu ou la commande de menu et choisir **Supprimer**.

Vous pouvez **réinitialiser** chacune de ces deux barres de menu (barres de menu par défaut et d'application) à son état d'installation original en sélectionnant le menu dans la liste de choix *Afficher les menus pour* et puis en cliquant sur la touche **Réinitialiser** en-dessous de la liste de choix.

Personnaliser les menus contextuels de l'application

Les menus contextuels sont les menus qui apparaissent lorsque vous cliquez avec la touche de droite sur certains objets dans l'interface de l'application. Chacun de ces menus contextuels peuvent être personnalisés en procédant comme suit :

1. Sélectionner le menu contextuel que vous souhaitez dans la liste de choix *Choisir menu contextuel*. Le menu contextuel s'ouvre.
2. Passer à l'onglet Commandes du dialogue Personnaliser.
3. Glisser une commande depuis la liste *Commandes* dans le menu contextuel.
4. Si vous souhaitez supprimer une commande depuis le menu contextuel, cliquer sur cette commande avec le bouton de droite dans le menu contextuel et cliquer sur **Supprimer**. En alternative, vous pouvez glisser la commande que vous souhaitez supprimer hors du menu contextuel.

Vous pouvez réinitialiser tout menu contextuel à son état d'installation original en le sélectionnant dans la liste de choix *Choisir un menu contextuel* puis en cliquant le bouton **Réinitialiser** en-dessous de la liste de choix.

Ombres de menu

Cliquer sur la case *Ombres de menu* pour ajouter une ombre à tous les menus.

© 2016-2022 Altova GmbH

13.9.8.6 Macros

L'onglet **Macros** vous permet de créer des commandes d'application pour macros qui ont été créés utilisant l'Éditeur de script Authentic Desktop. Ces commandes d'application (qui exécutent les macros associés) peuvent être mises à disposition par la suite dans les menus et barres d'outils, soit directement de l'onglet Macros ou en utilisant les mécanismes disponibles dans l'onglet Commandes du dialogue Personnaliser. En tant que commandes d'application, ils peuvent aussi être attribués à des raccourcis dans l'onglet Clavier du dialogue Personnaliser.

Comment les macros fonctionnent dans Authentic Desktop

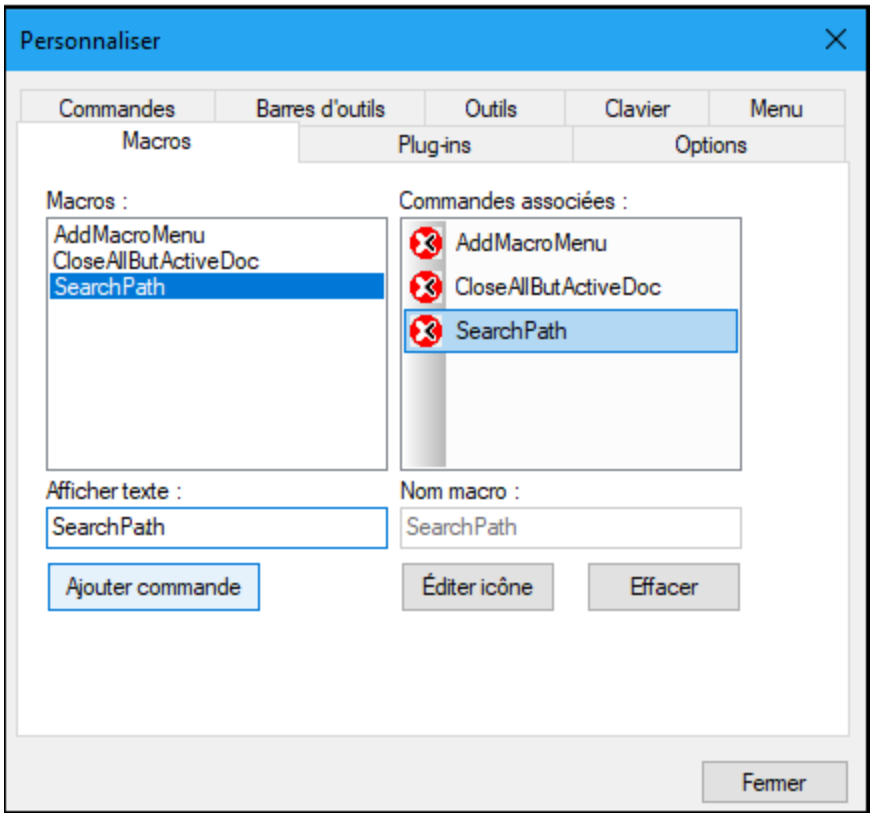
Dans Authentic Desktop, les macros fonctionnent comme suit :

- Les projets de script d'Altova (fichiers `.asprj`) sont créés dans l'Éditeur de script de l'Authentic Desktop ; Éditeur de script. Ce sont ces projets de script qui peuvent contenir les macros utilisés dans Authentic Desktop.
- Deux projets de script peuvent être actifs dans le même temps dans Authentic Desktop : (i) un projet de script d'application, qui est spécifié dans la section Script du dialogue Options, et (ii) le projet de script du projet XMLSpy projet Authentic Desktop actif, qui est spécifié dans le dialogue Paramètres de Script (**Projet | Paramètres du script**).
- Les macros dans ces deux projets de script sont disponibles dans l'application : dans le menu **Projet | Macros** (à partir duquel les macros peuvent être exécutés), et dans l'onglet Macros du dialogue Personnaliser (*capture d'écran ci-dessous*), dans lequel ils peuvent être définis en tant que commandes d'application. Une fois qu'un macro a été défini en tant que commande d'application, la commande peut être placée dans un menu et/ou une barre d'outils.

Créer une commande d'application pour un macro

Dans l'Éditeur de script (**Outils | Éditeur de script**), créer le macro que vous souhaitez et l'enregistrer dans un projet de script. Spécifier ce fichier pour qu'il soit le projet de script de l'application (via la section Script du dialogue des Options) ou du projet d'application actif (par le biais du dialogue Paramètres de Script (**Projet | Paramètres de Script**du projet d'application actif)). Les macros dans le projet de script apparaîtront maintenant dans le volet *Macros* de l'onglet Macros (*voir la capture d'écran ci-dessous*).

Pour créer une commande d'application pour un macro, sélectionner le macro dans le volet *Macros*, définir le texte de la commande dans la fenêtre *Afficher texte* et cliquer sur **Ajouter Commande** (*voir capture d'écran ci-dessous*). Une commande associée avec le macro sélectionné sera ajouté à la liste *Commandes* associées.



Pour éditer l'icône d'une commande associée, sélectionner la commande et cliquer sur **Éditer icône**. Pour supprimer une commande associée, cliquer sur **Supprimer**.

Placer une commande à association macro dans un menu ou une barre d'outils

Il existe deux moyens de placer une commande à association macro dans un menu ou une barre d'outils :

- Glisser la commande depuis la liste Commandes associées à l'emplacement désiré dans le menu ou la barre d'outils.
- Utiliser les mécanismes disponibles dans l'onglet Commandes du dialogue Personnaliser.

Dans les deux cas, la commande sera créée à l'emplacement désiré. Cliquer sur la commande dans le menu ou la barre d'outils permettra d'exécuter le macro.

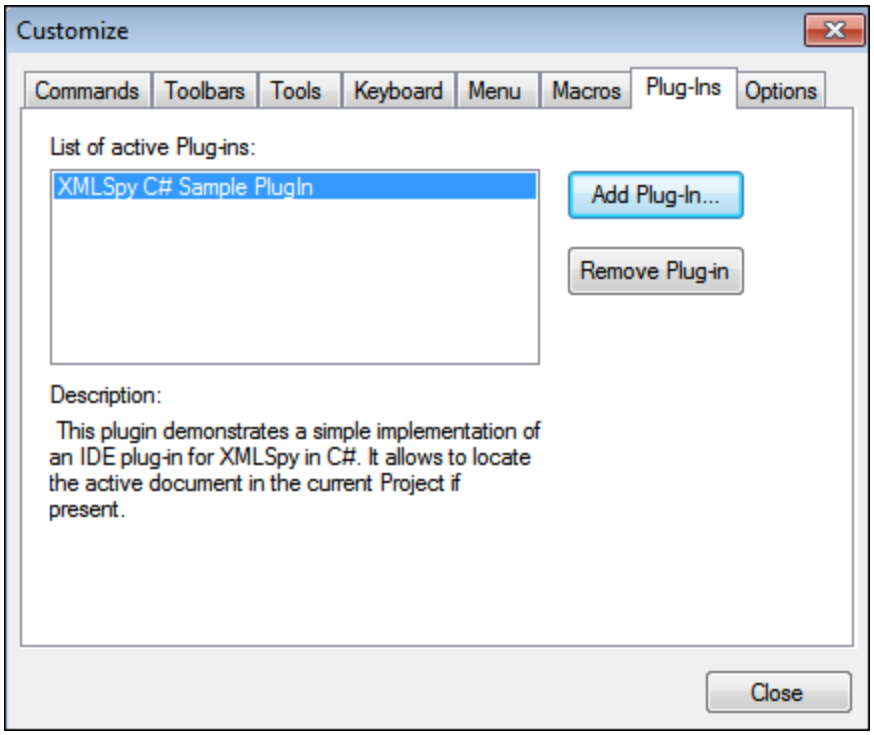
Note : Si un macro a été défini en tant que commande associée, vous pouvez définir un raccourci de clavier pour ce macro. Dans l'onglet Clavier du dialogue Personnaliser, choisir *Macros* dans la liste de choix *Catégorie*, puis sélectionner le macro requis et définir le raccourci. Vous devez définir un macro en tant que commande associée pour qu'elle soit disponible en tant que raccourci de clavier.

13.9.8.7 Plug-ins

L'onglet **Plug-Ins** vous permet d'intégrer les plug-ins et de placer des commandes, lorsque celles-ci ont été programmées de cette manière, dans un menu d'application et/ou barre d'outils. Dans l'onglet Plug-In (*capture*

© 2016-2022 Altova GmbH

d'écran ci-dessous), cliquer sur **Ajouter Plug-In**, et chercher le fichier DLL du plug-in (*voir 'Créer plug-ins' ci-dessous*). Cliquez sur **OK** pour ajouter le plug-in. De multiples plug-ins peuvent être ajoutés.



Une fois qu'un plug-in a bien été ajouté, une description du plug-in apparaît dans le dialogue et la touche **Supprimer le Plug-In** est activée. Si le code de plug-in crée des barres d'outils et des menus, ceux-ci seront immédiatement visibles dans l'interface d'application. Pour supprimer un plug-in, le sélectionner et cliquer sur **Supprimer le Plug-In**.

Créer des plug-ins

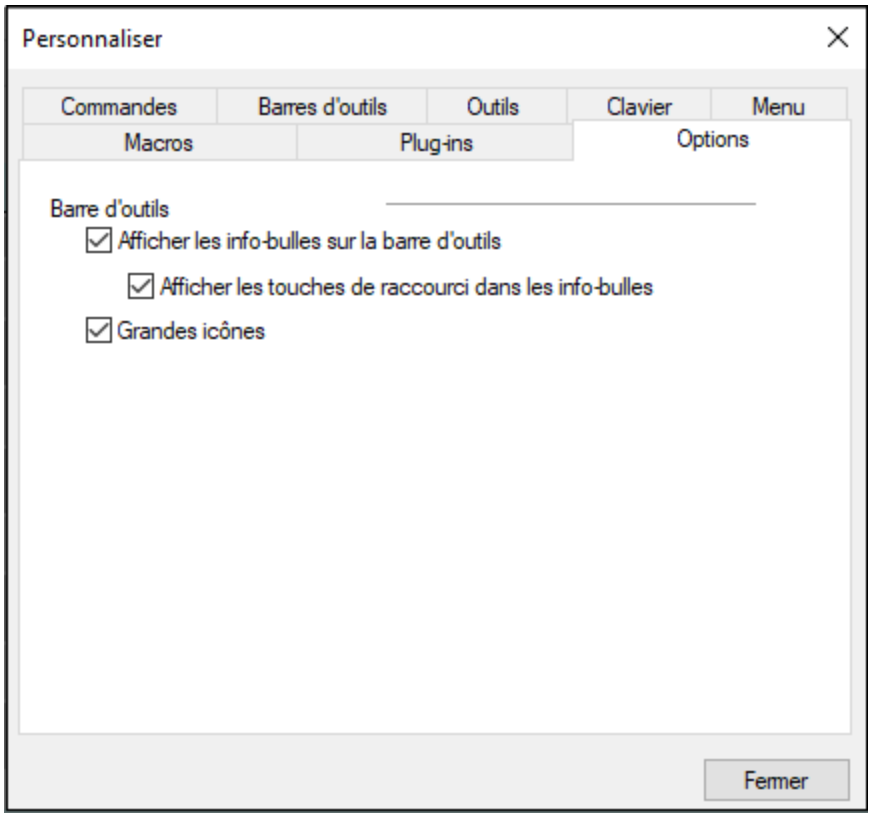
Le code de source pour les échantillons de plug-ins a été fourni dans le dossier (Mes) Documents de l'application : dossier `Examples\IDEPlugin`. Pour générer un plug-in à partir du code de source, procéder comme suit :

1. Ouvrir la solution que vous souhaitez générer en tant que plug-in dans Visual Studio.
2. Générer le plug-in avec la commande dans le menu Générer.
3. Le fichier DLL du plug-in sera créé dans le dossier `Bin` ou `Debug`. Ce fichier DLL est le fichier qui doit être ajouté comme plug-in (*voir ci-dessus*).

Pour plus d'informations sur les plug-ins, voir la section Plugins IDE.

13.9.8.8 Options

L'onglet **Options** vous permet de définir des paramètres d'environnement généraux.



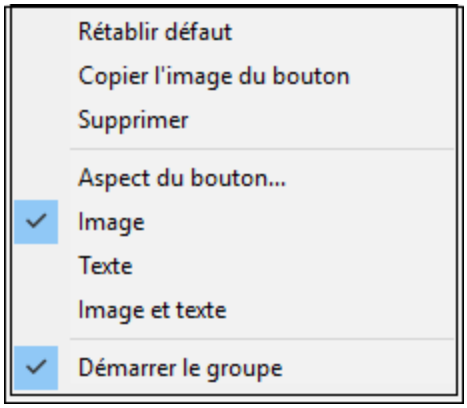
Cliquer sur les cases à cocher pour passer aux options suivantes :

- *Afficher les info-bulles sur la barre d'outils :* affiche un message popup lorsque le pointeur de la souris est placé au-dessus d'une icône dans n'importe quelle barre d'outils. Le message popup contient une brève description de la fonction de l'icône, ainsi que le raccourci clavier associé, dans le cas où un raccourci a été attribué et si l'option *Afficher les touches de raccourci* a été cochée.
- *Afficher les touches de raccourci dans les info-bulles :* définit si les informations de raccourci seront affichées dans les info-bulles.
- *Grandes icônes :* bascule la taille des icônes de la barre d'outils entre standard et grande.

13.9.8.9 Customize Context Menu

Le **menu contextuel Personnaliser** (*capture d'écran ci-dessous*) est le menu qui apparaît quand vous avez le dialogue Personnaliser ouvert, puis cliquez avec la touche de droite un menu d'application, une commande de menu ou une icône de barre d'outils.

© 2016-2022 Altova GmbH

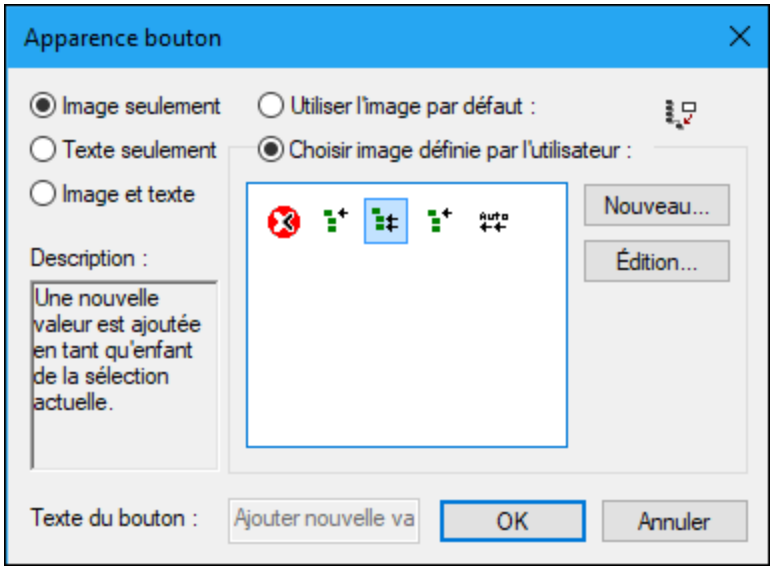


Les fonctions suivantes sont disponibles :

- *Réinitialiser au défaut :* Actuellement pas de fonction.
- *Copier Bouton Image :* Copie l'icône que vous cliquez avec la touche de droite dans le presse-papiers.
- *Supprimer :* Supprime le menu sélectionné, la commande de menu, ou l'icône de la barre d'outils. Pour information comment restaurer les items supprimés, voir ci-dessous.
- *Apparence Bouton :* Affiche le dialogue d'apparence du bouton (*voir la capture d'écran ci-dessous*), dans lequel vous pouvez définir les propriétés qui définissent l'apparence de l'icône de la barre d'outils sélectionnée. Voir la description ci-dessous pour les détails.
- *Image, Texte, Image et Texte* : Des options mutuellement exclusives qui déterminent si l'icône de la barre d'outils sélectionnée sera uniquement un icône, uniquement du texte ou les deux, à savoir icône et texte. Vous pouvez sélectionner une de ces options pour effectuer le changement. En alternative, vous pouvez effectuer ce changement dans le dialogue Apparence du bouton.
- *Groupe de démarrage :* Insère un séparateur de groupe vertical à gauche de l'icône de la barre d'outils sélectionnée. Cela permet de faire de l'icône de la barre d'outils sélectionnée, la première d'un groupe d'icônes.

Le dialogue Apparence du bouton

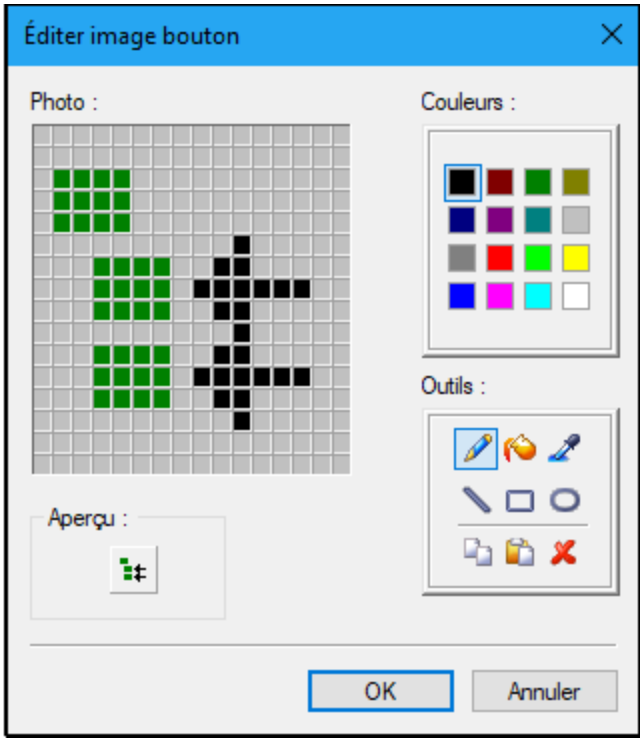
Cliquez avec la touche de droite sur une icône de barre d'outils et cliquez sur **Apparence Bouton** pour obtenir le dialogue Apparence Bouton (*voir la capture d'écran ci-dessous*). Ce dialogue vous permet d'éditer l'image d'icône de la barre outils, ainsi que son texte. Actuellement, uniquement les icônes de barre d'outils pour les macros et des plug-ins peuvent être édités en utilisant ce dialogue.



La fonction d'édition suivante est disponible pour l'icône de la barre d'outils sélectionnée (celle sur laquelle vous avez cliqué avec la touche de droite pour obtenir le menu contextuel Personnaliser) :

- *Image uniquement, Texte uniquement, Image et texte* : Sélectionnez le bouton radio désiré pour spécifier quelle forme l'icône de barre d'outils aura.
- *Édition de l'image :* Lorsque *Image uniquement* ou *Image et texte* est sélectionné, alors les options d'édition de l'image sont activées. Cliquer sur **Nouveau** pour créer une nouvelle image qui sera ajoutée aux images définies par l'utilisateur dans le volet d'images. Sélectionnez une image et cliquez sur **Éditer** pour l'éditer.

© 2016-2022 Altova GmbH



- *Sélection d'image :* Sélectionnez une image depuis le volet Images et cliquez sur OK pour utiliser l'image sélectionnée comme icône de barre d'outils.
- *Édition et sélection de texte :* Lorsque *Texte uniquement* ou *Image et texte* est sélectionné, alors la zone de texte *Texte de bouton* est activée. Saisir ou éditer le texte et cliquer sur **OK** pour en faire le texte de l'icône de la barre d'outils.

Note : le dialogue Apparence Bouton peut aussi être utilisé pour éditer le texte des commandes de menu. Cliquez avec la touche de droite sur la commande de menu (avec le dialogue Personnaliser ouvert), cliquez sur **Apparence Bouton**, puis éditez le texte de commande de menu dans la zone de texte *Texte du bouton*.

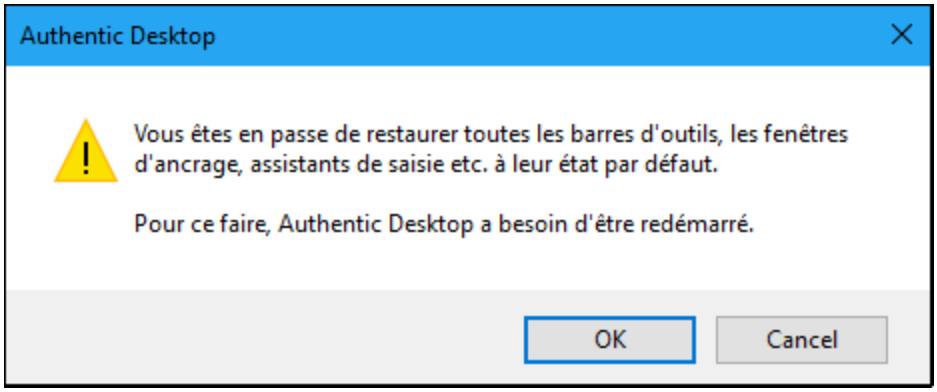
Restaurer les menus supprimés, les commandes de menu et les icônes de barres d'outils

Si un menu, une commande de menu, ou une icône de barre d'outils a été supprimée en utilisant la commande **Supprimer** dans le menu contextuel Personnaliser, ils peuvent être restaurés comme suit :

- *Menus :* Aller à **Outils | Personnaliser | Menu**, et cliquer sur la touche **Réinitialiser** dans le volet *Menus Cadre de l'application*. En alternative, aller à **Outils | Personnaliser | Barres outils**, choisir Barre de menus, et cliquer sur la touche **Réinitialiser**.
- *Commandes de menu :* Aller à **Outils | Personnaliser | Commandes**, et glisser la commande depuis la fenêtre Commandes dans le menu.
- *Icônes de barre d'outils :* Aller à **Outils | Personnaliser | Commandes**, et glisser la commande depuis la fenêtre Commandes dans le menu.

13.9.9 Restaurer les barres d'outils et les fenêtres

La commande **Restaure barres d'outils et fenêtres** ferme Authentic Desktop et la redémarre avec les paramètres par défaut. Avant de fermer le programme un dialogue s'ouvre et vous demande de confirmer si vous souhaitez fermer Authentic Desktop (*capture d'écran ci-dessous*).

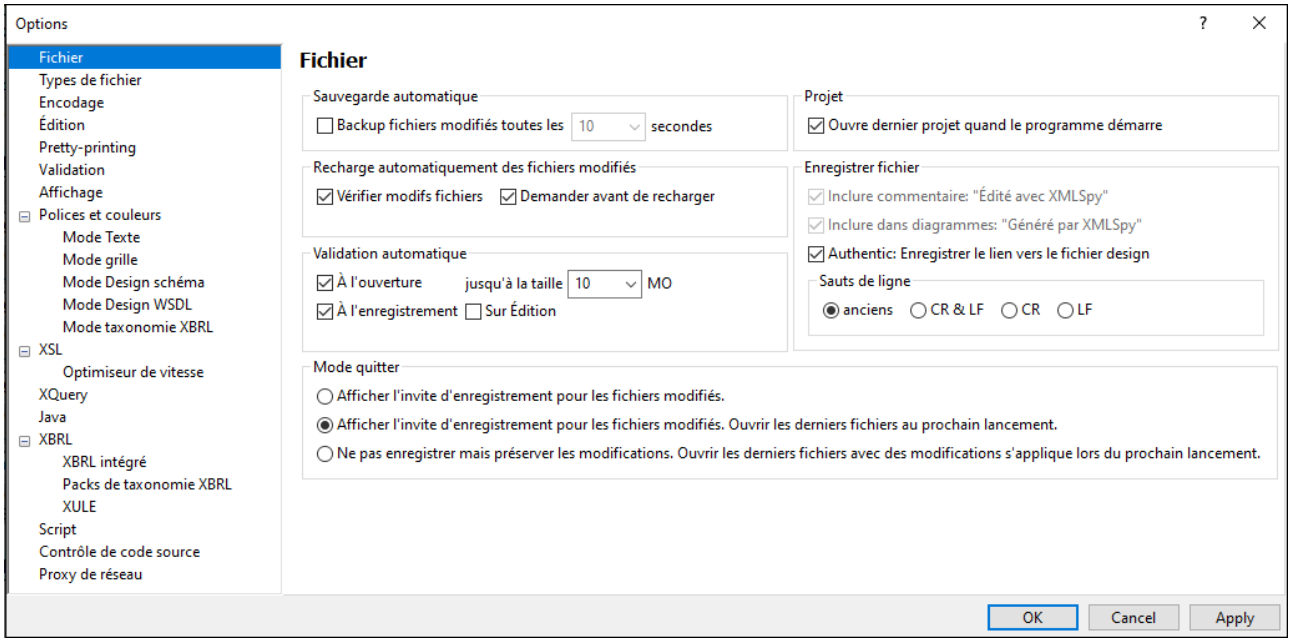


Cette commande est utile si vous avez redimensionné, déplacé ou dissimulé des barres d'outil ou des fenêtres et que vous souhaitez à présent avoir toutes les barres d'outils et les fenêtres à leur place originale.

13.9.10 Options

La commande **Outils** | **Options** vous permet de définir les paramètres d'application globaux. Ces paramètres sont organisés dans les sections (*voir le volet gauche dans la capture d'écran ci-dessous*). Par exemple, la section Fichier (*affichée dans la capture d'écran ci-dessous*) contient des options qui spécifient comment vous voulez Authentic Desktop pour ouvrir et enregistrer les fichiers. Pour spécifier les options d'une section particulière, sélectionnez cette section dans le panneau gauche et spécifiez les valeurs de propriété que vous souhaitez avoir. Le bouton **OK** enregistre les modifications dans le registre et ferme le dialogue. La touche **Appliquer** entraîne l'affichage des modifications dans les documents ouverts actuellement.

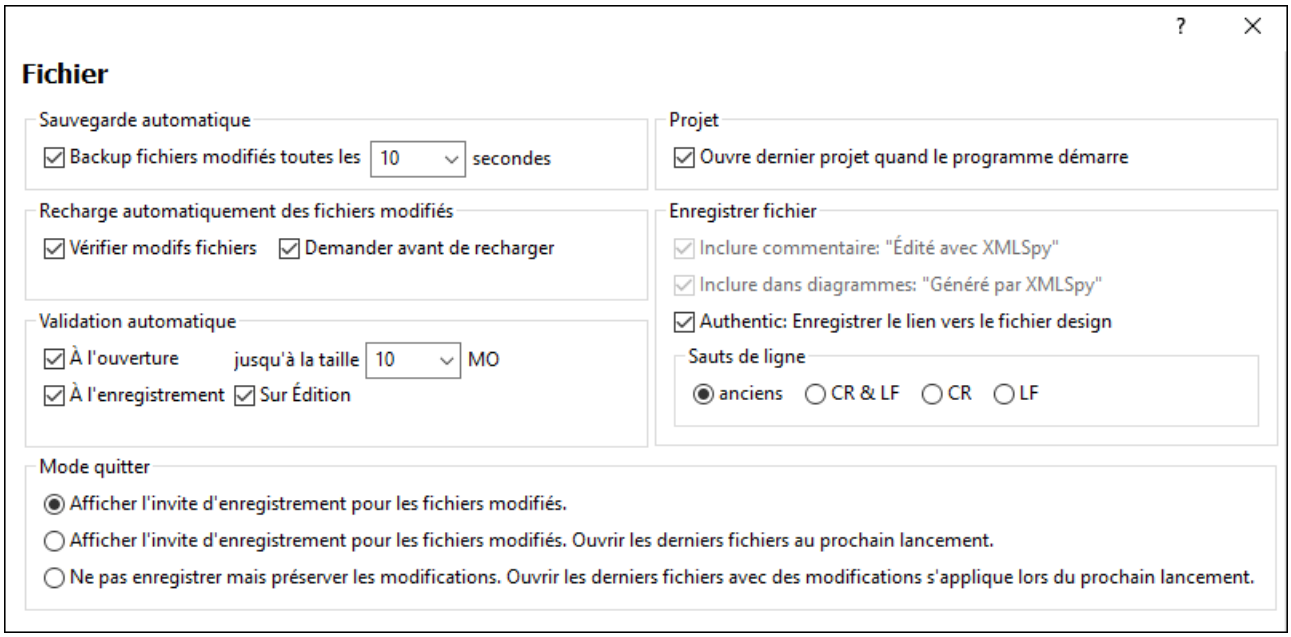
© 2016-2022 Altova GmbH



Chaque section du dialogue des Options est décrite en détail dans sa sous-section de cette section.

13.9.10.1 Fichier

La section **Fichier** définit la manière dont Authentic Desktop ouvre et enregistre des documents. Vous trouverez les paramètres liés dans la section Encodage.



Sauvegarde automatique

Les fichiers que vous éditez actuellement seront sauvegardés automatiquement si cette option est activée. Vous pouvez choisir une fréquence de sauvegarde allant de 5 à 60 secondes dans la liste de choix ou saisir une valeur individuelle de jusqu'à 300 secondes. Pour plus d'informations, voir la section *Sauvegarde automatique de fichiers*.

Le rechargement automatique des fichiers modifiés

Si vous utilisez un environnement à utilisateurs multiples, ou si vous travaillez sur des fichiers générés dynamiquement sur un serveur, vous pouvez détecter les modifications des fichiers actuellement ouverts dans l'interface. Chaque fois que Authentic Desktop détecte un changement dans un document ouvert, il vous demandera si vous souhaitez recharger le fichier modifié.

Validation automatique

Si vous utilisez des schémas DTD ou XML pour définir la structure de vos documents XML, vous pouvez valider automatiquement vos documents d'instance dans les situations suivantes :

- En ouvrant le fichier si le fichier a une taille inférieure à celle que vous spécifiez dans MB
- Sur Enregistrement du fichier
- En éditant le fichier.

Si le document n'est pas valide, un message d'erreur sera affiché. S'il est valide, un message aucun message ne sera affiché et l'opération sera traitée sans aucune notification.

Projet

Lorsque vous lancez Authentic Desktop, vous pouvez ouvrir le dernier projet utilisé en dernier automatiquement.

Enregistrer fichier

Lorsque vous enregistrez un document XML, Authentic Desktop inclut un commentaire succinct `<!-- Edited with Authentic Desktop http://www.altova.com -->` en haut du fichier. Cette option ne peut être désactivée par des utilisateurs de licence et prend effet lors de l'édition ou en enregistrant les fichiers dans la Grille améliorée ou le Mode Design de schéma.

Si StyleVision Power Stylesheet est associée à un fichier XML, l'option 'Authentic: save link to design file' fera que le lien vers StyleVision Power Stylesheet sera enregistré avec un fichier XML.

Sauts de ligne

Lorsque vous ouvrez un fichier, le codage de caractère pour les sauts de ligne dans celui-ci sera préservé si **Preserve old** est sélectionné. En alternative, vous pouvez choisir de coder les sauts de ligne dans un des trois codages : **CR&LF** (pour PC), **CR** (pour MacOS), ou **LF** (pour Unix).

Mode Quitter

Ces options déterminent comment gérer des fichiers qui sont ouverts lorsque Authentic Desktop a quitté. Les options suivantes sont disponibles :

© 2016-2022 Altova GmbH

- *Show save prompt for modified files*: Si un fichier ouvert contient des modifications non enregistrées, une invite apparaîtra demandant si vous souhaitez enregistrer les modifications de fichier. Dépendant de votre réponse, le fichier est enregistré ou pas enregistré et le programme est quitté par la suite.
- *Afficher invite enregistrer pour les fichiers modifiés. Open last files on the next launch*: Le dialogue Enregistrer apparaît pour les fichiers ouverts qui contiennent des modifications non enregistrées. L'utilisateur peut enregistrer un ou plusieurs fichiers ou pas. Lorsque le programme est relancé après avoir quitté, tous les fichiers qui ont été ouverts sur quitter seront ouverts sur redémarrage. (Si les modifications n'avaient pas été enregistrées, elles seraient perdues.)
- *Ne pas enregistrer mais préserver les modifications. Open last files with modifications applied on the next launch*: Le programme quitte directement sans enregistrer les modifications non enregistrées. En relançant le programme, tous les fichiers qui ont été ouverts en quittant seront ouverts en redémarrant, et ils contiendront les modifications non enregistrées. Ce serait une situation comme si vous continuez là où vous avez arrêté.

Si vous quittez le programme pour la première fois, les options en Mode Quitter sont présentées pour que vous puissiez choisir le comportement Quitter de votre choix. Ensuite, les options sont disponibles dans la section Fichier du dialogue Options.

Enregistrer et quitter

Après avoir défini les paramètres, cliquez sur **OK** pour terminer.

13.9.10.2 Types de fichier

La section **Types de fichier** *(capture d'écran ci-dessous)* vous permet de personnaliser le comportement de Authentic Desktop sur une base "par type de fichier". Sélectionnez un type de fichier depuis la zone de liste des Types de fichier, puis personnalisez les fonctions pour ce type de fichier particulier tel que décrit ci-dessous. Veuillez noter qu'il y a deux saisies spéciales dans la liste des types de fichier :

- <*default*> peut être utilisé pour spécifier le traitement de fichiers dotés d'une extension qui ne se trouve pas dans la liste du file-type.
- <*none*> peut être utilisé pour spécifier le traitement de fichiers qui n'ont aucune extension.

Paramètres Windows Explorer

Vous pouvez définir la description de txpe de fichier et le type de contenu conforme à MIME utilisé par Windows Explorer et si Authentic Desktop doit être l'éditeur par défaut pour des documents de ce type de fichier.

Conformité

Authentic Desktop fournit des fonctions d'édition intelligentes spécifiques, ainsi que d'autres fonctions, pour des types de fichier différents. Authentic Desktop définit les fonctions pour un type de fichier particulier sur la base de la conformité que vous avez définie dans cette option. Un grand nombre de types de fichiers est défini avec une conformité par défaut qui est appropriée pour le type de fichier. Nous vous recommandons de ne pas modifier ces paramètres à moins que vous souhaitiez ajouter un nouveau type de fichier ou que vous souhaitez volontairement définir une type de fichier sous une autre conformité.

Mode par défaut

Ce groupe vous permet de définir le mode par défaut à utiliser pour chaque type de fichier.

Mode Texte

La case à cocher *Coloration de la syntaxe* vous permet de définir la coloration de la syntaxe en mode activé ou désactivé pour différents types de fichier.

Désactiver la validation automatique

Cette option vous permet de désactiver la validation automatique par type de fichier. La validation automatique a généralement lieu lorsqu'un fichier est ouvert ou enregistré ou lorsqu'un mode est modifié.

Ajoute une nouvelle extension de fichier

Ajoute un nouveau type de fichier à la liste des Types de fichier. Vous devez ensuite définir les paramètres pour ce nouveau type de fichier en utilisant les autres options dans cet onglet.

Supprimer l'extension de fichier choisie

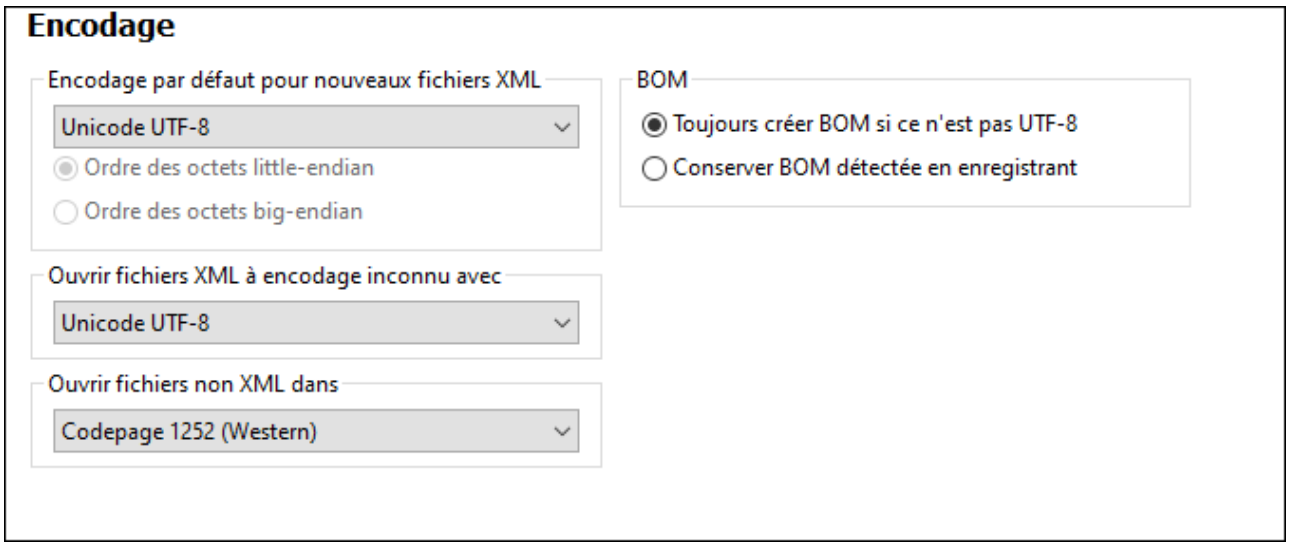
Supprime le type de fichier actuellement sélectionné et tous ses paramètres associés.

Enregistrer et quitter

Après avoir défini les paramètres, cliquez sur **OK** pour terminer.

13.9.10.3 Encodage

La section **Encoding** spécifie les options pour les encodages de fichier.



© 2016-2022 Altova GmbH

Encodage par défaut pour les nouveaux fichiers XML

L'encodage par défaut pour les nouveaux fichiers XML peut être défini en sélectionnant une option depuis la liste déroulante. Un nouveau document est créé avec une déclaration XML contenant la valeur encodée que vous avez spécifié ici. Si un encodage à deux-ou-quatre-byte est sélectionné comme encodage par défaut (par ex., UTF-16, UCS-2 ou UCS-4) vous pouvez aussi choisir entre un tri little-endian et big-endian byte.

L'encodage de fichiers XML existants sera retenu et ne peut être changé qu'avec la commande **Fichier | Encodage**.

Ouvrez un fichier XML avec un encodage inconnu comme

Si l'encodage d'un fichier XML ne peut pas être déterminé ou si le document XML n'a pas de spécification d'encodage, le fichier sera ouvert avec l'encodage que vous sélectionnez dans cette liste de choix.

Ouvrir fichiers non XML dans

Les fichiers existants et les nouveaux fichiers non-XML sont ouverts avec l'encodage que vous avez sélectionné dans cette liste de choix. Vous pouvez changer l'encodage du document en utilisant la commande **Fichier | Encodage**.

BOM (Byte Order Mark)

Lorsqu'un document à encodage de caractère deux octets ou quatre octets est enregistré, le document peut être enregistré soit avec (i) un ordre des octets little-endian et un BOM little-endian (*Toujours créer BOM si différent de UTF-8*); soit (ii) avec l'ordre des octets détecté et le BOM détecté (*Préserver BOM détecté à l'enregistrement*).

Enregistrer et quitter

Après avoir défini les paramètres, cliquez sur **OK** pour terminer.

13.9.10.4 Pretty Printing

La section **Pretty Printing** *(voir la capture d'écran ci-dessous)* vous permet de spécifier comment le texte est affiché dans le document XML représenté dans le Mode Authentic. Bien que Authentic Desktop ne fournit pas le mode texte du document XML, les paramètres que vous choisissez ici seront appliqués au texte brut du fichier XML que vous avez édité dans Authentic View.

Les paramètres sont :

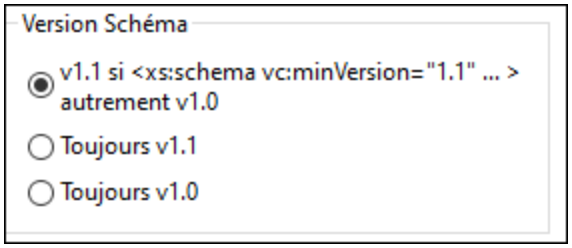
- Pour que vous puissiez choisir si utiliser les onglets ou espaces pour le pretting-printing.
- Pour savoir comment les éléments vides sont à écrire. Un élément de fermeture automatique est un élément dont les balises d'ouverture et de fermeture sont combinées en un, comme ceci : `<element/>` ou `<element />`.

Enregistrer et quitter

Après avoir défini les paramètres, cliquez sur **OK** pour terminer.

13.9.10.5 Validation

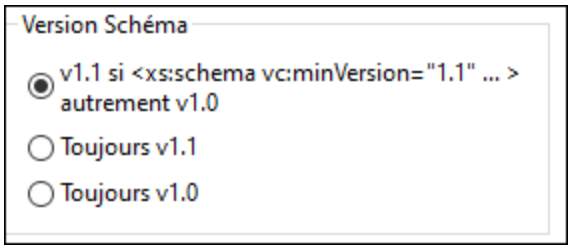
La section **Validation** vous permet de spécifier des options pour valider les documents XML.



XML

Authentic Desktop peut mettre en cache des fichiers de schéma DTD et XML en mémoire pour éviter un rechargement non nécessaire (par exemple, quand le schéma n'est pas local mais est accédé via un URL). Veuillez noter, toutefois, que si vous utilisez des versions en cache de schémas, les modifications apportées à votre schéma ne seront pas immédiatement reflétées quand vous les validez ; dans ce cas, vous devrez recharger le fichier XML ou redémarrer Authentic Desktop.

Version Schéma
Le mode XSD qui est activé dans le mode Schéma dépend de (i) la présence/l'absence — et, si présent, la valeur — de l'attribut `/xs:schema/@vc:minVersion` du document XSD, et (ii) l'option de la version de Schéma XML sélectionnée dans la section Fichier du dialogue des Options (**Outils | Options**, *voir la capture d'écran ci-dessous*).



Les situations suivantes sont possibles. La *Version de schéma XML* dans la table ci-dessous fait référence à la sélection dans le volet de la version de schéma XML ci-dessus. Les valeurs `vc:minVersion` dans la table se réfèrent à la valeur de l'attribut `xs:schema/@vc:minVersion` dans le document de schéma XML.

Version de Schéma XML	`vc:minVersion` attribut	Mode XSD
Toujours v1.0	est absent, ou est présent avec toute valeur	1.0
Toujours v1.1	Est absent, ou est présent avec toute valeur	1.1
Valeur de @vc:minVersion	Attribut a une valeur de `1.1`	1.1

© 2016-2022 Altova GmbH

Valeur de @vc:minVersion	Attribut est absent, ou attribut est présent avec une valeur autre que `1.1`	1.0

Limites de message

Ces options vous permettent de définir les limites séparées pour le nombre d'erreurs, inconsistances XBRL et avertissements qui sont affichés. Le nombre par défaut pour chaque catégorie est de 100. Modifiez-le au nombre souhaité.

Enregistrer et quitter

Après avoir défini les paramètres, cliquez sur **OK** pour terminer.

13.9.10.6 Mode

La section **Mode** vous permet de personnaliser la présentation des documents XML dans Authentic Desktop.

Logo du programme

Vous pouvez désactiver l'écran de démarrage sur démarrage du programme pour rendre l'application plus rapide. De même, si vous avez une licence achetée (comparé à, par exemple, une licence d'évaluation), vous aurez l'option de désactiver le logo de programme, la notice de copyright et les détails d'inscription lorsque vous imprimez un depuis XMLSpy.

Titre de fenêtre

Le titre de fenêtre pour chaque fenêtre de document peut contenir soit uniquement le nom de fichier ou le nom entier du chemin.

Moteur de navigation

Le moteur de navigation qui est utilisé dans Authentic View et Browser View est actuellement Internet Explorer (IE), et IE est donc le navigateur par défaut pour ces deux modes. En alternative, vous pouvez utiliser Microsoft Edge Web View 2 comme moteur pour le mode Navigateur. Si Edge n'est pas installé sur votre machine, allez à WebView2 page de téléchargement de laquelle vous pouvez installer Evergreen Bootstrapper. Ceci vous permettra d'utiliser Microsoft Edge WebView2 comme moteur pour le mode Navigateur.

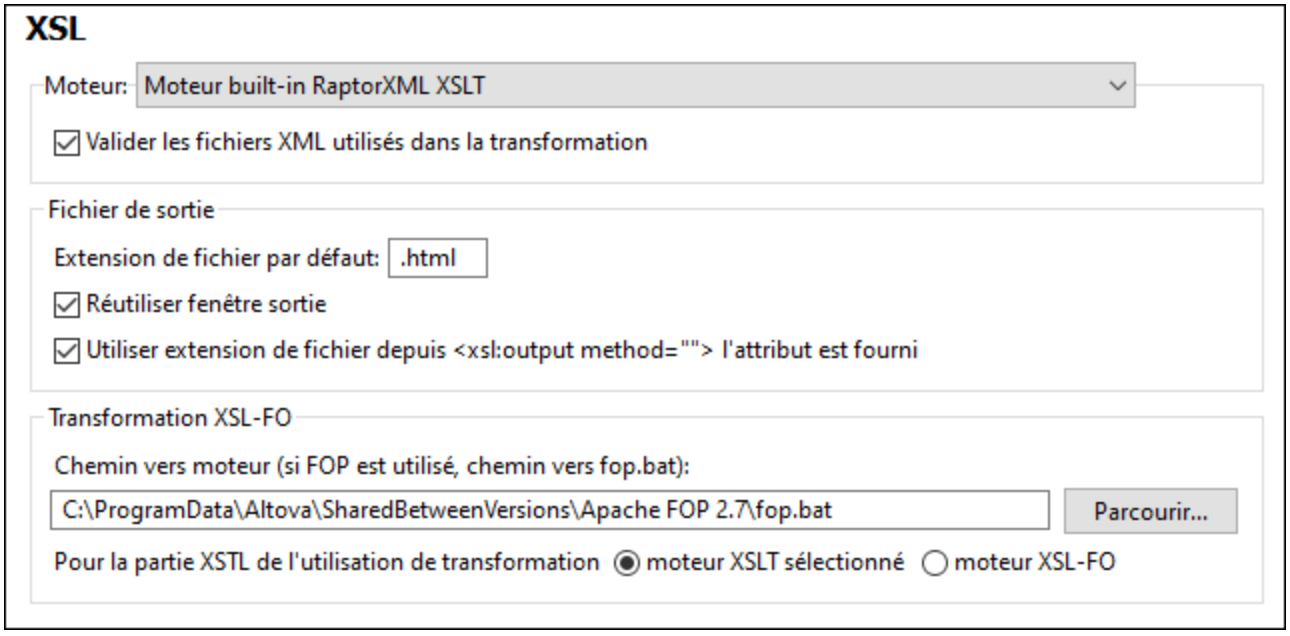
Voir le sujet Mode navigateur pour plus d'informations.

Enregistrer et quitter

Après avoir défini les paramètres, cliquez sur **OK** pour terminer.

13.9.10.7 XSL

La section **XSL** (*capture d'écran ci-dessous*) vous permet de définir des options pour les transformations XSLT et les transformations XSL-FO effectuées depuis l'application.



Paramètres de moteur

Vous pouvez établir un processeur XSLT pour exécuter les transformations XSLT lorsque la commande **Transformation XSLT** est invoquée.

Vous pouvez choisir parmi une des options de moteur XSLT suivantes :

- Moteur built-in RaptorXML XSLT
- Parseur Microsoft XML (MSXML)
- Moteur externe XSLT

Note: pour un débogage XSLT dans Authentic Desktop, le moteur built-in RaptorXML XSLT est toujours utilisé — même si un autre moteur XSLT est sélectionné ici pour les transformations.

Moteur Altova RaptorXML XSLT

Authentic Desktop contient les moteurs Altova RaptorXML XSLT 1.0, XSLT 2.0 et XSLT 3.0 que vous pouvez utiliser pour les transformations XSLT. Le moteur XSLT approprié (1.0, 2.0 ou 3.0) est utilisé (conformément à la valeur de l'attribut `version` de l'élément `xsl:stylesheet` ou `xsl:transform`). Cela vaut aussi bien pour les transformations XSLT que pour le débogage XSLT utilisant le Débogueur XSLT/XQuery de XMLSpy.

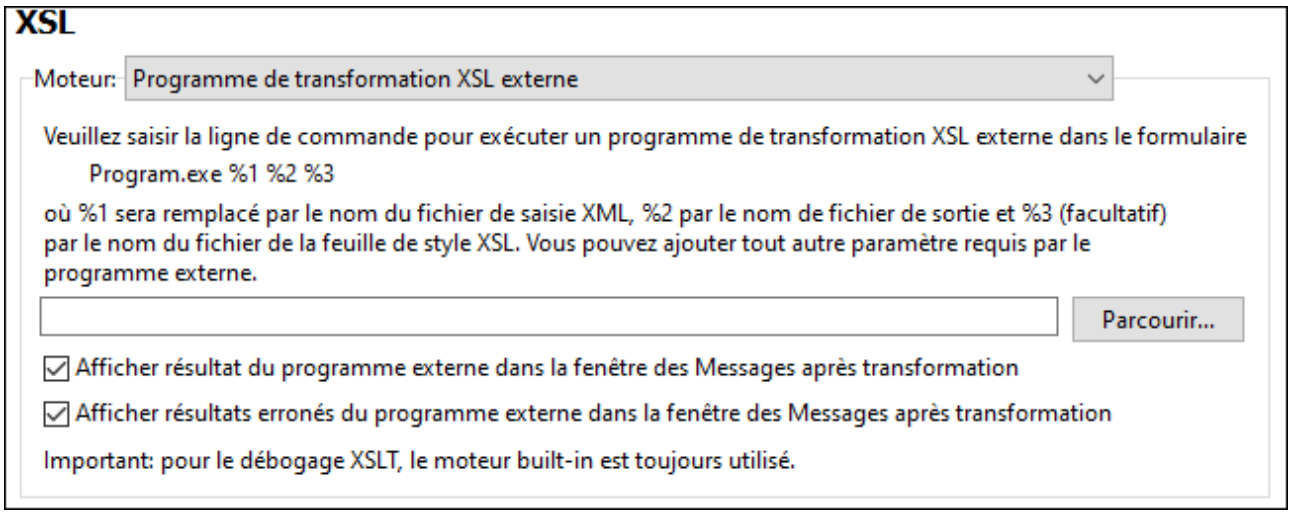
Si vous souhaitez valider les fichiers XML files utilisés dans la transformation, sélectionnez l'option *Valider* (*voir la capture d'écran ci-dessus*).

Parseur Microsoft XML (MSXML)

Un ou plus de parseurs de MSXML 3.0, 4.0 ou 6.0 seront préinstallés sur votre appareil. Si vous savez quelle version installée vous voulez utilisez, vous pourriez la sélectionner. Autrement, vous devriez laisser la sélection de la version automatiquement à Authentic Desktop. (L'option *Choisir version automatiquement* est active par défaut.) Dans ce cas, Authentic Desktop tente de choisir la version disponible la plus récente.

© 2016-2022 Altova GmbH

Moteur externe XSLT
Choisissez un processeur externe XSLT de votre choix en saisissant le chemin à son fichier exécutable.



Vous devez spécifier le string de ligne de commande que le processeur externe XSLT utilise pour exécuter une transformation. Vous pouvez créer le string de ligne de la commande avec les composants suivants :

`%1` = document XML à traiter
`%2` = Fichier de sortie à générer
`%3` = Feuille de style XSLT à utiliser (si le document XML ne contient pas de référence à la feuille de style)

Par exemple, vous avez un processeur qui utilise le modèle de commande suivant pour exécuter une transformation XSLT :

```
myxsltengine.exe -o <output.xml> <input.xml> <stylesheet.xslt> <param-name>=<param-value>?
```

Ensuite, dans Authentic Desktop, créez la ligne de commande utilisant les variables dans les emplacements corrects. Par exemple :

```
c:\MyEngine.exe -o %2 %1 %3 date=2023
```

Authentic Desktop vous enverra les fichiers d'entrée corrects au moteur externe pour le traitement renverra le/s fichier/s de sortie vers un emplacement de sortie s'il est précisé et/ou vers une fenêtre d'application.

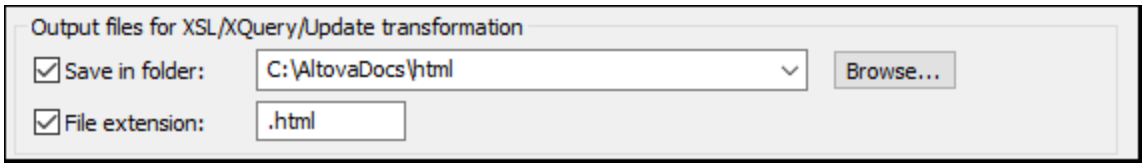
Cochez les cases respectives pour montrer les messages de sortie et d'erreur du programme externe dans la fenêtre Messages de Authentic Desktop.

Note : Les paramètres définis dans le dialogue des Paramètres d'entrée XSLT (accès via le menu **XSL**) sont passés aux Moteurs XSLT internes Altova uniquement. Ils ne sont pas passés dans un autre Moteur XSLT configuré en tant que processeur XSLT par défaut.

Paramètres fichier Output

Les options suivantes sont disponibles :

- *Extension de fichier par défaut :* Définit une extension de fichier par défaut pour les fichiers de sortie, qui peuvent être substitués par l'extension de fichier nommée dans l'élément XSLT `xsl:output` (*voir le dernier item de la liste*).
- *Réutiliser la fenêtre de sortie :* Entraîne des transformations subséquentes pour afficher le document de résultat dans la même fenêtre de sortie. Si le fichier d'entrée XML appartient à un projet et que l'option *Réutiliser la fenêtre de sortie* est désactivée, le paramètre prend effet si le chemin de fichier de sortie *Enregistrer dans le dossier* (*capture d'écran ci-dessous*) se trouvant dans les propriétés de projet pertinentes est **aussi** désactivé.



- *Utilisez l'extension de fichier de l'élément xsl:output:* Sélectionne si l'extension de fichier spécifié dans l'élément `xsl:output` de la feuille de style XSLT substituerait l'extension par défaut spécifiée dans la première option de cette liste.

Transformations XSL-FO

Les documents FO sont traités avec un processeur FO, et le chemin vers l'exécutable du moteur FO doit être spécifié dans zone de texte pour le moteur de transformation XSL-FO. La transformation est effectuée à l'aide de la commande de menu **Transformation XSL/XQuery | XSL-FO**. Si le fichier source (le document actif lorsque la commande est exécutée dans l'IDE) est un document XSL-FO, le moteur FO est invoqué pour la transformation. Si le document source est un document XML, une transformation XSLT est requise pour tout d'abord convertir le document XML en un document XSL-FO. Cette transformation XSLT peut être effectuée soit par le biais du moteur XSLT que vous avez spécifié en tant que moteur par défaut pour l'application (*voir ci-dessus*), ou par le moteur XSLT qui pourrait être intégré dans le moteur FO que vous avez spécifié en tant que le moteur FO par défaut pour l'application. Pour effectuer un choix entre ces deux options, cliquez sur le bouton radio approprié.

Après avoir défini les paramètres, cliquez sur **OK** pour terminer.

Note : À moins d'avoir désélectionné l'option pour installer le moteur FOP du Projet Apache XML, celui-ci sera installé dans le dossier `C:\ProgramData\Altova\SharedBetweenVersions`. En cas d'installation, le chemin y menant sera automatiquement saisi dans le champ d'entrée du Moteur XSL-FO. Vous pouvez déterminer le chemin vers n'importe lequel des moteurs FO que vous souhaitez utiliser. Veuillez noter, néanmoins, que le même chemin sera utilisé par d'autres produits Altova qui utilisent les moteurs FO et comportent des paramètres pour sélectionner le processeur FO (StyleVision et Authentic Desktop).

Enregistrer et quitter

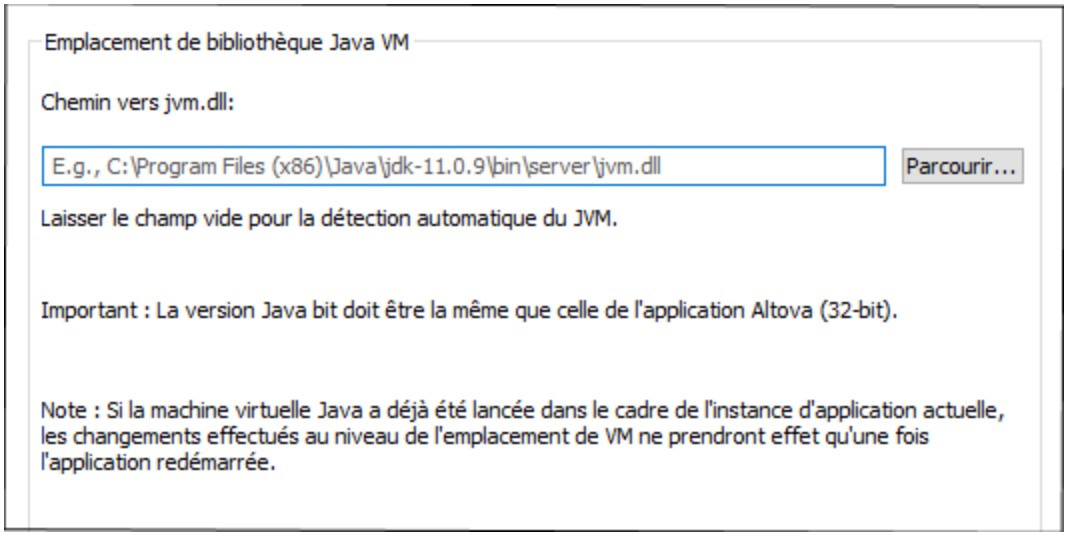
Après avoir défini les paramètres, cliquez sur **OK** pour terminer.

© 2016-2022 Altova GmbH

13.9.10.8 Java

Dans l'onglet *Java* (*voir la capture d'écran ci-dessous*), vous pouvez saisir en option un chemin vers une Java VM (Machine Virtuelle) sur votre système de fichier. Veuillez noter que le fait d'ajouter un chemin Java VM personnalisé n'est pas toujours nécessaire. Par défaut, Authentic Desktop tente de détecter le chemin Java VM automatiquement en lisant (dans cet ordre) le registre Windows et la variable d'environnement JAVA_HOME. Le chemin personnalisé ajouté dans ce dialogue prendra la priorité sur tout autre chemin Java VM détecté automatiquement.

Vous devrez éventuellement ajouter un chemin Java VM personnalisé, par exemple si vous utilisez une machine virtuelle Java qui ne possède pas de programme d'installation et ne crée pas d'entrées de registre (par exemple, OpenJDK d'Oracle). Vous pourrez également définir ce chemin si vous souhaitez contourner, pour une raison quelconque, tout chemin Java VM détecté automatiquement par Authentic Desktop.

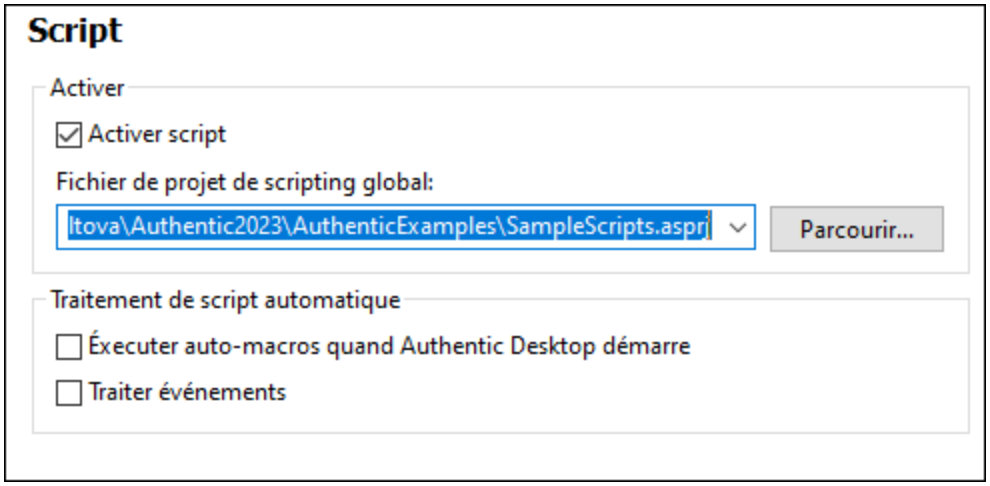


Veuillez noter les points suivants :

- Le chemin Java VM est partagé entre les applications de desktop (pas serveur) Altova. Par conséquent, si vous le modifiez dans une application, il s'appliquera automatiquement à toutes les autres applications Altova.
- Le chemin doit pointer vers le fichier **`jvm.dll`** provenant du répertoire **`\bin\server`** ou **`\bin\client`**, par rapport au répertoire sur lequel le JDK a été installé.
- La plateforme Authentic Desktop (32-bit, 64-bit) doit être la même que celle du JDK.
- Une fois avoir modifié le chemin Java VM, vous devrez éventuellement redémarrer Authentic Desktop pour que les nouveaux paramètres prennent effet.

13.9.10.9 Script

La section **Script** (*capture d'écran ci-dessous*) vous permet d'activer l'Environnement de script lors du démarrage de l'application. Cocher la case *Activer Script* pour ce faire. Vous pouvez ensuite spécifier le fichier de Projet de script global (*voir* capture d'écran ci-dessous).



Pour définir le projet de scripting global pour Authentic Desktop, vérifier la case à cocher *Activer le Scripting*, puis rechercher le fichier Altova Scripting Project (`.asprj`) que vous souhaitez. Vous pouvez aussi spécifier : (i) si les Auto-macros contenus dans le projet de script doivent être exécutés automatiquement au démarrage de Authentic Desktop, et (ii) si les scripts de gestionnaire d'événement d'application dans le projet doivent être exécutés automatiquement ou pas ; cocher ou décocher les cases selon vos besoins.

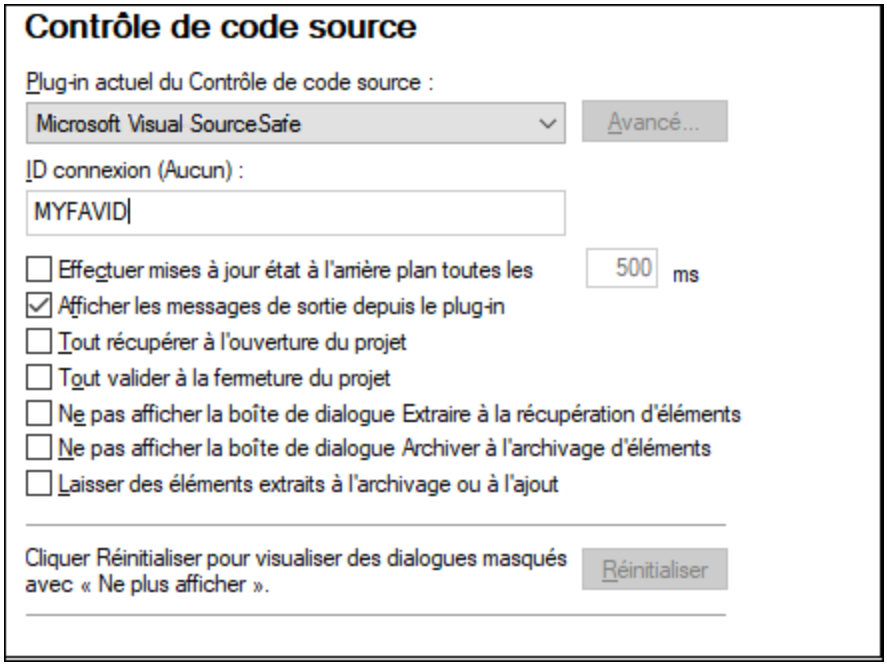
Enregistrer et quitter

Après avoir défini les paramètres, cliquez sur **OK** pour terminer. Les macros contenus dans le Projet de script global seront ensuite affichés dans le sous-menu de la commande **Macros**.

13.9.10.10 Contrôle de source

La section **Contrôle de source** (*voir la capture d'écran ci-dessous*) vous permet de spécifier le prestataire de contrôle de source ainsi que les paramètres et l'ID de connexion par défaut pour chaque prestataire de contrôle de source.

© 2016-2022 Altova GmbH



Plugin de Contrôle de source

Le plugin de contrôle de source actuel peut être sélectionné parmi les systèmes de contrôle de source installés. Ces systèmes sont listés dans la liste déroulante de la liste de choix. Une fois avoir sélectionné le contrôle de source requis, spécifier l'ID de connexion nécessaire dans le champ de saisie suivant. La touche **Avancé** ouvre un dialogue spécifique au plugin de contrôle de source sélectionné, dans lequel vous pouvez définir des paramètres pour ce plugin de contrôle de source. Ces paramètres sont différents pour les différents plugin de contrôle de source.

Préférences de l'utilisateur

Une gamme de préférences de l'utilisateur est disponible, y compris :

- Les mises à jour de statut peuvent être exécutées en arrière-plan au bout d'un intervalle défini par l'utilisateur, ou bien, elles peuvent être éteintes complètement. Les bases de données de contrôle de source très grandes pourraient consommer des ressource CPU et de réseau considérables. Le système peut néanmoins être accéléré en désactivant les mises à jour en arrière-plan du statut ou en augmentant les intervalles entre elles.
- Lors de l'ouverture et de la fermeture de projets, les fichiers peuvent être automatiquement extraits ou archivés, respectivement.
- L'affichage des dialogues Extraire et Archiver peut être supprimé.
- La touche **Réinitialiser** est activée si vous avez coché/activé l'option *Ne plus afficher* dans un des dialogues. Cliquer sur la touche **Réinitialiser** pour réactiver l'invite *Ne plus afficher*.

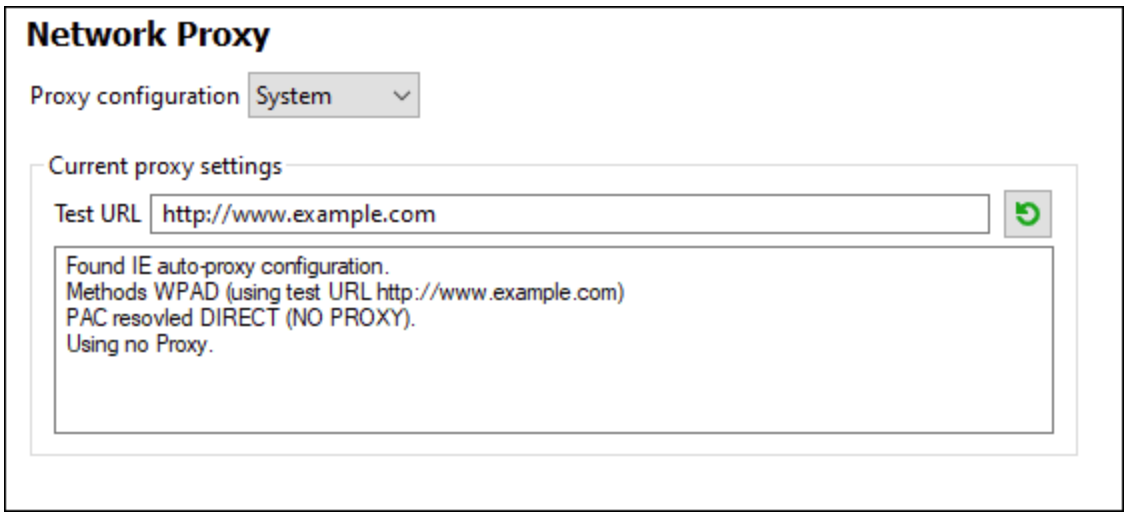
Enregistrer et quitter

Après avoir défini les paramètres, cliquez sur **OK** pour terminer.

13.9.10.11 Proxy de réseau

La section *Proxy de réseau* vous permet de configurer des paramètres de proxy personnalisés. Ces paramètres ont une incidence sur la manière dont l'application se connecte à Internet (par exemple, pour une validation XML). Par défaut, l'application utilise les paramètres proxy du système, dans la plupart des cas, vous n'aurez donc pas à changer les paramètres de proxy. Si vous souhaitez, toutefois, définir un proxy réseau alternatif, dans la zone de liste de *configuration Proxy*, sélectionnez *Automatic* ou *Manual* pour configurer les paramètres en conséquence.

Note : les paramètres de proxy de réseau sont partagés parmi toutes les applications de Altova MissionKit. Donc, si vous modifiez les paramètres dans une application, toutes les applications de MissionKit en seront touchées.



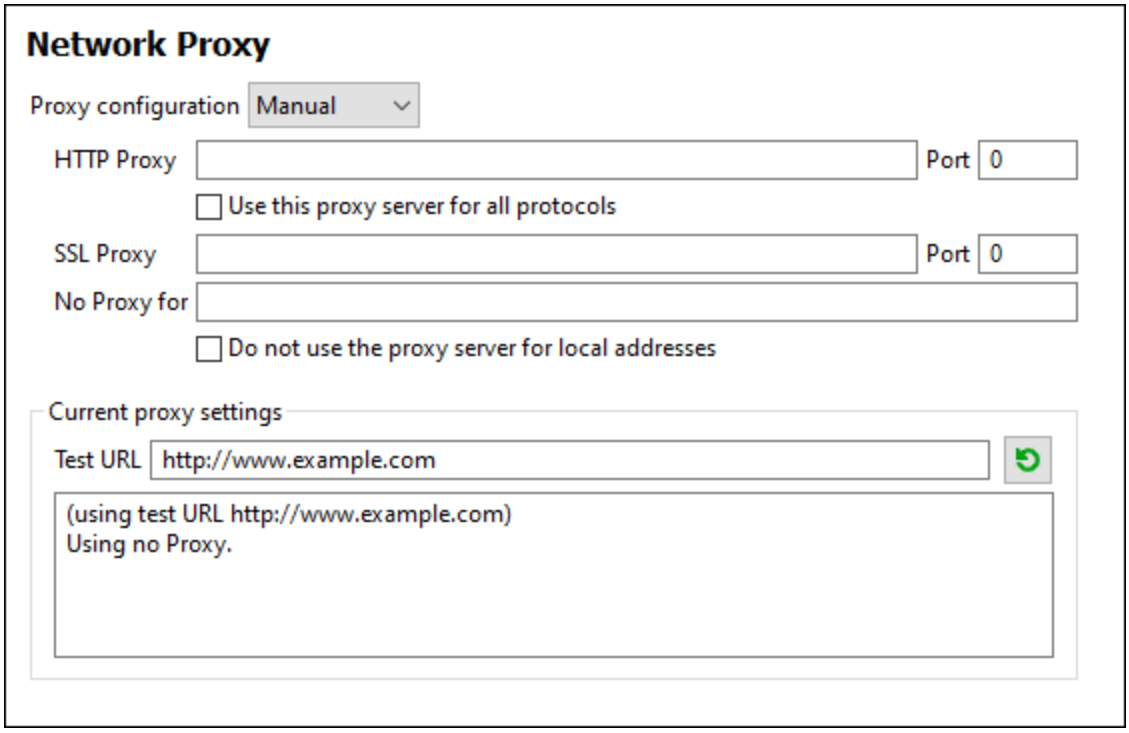
Utiliser les paramètres de proxy de système
Utilise les paramètres Internet Explorer (IE) configurables par le biais des paramètres de proxy du système. Effectue également une requête des paramètres configurés avec **`netsh.exe winhttp`**.

Configuration de proxy automatique
Les options suivantes sont possibles :

- *Paramètres auto-détection :* Consulte un script WPAD (**`http://wpad.LOCALDOMAIN/wpad.dat`**) par le biais de DHCP ou DNS, et utilise ce script pour une configuration proxy.
- *URL de script :* Spécifie une URL HTTP dans un script configuration-auto-proxy (**`.pac`**) qui doit être utilisé pour cette configuration de proxy.
- *Recharger :* Réinitialise et recharge la configuration automatique de proxy actuelle. Cette action requiert Windows 8 ou plus, et peut prendre jusqu'à 30 sec avant de prendre effet.

Configuration de proxy manuelle
Spécifier manuellement le nom d'hôte et le port entièrement qualifiés pour les proxies des protocoles respectifs. Un scheme pris en charge peut être inclus dans le nom d'hôte (par exemple : **`http://hostname`**). Il n'est pas exigé que le schéma soit le même que le protocole respectif si le proxy prend en charge le schéma.

© 2016-2022 Altova GmbH



Les options suivantes sont possibles :

- *Proxy HTTP :* Utilise le nom d'hôte spécifié et le port pour le protocole HTTP. Si *Utiliser le serveur proxy pour tous les protocoles* est sélectionné, alors le proxy HTTP spécifié est utilisé pour tous les protocoles.
- *Proxy SSL :* Utilise le nom d'hôte spécifié et le port pour le protocole SSL.
- *Pas de proxy pour :* Une liste séparée par point-virgule (`;`) de nom d'hôtes entièrement qualifiés, de noms de domaine, ou d'adresses IP pour des hôtes qui doivent être utilisés sans un proxy. Les adresses IP ne doivent pas être abrégées et les adresses IPv6 doivent être entourées de crochets (par exemple : `[2606:2800:220:1:248:1893:25c8:1946]`). Les noms de domaine doivent commencer avec un point (par exemple : `.example.com`).
- *Ne pas utiliser le serveur proxy pour les adresses locales :* Si cochées, ajoute `<local>` à la liste *Pas de proxy pour*. Si cette option est sélectionnée, les éléments suivants n'utiliseront pas le proxy : (i) `127.0.0.1`, (ii) `[::1]`, (iii) tous les noms d'hôte ne contenant pas de point (`.`).

Paramètres de proxy actuels
Fournit un journal verbeux de la détection de proxy. Il peut être réinitialisé avec la touche **Réinitialiser** située à droite du champ *Tester URL* (par exemple, en changeant l'URL de test, ou lorsque les paramètres de proxy ont été modifiés).

- *URL test :* Une URL test peut être utilisée pour voir quel proxy est utilisé pour cette URL spécifique. Aucun E/S n'est effectué avec cette URL. Ce champ ne doit pas être vide si configuration-auto-proxy est utilisé (soit par le biais de *Utiliser paramètres de proxy de système* soit *Configuration proxy automatique*).

13.10 Menu Fenêtre

Le menu **Fenêtre** contient des commandes qui vous laissent organiser des fenêtres d'application individuelle et de document au sein de la GUI. Vous pouvez mettre en cascade ou en mosaïque des fenêtres de document ouvertes et vous pouvez arranger des assistants de saisie et des fenêtres de sortie, de même que les masquer.

Cascade, Mosaïque horizontale/verticale

La commande **Cascade** arrange les fenêtres de document de telle façon qu'elles sont décalées en séquence de l'arrière vers l'avant.

Les **Mosaïque horizontale** et **Mosaïque verticale** arrangent les fenêtres de documents ouverts et non minimisés pour qu'elles soient redimensionnées en mosaïques qui sont toutes visibles dans la fenêtre d'application.

Fenêtre de projet, fenêtre d'info, assistants de saisie, fenêtres de sortie

Ces commandes activent/désactivent l'affichage respectivement des Fenêtre de projet, fenêtre d'Info, assistants de saisie et fenêtres de sortie.

Chacune de ces fenêtres est une fenêtre ancrable. Glisser sur la barre de titre de la fenêtre la détache de sa position actuelle et en fait une fenêtre flottante. Cliquer à droite de la barre de titre pour permettre l'ancrage ou dissimuler la fenêtre.

Projet et Aides à la saisie

Cette commande vous permet d'activer et de désactiver l'affichage de la Fenêtre de Projet et les Assistants de saisie ensemble. Ainsi, vous ne devez pas activer/désactiver l'affichage de ces fenêtres individuellement.

Tous activés/désactivés

Cette commande vous permet d'activer et de désactiver toutes les fenêtres (*liste ci-dessous*).

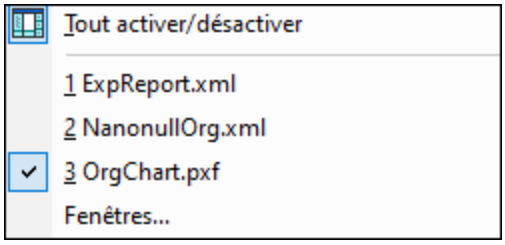
- Fenêtre de projet
- Fenêtre d'info
- Assistants de saisie
- Fenêtre Sortie

Ceci est utile si vous voulez masquer toutes les fenêtres non document rapidement, pour avoir un espace d'affichage maximum pour le(s) document(s) sur lesquels vous travaillez.

Liste de fenêtres actuellement ouvertes

Cette liste montre toutes les fenêtres ouvertes actuellement, et vous permet de basculer rapidement de l'une à l'autre.

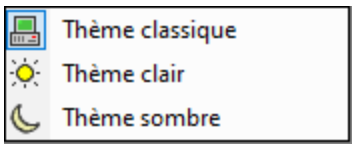
© 2016-2022 Altova GmbH



Vous pouvez aussi utiliser les raccourcis clavier **Ctrl+F6** pour passer dans les fenêtres ouvertes.

13.11 Menu Thème

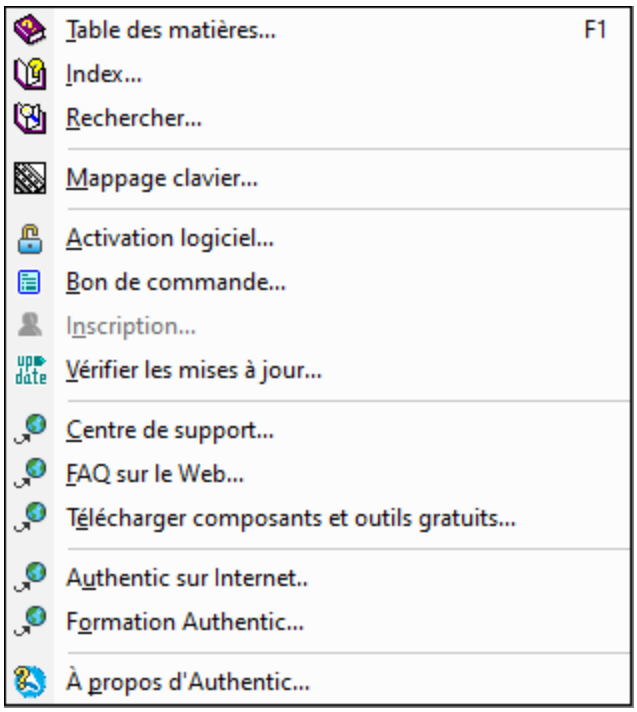
Authentic Desktop vous permet de choisir depuis les thèmes suivants : *Classique*, Clair et *Sombre*. Le thème par défaut est *Classique*.



© 2016-2022 Altova GmbH

13.12 Menu Aide

Le menu **Aide** contient les commandes nécessaires pour obtenir de l'aide ou des informations supplémentaires concernant <% appName%> ainsi que des liens vers des informations et des pages de support disponibles sur le serveur web d'Altova.



Le menu **Help** contient également le dialogue Inscription qui vous permet de saisir votre code-clé de licence obtenu au moment de l'achat du produit.

13.12.1 Sommaire, Index, Recherche

- Sommaire

  Ouvre le manuel d'aide sur écran de Authentic Desktop avec le Sommaire affiché dans le panneau de gauche de la fenêtre Aide. Le Sommaire présente un aperçu de l'ensemble du document Aide. Cliquer sur une entrée dans le Sommaire pour vous rendre dans cette section.

- Index

  Ouvre le manuel d'aide sur écran de Authentic Desktop avec l'Index de clavier affiché dans le panneau de gauche de la fenêtre Aide. L'index regroupe les mots-clés et vous permet de naviguer vers un thème en double-cliquant le mot-clé. Si un mot-clé est lié à plus d'un thème, une liste de ces thèmes s'affichera.

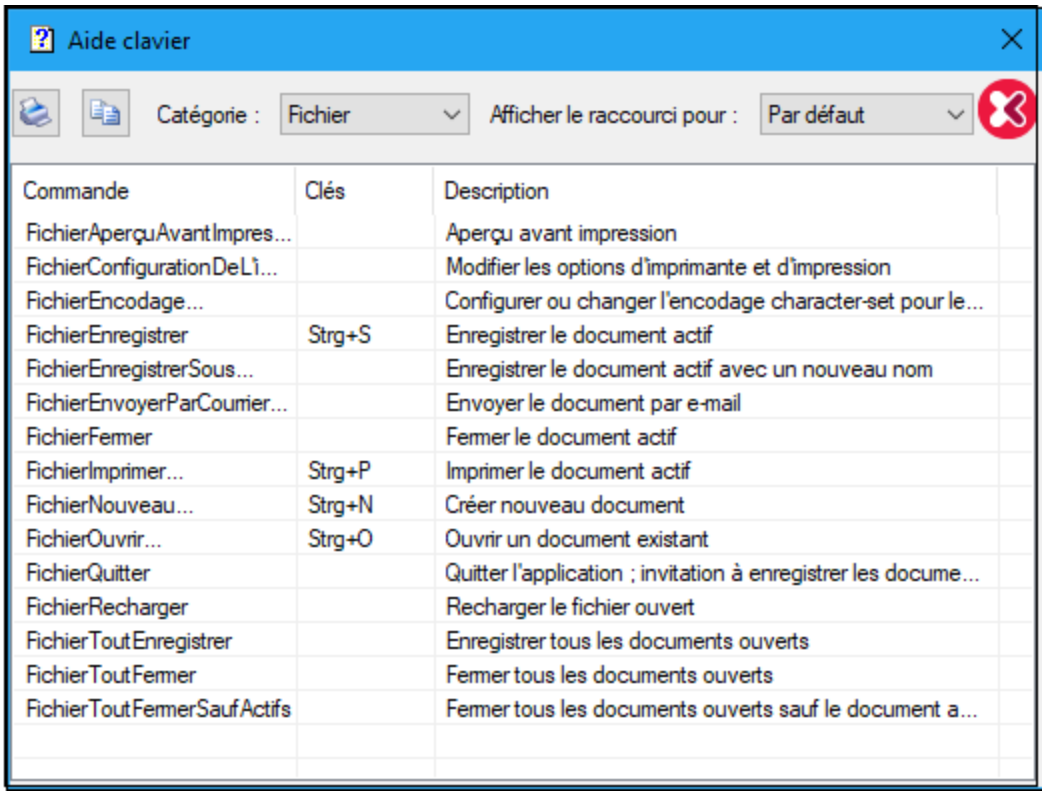
- Recherche

  Ouvre le manuel d'aide sur écran de Authentic Desktop avec le dialogue Recherche affiché dans le

panneau de gauche de la fenêtre Aide. Pour rechercher un terme, saisir le terme dans le champ de saisie et appuyer (i) sur Entrée ou (ii) sur List topics. Le système d'aide effectuera une recherche complète dans l'ensemble de la documentation d'aide et retourne une liste des rubriques trouvées. Double-cliquer sur un élément pour l'afficher.

13.12.2 Mappage clavier

La commande **Aide | Mappage clavier** fait qu'une case d'information est affichée qui contient une liste menu-par-menu de toutes les commandes dans Authentic Desktop. Les commandes de menu sont énumérées avec une description et des touches de clavier pour la commande.



Pour afficher les commandes dans un menu particulier, sélectionnez le nom de menu dans la zone de liste modifiable de la Catégorie. Vous pouvez imprimer la commande en cliquant sur l'icône d'imprimante.

13.12.3 Activation, Formulaire de commande, Inscription, Mises à jour

- Activation logiciel

  Mettre sous licence votre produit
  Après avoir téléchargé votre logiciel de produits Altova, vous pourrez acquérir une licence - ou l'activer - en

© 2016-2022 Altova GmbH

utilisant soit une clé d'évaluation gratuite ou en achetant une clé de licence permanente.

- ***Licence d'évaluation gratuite.*** Lorsque vous lancez le logiciel pour la première fois après l'avoir téléchargé et installé, le dialogue **Activation du logiciel** s'ouvrira. Vous y trouverez un bouton pour demander une licence d'évaluation gratuite. Saisir votre nom, entreprise et votre adresse e-mail dans le dialogue qui apparaît et cliquer sur **Requête.** Un fichier de licence est envoyé à l'adresse e-mail que vous avez saisi et devrait arriver dans quelques minutes. Enregistrer le fichier de licence dans un endroit approprié.

  Lorsque vous cliquez sur **Requête**, un champ de saisie apparaît en bas du dialogue Requête. Ce champ prend le chemin vers le fichier de licence. Chercher ou saisir le champ vers le fichier de licence et cliquer sur **OK**. (Dans le dialogue d'**Activation du logiciel**, vous pouvez aussi cliquer sur **Charger une Nouvelle Licence** pour accéder à un dialogue dans lequel le chemin vers le fichier de licence est saisi.) Le logiciel sera débloqué pour une période de 30 jours.

- ***Clé de licence permanente.*** Le dialogue **Activation du logiciel** contient un bouton pour acheter une clé de licence permanente. Cliquer sur ce bouton pour vous rendre à la boutique en ligne d'Altova, où vous pourrez acheter une clé de licence permanente pour votre produit. Votre licence vous sera envoyée par e-mail sous forme d'un fichier de licence contenant vos données de licence.

  Il existe trois types de licences permanentes : *installée*, *utilisateur simultané*, et *utilisateur nommé*. Une licence installée déverrouille le logiciel sur un seul ordinateur. Si vous achetez une licence installée pour*N* ordinateurs, la licence permettra une utilisation du logiciel sur jusqu'à *N* ordinateurs. Une licence utilisateur concomitant pour *N* utilisateurs concomitants permet à *N* utilisateurs d'exécuter le logiciel simultanément. (Le logiciel peut être installé sur 10N ordinateurs.) Une licence utilisateur nommé autorise un utilisateur spécifique d'utiliser le logiciel sur jusqu'à 5 ordinateurs différents. Pour activer votre logiciel, cliquer sur **Charger une Nouvelle licence**, et, dans le dialogue qui apparaît, chercher ou saisir le chemin vers le fichier de licence et cliquer sur **OK**.

Note : en ce qui concerne les licences utilisateurs multiples, chaque utilisateur sera invité à saisir son nom dans le champ Nom.

Votre e-mail de licence et les différents moyens de mise sous licence de votre produit Altova (activation) à votre disposition
L'e-mail de licence que vous avez reçu de la part d'Altova contient votre fichier de licence en pièce jointe. Le fichier package a une extension de fichier `.altova_licenses`.

Pour activer votre produit Altova, vous pouvez choisir une des étapes suivantes :

- Enregistrez le fchier de licence (`.altova_licenses`) vers un emplacement approprié, double-cliquez sur le fichier de licence, saisissez tout détail requis dans le dialogue qui apparaît, et terminez en cliquant sur **Appliquer clés**.
- Enregistrez le fichier de licence (`.altova_licenses`) vers un emplacement approprié. Dans votre produit Altova, sélectionnez la commande de menu **Aide | Activation Logiciel**, puis **Télécharger une nouvelle licence**. Recherchez ou saisissez le chemin du fichier de licence, et cliquez sur **OK**.
- Enregistrez le fichier de licence (`.altova_licenses`) vers un emplacement approprié et chargez-le depuis cet emplacement vers le pool de licences de votre Altova

LicenseServer Vous pouvez ensuite : (i) acquérir la licence depuis votre produit Altova via le dialogue d'Activation du logiciel du produit *(voir ci-dessous)* ou (ii) attribuer la licence au produit depuis l'Altova LicenseServer. *Pour plus d'informations concernant la mise sous licence via LicenseServer, lire le reste de cette rubrique.*

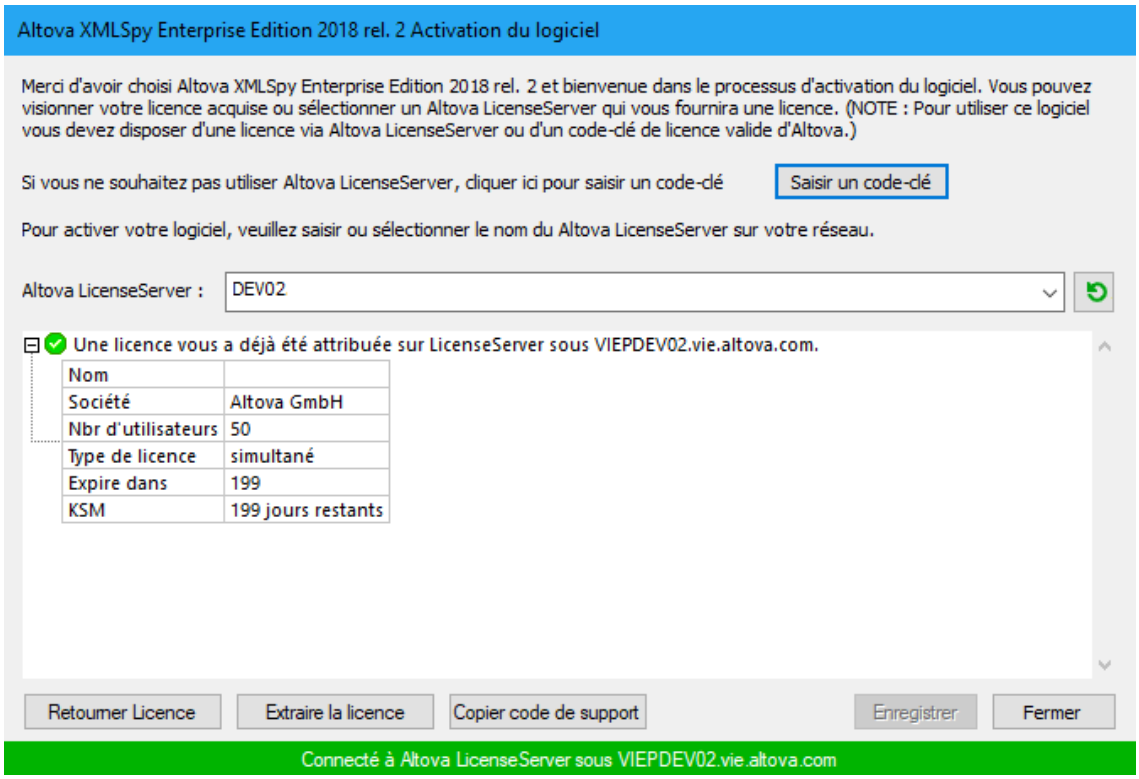
Le dialogue **Activation du logiciel** (*capture d'écran ci-dessous*) peut être accédé à tout moment en cliquant sur la commande **Aide | Activation du logiciel**.

Activer votre logiciel
Vous pouvez activer le logiciel en inscrivant le dialogue Activation du logiciel ou en enregistrant la licence par le biais du Serveur de licence Altova (*voir les détails ci-dessous*).

- *Enregistrant la licence dans le dialogue d'Activation du logiciel*. Dans le dialogue, cliquez sur **Charger une Nouvelle Licence**, puis cherchez et sélectionnez le fichier de licence. Cliquez sur **OK** pour confirmer le chemin vers le fichier de licence et confirmez toutes les données que vous avez saisies (votre nom dans le cas de licences multi-utilisateur). Terminez en cliquant **Enregistrer**.

- *Mise sous licence par le biais du Serveur de licence Altova sur votre réseau :* Pour acquérir une licence par le biais d'un Altova LicenseServer sur votre réseau, cliquez sur **Utiliser Altova LicenseServer**, situé en bas du dialogue **Activation du logiciel**. Choisissez l'appareil sur lequel le LicenseServer que vous souhaitez utiliser a été installé. Veuillez noter que l'auto-découverte des License Servers fonctionne par le biais d'une diffusion envoyée sur le LAN. Puisque les diffusions sont limitées à un sous-réseau, License Server doit se trouver sur le même sous-réseau que l'appareil client pour la découverte automatique afin de fonctionner. Si elle ne fonctionne pas, saisir le nom du serveur. L'Altova LicenseServer doit disposer d'une licence pour votre produit Altova dans son pool de licence. Si une licence est disponible dans le pool de LicenseServer, cela sera indiqué dans le dialogue d'**Activation du logiciel** (*la capture d'écran ci-dessous affiche le dialogue dans Altova XMLSpy*). Cliquez sur **Enregistrer** pour acquérir la licence.

© 2016-2022 Altova GmbH



Une fois qu'une licence spécifique aux appareils (aka installée) a été acquise depuis LicenseServer, elle ne peut pas être retournée au LicenseServer pour une période de sept jours. Après cette période, vous pouvez rendre la licence installée (cliquer sur **Retourner licence**) de manière à ce que la licence puisse être acquise depuis LicenseServer par un autre client. (Néanmoins, un administrateur de LicenseServer, peut désattribuer à tout moment une licence acquise par le biais de la Web UI du LicenseServer). Veuillez noter qu'un renvoi de la licence n'est applicable qu'aux seules licences sur appareil installées, pas aux licences concurrentes.

Extraire la licence

Vous pouvez consulter une licence du pool de licence pour une période de jusqu'à 30 jours pour que la licence puisse être stockée sur l'appareil de produit. Cela vous permet de travailler hors ligne, ce qui peut être utile, par exemple, si vous souhaitez travailler dans un environnement où vous ne pourrez pas accéder à votre Altova LicenseServer (par exemple, si votre produit Altova est installé sur un ordinateur portable et que vous vous trouvez en déplacement). Tant que la licence est extraite, LicenseServer affiche la licence comme étant utilisée ; elle ne peut donc pas être utilisée par une autre machine. La licence passe automatiquement à l'état d'archivage lorsque la période d'extraction expire. En alternative, une licence extraite peut être archivée à tout moment par le biais du bouton **Archiver** du dialogue d'**Activation du logiciel**.

Pour extraire une licence, procédez comme suit : (i) dans le dialogue d'A**ctivation du logiciel**, cliquez sur **Extraire licence** (*voir la capture d'écran ci-dessus*); (ii) dans le dialogue d'**extraction de la licence** qui apparaît, sélectionnez la période d'extraction que vous souhaitez et cliquez sur **Extraire**. La licence sera extraite. Après avoir extrait la licence, deux choses se produisent : (i) Le dialogue d'**Activation du logiciel** affichera les informations d'extraction, y compris l'heure à laquelle l'extraction expirera, (ii) le bouton **Extraire licence** dans le dialogue se transforme en un bouton **Archiver**. Vous pouvez archiver la licence à nouveau à tout moment en cliquant sur

Archiver. Étant donné que la licence passe automatiquement au statut Archiver à l'issue de la période d'extraction, assurez-vous que la période d'extraction que vous avez choisie couvre bien la période pendant laquelle vous travaillerez hors ligne.

Les enregistrements de licence doivent correspondre à la même version majeure du produit duquel la licence a été extraite. Donc veillez à enregistrer une licence avant de mettre à niveau votre produit Altova vers la prochaine version majeure.

Note : afin de pouvoir effectuer des extractions de licence, la fonction d'extraction doit être activée sur le LicenseServer. Si la fonction n'a pas été activée, vous recevrez un message d'erreur à cet effet lorsque vous essayez de faire le « check out ». Dans ce cas, veuillez contacter votre administrateur de LicenseServer.

Copier code de support
Cliquer sur **Copier code de support** pour copier des détails de licence dans le presse-papiers. Il s'agit des données que vous devrez fournir en cas de demande d'assistance avec le formulaire d'assistance en ligne.

Altova LicenseServer offre aux administrateurs IT un aperçu en temps réel de toutes les licences Altova sur un réseau, avec les détails de chaque licence, ainsi que les attributions clients et l'utilisation client des licences. L'avantage d'utiliser LicenseServer réside donc dans les fonctions administratives qu'il offre pour la gestion de licence à large volume d'Altova. Altova LicenseServer est disponible gratuitement depuis le site web Altova. Pour plus d'informations concernant Altova LicenseServer et la mise sous licence par le biais d'Altova LicenseServer, voir la documentation Altova LicenseServer.

- Formulaire de commande

  Lorsque vous êtes prêt pour commander une version de licence du produit de logiciel, vous pouvez soit utiliser la touche **Acheter une clé de licence permanente** dans le dialogue **Activation du logiciel** (*voir la section précédente*) ou la commande **Formulaire de commande** pour continuer vers la boutique en ligne Altova sécurisée.

- Inscription

  Ouvre la page d'enregistrement du produit Altova dans un onglet de votre navigateur. L'enregistrement de votre logiciel Altova vous aidera à vous assurer de toujours rester à jour avec les dernières informations du produit.

- Vérifier les mises à jour

  Contrôle sur le serveur Altova si une version plus récente que la vôtre est actuellement disponible et, dans l'affirmative, affiche un message approprié.

13.12.4 Autres commandes

- Centre de support

  Un lien qui vous mènera vers le Centre de support Altova sur Internet. Le Centre de support contient des FAQ, des forums de discussion pour toute sorte de problèmes et l'accès à l'équipe de support technique

© 2016-2022 Altova GmbH

d'Altova.

- FAQ sur le web

  Un lien menant à la base de données FAQ d'Altova sur Internet. La base de données FAQ est constamment mise à jour à la suite des problèmes rapportés par les clients.

- Télécharger les composants et les outils gratuits

  Un lien menant au Centre de téléchargement des composants Altova sur Internet. À partir de là, vous pouvez télécharger une variété de logiciels complémentaires à utiliser avec des produits Altova. Ces logiciels vont de processeurs XSLT et XSL-FO à des Plateformes de serveur d'application. Les logiciels disponibles dans le Centre de téléchargement des composants sont généralement gratuits.

- Authentic Desktop sur Internet

  Un lien menant au site web Altova sur Internet. Vous pouvez en apprendre plus sur Authentic Desktop, les technologies et produits liés le le site web Altova.

- À propos de Authentic Desktop

  Affiche la fenêtre d'accueil et le numéro de version de votre produit. Si vous utilisez la version 64-bit de Authentic Desktop, cela est indiqué par le suffixe (x64) placé après le nom de l'application. Il n'y a pas de suffixe pour la version 32-bit.

13.13 Ligne de commande

Certaines actions Authentic Desktop peuvent être réalisées depuis la ligne de commande. Ces commandes sont listées ci-dessous :

Ouvrir un fichier

authentic.exe file.xml

Ouvre le fichier file.xml dans Authentic Desktop

Ouvrir de multiples fichiers

authentic.exe file1.xml file2.xml

Ouvre les fichiers file1.xml et file2.xml dans Authentic Desktop

Attribue un fichier SPS vers un fichier XML pour l'édition du mode Authentic

authentic.exe myxml.xml /sps mysps.sps

Ouvre le fichier myxml.xml en mode Authentic avec mysps.sps comme son fichier SPS. L'indicateur /sps spécifie que le fichier SPS qui suit doit être utilisé avec un fichier XML qui précède l'indicateur /sps (pour l'édition du mode Authentic).

Ouvrir un nouveau modèle de fichier XML par le biais d'un fichier SPS

authentic.exe mysps.sps

Ouvrir le fichier XML dans Authentic View. L'affichage sera basé sur le SPS et le nouveau fichier XML aura une structure squelletique basée sur le schéma SPS. Le nom du nouveau fichier XML doit être assigné lorsque vous enregistrez le fichier XML.

© 2016-2022 Altova GmbH

14 Programmers' Reference

Authentic Desktop is an Automation Server. It exposes programmable objects to other applications called Automation Clients. As a result, an Automation Client can directly access the objects and functionality that the Automation Server makes available. An Automation Client of Authentic Desktop, can use the XML validation functionality of Authentic Desktop. Developers can thus enhance their applications with the ready-made functionality of Authentic Desktop.

The programmable objects of Authentic Desktop are made available to Automation Clients via the Application API of Authentic Desktop, which is a COM API. The object model of the API and a complete description of all available objects are provided in this documentation (see the section Application API).

The API can be accessed from within the following environments:

- Scripting Editor
- IDE Plug-ins
- External programs
- ActiveX Integration

Each of these environments is described briefly below.

Scripting Editor: Customizing and modifying Authentic Desktop functionality

You can customize your installation of Authentic Desktop by modifying and adding functionality to it. You can also create Forms for user input and modify the user interface so that it contains new menu commands and toolbar shortcuts. All these features are achieved by writing scripts that interact with objects of the Application API. To aid you in carrying out these tasks efficiently, Authentic Desktop offers you an in-built Scripting Editor. A complete description of the functionality available in the Scripting Editor and how it is to be used is given in the Scripting Editor section of this documentation. The supported programming languages are **JScript** and **VBScript**.

IDE Plug-ins: Creating plug-ins for Authentic Desktop

Authentic Desktop enables you to create your own plug-ins and integrate them into Authentic Desktop. You can do this using Authentic Desktop's special interface for plug-ins. A description of how to create plug-ins is given in the section Authentic Desktop IDE Plug-ins.

An application object gets passed to most methods that must be implemented by an IDE plug-in and gets called by the application. Typical languages used to implement an IDE plug-in are **C#** and **C++**. For more information, see the section Authentic Desktop IDE Plugins.

External programs

Additionally, you can manipulate Authentic Desktop with external scripts. For example, you could write a script to open Authentic Desktop at a given time, then open an XML file in Authentic Desktop, validate the file, and print it out. External scripts would again make use of the Application API to carry out these tasks. For a description of the Application API, see the section Application API.

Using the Application API from outside Authentic Desktop requires an instance of Authentic Desktop to be started first. How this is done depends on the programming language used. See the section, Programming Languages, for information about individual languages.

Essentially, Authentic Desktop will be started via its COM registration. Then the `Application` object associated with the Authentic Desktop instance is returned. Depending on the COM settings, an object associated with an already running Authentic Desktop can be returned. Any programming language that supports creation and invocation of COM objects can be used. The most common of these are listed below.

- JScript and VBScript script files have a simple syntax and are designed to access COM objects. They can be run directly from a DOS command line or with a double click on Windows Explorer. They are best used for simple automation tasks.
- C# is a a full-fledged programming language that has a wide range of existing functionality. Access to COM objects can be automatically wrapped using C#..
- C++ provides direct control over COM access but requires relatively larger amounts of code than the other languages.
- Java: Altova products come with native Java classes that wrap the Application API and provide a full Java look-and-feel.
- Other programming languages that make useful alternatives are: Visual Basic for Applications, Perl, and Python.

ActiveX Integration

A special case of accessing the Application API is via the Authentic Desktop ActiveX control. This feature is only available if the Authentic Desktop integration package is installed. Every ActiveX Control has a property that returns a corresponding COM object for its underlying functionality. The manager control provides an `Application` object, the document control a `Document` object, and the placeholder object, in cases where it contains the project tree, returns the `Project` object. The methods supported by these objects are exactly as described in the Interfaces section of the Application API. Care must be taken not to use methods that do not make sense in the context of ActiveX control integration. For details see ActiveX Integration.

About Programmers' Reference

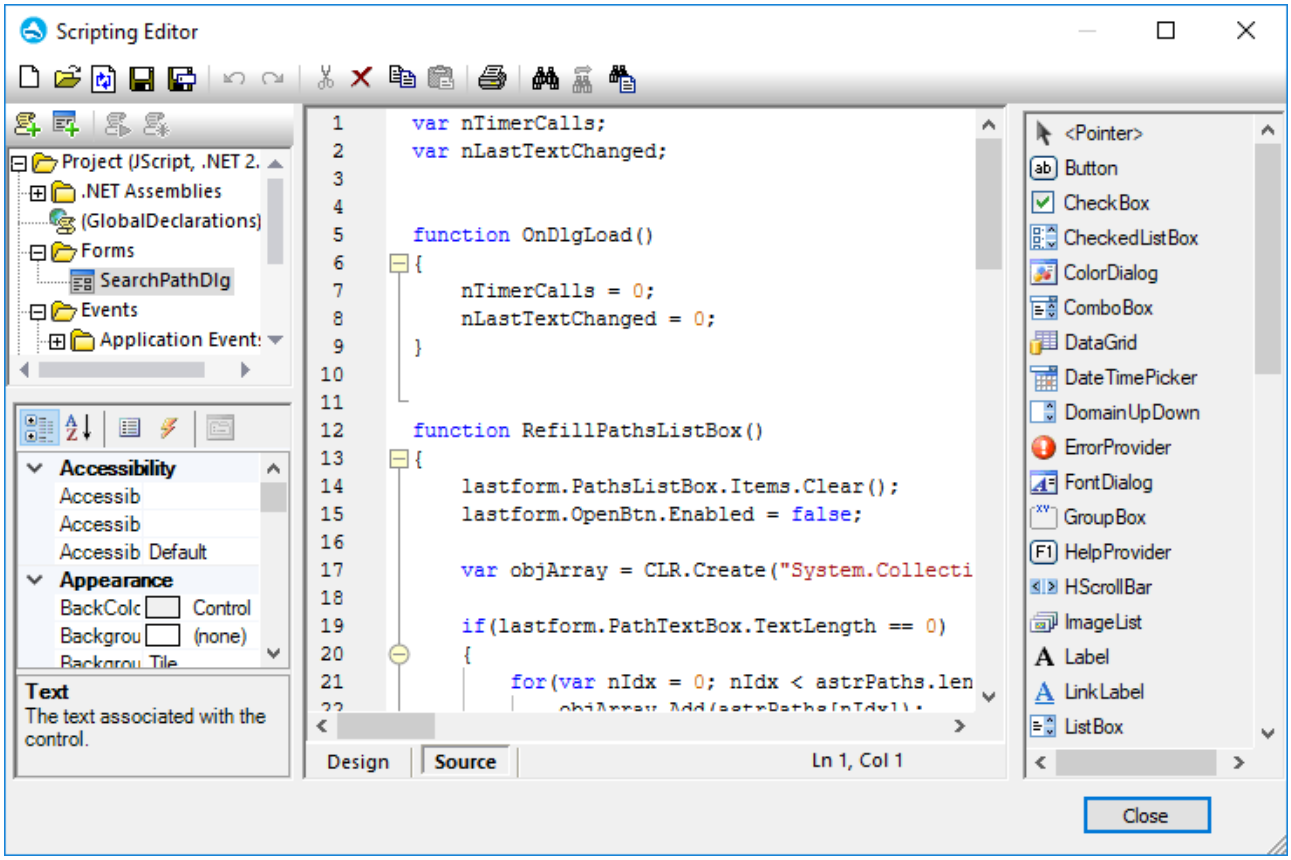
The documentation contained in the Programmers' Reference for Authentic Desktop consists of the following sections:

- Scripting Editor: a user reference for the Scripting Environment available in Authentic Desktop
- IDE Plug-ins: a description of how to create plug-ins for Authentic Desktop
- Application API: a reference for the Application API
- ActiveX Integration: a guide and reference for how to integrate the Authentic Desktop GUI and Authentic Desktop functionality using an ActiveX control

© 2016-2022 Altova GmbH

14.1 Scripting Editor

Scripting Editor is a development environment built into Authentic Desktop from where you can customize the functionality of Authentic Desktop with the help of JScript or VBScript scripts. For example, you can add a new menu item to perform a custom project task, or you can have Authentic Desktop trigger some behavior each time when a document is opened or closed. To make this possible, you create scripting projects—files with .asprj extension (Altova Scripting Project).



Scripting Editor

Scripting projects typically include one or several macros—these are programs that perform miscellaneous custom tasks when invoked. You can run macros either explicitly from a menu item (or a toolbar button, if configured), or you can set up a macro to run automatically whenever Authentic Desktop starts. The scripting environment also integrates with the Authentic Desktop COM API. For example, your VBScript or JScript scripts can handle application or document events such as starting or shutting down Authentic Desktop, opening or closing a project, and so on. Scripting projects can include Windows Forms that you can design visually, in a way similar to Visual Studio. In addition, several built-in commands are available that help you instantiate and use .NET classes from VBScript or JScript code.

Once your scripting project is complete, you can enable it either globally in Authentic Desktop, or only for specific projects.

Scripting Editor requires .NET Framework 2.0 or later to be installed before Authentic Desktop is installed.

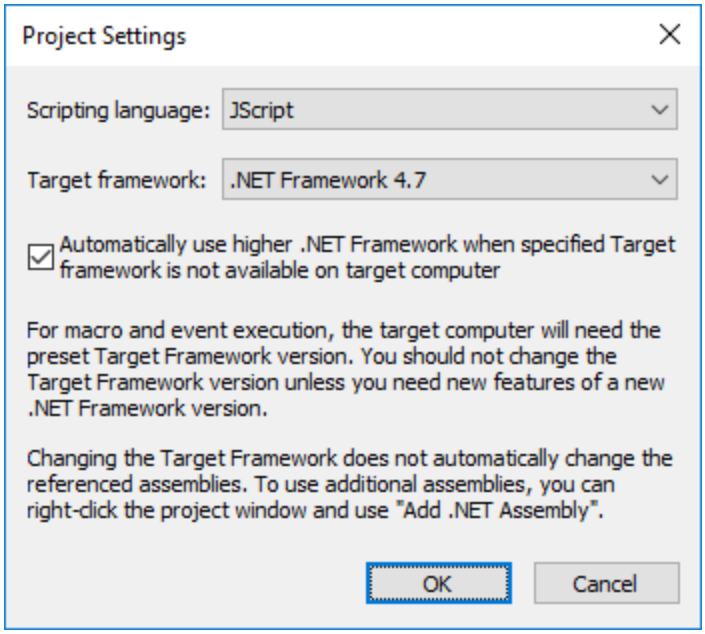
14.1.1 Creating a Scripting Project

All scripts and scripting information created in the Scripting Editor are stored in Altova Scripting styleclass="Code Bold" (.asprj files). A scripting project may contain macros, application event handlers, and forms (which can have their own event handlers). In addition, you can add global variables and functions to a "Global Declarations" script—this makes such variables and functions accessible across the entire project.

To start a new project, run the menu command **Tools | Scripting Editor**.

The languages supported for use in a scripting project are JScript and VBScript (not to be confused with Visual Basic, which is not supported). These scripting engines are available by default on Windows and have no special requirements to run. You can select a scripting language as follows:

1. Right-click the **Project** item in the upper-left pane, and select **Project settings** from the context menu.
2. Select a language (JScript or VBScript), and click **OK**.



From the Project settings dialog box above, you can also change the target .NET Framework version. This is typically necessary if your scripting project requires features available in a newer .NET Framework version. Note that any clients using your scripting project will need to have the same .NET Framework version installed (or a later compatible version).

By default, a scripting project references several .NET assemblies, like `System`, `System.Data`, `System.Windows.Forms`, and others. If necessary, you can import additional .NET assemblies, including assemblies from .NET Global Assembly Cache (GAC) or custom .dll files. You can import assemblies as follows:

1. Statically, by adding them manually to the project. Right-click **Project** in the top-left pane, and select **Add .NET Assembly** from the context menu.
2. Dynamically, at runtime, by calling the `CLR.LoadAssembly` command from the code.

© 2016-2022 Altova GmbH

You can create multiple scripting projects if necessary. You can save a scripting project to the disk, and then load it back into the Scripting Editor later. To do this, use the standard Windows buttons available in the toolbar: **New**, **Open**, **Save**, **Save As**. Once the scripting project has been tested and is ready for deployment, you can load it into Authentic Desktop and run any of its macros or event handlers. For more information, see Enabling Scripts and Macros.

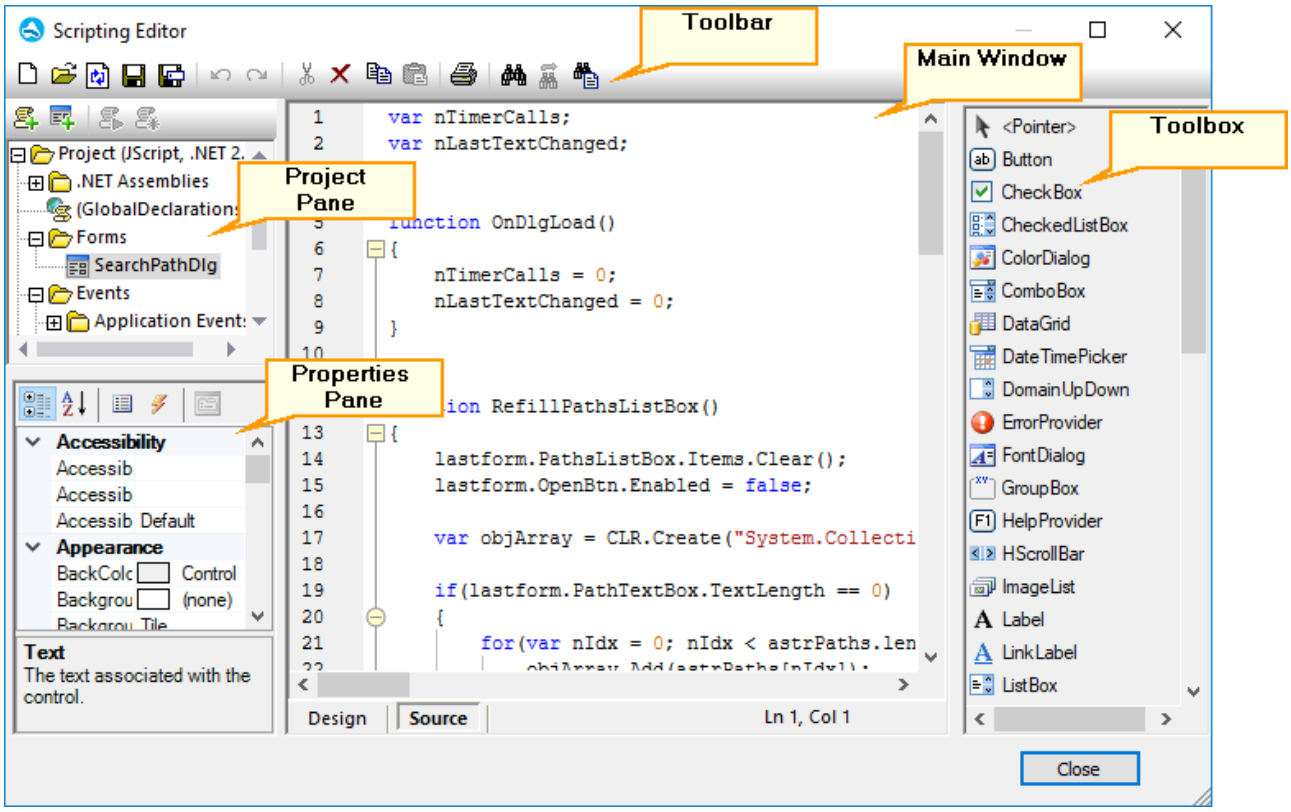
You can also find an example scripting project at the following path: **C:\Users\<user>\Documents\Altova\Authentic2023\AuthenticExamples\SampleScripts.asprj**.

The next sections focus on the parts that your scripting project may need: global declarations, macros, forms, and events.

14.1.1.1 Overview of the Environment

The Scripting Editor consists of the following parts:

- Toolbar
- Project pane
- Properties pane
- Main window
- Toolbox



Toolbar

The toolbar includes standard Windows file management commands (**New**, **Open**, **Save**, **Save As**) commands and editor commands (**Copy**, **Cut**, **Delete**, **Paste**). When editing source code, the **Find** and **Replace** commands are additionally available, as well as the **Print** command.

Project pane

The project pane helps you view and manage the structure of the project. A scripting project consists of several components that can work together and may be created in any order:

- A *"Global Declarations"* script. As the name suggests, this script stores information available globally across the project. You can declare in this script any variables or functions that you need to be available in all forms, event handler scripts, and macros.
- *Forms.* Forms are typically necessary to collect user input, or provide some informative dialog boxes. A form is invoked by a call to it either within a function (in the Global Declarations script) or directly in a macro.
- *Events.* The "Events" folder displays Authentic Desktop application events provided by the COM API. To write a script that will be executed when an event occurs, double-click any event, and then type the handling code in the editor. The application events should not be confused with form events; the latter are handled at form level, as further detailed below.
- *Macros*. A macro is a script that can be invoked either on demand from a context menu or be executed automatically when Authentic Desktop starts. Macros do not have parameters or return values. A macro can access all variables and functions declared in the Global Declarations script and it can also display forms.

Right-click any of the components to see the available context menu commands and their shortcuts. Double-click any file (such as a form or a script) to open it in the main window.

The toolbar buttons provide the following quick commands:

	**New macro**	Adds a new macro to the project, in the **Macros** directory.
	**New form**	Adds a new form to the project, in the **Forms** directory.
	**Run macro**	Runs the selected macro.
	**Debug macro**	Runs the selected macro in debug mode.

Properties pane

The Properties pane is very similar to the one in Visual Studio. It displays the following:

- Form properties, when a form is selected
- Object properties, when an object in a form is selected
- Form events, when a form is selected
- Object events, when an object in a form is selected

To switch between the properties and events of the selected component, click the **Properties**  or **Events**  buttons, respectively.

© 2016-2022 Altova GmbH

The **Categorized** and **Alphabetical** icons display the properties or events either organized by category or organized in ascending alphabetical order.

When a property or event is selected, a short description of it is displayed at the bottom of the Properties pane.

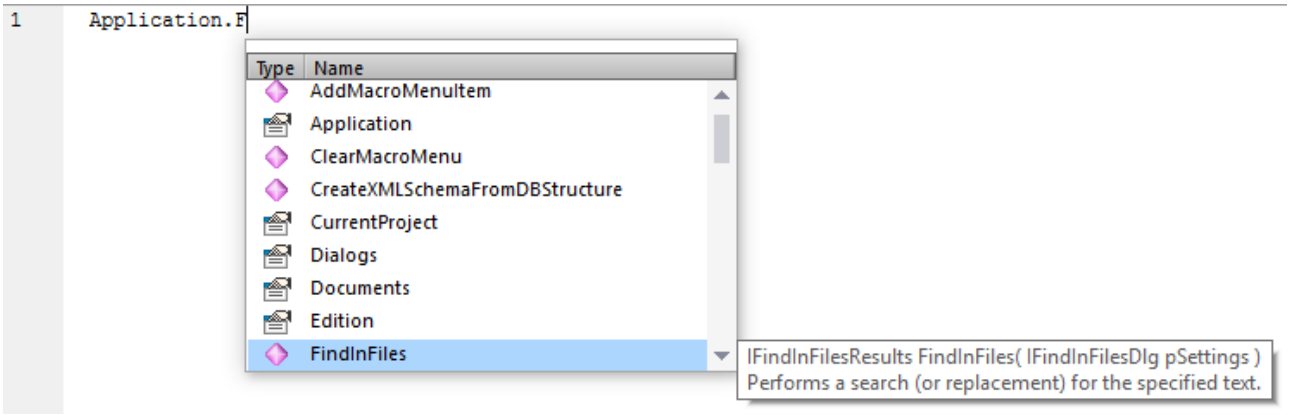
Main window

The main window is the working area where you can enter source code or modify the design of the form. When editing forms, you can work in two tabs: the **Design** tab and the **Source** tab. The **Design** tab shows the layout of the form, while the **Source** tab contains the source code such as handler methods for the form events.

The source code editor provides code editing aids such as syntax coloring, source code folding, highlighting of starting and ending braces, zooming, autocompletion suggestions, bookmarks.

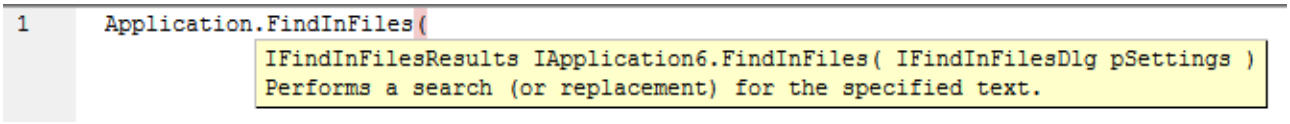
Autocompletion suggestions

JScript and VBScript are untyped languages, so autocompletion is limited to COM API names and Authentic Desktop built-in commands. The full method or property signature is shown next to the autocompletion entry helper.



If names start with `objDocument`, `objProject`, `objXMLData`, or `objAuthenticRange`, members of the corresponding interface will be shown.

Placing the mouse over a known method or property displays its signature (and documentation if available), for example:



The auto-completion entry helper is normally shown automatically during editing, but it can also be obtained on demand by pressing **Ctrl+Space**.

Bookmarks

- To set or remove a bookmark, click inside a line, and then press **Ctrl+F2**
- To navigate to the next bookmark, press **F2**
- To navigate to the previous bookmark, press **Shift+F2**
- To delete all bookmarks, press **Ctrl+Shift+F2**

Zooming in/out

- To zoom in or out, hold the **Ctrl** key pressed and then press the "+" or "-" keys or rotate the mouse wheel.

Text view settings

To trigger text settings, right-click inside the editor, and select **Text View Settings** from the context menu.

Font settings

To change the font, right-click inside the editor, and select **Text View Font** from the context menu.

Toolbox

The Toolbox contains all the objects that are available for designing forms, such as buttons, text boxes, combo boxes, and so on.

To add a Toolbox item to a form:

1. Create or open a form and make sure that the **Design** tab is selected.
2. Click the Toolbox object (for example, **Button**), and then click at the location in the form where you wish to insert it. Alternatively, drag the object directly onto the form.

Some objects such as `Timer` are not added to the Form but are created in a tray at the bottom of the main window. You can select the object in the tray and set properties and event handlers for the object from the Properties pane. For an example of handling tray components from the code, see Handling form events.

You can also add registered ActiveX controls to the form. To do this, right-click the Toolbox area and select **Add ActiveX Control** from the context menu.

14.1.1.2 Global Declarations

The "Global Declarations" script is present by default in any scripting project; you do not need to create it explicitly. Any variables or functions that you add to this script are considered global across the entire project. Consequently, you can refer to such variables and functions from any of the project's macros and events. The following is an example of a global declarations script that imports the `System.Windows.Forms` namespace into the project. To achieve that, the code below invokes the `CLR.Import` command built into Scripting Editor.

```
// import System.Windows.Forms namespace for all macros, forms and events:
CLR.Import( "System.Windows.Forms" );
```

Note: Every time a macro is executed or an event handler is called, the global declarations are re-initialized.

14.1.1.3 Macros

Macros are scripts that contain JScript (or VBScript, depending on your project's language) statements, such as variable declarations and functions.

© 2016-2022 Altova GmbH

If your projects should use macros, you can add them as follows: right-click inside the Project pane, select **Add Macro** from the context menu, and then enter the macro's code in the main form. The code of a macro could be as simple as an alert, for example:

```
alert("Hello, I'm a macro!");
```

More advanced macros can contain variables and local functions. Macros can also contain code that invokes forms from the project. The listing below illustrates an example of a macro that shows a form. It is assumed that this form has already been created in the "Forms" folder and has the name "SampleForm", see also Forms.

```
// display a form
ShowForm( "SampleForm" );
```

In the code listing above, `ShowForm` is a command built into Scripting Editor. For reference to other similar commands that you can use to work with forms and .NET objects, the Built-in Commands.

You can add multiple macros to the same project, and you can designate any macro as "auto-macro". When a macro is designated as "auto-macro", it runs automatically when Authentic Desktop starts. To designate a macro as auto-macro, right-click it, and select **Set as Auto-Macro** from the context menu.

Only one macro can be run at a time. After a macro (or event) is executed, the script is closed and global variables lose their values.

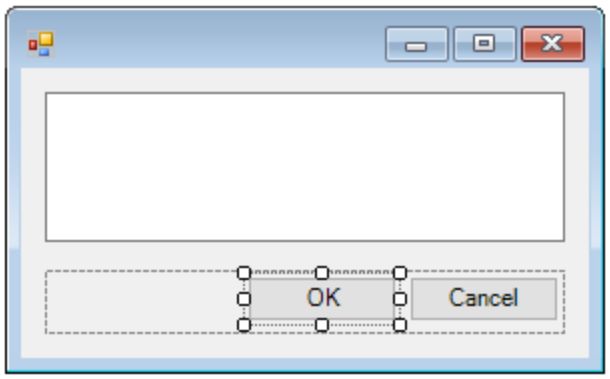
To run a macro directly in Script Editor, click **Run Macro**  . To debug a macro using the Visual Studio debugger, click **Debug Macro**  . For information about enabling and running macros in Authentic Desktop, see Enabling Scripts and Macros.

14.1.1.4 Forms

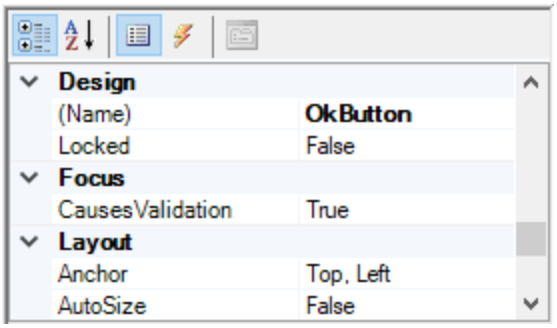
Forms are particularly useful if you need to collect input data from users or display data to users. A form can contain miscellaneous controls to facilitate this, such as buttons, check boxes, combo boxes, and so on.

To add a form, right-click inside the Project pane, and then select **Add Form** from the context menu. To add a control to a form, drag it from the Toolbox available to the right side of Scripting Editor and drop it onto the form.

You can change the position and size of the controls directly on the form, by using the handles that appear when you click any control, for example:



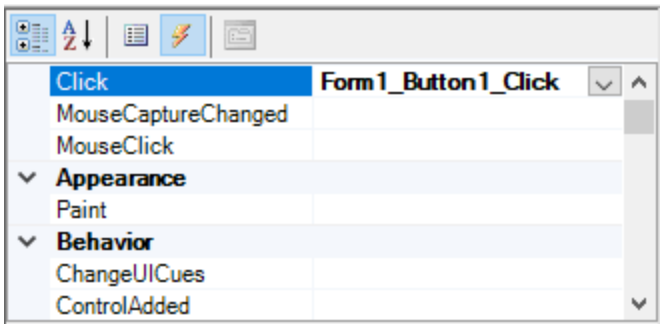
All form controls have properties that you can easily adjust in the Properties pane. To do this, first select the control on the form, and then edit the required properties in the Properties pane.



Handling form events

Each form control also exposes various events to which your scripting project can bind. For example, you might want to invoke some Authentic Desktop COM API method whenever a button is clicked. To create a function that binds to a form event, do the following:

1. In the Properties pane, click **Events** .
2. In the **Action** column, double-click the event where you need the method (for example, in the image below, the handled event is "Click").



© 2016-2022 Altova GmbH

You can also add handler methods by double-clicking a control on the form. For example, double-clicking a button in the form design generates a handler method for the "Click" event of that button.

Once the body of the handler method is generated, you can type code that handles this event, for example:

```
//Occurs when the component is clicked.
function MyForm_ButtonClick( objSender, e_EventArgs )
{
   alert("A button was clicked");
}
```

To display a work-in-progress form detached from the Scripting Editor, right-click the form, and select **Test Form** from the context menu. Note that the **Test Form** command just displays the form; the form's events (such as button clicks) are still disabled. To have the form react to events, call it from a macro, for example:

```
// Instantiate and display a form
ShowForm( "SampleForm" );
```

Accessing form controls

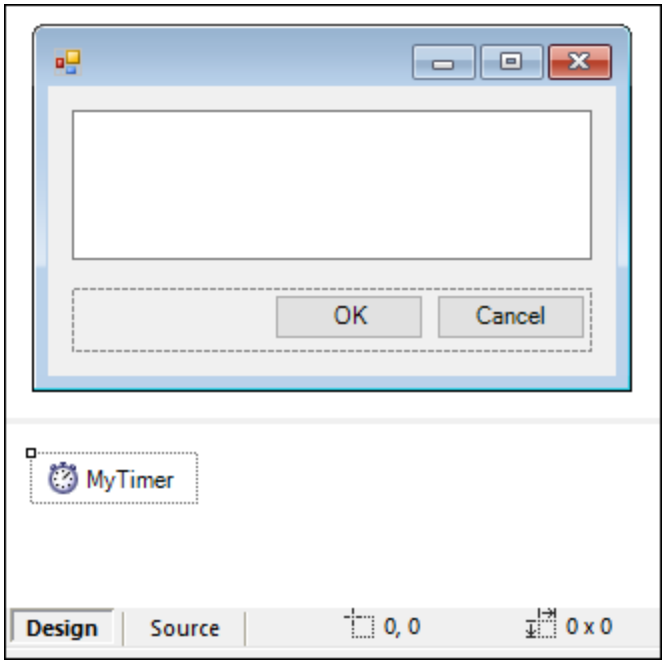
You can access any components on a form from your code by using field access syntax. For example, suppose there is a form designed as follows:

```
// MyForm
//   ButtonPanel
//      OkButton
//      CancelButton
//   TextEditor
//      AxMediaPlayer1
// TrayComponents
//   MyTimer
```

The code below shows how to instantiate the form, access some of its controls using field access syntax, and then display the form:

```
// Instantiate the form
var objForm = CreateForm("MyForm");
// Disable the OK button
objForm.ButtonPanel.OkButton.Enabled = false;
// Change the text of TextEditor
objForm.TextEditor.Text = "Hello";
// Show the form
objForm.ShowDialog();
```

When you add certain controls such as timers to the form, they are not displayed on the form; instead, they are shown as tray components at the base of the form design, for example:



To access controls from the tray, use the `GetTrayComponent` method on the form object, and supply the name of the control as argument. In this example, to get a reference to `MyTimer` and enable it, use the following code:

```
var objTimer = objForm.GetTrayComponent("MyTimer");
objTimer.Enabled = true;
```

For ActiveX Controls, you can access the underlying COM object via the OCX property:

```
var ocx = lastform.AxMediaPlayer1.OCX; // get underlying COM object
ocx.enableContextMenu = true;
ocx.URL = "mms://apasf.apa.at/fm4_live_worldwide";
```

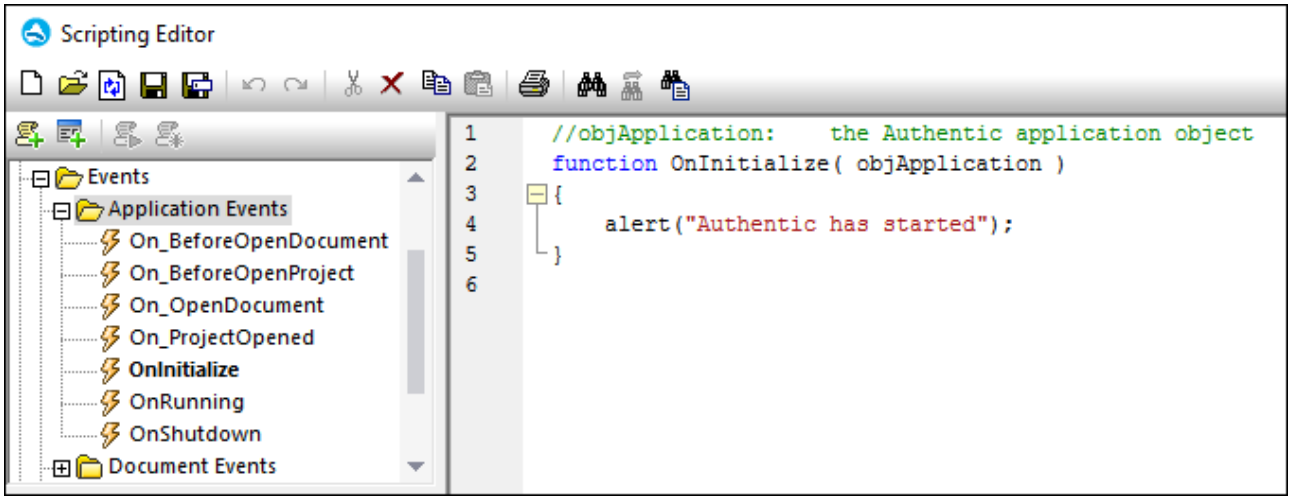
14.1.1.5 Events

Your scripting project may optionally include scripts that handle Authentic Desktop events such as opening, closing, or saving a document, starting or closing Authentic Desktop, adding an element to a diagram, and others. These events are provided by the Authentic Desktop COM API, and you can find them in the "Events" folder of your scripting project. Note that these events are Authentic Desktop-specific, as opposed to form events. Events are organized into folders as follows:

- Application Events
- Document Events
- AuthenticView Events
- GridView Events
- TextView Evets

© 2016-2022 Altova GmbH

To create an event handler script, right-click an event, and select **Open** from the context menu (or double-click the event). The event handler script is displayed in the main window, where you can start editing it. For example, the event handler illustrated below displays an alert each time Authentic Desktop starts:



Note the following:

- The `alert` command is applicable to JScript. The VBScript equivalent is `MsgBox`. See also alert.
- The name of the event handler function must not be changed; otherwise, the event handler script will not be called.
- In order for events to be processed, the **Process Events** check box must be selected when you enable the scripting project in Authentic Desktop. For more information, see Enabling Scripts and Macros.

You can optionally define local variables and helper functions within event handler scripts, for example:

```
var local;

function OnInitialize( objApplication )
{
   local = "OnInitialize";
   Helper();
}

function Helper()
{
   alert("I'm a helper function for " + local);
}
```

14.1.1.6 JScript Programming Tips

Below are a few JScript programming tips that you may find useful while developing a scripting project in Authentic Desktop Scripting Editor.

Out parameters

Out parameters from methods of the.NET Framework require special variables in JScript. For example:

```
var dictionary =
CLR.Create("System.Collections.Generic.Dictionary<System.String,System.String>");
dictionary.Add("1", "A");
dictionary.Add("2", "B");

// use JScript method to access out-parameters
var strOut = new Array(1);
if ( dictionary.TryGetValue("1", strOut) ) // TryGetValue will set the out parameter
  alert( strOut[0] ); // use out parameter
```

Integer arguments

.NET Methods that require integer arguments should not be called directly with JScript number objects which are floating point values. For example, instead of:

```
var objCustomColor = CLR.Static("System.Drawing.Color").FromArgb(128,128,128);
```

use:

```
var objCustomColor =
CLR.Static("System.Drawing.Color").FromArgb(Math.floor(128),Math.floor(128),Math.floor(12
8));
```

Iterating .NET collections

To iterate .NET collections, the JScript Enumerator as well as the .NET iterator technologies can be used, for example:

```
// iterate using the JScript iterator
var itr = new Enumerator( coll );
for ( ; !itr.atEnd(); itr.moveNext() )
  alert( itr.item() );

// iterate using the .NET iterator
var itrNET = coll.GetEnumerator();
while( itrNET.MoveNext() )
  alert( itrNET.Current );
```

.NET templates

.NET templates can be instantiated as shown below:

```
var coll = CLR.Create( "System.Collections.Generic.List<System.String>" );
```


© 2016-2022 Altova GmbH

or

```
CLR.Import( "System" );
CLR.Import( "System.Collections.Generic" );
var dictionary = CLR.Create( "Dictionary<String,Dictionary<String,String>>" );
```

.NET enumeration values

.NET enumeration values are accessed as shown below:

```
var enumValStretch = CLR.Static( "System.Windows.Forms.ImageLayout" ).Stretch;
```

Enumeration literals

The enumeration literals from the Authentic Desktop API can be accessed as shown below (there is no need to know their numerical value).

```
objExportXMIFileDlg.XMIType = eXMI21ForUML23;
```

14.1.1.7 Example Scripting Project

A demo project that illustrates scripting with Authentic Desktop is available at the following path: **C:\Users\<user>\Documents\Altova\Authentic2023\AuthenticExamples\SampleScripts.asprj**. This scripting project consists of a few macros and a Windows form.

To load the scripting project into Scripting Editor:

1. On the **Tools** menu, click **Scripting Editor**.
2. Click **Open** and browse for the **SampleScripts.asprj** file from the path above.

The project contains several macros in the "Macros" directory.

Macro	Description
AddMacroMenu	This macro adds a new menu item to Authentic Desktop, by invoking the `Application.AddMacroMenuItem` method of the COM API. The first argument of the `AddMacroMenuItem` method is the name of the macro to be added (in this example, "CloseAllButActiveDoc") and the second argument is the display text for the menu item. Whenever this macro is run, a new menu command called "CloseAllButActiveDoc") is added under the **Tools** menu. To clear macro menu items created previously, either restart Authentic Desktop or create a macro that calls the `Application.ClearMacroMenu` API method.
CloseAllButActiveDocument	When executed, the macro iterates though the currently open documents in Authentic Desktop and closes all of them, except for the active

	document.
SearchPath	This macro displays a form that lets users perform search for files within the current project. The form is available in the "Forms" directory, where you can view its design and the associated event handlers. The `GetAllPathsFromProject()` method returns all the file paths that belong to the currently opened project, as an array. The definition of this method is in the **GlobalDeclarations** script of the project. The `InsertStringInArrayUnique` method ensures that only unique paths are added to the array. Next, the form is initialized with CreateForm. Finally, the array is converted to a .NET type with the help of the CLR.Create method and the form is populated with the resulting `ArrayList` collection. The **Open** button of the form has a handler that calls the `Application.Documents.OpenFile` API method to open the currently selected file.

To enable the scripting project as global Authentic Desktop scripting project:

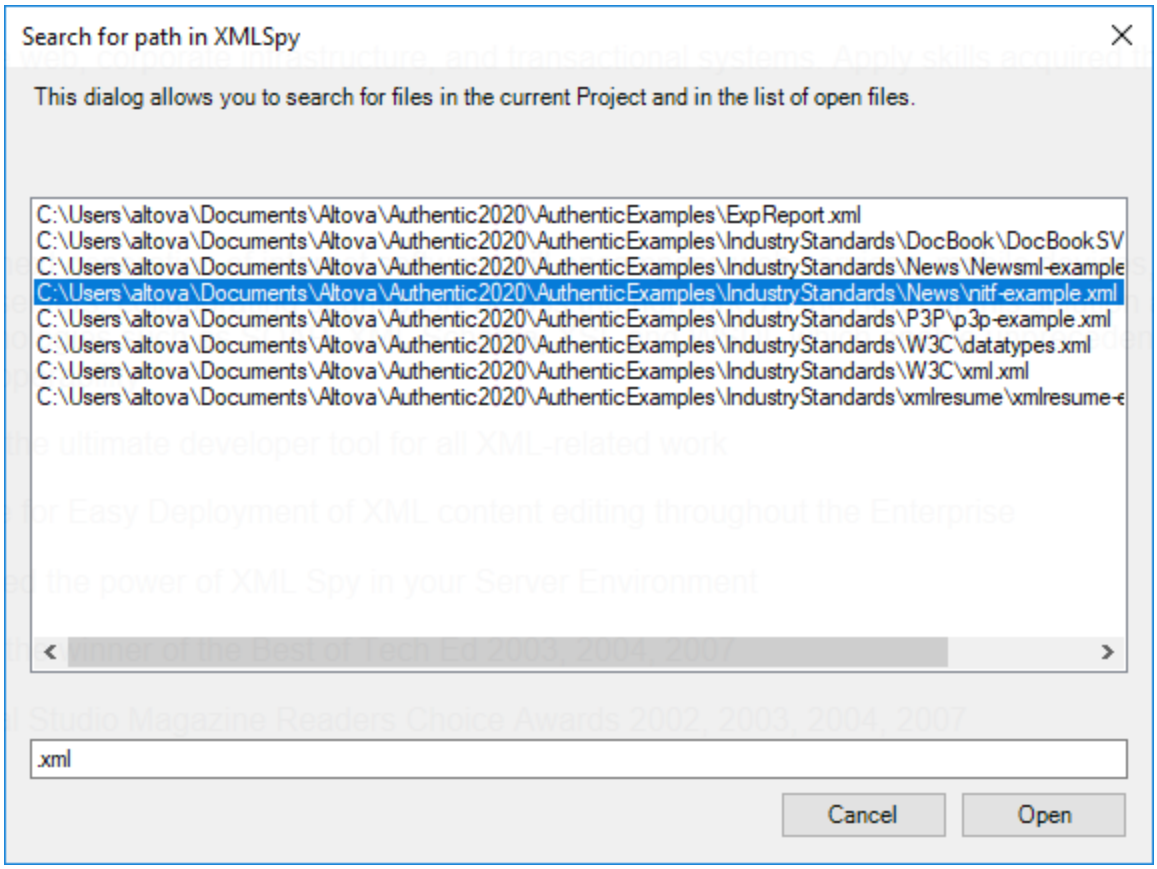
1. On the **Tools** menu, click **Options**.
2. Click the **Scripting** tab.
3. Under "Global scripting project file", click **Browse** and select the **SampleScripts.asprj** file from the path above.
4. This scripting project does not have auto-macros and application event handlers; therefore, you don't need to select either the **Run auto-macros...** or **Process events** check boxes.
5. Click **Apply**.

At this stage, several new menu items (one for each macro) become available under the **Tools | Macros** menu.

To run the "SearchPath" macro:

1. Open an Authentic Desktop project that contains several files (in this example, **C:\Users\<user>\Documents\Altova\Authentic2023\AuthenticExamples\Examples.spp**).
2. On the **Tools** menu, click **Macros**, and then click **Search Path**.
3. Type the search term (in this example, ".xml").

© 2016-2022 Altova GmbH



As shown above, all file names that contain the search term are now listed. You can click any element in the list, and then click **Open** to display it in the main editor.

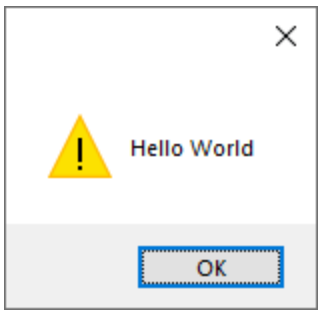
14.1.2 Built-in Commands

This section provides reference to all the commands you can use in the Authentic Desktop Scripting Editor.

- alert
- confirm
- CLR.Create
- CLR.Import
- CLR.LoadAssembly
- CLR.ShowImports
- CLR.ShowLoadedAssemblies
- CLR.Static
- CreateForm
- doevents
- lastform
- prompt
- ShowForm
- watchdog

14.1.2.1 alert

Displays a message box that shows a given message and the "OK" button. To proceed, the user will have to click "OK".



Signature

For JScript, the signature is:

```
alert(strMessage : String) -> void
```

For VBScript, the signature is:

```
MsgBox(strMessage : String) -> void
```

Example

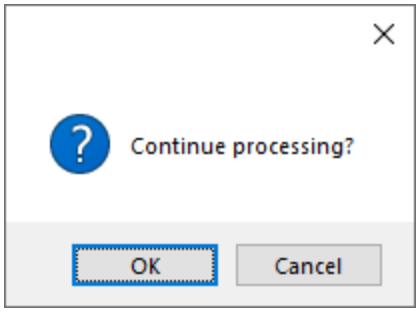
The following JScript code displays a message box with the text "Hello World".

```
alert("Hello World");
```

14.1.2.2 confirm

Opens a dialog box that shows a given message, a confirmation button, and a cancel button. The user will have to click either "OK" or "Cancel" to proceed. Returns a Boolean that represents the user's answer. If the user clicked "OK", the function returns **true**; if the user clicked "Cancel", the function returns **false**.

© 2016-2022 Altova GmbH



Signature

```
confirm(strMessage : String) -> result : Boolean
```

Example (JScript)

```
if ( confirm( "Continue processing?" ) == false )
  alert("You have cancelled this action");
```

Example (VBScript)

```
If ( confirm( "Continue processing?" ) = false ) Then
  MsgBox("You have cancelled this action")
End If
```

14.1.2.3 CLR.Create

Creates a new .NET object instance of the type name supplied as argument. If more than one argument is passed, the successive arguments are interpreted as the arguments for the constructor of the .NET object. The return value is a reference to the created .NET object

Signature

```
CLR.Create(strTypeNameCLR : String, constructor arguments ... ) -> object
```

Example

The following JScript code illustrates how to create instances of various .NET classes.

```
// Create an ArrayList
var objArray = CLR.Create("System.Collections.ArrayList");
// Create a ListViewItem
var newItem = CLR.Create( "System.Windows.Forms.ListViewItem", "NewItemText" );
// Create a List<string>
```

```
var coll = CLR.Create( "System.Collections.Generic.List<System.String>" );
// Import required namespaces and create a Dictionary object
CLR.Import( "System" );
CLR.Import( "System.Collections.Generic" );
var dictionary = CLR.Create( "Dictionary< String, Dictionary< String, String > >" );
```

14.1.2.4 CLR.Import

Imports a namespace. This is the scripting equivalent of C# `using` and VB.Net `imports` keyword. Calling `CLR.Import` makes it possible to leave out the namespace part in subsequent calls like `CLR.Create()` and `CLR.Static()`.

Note: Importing a namespace does not add or load the corresponding assembly to the scripting project. You can add assemblies to the scripting project dynamically (at runtime) in the source code by calling CLR.LoadAssembly.

Signature

```
CLR.Import(strNamespaceCLR : String) -> void
```

Example

Instead of having to use fully qualified namespaces like:

```
if ( ShowForm( "FormName" ) == CLR.Static( "System.Windows.Forms.DialogResult" ).OK )
{
    var sName = lastform.textboxFirstName.Text + " " + lastform.textboxLastName.Text;
    CLR.Static( "System.Windows.Forms.MessageBox" ).Show( "Hello " + sName );
}
```

One can import namespaces first and subsequently use the short form:

```
CLR.Import( "System.Windows.Forms" );

if ( ShowForm( "FormName" ) == CLR.Static( "DialogResult" ).OK )
{
    var sName = lastform.textboxFirstName.Text + " " + lastform.textboxLastName.Text;
    CLR.Static( "MessageBox" ).Show( "Hello " + sName );
}
```

14.1.2.5 CLR.LoadAssembly

Loads the .NET assembly with the given long assembly name or file path. Returns Boolean **true** if the assembly could be loaded; **false** otherwise.

© 2016-2022 Altova GmbH

Signature

```
CLR.LoadAssembly(strAssemblyNameCLR : String, showLoadErrors : Boolean) -> result :
Boolean
```

Example

The following JScript code attempts to set the clipboard text by loading the required assembly dynamically.

```
// set clipboard text (if possible)
// System.Windows.Clipboard is part of the PresentationCore assembly, so load this
assembly first:
if ( CLR.LoadAssembly( "PresentationCore, Version=3.0.0.0, Culture=neutral,
PublicKeyToken=31bf3856ad364e35", true ) )
{
    var clipboard = CLR.Static( "System.Windows.Clipboard" );
    if ( clipboard != null )
        clipboard.SetText( "HelloClipboard" );
}
```

14.1.2.6 CLR.ShowImports

Opens a message box that shows the currently imported namespaces. The user will have to click "OK" to proceed.

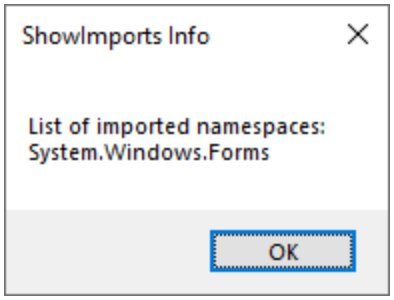
Signature

```
CLR.ShowImports() -> void
```

Example

The following JScript code first imports a namespace, and then displays the list of imported namespaces:

```
CLR.Import( "System.Windows.Forms");
CLR.ShowImports();
```



14.1.2.7 CLR.ShowLoadedAssemblies

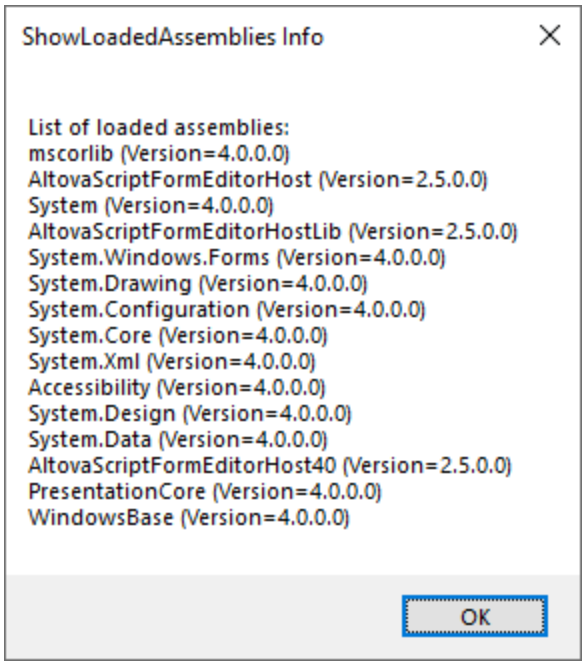
Opens a message box that shows the currently loaded assemblies. The user will have to click "OK" to proceed.

Signature

```
CLR.ShowLoadedAssemblies() -> void
```

Example

```
CLR.ShowLoadedAssemblies();
```



14.1.2.8 CLR.Static

Returns a reference to a static .NET object. You can use this function to get access to .NET types that have no instances and contain only static members.

Signature

```
CLR.Static(strTypeNameCLR : String) -> object
```


© 2016-2022 Altova GmbH

Example (JScript)

```
// Get the value of a .NET Enum into a variable
var enumValStretch = CLR.Static( "System.Windows.Forms.ImageLayout" ).Stretch

// Set the value of the Windows clipboard
var clipboard = CLR.Static( "System.Windows.Clipboard" );
clipboard.SetText( "HelloClipboard" );

// Check the buttons pressed by the user on a dialog box
if ( ShowForm( "FormName" ) == CLR.Static( "System.Windows.Forms.DialogResult" ).OK )
    alert( "ok" );
else
    alert( "cancel" );
```

14.1.2.9 CreateForm

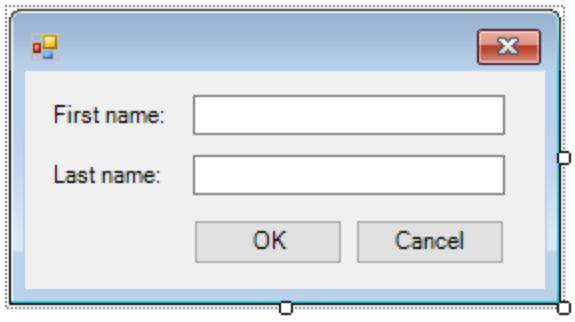
Instantiates the `Form` object identified by the name supplied as argument. The form must exist in the "Forms" folder of the scripting project. Returns the form object (`System.Windows.Forms.Form`) corresponding to the given name, or `null` if no form with such name exists.

Signature

```
CreateForm (strFormName : String) -> System.Windows.Forms.Form | null
```

Example

Let's assume that a form called "FormName" exists in the scripting project.



The following JScript code instantiates the form with some default values and displays it to the user.

```
var myForm = CreateForm( "FormName" );
if ( myForm != null )
{
   myForm.textboxFirstName.Text = "Daniela";
   myForm.textboxLastName.Text = "Heidegger";
```

```
   var dialogResult = myForm.ShowDialog();
}
```

The `dialogResult` can subsequently be evaluated as follows:

```
if ( dialogResult == CLR.Static( "System.Windows.Forms.DialogResult" ).OK )
    alert( "ok" );
else
    alert( "cancel" );
```

Note: The code above will work only if the **DialogResult** property of the "OK" and "Cancel" buttons is set correctly from the Properties pane (for example, it must be **OK** for the "OK" button).

14.1.2.10 doevents

Processes all Windows messages currently in the message queue.

Signature

```
doevents() -> void
```

Example (JScript)

```
for ( i=0; i < nLongLastingProcess; ++i )
{
    // do long lasting process

    doevents(); // process Windows messages; give UI a chance to update
}
```

14.1.2.11 lastform

This is a global field that returns a reference to the last form object that was created via `CreateForm()` or `ShowForm()`.

Signature

```
lastform -> formObj : System.Windows.Forms.Form
```

Example

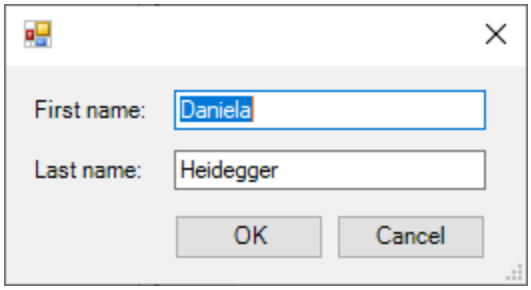
The following JScript code shows the form "FormName" as a dialog box.

```
CreateForm( "FormName" );
if ( lastform != null )
```


© 2016-2022 Altova GmbH

```
{
    lastform.textboxFirstName.Text = "Daniela";
    lastform.textboxLastName.Text = "Heidegger";
    var dialogResult = lastform.ShowDialog();
}
```

The values of both textbox controls are initialized with the help of `lastform`.



14.1.2.12 prompt

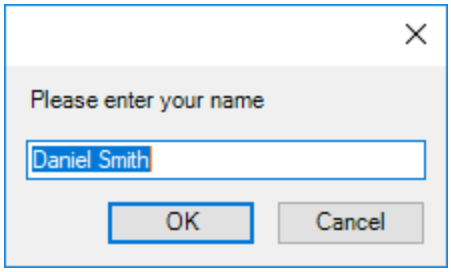
Opens a dialog box that shows a message and a textbox control with a default answer. This can be used to let the user input a simple string value. The return value is a string that contains the textbox value or null if the user selected "Cancel".

Signature

```
prompt(strMessage : String, strDefault : String) -> val : String
```

Example

```
var name = prompt( "Please enter your name", "Daniel Smith" );
if ( name != null )
    alert( "Hello " + name + "!" );
```



14.1.2.13 ShowForm

Instantiates a new form object from the given form name and immediately shows it as dialog box. The return value is an integer that represents the generated `DialogResult` (`System.Windows.Forms.DialogResult`). For the list of possible values, refer to the documentation of the `DialogResult` Enum (https://docs.microsoft.com/en-us/dotnet/api/system.windows.forms.dialogresult?view=netframework-4.8).

Signature

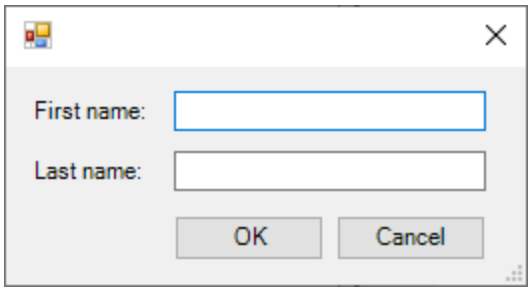
```
ShowForm(strFormName : String) -> result : Integer
```

Example

The following JScript code

```
var dialogResult = ShowForm( "FormName" );
```

Shows the form "FormName" as a dialog box:



The `DialogResult` can subsequently be evaluated, for example:

```
if ( dialogResult == CLR.Static( "System.Windows.Forms.DialogResult" ).OK )
    alert( "ok" );
else
    alert( "cancel" );
```

Note: The code above will work only if the **DialogResult** property of the "OK" and "Cancel" buttons is set correctly from the Properties pane (for example, it must be **OK** for the "OK" button).

14.1.2.14 watchdog

Long running CPU-intensive scripts may ask the user if the script should be terminated. The `watchdog()` method is used to disable or enable this behavior. By default, the watchdog is enabled.

Calling `watchdog(true)` can also be used to reset the watchdog. This can be useful before executing long running CPU-intensive tasks to ensure they have the maximum allowed script processing quota.

© 2016-2022 Altova GmbH

Signature

```
watchdog(bEnable : boolean) -> void
```

Example

```
watchdog( false );  // disable watchdog - we know the next statement is CPU intensive but
it will terminate for sure
doCPUIntensiveScript();
watchdog( true );   // re-enable watchdog
```

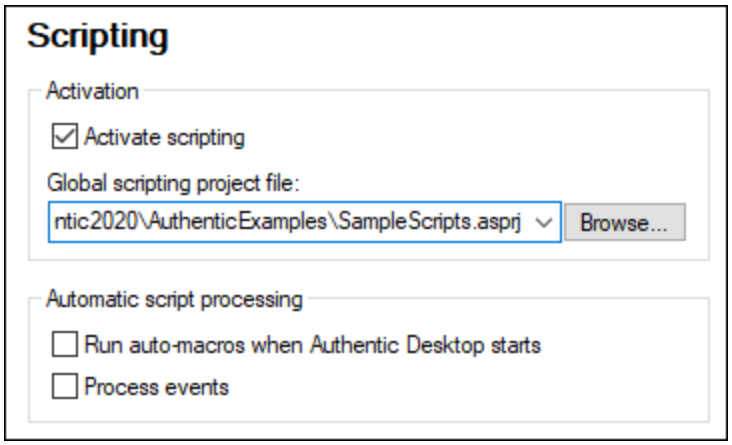
14.1.3 Enabling Scripts and Macros

Once a scripting project is complete and tested, you can use it in the following ways:

1. As the global scripting project for Authentic Desktop. This means that all the scripts and macros from the scripting project are available to Authentic Desktop.
2. At project level. This means that a reference to the .asprj file is saved together with the Authentic Desktop project. When the Authentic Desktop project is opened, its associated scripts and macros can be called.

To set a scripting project as global:

1. On the **Tools** menu, click **Options**.
2. Click the **Scripting** tab.
3. Select the **Activate scripting** check box and browse for the .asprj file to be used as global scripting project.



You can optionally enable the following additional script processing options:

Run auto-macros when Authentic Desktop starts	If you select this check box, any macros that were set as "Auto-macro" in the project will be triggerred automatically when Authentic Desktop starts.

Process events	Select this check box if your scripts bind to any application events. Clear the check box to prevent the scripts from reacting to events.

To enable a scripting project at project level:

1. Open the project.
2. On the **Project** menu, click **Script Settings**.
3. Select the **Activate project scripts** check box and browse for the .asprj file.

The **Run-auto macros...** check box has the same meaning as already described above.

14.1.3.1 Running Macros

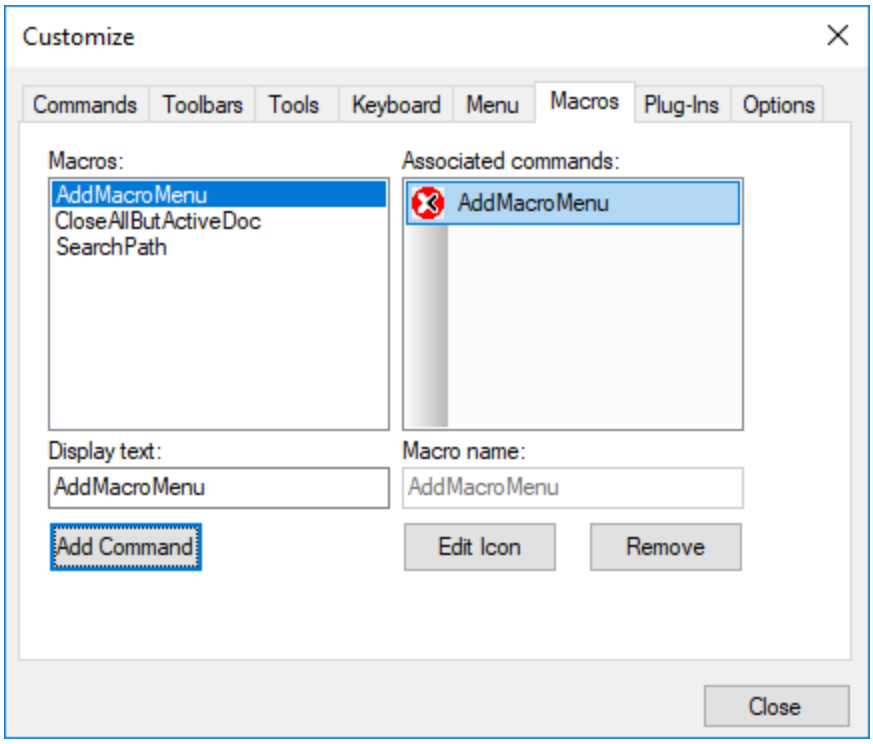
When a scripting project is active in Authentic Desktop, any macros available in that project are displayed in the **Tools | Macros** menu. Therefore, you can run a macro at any time, by triggering the respective menu command, for example **Tools | Macros | <SomeMacro>**.

Macros that were configured as auto-macros will run automatically whenever Authentic Desktop starts, provided that this behavior is enabled from options, as described in Enabling Scripts and Macros.

For convenience, you can create toolbar buttons for macros, as follows:

1. On the **Tools** menu, click **Customize**.
2. Click the **Macros** tab. Any macros that are available at application level (in the *global* scripting project) are listed.
3. Click **Add Command**.

© 2016-2022 Altova GmbH



4. Optionally, click **Edit icon** and draw a new icon for the new macro. You can also assign a shortcut to the macro, from the **Keyboard** tab.
5. Drag the macro from the **Associated commands** pane onto the toolbar where you would like it to appear.

To remove a macro from a toolbar:

1. On the **Tools** menu, click **Customize**.
2. Click the **Macros** tab.
3. Drag the macro from the toolbar where it appears back into the **Associated commands** pane.

14.2 IDE Plugins

Authentic Desktop allows you to create your own IDE plug-ins and integrate them into Authentic Desktop.

Use plug-ins to:

- Configure your version of Authentic Desktop, add commands through menus, icons, buttons etc.
- React to events from Authentic Desktop.
- Run your specific code within Authentic Desktop with access to the complete Authentic Desktop API

Authentic Desktop expects your plug-in to implement the IXMLSpyPlugIn interface. VB.NET, C# and C++ examples are included with your installation package and are located in the `Authentic2023\AuthenticExamples\IDEPlugin` folder of your Authentic Desktop installation.

Windows 7, 8, 10, 11	`C:/Users/<username>/Documents`

See ATL sample files for an example using C++.

14.2.1 Registration of IDE PlugIns

Authentic Desktop maintains a specific key in the Registry where it stores all registered IDE plug-ins:

HKEY_CURRENT_USER\Software\Altova\XML Spy\PlugIns

All values of this key are treated as references to registered plug-ins and must conform to the following format:

Value name:	ProgID of the plug-in
Value type:	must be REG_SZ
Value data:	CLSID of the component

Each time the application starts the values of the "PlugIns" key is scanned, and the registered plug-ins are loaded.

Register plug-in manually

To register a plug-in manually, use the "Customize" dialog box of the Authentic Desktop "Tools" menu. Use the "Add Plug-In..." button to specify the DLL that implements your plug-in. Authentic Desktop registers the DLL as a COM server and adds the corresponding entry in its "PlugIns" key.

If you experience problems with manual registration you can check if the CLSID of your plug-in is correctly registered in the "PlugIns" key. If this is not the case, the name of your plug-in DLL was probably not sufficiently unique. Use a different name or perform direct registration.

Register plug-in directly

A plug-in can be directly registered as an IDE plug-in by first registering the DLL and then adding the appropriate value to the "PlugIns" key of Authentic Desktop during plug-in setup for example. The new plug-in will be activated the next time Authentic Desktop is launched.

© 2016-2022 Altova GmbH

Creating plug-ins

Source code for sample plug-ins has been provided in the application's (My) Documents folder: `Examples\IDEPlugin` folder. To build a plug-in from such source code, do the following:

1. Open the solution you want to build as a plug-in in Visual Studio.
2. Build the plug-in with the command in the Build menu.
3. The plug-in's DLL file will be created in the `Bin` or `Debug` folder. This DLL file is the file that must be added as a plug-in (*see above*).

14.2.2 ActiveX Controls

ActiveX controls are supported. Any IDE PlugIn which is also an ActiveX control will be displayed in a Dialog Control Bar. A sample PlugIn that is also an ActiveX control is included in the `IDEPlugin` folder in the `Examples` folder of your application folder.

14.2.3 Configuration XML

The IDE plug-in allows you to change the user interface (UI) of Authentic Desktop. This is done by describing each separate modification using an XML data stream. The XML configuration is passed to Authentic Desktop using the GetUIModifications method of the IXMLSpyPlugIn interface.

The XML file containing the UI modifications for the IDE PlugIn, must have the following structure:

```
<ConfigurationData>
	<ImageFile>path To image file</ImageFile>
	<Modifications>
		<Modification>
			...
		</Modification>
		...
	</Modifications>
</ConfigurationData>
```

You can define icons or toolbar buttons for the new menu items which are added to the UI of Authentic Desktop by the plug-in. The path to the file containing the images is set using the ImageFile element. Each image must be 16 x 16 pixels using max. 256 colors. The image references must be arranged from left to right in a single (<ImageFile>...) line. The rightmost image index value is zero.

The Modifications element can have any number of Modification child elements. Each Modification element defines a specific change to the standard UI of Authentic Desktop. Starting with version 4.3, it is also possible to remove UI elements from Authentic Desktop.

Structure of Modification elements

All Modification elements consist of the following two child elements:

```
<Modification>
	<Action>Type of action</Action>
```

```
        <UIElement Type="type of UI element">
        </UIElement>
</Modification>
```

Valid values for the Action element are:

Add - to add the following UI element to Authentic Desktop
Hide - to hide the following UI element in Authentic Desktop
Remove - to remove the UI element from the "Commands" list box, in the customize dialog

You can combine values of the Action element e.g. "Hide Remove"

The UIElement element describes any new or existing UI element for Authentic Desktop. Possible elements are currently: new toolbars, buttons, menus or menu items. The **type** attribute defines which UI element is described by the XML element.

Common UIElement children
The ID and Name elements are valid for all different types of XML UIElement fragments. It is, however, possible to ignore one of the values for a specific type of UIElement. For example, Name is ignored for a separator.

```
<ID></ID>
<Name></Name>
```

If UIElement describes an existing element of the UI. The value of the ID element is predefined by Authentic Desktop. Normally these ID values are not known to the public. If the XML fragment describes a new part of the UI, then the ID is arbitrary and the value should be less than 1000. The Name element sets the textual value. Existing UI elements can be identified just by name, for example, menus and menu items with associated sub menus. For new UI elements, the Name element sets the caption (for example, the title of a toolbar) or text for a menu item.

Toolbars and Menus
To define a toolbar it is necessary to specify the ID and/or the name of the toolbar. An existing toolbar can be specified using only the name or ID (if the latter is known). To create a **new** toolbar both values must be set. The **type** attribute must be equal to "ToolBar".

```
<UIElement Type="ToolBar">
        <ID>1</ID>
        <Name>TestPlugIn</Name>
</UIElement>
```

To specify an Authentic Desktop menu you need two parameters:

- The ID of the menu bar which contains the menu. If no XML documents are open in the main window, the menu bar ID is 128. If one or more XML documents are open, the menu bar ID is 129.
- The menu name. Menus do not have an associated ID value. The following example defines the "Edit" menu of the menu bar which is active, when at least one XML document is open:

```
<UIElement Type="Menu">
        <ID>129</ID>
        <Name>Edit</Name>
</UIElement>
```

An additional element is used if you want to create a new menu. The Place element defines the position of the new menu in the menu bar:

© 2016-2022 Altova GmbH

```
<UIElement Type="Menu">
	<ID>129</ID>
	<Name>PlugIn Menu</Name>
	<Place>12</Place>
</UIElement>
```

A value of -1 for the Place element sets the new button or menu item at the end of the menu or toolbar.

Commands
If you add a new command (through a toolbar button or a menu item), the UIElement fragment can contain any of these sub elements:

```
<MacroName></MacroName>
<Info></Info>
<ImageID></ImageID>
```

If MacroName is specified, Authentic Desktop searches for a macro with the same name in the scripting environment and executes it each time this command is processed. The Info element contains a short description string which is displayed in the status bar when the mouse pointer is over the associated command (button or menu item). ImageID defines the index of the icon in the image file. Please note that all icons are stored in one image file.

To define a toolbar button create an UIElement with this structure:

```
<UIElement Type="ToolBarItem">
	<!--don't reuse local IDs even the commands do the same-->
	<ID>5</ID>
	<Name>Open file from repository...</Name>
	<!--Set Place To -1 If this is the first button To be inserted-->
	<Place>-1</Place>
	<ImageID>0</ImageID>
	<ToolBarID>1</ToolBarID>
	<!--instead of the toolbar ID the toolbar name could be used-->
	<ToolBarName>TestPlugIn</ToolBarName>
</UIElement>
```

Additional elements to declare a toolbar button are Place, ToolBarID and ToolBarName. ToolBarID and ToolBarName are used to identify the toolbar which contains the new or existing button. The textual value of ToolBarName is case-sensitive. The (UIElement) **type** attribute must be "ToolBarItem".

To define a menu item, the elements MenuID, Place and Parent are available in addition to the standard elements used to declare a command. MenuID can be either 128 or 129. Please see "Toolbars and Menus" for more information on these values.

The Parent element is used to identify the **menu** where the new menu entry should be inserted. As sub menu items have no unique Windows ID, we need some other way to identify the parent of the menu item.

The value of the Parent element is a path to the menu item. The text value of the Parent element, must equal the **parent menu name** of the submenu, where the submenu name is separated by a colon. If the menu has no parent, because it is not a submenu, add a colon to the beginning of the name. The **type** attribute must be set to "MenuItem". Example for an UIElement defining a menu item:

```
<UIElement Type="MenuItem">
```

```
        <!--the following element is a Local command ID-->
        <ID>3</ID>
        <Name>Open file from repository...</Name>
        <Place>-1</Place>
        <MenuID>129</MenuID>
        <Parent>:PlugIn Menu</Parent>
        <ImageID>0</ImageID>
    </UIElement>
```

Authentic Desktop makes it possible to add toolbar separators and menus if the value of the ID element is set to 0.

14.2.4 ATL sample files

The following pages show how to create a simple Authentic Desktop IDE plug-in DLL using ATL. To build the DLL it is necessary to know about ATL, the wizards that generate new ATL objects, as well as MS VisualStudio.

To access the API the implementation imports the Type Library of Authentic Desktop. The code reads various properties and calls methods using the smart pointers provided by the #import statement.

In addition, the sample code uses the MFC class CString and the ATL conversion macros such as W2T.

At a glance the steps to create an ATL DLL are as follows:

1. Open VisualStudio and select "New..." from the "File" menu.
2. Select the "styleclass="Code Bold"" tab.
3. Select "ATL COM AppWizard" and type in a project name.
4. Select "Support for MFC" if you want to use MFC classes, or if you want to create a project for the sample code.

   Having created the project files you can add an ATL object to implement the IXMLSpyPlugIn interface:

1. Select "New ATL Object..." from the "Insert" menu.
2. Select "Simple Object" from the wizard and click "Next".
3. Type in a name for the object.
4. On the "Attributes" tab, select "Custom" for the type of interface, and disable Aggregation.

These steps produce the skeleton code for the implementation of the IDE plug-in interface. Please see the following pages on how to modify the code and achieve some basic functionality.

14.2.4.1 Interface description (IDL)

The IDL of the newly created ATL object contains a declaration for one COM interface.

- This interface declaration must be replaced by the declaration of IXMLSpyPlugIn as shown below.
- The IDL must also contain the definition of the SPYUpdateAction enumeration.
- Replace the generated default interface name, (created by the wizard) with "IXMLSpyPlugIn" in the coclass declaration. The IDL should then look something like the example code below:

© 2016-2022 Altova GmbH

Having created the ATL object, you then need to implement the IDE plug-in interface of Authentic Desktop.

```
import "oaidl.idl";
import "ocidl.idl";

// ----- please insert the following block into your IDL file -----
 typedef enum {
  spyEnable = 1,
  spyDisable = 2,
  spyCheck = 4,
  spyUncheck = 8
 } SPYUpdateAction;

// ----- end insert block ----


// ----- E.g. Interface entry automatically generated by the ATL wizard -----
// [
//   object,
//   uuid(AB7CD86A-8145-429A-A1F3-270692EO8AFC),

//   helpstring("IXMLSpyPlugIn Interface")
//   pointer_default(unique)
// ]
// interface IXMLSpyPlugIn : IUnknown
// {
// };

// ----- end automatically generated Interface Entry


// ----- replace the Interface Entry (shown above) generated for you by the ATL wizard,
with the following block -----

 [
  odl,
  uuid(88F2A622-4B7E-42CD-8D04-3C0E5389DD85),
  helpstring("IXMLSpyPlugIn Interface")
 ]
 interface IXMLSpyPlugIn : IUnknown
  {
   HRESULT _stdcall OnCommand([in] long nID, [in] IDispatch* pXMLSpy);

   HRESULT _stdcall OnUpdateCommand([in] long nID, [in] IDispatch* pXMLSpy, [out, retval]
SPYUpdateAction* pAction);

   HRESULT _stdcall OnEvent([in] long nEventID, [in] SAFEARRAY(VARIANT)* arrayParameters,
[in] IDispatch* pXMLSpy, [out, retval] VARIANT* pReturnValue);

   HRESULT _stdcall GetUIModifications([out, retval] BSTR* pModificationsXML);

   HRESULT _stdcall GetDescription([out, retval] BSTR* pDescription);
  };
```

```
// ----- end replace block -----


// ----- The code below is automatically generated by the ATL wizard and will look slightly
different in your case -----

 [
 uuid(24FE0D1B-3FC0-494E-B36E-1D4CE412B014),
 version(1.0),
 helpstring("XMLSpyIDEPlugInDLL 1.0 Type Library")
 ]
 library XMLSPYIDEPLUGINDLLLib
 {
 importlib("stdole32.tlb");
 importlib("stdole2.tlb");

 [
  uuid(3800E791-7F6B-4ACD-9E32-2AC184444501),
  helpstring("XMLSpyIDEPlugIn Class")
 ]
 coclass XMLSpyIDEPlugIn
 {
  [default] interface IXMLSpyPlugIn;   // ----- define IXMLSpyPlugIn as the default
interface -----
 };
};
```

14.2.4.2 Class definition

In the class definition of the ATL object, several changes must be made. The class has to derive from IXMLSpyPlugIn, the "Interface Map" needs an entry for IXMLSpyPlugIn, and the methods of the IDE plug-in interface must be declared:

```
#ifndef __XMLSPYIDEPLUGIN_H_
#define __XMLSPYIDEPLUGIN_H_

#include "resource.h"       // main symbols

/////////////////////////////////////////////////////////////////////////////
// CXMLSpyIDEPlugIn
class ATL_NO_VTABLE CXMLSpyIDEPlugIn :
 public CComObjectRootEx<CComSingleThreadModel>,
 public CComCoClass<CXMLSpyIDEPlugIn, &CLSID_XMLSpyIDEPlugIn>,
 public IXMLSpyPlugIn
{
public:
 CXMLSpyIDEPlugIn()
 {
 }

DECLARE_REGISTRY_RESOURCEID(IDR_XMLSPYIDEPLUGIN)
```


© 2016-2022 Altova GmbH

```
DECLARE_NOT_AGGREGATABLE(CXMLSpyIDEPlugIn)

DECLARE_PROTECT_FINAL_CONSTRUCT()

BEGIN_COM_MAP(CXMLSpyIDEPlugIn)
 COM_INTERFACE_ENTRY(IXMLSpyPlugIn)
END_COM_MAP()

// IXMLSpyIDEPlugIn
public:
 virtual HRESULT _stdcall OnCommand(long nID, IDispatch* pXMLSpy);

 virtual HRESULT _stdcall OnUpdateCommand(long nID, IDispatch* pXMLSpy, SPYUpdateAction*
pAction);

 virtual HRESULT _stdcall OnEvent(long nEventID, SAFEARRAY **arrayParameters, IDispatch*
pXMLSpy, VARIANT* pReturnValue);

 virtual HRESULT _stdcall GetUIModifications(BSTR* pModificationsXML);

 virtual HRESULT _stdcall GetDescription(BSTR* pDescription);
};

#endif //__XMLSPYIDEPLUGIN_H_
```

14.2.4.3 Implementation

The code below shows a simple implementation of an Authentic Desktop IDE plug-in. It adds a menu item and a separator (available with Authentic Desktop) to the Tools menu. Inside the OnUpdateCommand() method, the new command is only enabled when the active document is displayed using the Grid View. The command searches for the XML element which has the current focus, and opens any URL starting with "http://", from the textual value of the element.

```
////////////////////////////////////////////////////////////////////////
// CXMLSpyIDEPlugIn

#import "XMLSpy.tlb"
using namespace XMLSpyLib;

HRESULT CXMLSpyIDEPlugIn::OnCommand(long nID, IDispatch* pXMLSpy)
{
 USES_CONVERSION;

 if(nID == 1) {
  IApplicationPtripSpyApp;

  if(pXMLSpy) {
   if(SUCCEEDED(pXMLSpy->QueryInterface(__uuidof(IApplication),(void **)&ipSpyApp))) {
    IDocumentPtr ipDocPtr = ipSpyApp->ActiveDocument;

    // we assume that grid view is active
```

```
    if(ipDocPtr) {
     IGridViewPtr ipGridPtr = ipDocPtr->GridView;

     if(ipGridPtr){
      IXMLDataPtr  ipXMLData = ipGridPtr->CurrentFocus;

      CString strValue = W2T(ipXMLData->TextValue);

      if(!strValue.IsEmpty() && (strValue.Left(7) == _T("http://")))
       ::ShellExecute(NULL,_T("open"),W2T(ipXMLData->TextValue),NULL,NULL,SW_SHOWNORMAL);
     }
    }
   }
  }
 }

 return S_OK;
}


HRESULT CXMLSpyIDEPlugIn::OnUpdateCommand(long nID, IDispatch* pXMLSpy, SPYUpdateAction*
pAction)
{
 *pAction = spyDisable;

 if(nID == 1) {
  IApplicationPtripSpyApp;

  if(pXMLSpy) {
   if(SUCCEEDED(pXMLSpy->QueryInterface(__uuidof(IApplication),(void **)&ipSpyApp)))     {
    IDocumentPtr ipDocPtr = ipSpyApp->ActiveDocument;

    // only enable if grid view is active
    if((ipDocPtr != NULL) && (ipDocPtr->CurrentViewMode == spyViewGrid))
     *pAction = spyEnable;
   }
  }
 }

 return S_OK;
}


HRESULT CXMLSpyIDEPlugIn::OnEvent(long nEventID, SAFEARRAY **arrayParameters, IDispatch*
pXMLSpy, VARIANT* pReturnValue)
{
 return S_OK;
}

HRESULT CXMLSpyIDEPlugIn::GetUIModifications(BSTR* pModificationsXML)
{
 CComBSTR bstrMods = _T(" \
```


© 2016-2022 Altova GmbH

```
    <ConfigurationData>\
     <Modifications>");
 // add "Open URL..." to Tools menu
 bstrMods.Append (_T(" \
      <Modification> \
       <Action>Add</Action> \
       <UIElement type=\"MenuItem\"> \
        <ID>1</ID> \
        <Name>Open URL...</Name> \
        <Place>0</Place> \
        <MenuID>129</MenuID> \
        <Parent>:Tools</Parent> \
       </UIElement> \
      </Modification>"));
 // add Seperator to Tools menu
 bstrMods.Append (_T(" \
      <Modification> \
       <Action>Add</Action> \
       <UIElement type=\"MenuItem\"> \
        <ID>0</ID> \
        <Place>1</Place> \
        <MenuID>129</MenuID> \
        <Parent>:Tools</Parent> \
       </UIElement> \
      </Modification>"));
 // finish modification description
 bstrMods.Append (_T(" \
     </Modifications> \
    </ConfigurationData>"));

 return bstrMods.CopyTo(pModificationsXML);
}


HRESULT CXMLSpyIDEPlugIn::GetDescription(BSTR* pDescription)
{
 CComBSTR bstrDescr = _T("ATL C++ XMLSpy IDE PlugIn;This PlugIn demonstrates the
implementation of a simple ATL DLL as a IDE PlugIn for XMLSpy.");
 return bstrDescr.CopyTo(pDescription);
}
```

14.2.5 IXMLSpyPlugIn

See also

Methods
OnCommand
OnUpdateCommand
OnEvent
GetUIModifications
GetDescription

Description
If a DLL is added to Authentic Desktop as an IDE plug-in, it is necessary that it registers a COM component that answers to an IXMLSpyPlugIn interface with the reserved uuid(88F2A622-4B7E-42CD-8D04-3C0E5389DD85), for it to be recognized as a plug-in.

14.2.5.1 OnCommand

See also

Declaration: `OnCommand(nID as long, pXMLSpy as IDispatch)`

Description
The `OnCommand()` method of the interface implementation, is called each time a command added by the the IDE plug-in (menu item or toolbar button) is processed. nID stores the command ID defined by the `ID` element of the respective `UIElement`.

`pXMLSpy` holds a reference to the dispatch interface of the `Application` object of Authentic Desktop.

Example

```
Public Sub IXMLSpyPlugIn_OnCommand(ByVal nID As Long, ByVal pXMLSpy As Object)
    If (Not (pXMLSpy Is Nothing)) Then
        Dim objDlg
        Dim objDoc As XMLSpyLib.Document
        Dim objSpy As XMLSpyLib.Application
        Set objSpy = pXMLSpy

        If nID = 3 Or nID = 5 Then
            Set objDlg = CreateObject("MSComDlg.CommonDialog")
            objDlg.Filter = "XML Files (*.xml)|*.xml|All Files (*.*)|*.*||"
            objDlg.FilterIndex = 1
            objDlg.ShowOpen

            If Len(objDlg.FileName) > 0 Then
                Set objDoc = objSpy.Documents.OpenFile(objDlg.FileName, False)
                Set objDoc = Nothing
            End If
        End If

        If nID = 4 Or nID = 6 Then
            Set objDlg = CreateObject("MSComDlg.CommonDialog")
            objDlg.Filter = "All Files (*.*)|*.*||"
            objDlg.Flags = cdlOFNPathMustExist
            objDlg.ShowSave

            If Len(objDlg.FileName) > 0 Then
                Set objDoc = objSpy.ActiveDocument

                If Not (objDoc Is Nothing) Then
                    objDoc.SetPathName objDlg.FileName
                    objDoc.Save
                    Set objDoc = Nothing
```


© 2016-2022 Altova GmbH

```
                End If
            End If
        End If

        Set objSpy = Nothing
    End If
End Sub
```

14.2.5.2 OnUpdateCommand

See also

Declaration: `OnUpdateCommand(nID as long, pXMLSpy as IDispatch) as SPYUpdateAction`

Description

The `OnUpdateCommand()` method is called each time the visible state of a button or menu item needs to be set. `nID` stores the command ID defined by the `ID` element of the respective `UIElement`.

`pXMLSpy` holds a reference to the dispatch interface of the `Application` object.

Possible return values to set the update state are:

```
spyEnable         = 1
spyDisable        = 2
spyCheck          = 4
spyUncheck        = 8
```

Example

```
Public Function IXMLSpyPlugIn_OnUpdateCommand(ByVal nID As Long, ByVal pXMLSpy As Object)
As SPYUpdateAction
    IXMLSpyPlugIn_OnUpdateCommand = spyDisable

    If (Not (pXMLSpy Is Nothing)) Then
        Dim objSpy As XMLSpyLib.Application
        Set objSpy = pXMLSpy

        If nID = 3 Or nID = 5 Then
            IXMLSpyPlugIn_OnUpdateCommand = spyEnable
        End If
        If nID = 4 Or nID = 6 Then
            If objSpy.Documents.Count > 0 Then
                IXMLSpyPlugIn_OnUpdateCommand = spyEnable
            Else
                IXMLSpyPlugIn_OnUpdateCommand = spyDisable
            End If
        End If
    End If
End Function
```

14.2.5.3 OnEvent

See also

Declaration: `OnEvent(nEventID as long, arrayParameters as SAFEARRAY(VARIANT), pXMLSpy as IDispatch) as VARIANT`

Description

`OnEvent()` is called each time an event is raised from Authentic Desktop.

Possible values for `nEventID` are:

On_BeforeStartEditing	= 1
On_EditingFinished	= 2
On_FocusChanged	= 3
On_Beforedrag	= 4
On_BeforeDrop	= 5
On_OpenProject	= 6
On_OpenDocument	= 7
On_CloseDocument	= 8
On_SaveDocument	= 9

Events available since Authentic Desktop 4r4:

On_DocEditDragOver	= 10
On_DocEditDrop	= 11
On_DocEditKeyDown	= 12
On_DocEditKeyUp	= 13
On_DocEditKeyPressed	= 14
On_DocEditMouseMove	= 15
On_DocEditButtonUp	= 16
On_DocEditButtonDown	= 17
On_DocEditContextMenu	= 18
On_DocEditPaste	= 19
On_DocEditCut	= 20
On_DocEditCopy	= 21
On_DocEditClear	= 22
On_DocEditSelectionChanged	= 23

Events available since Authentic Desktop 2004:

On_DocEditDragOver	= 10

© 2016-2022 Altova GmbH

Events available since Authentic Desktop 2004r4 (type library version 1.4):

On_BeforeOpenProject	= 25
On_BeforeOpenDocument	= 26
On_BeforeSaveDocument	= 27
On_BeforeCloseDocument	= 28
On_ViewActivation	= 29
On_DocEditKeyboardEvent	= 30
On_DocEditMouseEvent	= 31

Events available since Authentic Desktop 2006 SP1 (type library version 1.5):

On_BeforeValidate	= 32

Events available since Authentic Desktop 2007 (type library version 1.6):

On_BeforeShowSuggestions	= 33
On_ProjectOpened	= 34
On_Char	= 35

Events available since Authentic Desktop 2009 (type library version 2.2):

On_Initialize	= 36
On_Running	= 37
On_Shutdown	= 38

Events available since Authentic Desktop 2012 (type library version 2.8):

On_AuthenticBeforeSave	= 39
On_AuthenticContextMenuActivated	= 40
On_AuthenticLoad	= 41
On_AuthenticToolbarButtonClicked	= 42
On_AuthenticToolbarButtonExecuted	= 43
On_AuthenticUserAddedXMLNode	= 44

The names of the events are the same as they appear in the Scripting Environment of Authentic Desktop. For IDE plug-ins the names used are immaterial. The events are identified using the `ID` value.

`arrayParameters` is an array which is filled with the parameters of the currently raised event. Order, type and meaning of the single parameters are available through the scripting environment of Authentic Desktop. The events module of a scripting project, contains predefined functions for all events prior to version 4.4. The parameters passed to the predefined functions are identical to the array elements of the `arrayParameters` parameter.

Events raised from the Authentic View of Authentic Desktop do not pass any parameters directly. An "event" object is used instead. The event object can be accessed through the Document object of the active document.

`pXMLSpy` holds a reference to the dispatch interface of the `Application` object of Authentic Desktop.

If the return value of `OnEvent()` is set, then neither the IDE plug-in, nor an event handler inside of the scripting environment will get this event afterwards. Please note that all IDE plug-ins get/process the event before the Scripting Environment does.

14.2.5.4 GetUIModifications

See also

Declaration: `GetUIModifications()` as String

Description
The `GetUIModifications()` method is called during initialization of the plug-in, to get the configuration XML data that defines the changes to the UI of Authentic Desktop. The method is called when the plug-in is loaded for the first time, and at every start of Authentic Desktop.

See also Configuration XML for a detailed description how to change the UI.

Example

```
Public Function IXMLSpyPlugIn_GetUIModifications() As String
    ' GetUIModifications() gets the XML file with the specified modifications of
    ' the UI from the config.xml file in the plug-in folder
    Dim strPath As String
    strPath = App.Path

    If Len(strPath) > 0 Then
        Dim fso As New FileSystemObject
        Dim file As file

        Set file = fso.GetFile(strPath & "\config.xml")

        If (Not (file Is Nothing)) Then
            Dim stream As TextStream
            Set stream = file.OpenAsTextStream(ForReading)

            ' this replaces the token '**path**' from the XML file with
            ' the actual installation path of the plug-in to get the image file
            Dim strMods As String
            strMods = stream.ReadAll
            strMods = Replace(strMods, "**path**", strPath)

            IXMLSpyPlugIn_GetUIModifications = strMods
        Else
            IXMLSpyPlugIn_GetUIModifications = ""
        End If
    End If
End Function
```

14.2.5.5 GetDescription

See also

© 2016-2022 Altova GmbH

Declaration: `GetDescription()` as String

Description

`GetDescription()` is used to define the description string for the plug-in entries visible in the Customize dialog box.

Example

```
Public Function IXMLSpyPlugIn_GetDescription() As String
    IXMLSpyPlugIn_GetDescription = "Sample Plug-in for XMLSpy;This Plug-in demonstrates the
implementation of a simple VisualBasic DLL as a Plug-in for XMLSpy."
End Function
```

14.3 Application API

The COM-based API of Authentic Desktop (also called the Application API from now on) enables other applications to use the functionality of Authentic Desktop. As a result, it is possible to automate a wide range of tasks, from validating an XML file to modifying complex XML content (with the XMLData interface).

Authentic Desktop and its Application API follow the common specifications for automation servers set out by Microsoft. It is possible to access the methods and properties of the Application API from common development environments, such as those using C#, C++, VisualBasic, and Delphi, and with scripting languages like JScript and VBScript.

Execution environments for the Application API

The Application API can be accessed from the following execution environments:

- External programs (described below and in the Overview part of this section)
- From within the built-in Scripting Editor of Authentic Desktop. For a description of the scripting environment, see the section, Scripting Editor.
- Authentic Desktop allows you to create and integrate your own plug-ins into the application using a special interface for plug-ins. A description of how to create plug-ins is given in the section IDE Plug-ins.
- Via an ActiveX Control, which is available if the integration package is installed. For more information, see the section ActiveX Integration.

External programs

In the Overview part of this section, we describe how the functionality of Authentic Desktop can be accessed and automated from external programs.

Using the Application API from outside Authentic Desktop requires an instance of Authentic Desktop to be started first. How this is done depends on the programming language used. See the section, Programming Languages, for information about individual languages.

Essentially, Authentic Desktop will be started via its COM registration. Then the `Application` object associated with the Authentic Desktop instance is returned. Depending on the COM settings, an object associated with an already running Authentic Desktop can be returned. Any programming language that supports creation and invocation of COM objects can be used. The most common of these are listed below.

- JScript and VBScript script files have a simple syntax and are designed to access COM objects. They can be run directly from a DOS command line or with a double click on Windows Explorer. They are best used for simple automation tasks.
- C# is a full-fledged programming language that has a wide range of existing functionality. Access to COM objects can be automatically wrapped using C#..
- C++ provides direct control over COM access but requires relatively larger amounts of code than the other languages.
- Java: Altova products come with native Java classes that wrap the Application API and provide a full Java look-and-feel.
- Other programming languages that make useful alternatives are: Visual Basic for Applications, Perl, and Python.

© 2016-2022 Altova GmbH

Programming points

The following limitations must be considered in your client code:

- Be aware that if your client code crashes, instances of Authentic Desktop may still remain in the system.
- Don't hold references to objects in memory longer than you need them, especially those from the `XMLData` interface. If the user interacts between two calls of your client, then there is no guarantee that these references are still valid.
- Don't forget to disable dialogs if the user interface is not visible.
- See Error handling in JScript (and in C# and Java) for details of how to avoid annoying error messages.
- Free references explicitly if you are using C# or C++.

This documentation

This documentation section about the Application API is broadly divided into two parts.

- The first part consists of an Overview, which describes the object model for the API and explains how the API is accessed via various programming languages.
- The second part is a reference section (Interfaces and Enumerations) that contains descriptions of the interface objects of the Application API.

14.3.1 Overview

This overview of the Application API is organized as follows:

- The Object Model describes the relationships between the objects of the Application API.
- Programming Languages explains how the most commonly used programming languages (JScript, VBScript, C#, and Java) can be used to access the functionality of the Application API. Code listings from the example files supplied with your application package are used to describe basic mechanisms.

14.3.1.1 Object Model

The starting point for every application which uses the Application API is the `Application` object. This object contains general methods like import/export support and references to the open documents and any open project.

The `Application` object is created differently in various programming languages. In scripting languages such as JScript or VBScript, this involves calling a function which initializes the application's COM object. For examples, see the Programming Languages section.

XMLSpy.Application or AuthenticDesktop.Application

Authentic Desktop installs a TypeLibrary containing the XMLSpyLib. If this TypeLibrary has been added to the development environment (VB development environment, for example) then an object of the `Application` type can be created with:

```
Set objSpy = New XMLSpyLib.Application
```

If only Authentic Desktop is installed (and not XMLSpy), then

```
Set objSpy = GetObject("", "XMLSpy.Application")
```

does not work, because there won't be any object registered in the Registry with a ProgID of `XMLSpy.Application`. In this case, the registered object is `AuthenticDesktop.Application`.

The code listings in this documentation assume that both Authentic Desktop and XMLSpy have been installed. If, however, only Authentic Desktop has been installed, then please modify code fragments to take account of this difference.
The application object consists of the following parts:

1. Document collection and reference to the active document.
2. Reference to current project and methods for creating and opening projects.
3. Methods to support the export to and import from databases, text files, and Word documents.
4. URL management.
5. Methods for macro menu items.

Once you have created an Application object you can start using the functionality of Authentic Desktop. In most cases, you either open a project and access the documents from there or you directly open a document via the Documents interface.

14.3.1.2 Programming Languages

Programming languages differ in the way they support COM access. A few examples for the most frequently used languages (*links below*) will help you get started. The code listings in this section show how basic functionality can be accessed. The files in the `API` subfolder of the `Examples` folder can be used to test this functionality:

Windows 7, Windows 8, Windows 10, Windows 11	`C:\Users\<username>\Documents\Altova\Authentic\2023\%APPNAME%>Examples`

JScript

The JScript listings demonstrate the following basic functionality:

- Start application or attach to a running instance
- Simple document access
- Iteration
- Error handling
- Events

VBScript

VBScript is different than JScript only syntactically; otherwise it works in the same way. The listings below describe is an example of how VBScript can be used. For more information, refer to the JScript examples.

© 2016-2022 Altova GmbH

- Events: Shows how events are handled using VBScript.

C#

C# can be used to access the Application API functionality. The code listings show how to access the API for certain basic functionality.

- Start Authentic Desktop: Starts Authentic Desktop, which is registered as an automation server, or activates the application if it is already running.
- Open OrgChart.pxf: Locates one of the example documents installed with Authentic Desktop and opens it. If this document is already open it becomes the active document.
- OnDocumentOpened Event On/Off: Shows how to listen to Authentic Desktop events. When turned on, a message box will pop up after a document has been opened.
- Open ExpReport.xml: Opens another example document.
- Toggle View Mode: Changes the view of all open documents between Browser View and Authentic View. The code shows how to iterate through open documents.
- Validate: Validates the active document and shows the result in a message box. The code shows how to handle errors and COM output parameters.
- Shutdown Authentic Desktop: Stops Authentic Desktop.

Java

The Authentic Desktop API can be accessed from Java code. The Java sub-section of this section explains how some basic Authentic Desktop functionality can be accessed from Java code. It is organized into the following sub-sections:

- Mapping Rules for the Java Wrapper
- Example Java Project
- Application Startup and Shutdown
- Simple Document Access
- Iterations
- Use of Out-Parameters
- Event Handlers

14.3.1.2.1 JScript

This section contains listings of JScript code that demonstrate the following basic functionality:

- Start application or attach to a running instance
- Simple document access
- Iteration
- Error handling
- Events

Example files

The code listings in this section are available in example files that you can test as is or modify to suit your needs. The JScript example files are located in the `JScript` folder of the API Examples folder:

Windows 7, Windows 8, Windows 10, Windows 11	C:\Users\<username>\Documents\Altova\Authentic\2023\%APPNAME%>Examples

The example files can be run in one of two ways:

- *From the command line:* Open a command prompt window, change the directory to the path above, and type the name of one of the example scripts (for example, `Start.js`).
- *From Windows Explorer:* In Windows Explorer, browse for the JScript file and double-click it.

The script is executed by Windows Script Host that is packaged with Windows operating system. For more information about Windows Script Host, refer to MSDN documentation (https://msdn.microsoft.com).

14.3.1.2.1.1 Start Application

The JScript below starts the application and shuts it down. If the COM object of the 32-bit Authentic Desktop cannot be found, the code attempts to get the COM object of the 64-bit application; otherwise, an error is thrown. If an instance of the application is already running, the running instance will be returned.

Note : Pour Authentic Desktop 32-bit, le nom enregistré ou l'identificateur programmatique (ProgId) de l'objet COM est `AuthenticDesktop.Application`. Pour Authentic Desktop 64-bit, le nom est `AuthenticDesktop_x64.Application`. Néanmoins, n'oubliez pas que le programme d'appel accédera aux entrée de registre CLASSES dans sa propre ruche de registre, ou groupe (32-bit ou 64-bit). C'est pourquoi, si vous exécutez des scripts en utilisant l'invite de commande standard et Windows Explorer sur Windows 64-bit, les entrées de registre 64-bit seront accédées, et pointeront vers le Authentic Desktop 64-bit. Pour cette raison, si les deux Authentic Desktop 32-bit et 64-bit sont installés, une gestion spéciale est nécessaire afin de pouvoir appeler le Authentic Desktop 32-bit. Par exemple, si on part du principe que Windows Scripting Host est le programme d'appel, procéder comme suit :

1. Changer le répertoire actuel à **C:\Windows\SysWOW64**.
2. Dans la ligne de commande, saisir **wscript.exe** suivi du chemin vers le script que vous souhaitez exécuter, par exemple :

```
wscript.exe "C:\Users\...
\Documents\Altova\Authentic2023\AuthenticExamples\API\JScript\start.js"
```

```
// Initialize application's COM object. This will start a new instance of the application
and
// return its main COM object. Depending on COM settings, the main COM object of an
already
// running application might be returned.

try {   objAuthentic = WScript.GetObject("", "AuthenticDesktop.Application");   }
catch(err) {}

if( typeof( objAuthentic ) == "undefined" )
{
```


© 2016-2022 Altova GmbH

```
   try   {   objAuthentic = WScript.GetObject("", "AuthenticDesktop_x64.Application")   }
   catch(err)
   {
      WScript.Echo( "Can't access or create AuthenticDesktop.Application" );
      WScript.Quit();
   }
}

// if newly started, the application will start without its UI visible. Set it to
visible.
objAuthentic.Visible = true;

WScript.Echo(objAuthentic.Edition + " has successfully started. ");

objAuthentic.Visible = false; // will shutdown application if it has no more COM
connections
//objAuthentic.Visible = true;   // will keep application running with UI visible
```

The JScript code listed above is available in the sample file `Start.js` (see Example Files).

14.3.1.2.1.2 Simple Document Access

After you have started the application as shown in Start Application, you will most likely want to programmatically open a document in order to work with it. The JScript code listing below illustrates how to open two documents from the Authentic Desktop Examples folder and set one of them as the active document.

```
// Locate examples via USERPROFILE shell variable. The path needs to be adapted to major
release versions.
objWshShell = WScript.CreateObject("WScript.Shell");
majorVersionYear = objAuthentic.MajorVersion + 1998
strExampleFolder = objWshShell.ExpandEnvironmentStrings("%USERPROFILE%") + "\\My
Documents\\Altova\\Authentic" + majorVersionYear + "\\AuthenticExamples\\";

// Tell Authentic to open two documents. No dialogs
objDoc1 = objAuthentic.Documents.OpenFile(strExampleFolder + "OrgChart.pxf", false);
objAuthentic.Documents.OpenFile(strExampleFolder + "ExpReport.xml", false);

// The document currently active can be easily located.
objDoc2 = objAuthentic.ActiveDocument;

// Let us make sure that the document is shown in Authentic view.
objDoc2.SwitchViewMode(5);   // SPYViewModes.spyViewAuthentic = 5

// Now switch back to the document opened first
objDoc1.SetActiveDocument();
```

The JScript code listed above is available in the sample file `DocumentAccess.js` (see Example Files).

14.3.1.2.1.3 Iteration

The JScript listing below shows how to iterate through the open documents. It is assumed that you have already started the application and opened some documents as shown in the previous sections.

```
// go through all open documents using a JScript Enumerator
bRequiresSaving = false;
for (var iterDocs = new Enumerator(objAuthentic.Documents); !iterDocs.atEnd();
iterDocs.moveNext())
{
   if (iterDocs.item().IsModified)
      bRequiresSaving = true;

   var strErrorText = new Array(1);
   var nErrorNumber = new Array(1);
   var errorData = new Array(1);

   if (!iterDocs.item().IsValid(strErrorText, nErrorNumber, errorData))
   {
       var text = strErrorText;
      // access that XMLData object only if filled in
       if (errorData[0] != null)
         text += "(" + errorData[0].Name + "/" + errorData[0].TextValue + ")";

      WScript.Echo("Document \"" + iterDocs.item().Name +"\" validation error[" +
nErrorNumber + "]: " + text);
   }
   else
   {
      // The COM call succeeded and the document is valid.
      WScript.Echo("Document \"" + iterDocs.item().Name + "\" is valid.");
   }
}

// go through all open documents using index-based access to the document collection
for (i = objAuthentic.Documents.Count; i > 0; i--)
   objAuthentic.Documents.Item(i).Close(false);
```

The JScript code listed above is available in the sample file `DocumentAccess.js` (see Example Files).

14.3.1.2.1.4 Error Handling

The Application API returns errors in two different ways:

- The `HRESULT` returned by every API method
- The `IErrorInfo` interface of the Application API

© 2016-2022 Altova GmbH

Every API method returns an HRESULT. This return value gives the caller information about errors during execution of the method. If the call was successful, the return value is S_OK. The HRESULT option is commonly used in C/C++ programs.

However, programming languages such as VisualBasic and scripting languages (and other high-level development environments) don't give the programmer access to the HRESULT return of a COM call. Such languages use the IErrorInfo interface, which is also supported by the Application API. If an error occurs, the Application API creates a new object that implements the IErrorInfo interface. The information provided by the IErrorInfo interface is imported by the development environment into its own error-handling mechanism.

For example, the JScript code listing below causes an error to be thrown by incorrectly declaring an array. Additional information about the error object is provided by its properties number and description.

```
try {
   var arr = new Array(-1);
}
catch (err) {
   WScript.Echo("Error : (" + (err.number & 0xffff) + ")" + err.description);
}
```

14.3.1.2.1.5 Events

COM specifies that a client must register itself at a server for callbacks using the connection point mechanism. The automation interface for XMLSpy defines the necessary event interfaces. The way to connect to those events depends on the programming language you use in your client. The following code listing shows how this is done using JScript.

The method WScript.ConnectObject is used to receive events.

```
// The event-handler function
function DocEvent_OnBeforeCloseDocument(objDocument)
{
   WScript.Echo("Received event - before closing document");
}

// Create or connect to XMLSpy (or Authentic Desktop)
try
{
   // Create the environment and XMLSpy (or Authentic Desktop)
   objWshShell = WScript.CreateObject("WScript.Shell");
   objFSO = WScript.CreateObject("Scripting.FileSystemObject");
   objSpy = WScript.GetObject("", "XMLSpy.Application");

// If only Authentic Desktop is installed (and XMLSpy is not installed) use:
// objSpy = WScript.GetObject("", "AuthenticDesktop.Application")

}
catch(err)
   { WScript.Echo ("Can't create WScript.Shell object or XMLSpy"); }
```

```
// Create document object and connect to its events
objSpy.Visible = true;
majorVersionYear = objSpy.MajorVersion + 1998
docPath = objWshShell.ExpandEnvironmentStrings("%USERPROFILE%") + "\\Documents\\Altova\
\XMLSpy" + majorVersionYear + "\\Examples\\ExpReport.xml";
objDoc = objSpy.Documents.OpenFile (docPath, false);
WScript.ConnectObject(objDoc, "DocEvent_");

// Keep running while waiting for the event
// In the meanwhile close this document in XMLSpy (or Authentic Desktop) manually
WScript.Echo ("Sleeping for 10 seconds ...");
WScript.Sleep (10000);

objDoc = null;
WScript.Echo ("Stopped listening for event");
objSpy.Quit();
```

14.3.1.2.2 VBScript

VBScript is syntactically different than JScript but works in the same way. This section contains a listing showing how events are used with VBScript and an example.

For information about other functionality, refer to the JScript examples listed below:

- Start application or attach to a running instance
- Simple document access
- Iteration
- Error handling

14.3.1.2.2.1 Events

COM specifies that a client must register itself at a server for callbacks using the connection point mechanism. The automation interface for XMLSpy defines the necessary event interfaces. The way to connect to those events depends on the programming language you use in your client. The following code listing shows how this is done using VBScript.

The method `WScript.ConnectObject` is used to receive events.

To run this code, paste it into a file with .vbs extension, and either double-click in Windows Explorer, or run it from a command prompt.

```
' the event handler function
Function DocEvent_OnBeforeCloseDocument(objDocument)
   Call WScript.Echo("received event - before closing document")
End Function

' create or connect to XmlSpy
```


© 2016-2022 Altova GmbH

```
Set objWshShell = WScript.CreateObject("WScript.Shell")
Set objFSO = WScript.CreateObject("Scripting.FileSystemObject")
Set objSpy = WScript.GetObject("", "XMLSpy.Application")
' If only Authentic is installed (and XMLSpy is not installed) use:
' Set objSpy = WScript.GetObject("", "AuthenticDesktop.Application")
' If only XMLSpy 64-bit is intalled, use:
' Set objSpy = WScript.GetObject("", "XMLSpy_x64.Application")

' create document object and connect to its events
objSpy.Visible = True

' Find out user's personal folder and locate one of the installed examples.
personalFolder = objWshShell.ExpandEnvironmentStrings("%UserProfile%")
majorVersionYear = objSpy.MajorVersion + 1998
xmlspyExamplesFolder = personalFolder & "\Documents\Altova\XMLSpy" & majorVersionYear  &
"\Examples\"
docPath = xmlspyExamplesFolder & "ExpReport.xml"

' open a document
Set objDoc = objSpy.Documents.OpenFile (docPath, False)
Call WScript.ConnectObject(objDoc, "DocEvent_")

' keep running while waiting on the event
' in the meantime close the document in XMLSPY manually
Call WScript.Echo ("sleeping for 10 seconds ...")
Call WScript.Sleep (10000)

Set objDoc = Nothing
Call WScript.Echo ("stopped listening for event")
Call objSpy.Quit
```

Note : Pour Authentic Desktop 32-bit, le nom enregistré ou l'identificateur programmatique (ProgId) de l'objet COM est `AuthenticDesktop.Application`. Pour Authentic Desktop 64-bit, le nom est `AuthenticDesktop_x64.Application`. Néanmoins, n'oubliez pas que le programme d'appel accédera aux entrée de registre CLASSES dans sa propre ruche de registre, ou groupe (32-bit ou 64-bit). C'est pourquoi, si vous exécutez des scripts en utilisant l'invite de commande standard et Windows Explorer sur Windows 64-bit, les entrées de registre 64-bit seront accédées, et pointeront vers le Authentic Desktop 64-bit. Pour cette raison, si les deux Authentic Desktop 32-bit et 64-bit sont installés, une gestion spéciale est nécessaire afin de pouvoir appeler le Authentic Desktop 32-bit. Par exemple, si on part du principe que Windows Scripting Host est le programme d'appel, procéder comme suit :

1. Changer le répertoire actuel à **C:\Windows\SysWOW64**.
2. Dans la ligne de commande, saisir **wscript.exe** suivi du chemin vers le script que vous souhaitez exécuter, par exemple :

```
wscript.exe "C:\Users\...
\Documents\Altova\Authentic2023\AuthenticExamples\API\JScript\start.js"
```

14.3.1.2.2.2 Example: Using Events

Authentic View supports event connection on a per-object basis. Implementation of this feature is based on COM connection points and is available in environments that support this mechanism.

The following example is a VBScript code example that shows how to use events from within a VBScript project.

```
' ------------------------------------------------------------------------
' VBScript example that demonstrates how to use events.
' ------------------------------------------------------------------------

' Event handler for OnSelectionChanged event of AuthenticView
Function AuthenticViewEvent_OnSelectionChanged(objAuthenticRange)
    If objAuthenticRange.FirstTextPosition <> objAuthenticRange.LastTextPosition Then
        Call WScript.Echo("Selection: " & objAuthenticRange.Text & vbNewLine & vbNewLine
& "Close this dialog.")
    Else
        Call WScript.Echo("Cursor position: " & objAuthenticRange.FirstTextPosition &
vbNewLine & vbNewLine & "Close this dialog.")
    End If
End Function

' Start/access XMLSpy and connect to its automation interface.
Set WshShell = WScript.CreateObject("WScript.Shell")
Set objSpy = GetObject("", "XMLSpy.Application")
' Make the UI of XMLSpy visible.
objSpy.Visible = True

' Find out user's personal folder and locate one of the installed XMLSpy examples.
personalFolder = WshShell.ExpandEnvironmentStrings("%UserProfile%")
majorVersionYear = objSpy.MajorVersion + 1998
xmlspyExamplesFolder = personalFolder & "\Documents\Altova\XMLSpy" & majorVersionYear  &
"\Examples\"
docPath = xmlspyExamplesFolder & "ExpReport.xml"

' Create object to access windows file system and test if the our document exists.
Set fso = CreateObject("Scripting.FileSystemObject")
If fso.FileExists(docPath) Then
    ' open the document
    Call objSpy.Documents.OpenFile(docPath, False)
    set objDoc = objSpy.ActiveDocument

    ' switch active document to authentic view
    objDoc.SwitchViewMode 4 ' spyViewAuthentic

    ' Register for connection point events on the authentic view of the active document.
    ' Any function with a valid event name prefixed with "AuthenticViewEvent_" will
    ' be called when the corresponding event gets triggered on the specified object.
    set objView = objDoc.AuthenticView
    Call WScript.ConnectObject(objView, "AuthenticViewEvent_")
    Call WScript.Echo("Events are connected." & vbNewLine & vbNewLine & "Now set or move
```


© 2016-2022 Altova GmbH

```
the cursor in XMLSpy." & vbNewLine & vbNewLine & "Close this dialog to shut down
XMLSpy.")

    ' To disconnect from the events delete the reference to the object.
    set objView = Nothing
Else
    Call WScript.Echo("The file " & docPath & " does not exist.")
End If

' shut down XMLSpy when this script ends
objSpy.Visible = False
```

14.3.1.2.3 C#

The C# programming language can be used to access the Application API functionality. You could use Visual Studio 2012/2013/2015/2017/2019/2022 to create the C# code, saving it in a Visual Studio project. Create the project as follows:

1. In Microsoft Visual Studio, add a new project using **File | New | Project**.
2. Add a reference to the Authentic Desktop Type Library by clicking **Project | Add Reference**. The Add Reference dialog appears. Browse for the Authentic Desktop Type Library component, which is located in the Authentic Desktop application folder, and add it.
3. Enter the code you want.
4. Compile the code and run it.

Example C# project

Your Authentic Desktop package contains an example C# project, which is located in the `API\C#` subfolder of the `Examples` folder :

Windows 7, Windows 8, Windows 10, Windows 11	`C:\Users\<username>\Documents\Altova\Authentic\2023\%APPNAME%>Examples`

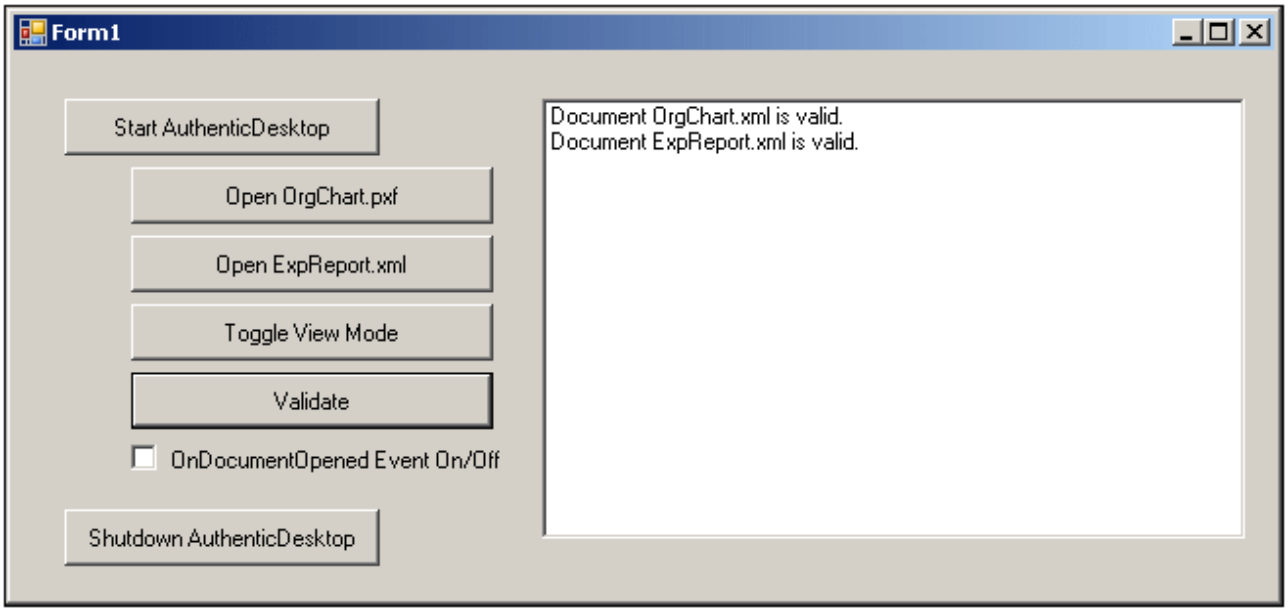
You can compile and run the project from within Visual Studio 2012/2013/2015/2017/2019/2022. The code listing below shows how basic application functionality can be used. This code is similar to the example C# project in the API Examples folder of your application package, but might differ slightly.

Platform configuration

If you have a 64-bit operating system and are using a 32-bit installation of Authentic Desktop, you must add the x86 platform in the solution's Configuration Manager and build the sample using this configuration. A new x86 platform (for the active solution in Visual Studio) can be created in the New Solution Platform dialog (**Build | Configuration Manager | Active solution platform | <New...>**).

What the code listing below does

The example code listing below creates a simple user interface (*screenshot below*) with buttons that invoke basic Authentic Desktop operations:



- Start Authentic Desktop: Starts Authentic Desktop, which is registered as an automation server, or activates the application if it is already running.
- Open OrgChart.pxf: Locates one of the example documents installed with Authentic Desktop and opens it. If this document is already open it becomes the active document.
- Open ExpReport.xml: Opens another example document.
- Toggle View Mode: Changes the view of all open documents between Text View and Authentic View. The code shows how to iterate through open documents.
- Validate: Validates the active document and shows the result in a message box. The code shows how to handle errors and COM output parameters.
- Shut down Authentic Desktop: Stops Authentic Desktop.

You can modify the code (of the code listing below or of the example C# project in the API Examples folder) in any way you like and run it.

Compiling and running the example

In the API Examples folder, double-click the file `AutomateAuthenticDesktop_VS2008.sln` or the file `AutomateAuthenticDesktop_VS2010.sln` (to open in Visual Studio 2012/2013/2015/2017/2019/2022). Alternatively the file can be opened from within Visual Studio (with **File | Open | Project/Solution**). To compile and run the example, select **Debug | Start Debugging** or **Debug | Start Without Debugging**.

Code listing of the example

Given below is the C# code listing of the basic functionality of the form (`Form1.cs`) created in the `AutomateAuthenticDesktop` example. Note that the code listed below might differ slightly from the code in the API Examples form.The listing below is commented for ease of understanding. Parts of the code are also presented separately in the sub-sections of this section, according to the Application API functionality they access.

The code essentially consists of a series of handlers for the buttons in the user interface shown in the screenshot above.

© 2016-2022 Altova GmbH

```
namespace WindowsFormsApplication2
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
        }

        // An instance of AuthenticDesktop is accessed via its automation interface.
        XMLSpyLib.Application AuthenticDesktop;

        // Location of examples installed with AuthenticDesktop
        String strExamplesFolder;

        private void Form1_Load(object sender, EventArgs e)
        {
            // Locate examples installed with AuthenticDesktop.
            // REMARK: You might need to adapt this if you have a different major version
of the product.
            strExamplesFolder = Environment.GetEnvironmentVariable("USERPROFILE") + "\\My
Documents\\Altova\\Authentic2012\\AuthenticExamples\\";
        }

        // Handler for the "Start AuthenticDesktop" button
        private void StartAuthenticDesktop_Click(object sender, EventArgs e)
        {
            if (AuthenticDesktop == null)
            {
                Cursor.Current = Cursors.WaitCursor;

                // If there is no AuthenticDesktop instance, create one and make it
visible.
                AuthenticDesktop = new XMLSpyLib.Application();
                AuthenticDesktop.Visible = true;

                Cursor.Current = Cursors.Default;
            }
            else
            {
                // If an AuthenticDesktop instance is already running, make sure it's
visible.
                if (!AuthenticDesktop.Visible)
                    AuthenticDesktop.Visible = true;
            }
        }

        // Handler for the "Open OrgChart.pxf" button
        private void openOrgChart_Click(object sender, EventArgs e)
        {
            // Make sure there's a running Authentic Desktop instance, and that it's
visible
            StartAuthenticDesktop_Click(null, null);

            // Open a sample file installed with the product.
```

```
            AuthenticDesktop.Documents.OpenFile(strExamplesFolder + "OrgChart.pxf", false);
        }

        // Handler for the "Open ExpReport.xml" button
        private void openExpReport_Click(object sender, EventArgs e)
        {
            // Make sure there's a running Authentic Desktop instance, and that it's
visible
            StartAuthenticDesktop_Click(null, null);

            // Open a sample file installed with the product.
            AuthenticDesktop.Documents.OpenFile(strExamplesFolder + "ExpReport.xml",
false);
        }

        // Handler for the "Toggle View Mode" button
        private void toggleView_Click(object sender, EventArgs e)
        {
            // Make sure there's a running Authentic Desktop instance, and that it's
visible
            StartAuthenticDesktop_Click(null, null);

            // Iterate through all open documents and toggle the current view between Text
View and Authentic View.
            foreach (XMLSpyLib.Document doc in AuthenticDesktop.Documents)
                if (doc.CurrentViewMode == XMLSpyLib.SPYViewModes.spyViewAuthentic)
                    doc.SwitchViewMode(XMLSpyLib.SPYViewModes.spyViewBrowser);
                else
                    doc.SwitchViewMode(XMLSpyLib.SPYViewModes.spyViewAuthentic);
        }

        // Handler for the "Shut down AuthenticDesktop" button
        // Shut down application instance by explicitely releasing the COM object.
        private void shutdownAuthenticDesktop_Click(object sender, EventArgs e)
        {
            if (AuthenticDesktop != null)
            {
                // Allow shut down of AuthenticDesktop by releasing the UI
                AuthenticDesktop.Visible = false;

                // Explicitly release the COM object
                try
                {
                    while
(System.Runtime.InteropServices.Marshal.ReleaseComObject(AuthenticDesktop) > 0) ;
                }
                finally
                {
                    // Avoid subsequent access to this object.
                    AuthenticDesktop = null;
                }
            }
        }

        // Handler for the "Validate" button
        private void validate_Click(object sender, EventArgs e)
```

 © 2016-2022 Altova GmbH

```
        {
            // COM errors get returned to C# as exceptions. Use a try/catch block to handle
them.
            try
            {
                // Method 'IsValid' is one of the few functions that use output parameters.
                // Use 'object' type for these parameters.
                object strErrorText = "";
                object nErrorNumber = 0;
                object errorData = null;

                if (!AuthenticDesktop.ActiveDocument.IsValid(ref strErrorText, ref
nErrorNumber, ref errorData))
                {
                    // The COM call succeeds but the document is not valid.
                    // A detailed description of the problem is returned in strErrorText,
nErrorNumber and errorData.
                    listBoxMessages.Items.Add("Document " +
AuthenticDesktop.ActiveDocument.Name + " is not valid.");
                    listBoxMessages.Items.Add("\tErrorText  : " + strErrorText);
                    listBoxMessages.Items.Add("\tErrorNumber: " + nErrorNumber);
                    listBoxMessages.Items.Add("\tElement    : " + (errorData != null ?
((XMLSpyLib.XMLData)errorData).TextValue : "null"));
                }
                else
                {
                    // The COM call succeeds and the document is valid.
                    listBoxMessages.Items.Add("Document " +
AuthenticDesktop.ActiveDocument.Name + " is valid.");
                }
            }
            catch (Exception ex)
            {
                // The COM call was not successful.
                // Probably no application instance has been started or no document is
open.
                listBoxMessages.Items.Add("Error validating active document: " +
ex.Message);
            }
        }

        delegate void addListBoxItem_delegate(string sText);
        // Called from the UI thread
        private void addListBoxItem(string sText)
        {
            listBoxMessages.Items.Add(sText);
        }
        // Wrapper method to call UI control methods from a worker thread
        void syncWithUIthread(Control ctrl, addListBoxItem_delegate methodToInvoke, String
sText)
        {
            // Control.Invoke: Executes on the UI thread, but calling thread waits for
completion before continuing.
            // Control.BeginInvoke: Executes on the UI thread, and calling thread doesn't
wait for completion.
            if (ctrl.InvokeRequired)
```

```
                ctrl.BeginInvoke(methodToInvoke, new Object[] { sText });
        }

        // Event handler for OnDocumentOpened event
        private void handleOnDocumentOpened(XMLSpyLib.Document i_ipDocument)
        {
            String sText = "";

            if (i_ipDocument.Name.Length > 0)
                sText = "Document " + i_ipDocument.Name + " was opened!";
            else
                sText = "An empty document was created.";

            // Synchronize the calling thread with the UI thread because
            // COM events are triggered from a working thread
            addListBoxItem_delegate methodToInvoke = new
addListBoxItem_delegate(addListBoxItem);
            // Call syncWithUIthread with the following arguments:
            // 1 - listBoxMessages - list box control to display messages from COM events
            // 2 - methodToInvoke  - a C# delegate which points to the method which will be
called from the UI thread
            // 3 - sText           - the text to be displayed in the list box
            syncWithUIthread(listBoxMessages, methodToInvoke, sText);
        }


        private void checkBoxEventOnOff_CheckedChanged(object sender, EventArgs e)
        {
            if (AuthenticDesktop != null)
            {
                if (checkBoxEventOnOff.Checked)
                    AuthenticDesktop.OnDocumentOpened += new
XMLSpyLib._IApplicationEvents_OnDocumentOpenedEventHandler(handleOnDocumentOpened);
                else
                    AuthenticDesktop.OnDocumentOpened -= new
XMLSpyLib._IApplicationEvents_OnDocumentOpenedEventHandler(handleOnDocumentOpened);
            }
        }
    }
}
```

14.3.1.2.3.1 Add Reference to Authentic Desktop API

Add the application's type library as a reference in a .NET project as follows: With the .NET project open, click **Project | Add Reference**. Then browse for the type library, which is called `Authentic Desktop.tlb`, and is located in the Authentic Desktop application folder.

Then declare a variable to access the Authentic Desktop API:

```
    // An instance of Authentic Desktop is accessed via its automation interface.
       XMLSpyLib.Application Authentic Desktop;
```


© 2016-2022 Altova GmbH

14.3.1.2.3.2 Application Startup and Shutdown

In the code snippets below, the methods `StartAuthenticDesktop_Click` and `ShutdownAuthenticDesktop_Click` are those assigned to buttons in the AutomateAuthenticDesktop example that, respectively, start up and shut down the application. This example is located in the C# folder of the API Examples folder (*see the file Form1.cs*):

Windows 7, Windows 8, Windows 10, Windows 11	`C:\Users\<username>\Documents\Altova\Authentic\2023\%APPNAME%>Examples`

You can compile and run the project from within Visual Studio 2012/2013/2015/2017/2019/2022.

Starting Authentic Desktop

The following code snippet from the AutomateAuthenticDesktop example shows how to start up the application.

```
        // Handler for the "Start AuthenticDesktop" button
        private void StartAuthenticDesktop_Click(object sender, EventArgs e)
        {
            if (AuthenticDesktop == null)
            {
                Cursor.Current = Cursors.WaitCursor;

                // If there is no AuthenticDesktop instance, create one and make it
visible.
                AuthenticDesktop = new XMLSpyLib.Application();
                AuthenticDesktop.Visible = true;

                Cursor.Current = Cursors.Default;
            }
            else
            {
                // If an instance of Authentic Desktop is already running, make sure it's
visible
                if (!AuthenticDesktop.Visible)
                    AuthenticDesktop.Visible = true;
            }
        }
```

Shutting down Authentic Desktop

The following code snippet from the AutomateAuthenticDesktop example shows how to shut down the application.

```
        // Handler for the "Shut down AuthenticDesktop" button
        // Shut down application instance by explicitely releasing the COM object.
        private void shutdownAuthenticDesktop_Click(object sender, EventArgs e)
        {
            if (AuthenticDesktop != null)
            {
                // Allow shut down of AuthenticDesktop by releasing the UI
                AuthenticDesktop.Visible = false;
```

```
                // Explicitly release the COM object
                try
                {
                    while
(System.Runtime.InteropServices.Marshal.ReleaseComObject(AuthenticDesktop) > 0) ;
                }
                finally
                {
                    // Avoid subsequent access to this object.
                    AuthenticDesktop = null;
                }
            }
        }
```

14.3.1.2.3.3 Opening Documents

The code snippets below (from the AutomateAuthenticDesktop example) show how two files are opened via two separate methods assigned to two buttons in the user interface. Both methods use the same Application API access mechanism: Documents.OpenFile(string, boolean).

The AutomateAuthenticDesktop example (*see the file Form1.cs*) is located in the C# folder of the API Examples folder:

Windows 7, Windows 8, Windows 10, Windows 11	C:\Users\<username>\Documents\Altova\Authentic\2023\%APPNAME%>Examples

You can compile and run the project from within Visual Studio 2012/2013/2015/2017/2019/2022.

Code snippet

```
       // Handler for the "Open OrgChart.pxf" button
        private void openOrgChart_Click(object sender, EventArgs e)
        {
            // Make sure there's a running Authentic Desktop instance, and that it's
visible
            StartAuthenticDesktop_Click(null, null);

            // Open a sample file installed with the product.
            AuthenticDesktop.Documents.OpenFile(strExamplesFolder + "OrgChart.pxf", false);
        }

        // Handler for the "Open ExpReport.xml" button
        private void openExpReport_Click(object sender, EventArgs e)
        {
            // Make sure there's a running Authentic Desktop instance, and that it's
visible
            StartAuthenticDesktop_Click(null, null);

            // Open a sample file installed with the product.
```

 © 2016-2022 Altova GmbH

```
            AuthenticDesktop.Documents.OpenFile(strExamplesFolder + "ExpReport.xml",
false);
        }
```

The file opened last will be the active file.

14.3.1.2.3.4 Iterating through Open Documents

The code snippet below (from the AutomateAuthenticDesktop example; *see the file Form1.cs*) shows how to iterate through open documents. A condition is then tested within the iteration loop, and the document view is switched between Text View and Authentic View.

```
       // Handler for the "Toggle View Mode" button
        private void toggleView_Click(object sender, EventArgs e)
        {
            // Make sure there's a running Authentic Desktop instance, and that it's
visible
            StartAuthenticDesktop_Click(null, null);

            // Iterate through all open documents and toggle the current view between Text
View and Authentic View.
            foreach (XMLSpyLib.Document doc in AuthenticDesktop.Documents)
                if (doc.CurrentViewMode == XMLSpyLib.SPYViewModes.spyViewAuthentic)
                    doc.SwitchViewMode(XMLSpyLib.SPYViewModes.spyViewBrowser);
                else
                    doc.SwitchViewMode(XMLSpyLib.SPYViewModes.spyViewAuthentic);
        }
```

The AutomateAuthenticDesktop example example is located in the C# folder of the API Examples folder:

Windows 7, Windows 8, Windows 10, Windows 11	C:\Users\<username>\Documents\Altova\Authentic\2023\%APPNAME%>Examples

You can compile and run the project from within Visual Studio 2012/2013/2015/2017/2019/2022.

14.3.1.2.3.5 Errors and COM Output Parameters

The code snippet below (from the AutomateAuthenticDesktop example) shows how to handle errors and COM output parameters. The method AuthenticDesktop.ActiveDocument.IsValid(ref strErrorText, ref nErrorNumber, ref errorData) uses output parameters that are used, in the code snippet below, to generate an error-message text.

The AutomateAuthenticDesktop example (*see the file Form1.cs*) is located in the C# folder of the API Examples folder:

Windows 7, Windows 8, Windows 10, Windows 11	C:\Users\<username>\Documents\Altova\Authentic\2023\%APPNAME%>Examples

You can compile and run the project from within Visual Studio 2012/2013/2015/2017/2019/2022.

Code snippet

```
        // Handler for the "Validate" button
        private void validate_Click(object sender, EventArgs e)
        {
            // COM errors get returned to C# as exceptions. Use a try/catch block to handle
them.
            try
            {
                // Method 'IsValid' is one of the few functions that use output parameters.
                // Use 'object' type for these parameters.
                object strErrorText = "";
                object nErrorNumber = 0;
                object errorData = null;

                if (!AuthenticDesktop.ActiveDocument.IsValid(ref strErrorText, ref
nErrorNumber, ref errorData))
                {
                    // The COM call succeeds but the document is not valid.
                    // A detailed description of the problem is returned in strErrorText,
nErrorNumber and errorData.
                    listBoxMessages.Items.Add("Document " +
AuthenticDesktop.ActiveDocument.Name + " is not valid.");
                    listBoxMessages.Items.Add("\tErrorText  : " + strErrorText);
                    listBoxMessages.Items.Add("\tErrorNumber: " + nErrorNumber);
                    listBoxMessages.Items.Add("\tElement    : " + (errorData != null ?
((XMLSpyLib.XMLData)errorData).TextValue : "null"));
                }
                else
                {
                    // The COM call succeeds and the document is valid.
                    listBoxMessages.Items.Add("Document " +
AuthenticDesktop.ActiveDocument.Name + " is valid.");
                }
            }
            catch (Exception ex)
            {
                // The COM call was not successful.
                // Probably no application instance has been started or no document is
open.
                listBoxMessages.Items.Add("Error validating active document: " +
ex.Message);
            }
        }
```


© 2016-2022 Altova GmbH

14.3.1.2.3.6 Events

The code snippet below (from the AutomateAuthenticDesktop example) lists the code for two event handlers. The AutomateAuthenticDesktop example (*see the file Form1.cs*) is located in the C# folder of the API Examples folder:

Windows 7, Windows 8, Windows 10, Windows 11	C:\Users\<username>\Documents\Altova\Authentic\2023\%APPNAME%>Examples

You can compile and run the project from within Visual Studio 2012/2013/2015/2017/2019/2022.

Code snippet

```
        delegate void addListBoxItem_delegate(string sText);
        // Called from the UI thread
        private void addListBoxItem(string sText)
        {
            listBoxMessages.Items.Add(sText);
        }
        // Wrapper method to call UI control methods from a worker thread
        void syncWithUIthread(Control ctrl, addListBoxItem_delegate methodToInvoke, String
sText)
        {
            // Control.Invoke: Executes on the UI thread, but calling thread waits for
completion before continuing.
            // Control.BeginInvoke: Executes on the UI thread, and calling thread doesn't
wait for completion.
            if (ctrl.InvokeRequired)
                ctrl.BeginInvoke(methodToInvoke, new Object[] { sText });
        }

        // Event handler for OnDocumentOpened event
        private void handleOnDocumentOpened(XMLSpyLib.Document i_ipDocument)
        {
            String sText = "";

            if (i_ipDocument.Name.Length > 0)
                sText = "Document " + i_ipDocument.Name + " was opened!";
            else
                sText = "An empty document was created.";

            // Synchronize the calling thread with the UI thread because
            // COM events are triggered from a working thread
            addListBoxItem_delegate methodToInvoke = new
addListBoxItem_delegate(addListBoxItem);
            // Call syncWithUIthread with the following arguments:
            // 1 - listBoxMessages - list box control to display messages from COM events
            // 2 - methodToInvoke  - a C# delegate which points to the method which will be
called from the UI thread
            // 3 - sText           - the text to be displayed in the list box
            syncWithUIthread(listBoxMessages, methodToInvoke, sText);
```

```
        }


        private void checkBoxEventOnOff_CheckedChanged(object sender, EventArgs e)
        {
            if (AuthenticDesktop != null)
            {
                if (checkBoxEventOnOff.Checked)
                    AuthenticDesktop.OnDocumentOpened += new
XMLSpyLib._IApplicationEvents_OnDocumentOpenedEventHandler(handleOnDocumentOpened);
                else
                    AuthenticDesktop.OnDocumentOpened -= new
XMLSpyLib._IApplicationEvents_OnDocumentOpenedEventHandler(handleOnDocumentOpened);
            }
        }
```

14.3.1.2.4 Java

The Application API can be accessed from Java code. To allow accessing the Authentic Desktop automation server directly from Java code, the libraries listed below must reside in the `classpath`. They are installed in the folder: `JavaAPI` in the Authentic Desktop application folder.

- `AltovaAutomation.dll`: a JNI wrapper for Altova automation servers (`AltovaAutomation_x64.dll` in the case of 64-bit versions)
- `AltovaAutomation.jar`: Java classes to access Altova automation servers
- `AuthenticAPI.jar`: Java classes that wrap the Authentic Desktop automation interface
- `AuthenticAPI_JavaDoc.zip`: a `Javadoc` file containing help documentation for the Java API

Note: In order to use the Java API, the DLL and Jar files must be on the Java Classpath.

Example Java project

An example Java project is supplied with your product installation. You can test the Java project and modify and use it as you like. For more details of the example Java project, see the section, Example Java Project.

Rules for mapping the Application API names to Java

The rules for mapping between the Application API and the Java wrapper are as follows:

- ***Classes and class names***
  For every interface of the Authentic Desktop automation interface a Java class exists with the name of the interface.

- ***Method names***
  Method names on the Java interface are the same as used on the COM interfaces but start with a small letter to conform to Java naming conventions. To access COM properties, Java methods that prefix the property name with `get` and `set` can be used. If a property does not support write-access, no setter method is available. Example: For the `Name` property of the `Document` interface, the Java methods `getName` and `setName` are available.

- ***Enumerations***

© 2016-2022 Altova GmbH

For every enumeration defined in the automation interface, a Java enumeration is defined with the same name and values.

- ***Events and event handlers***
  For every interface in the automation interface that supports events, a Java interface with the same name plus '`Event`' is available. To simplify the overloading of single events, a Java class with default implementations for all events is provided. The name of this Java class is the name of the event interface plus '`DefaultHandler`'. For example:
  `Application`: Java class to access the application
  `ApplicationEvents`: Events interface for the Application
  `ApplicationEventsDefaultHandler`: Default handler for `ApplicationEvents`

Exceptions to mapping rules

There are some exceptions to the rules listed above. These are listed below:

Interface	Java name
`Document`, method `SetEncoding`	`setFileEncoding`
`AuthenticView`, method `Goto`	`gotoElement`
`AuthenticRange`, method `Goto`	`gotoElement`
`AuthenticRange`, method `Clone`	`cloneRange`

This section

This section explains how some basic Authentic Desktop functionality can be accessed from Java code. It is organized into the following sub-sections:

- Example Java Project
- Application Startup and Shutdown
- Simple Document Access
- Iterations
- Use of Out-Parameters
- Event Handlers

14.3.1.2.4.1 Example Java Project

The Authentic Desktop installation package contains an example Java project, located in the the `API\Java` subfolder of the `Examples` folder :

Windows 7, Windows 8, Windows 10, Windows 11	`C:\Users\<username>\Documents\Altova\Authentic\2023\%APPNAME%>Examples`

This folder contains Java examples for the Authentic Desktop API. You can test it directly from the command line using the batch file `BuildAndRun.bat`, or you can compile and run the example project from within Eclipse. See below for instructions on how to use these procedures.

File list

The Java examples folder contains all the files required to run the example project. These files are listed below. If you are using a 64-bit version of the application, some filenames contain `_x64` in the name. These filenames are indicated with `(_x64)`.

`AltovaAutomation(_x64).dll`	Java-COM bridge: DLL part
`AltovaAutomation.jar`	Java-COM bridge: Java library part
`AuthenticAPI.jar`	Java classes of the Authentic Desktop API
`RunAuthenticDesktop.java`	Java example source code
`BuildAndRun.bat`	Batch file to compile and run example code from the command line prompt. Expects folder where Java Virtual Machine resides as parameter.
`.classpath`	Eclipse project helper file
`.project`	Eclipse project file
`Authentic_JavaDoc.zip`	`Javadoc` file containing help documentation for the Java API

What the example does

The example starts up Authentic Desktop and performs a few operations, including opening and closing documents. When done, Authentic Desktop stays open. You must close it manually.

- Start Authentic Desktop: Starts Authentic Desktop, which is registered as an automation server, or activates Authentic Desktop if it is already running.
- Open example files: Locates example documents installed with Authentic Desktop and opens them.
- Iteration and Changing the View Mode: Changes the view of all open documents to Text View. The code also shows how to iterate through open documents.
- Iteration, validation, output parameters: Validates the active document and shows the result in a message box. The code shows how to use output parameters.
- Event Handling: Shows how to handle Authentic Desktop events.
- Shut down Authentic Desktop: Shuts down Authentic Desktop.

You can modify the example in any way you like and run it.

Running the example from the command line

To run the example from the command line, open a command prompt window, go to the Java folder of the API Examples folder (*see above for location*), and then type:

```
buildAndRun.bat "<Path-to-the-Java-bin-folder>"
```

The Java binary folder must be that of a JDK 1.5 or later installation on your computer. Press the **Return** key. The Java source in `RunAuthenticDesktop.java` will be compiled and then executed.

Loading the example in Eclipse

Open Eclipse and use the **Import | Existing Projects into Workspace** command to add the Eclipse project file (`.project`) located in the Java folder of the API Examples folder (*see above for location*). The project

© 2016-2022 Altova GmbH

`RunAuthenticDesktop` will then appear in your Package Explorer or Navigator. Select the project and then the command **Run as | Java Application** to execute the example.

Note: You can select a class name or method of the Java API and press F1 to get help for that class or method.

Java source code listing

The Java source code in the example file `RunAuthenticDesktop.java` is listed below with comments.

```
01 // Access general JAVA-COM bridge classes
02 import com.altova.automation.libs.*;
03
04 // Access AuthenticDesktop Java-COM bridge
05 import com.altova.automation.AuthenticDesktop.*;
06 import com.altova.automation.AuthenticDesktop.Enums.SPYViewModes;
07
08 /**
09  * A simple example that starts AuthenticDesktop COM server and performs a view
operations on it.
10  * Feel free to extend.
11  */
12 public class RunAuthenticDesktop
13 {
14   public static void main(String[] args)
15   {
16     // An instance of the application.
17     Application authenticDesktop = null;
18
19     // Instead of COM error-handling, use Java exception mechanism.
20     try
21     {
22       // Start AuthenticDesktop as COM server.
23       authenticDesktop = new Application();
24       // COM servers start up invisible so we make it visible
25       authenticDesktop.setVisible(true);
26
27       // Locate samples installed with the product.
28       String strExamplesFolder = System.getenv("USERPROFILE") + "\\My Documents\\Altova\
\Authentic2012\\AuthenticExamples\\";
29
30       // Open two files from the product samples.
31       authenticDesktop.getDocuments().openFile(strExamplesFolder + "OrgChart.pxf",
false);
32       authenticDesktop.getDocuments().openFile(strExamplesFolder + "ExpReport.xml",
false);
33
34       // Iterate through all open documents and set the View Mode to 'Text'.
35       for (Document doc:authenticDesktop.getDocuments())
36         if ( doc.getCurrentViewMode() != SPYViewModes.spyViewText)
37           doc.switchViewMode(SPYViewModes.spyViewText);
38
39       // An alternative iteration mode is index-based. COM indices are typically zero-
based.
40       Documents documents = authenticDesktop.getDocuments();
```

```
41       for (int i = 1; i <= documents.getCount(); i++)
42       {
43         Document doc = documents.getItem(i);
44
45         // Validation is one of the few methods that have output parameters.
46         // The class JVariant is the correct type for parameters in these cases.
47         // To get values back mark them with the by-reference flag.
48         JVariant validationErrorText = new JVariant.JStringVariant("");
validationErrorText.setByRefFlag();
49         JVariant validationErrorCount = new JVariant.JIntVariant(0);
validationErrorCount.setByRefFlag();
50         JVariant validationErrorXMLData = new JVariant.JIDispatchVariant(0);
validationErrorXMLData.setByRefFlag();
51         if (!doc.isValid(validationErrorText, validationErrorCount,
validationErrorXMLData))
52           System.out.println("Document " + doc.getName() + " is not wellformed - " +
validationErrorText.getStringValue());
53         else
54           System.out.println("Document " + doc.getName() + " is wellformed.");
55       }
56
57       // The following lines attach to the document events using a default
implementation
58       // for the events and override one of its methods.
59       // If you want to override all document events it is better to derive your
listener class
60       // from DocumentEvents and implement all methods of this interface.
61       Document doc = authenticDesktop.getActiveDocument();
62       doc.addListener(new DocumentEventsDefaultHandler()
63       {
64         @Override
65         public boolean onBeforeCloseDocument(Document i_ipDoc) throws
AutomationException
66         {
67           System.out.println("Document " + i_ipDoc.getName() + " requested closing.");
68
69           // Allow closing of document
70           return true;
71         }
72       });
73       doc.close(true);
74       doc = null;
75
76       System.out.println("Watch AuthenticDesktop!");
77     }
78     catch (AutomationException e)
79     {
80       // e.printStackTrace();
81     }
82     finally
83     {
84       // Make sure that AuthenticDesktop can shut down properly.
85       if (authenticDesktop != null)
86         authenticDesktop.dispose();
87
```

 © 2016-2022 Altova GmbH

```
88       // Since the COM server was made visible and still is visible, it will keep
running
89       // and needs to be closed manually.
90       System.out.println("Now close AuthenticDesktop!");
91     }
92   }
93 }
```

14.3.1.2.4.2 Application Startup and Shutdown

The code listings below show how the application can be started up and shut down.

Application startup

Before starting up the application, the appropriate classes must be imported (*see below*).

```
01 // Access general JAVA-COM bridge classes
02 import com.altova.automation.libs.*;
03
04 // Access AuthenticDesktop Java-COM bridge
05 import com.altova.automation.AuthenticDesktop.*;
06 import com.altova.automation.AuthenticDesktop.Enums.SPYViewModes;
07
08 /**
09  * A simple example that starts AuthenticDesktop COM server and performs a view
operations on it.
10  * Feel free to extend.
11  */
12 public class RunAuthenticDesktop
13 {
14   public static void main(String[] args)
15   {
16     // An instance of the application.
17     Application authenticDesktop = null;
18
19     // Instead of COM error-handling, use Java exception mechanism.
20     try
21     {
22       // Start AuthenticDesktop as COM server.
23       authenticDesktop = new Application();
24       // COM servers start up invisible so we make it visible
25       authenticDesktop.setVisible(true);
26
27 ...
28     }
29   }
30 }
```

Application shutdown

The application can be shut down as shown below.

```
1 {
```

```
2       // Make sure that AuthenticDesktop can shut down properly.
3       if (authenticDesktop != null)
4         authenticDesktop.dispose();
5
6       // Since the COM server was made visible and still is visible, it will keep running
7       // and needs to be closed manually.
8       System.out.println("Now close AuthenticDesktop!");
9 }
```

14.3.1.2.4.3 Simple Document Access

The code listing below shows how to open a document.

```
1  // Locate samples installed with the product.
2  String strExamplesFolder = System.getenv("USERPROFILE") + "\\My Documents\\Altova\
\Authentic2012\\AuthenticExamples\\";
3
4  // Open two files from the product samples.
5  authenticDesktop.getDocuments().openFile(strExamplesFolder + "OrgChart.pxf", false);
6  authenticDesktop.getDocuments().openFile(strExamplesFolder + "ExpReport.xml", false);
```

14.3.1.2.4.4 Iterations

The listing below shows how to iterate through open documents.

```
01 // Iterate through all open documents and set the View mode to 'Text'.
02 for (Document doc:authenticDesktop.getDocuments())
03   if ( doc.getCurrentViewMode() != SPYViewModes.spyViewText)
04           doc.switchViewMode(SPYViewModes.spyViewText);
05
06 // An alternative iteration mode is index-based. COM indices are typically zero-based.
07 Documents documents = authenticDesktop.getDocuments();
08 for (int i = 1; i <= documents.getCount(); i++)
09   {
10     Document doc = documents.getItem(i);
11     ...
12   }
```

14.3.1.2.4.5 Use of Out-Parameters

The code listing below iterates through open documents and validates each of them. For each validation, a message is generated using the output parameters of the Validation method.

```
01 // Iterate through all open documents and set the View mode to 'Text'.
```


© 2016-2022 Altova GmbH

```
02 for (Document doc:authenticDesktop.getDocuments())
03   if ( doc.getCurrentViewMode() != SPYViewModes.spyViewText)
04        doc.switchViewMode(SPYViewModes.spyViewText);
05
06 // An alternative iteration mode is index-based. COM indices are typically zero-based.
07 Documents documents = authenticDesktop.getDocuments();
08 for (int i = 1; i <= documents.getCount(); i++)
09 {
10   Document doc = documents.getItem(i);
11
12 // Validation is one of the few methods that have output parameters.
13 // The class JVariant is the correct type for parameters in these cases.
14 // To get values back, mark them with the by-reference flag.
15 JVariant validationErrorText = new JVariant.JStringVariant("");
validationErrorText.setByRefFlag();
16 JVariant validationErrorCount = new JVariant.JIntVariant(0);
validationErrorCount.setByRefFlag();
17 JVariant validationErrorXMLData = new JVariant.JIDispatchVariant(0);
validationErrorXMLData.setByRefFlag();
18   if (!doc.isValid(validationErrorText, validationErrorCount, validationErrorXMLData))
19     System.out.println("Document " + doc.getName() + " is not wellformed - " +
validationErrorText.getStringValue());
20   else
21     System.out.println("Document " + doc.getName() + " is wellformed.");
22 }
```

14.3.1.2.4.6 Event Handlers

The listing below shows how to listen for and use events.

```
01 // The following lines attach to the document events using a default implementation
02 // for the events and override one of its methods.
03 // If you want to override all document events, it is better to derive your listener
class
04 // from DocumentEvents and implement all methods of this interface.
05 Document doc = authenticDesktop.getActiveDocument();
06 doc.addListener(new DocumentEventsDefaultHandler()
07 {
08   @Override
09   public boolean onBeforeCloseDocument(Document i_ipDoc) throws AutomationException
10   {
11     System.out.println("Document " + i_ipDoc.getName() + " requested closing.");
12
13     // allow closing of document
14     return true;
15   }
16 });
17 doc.close(true);
18 doc = null;
```

14.3.2 Interfaces

Object Hierarchy

Application
- SpyProject
  - SpyProjectItems
    - SpyProjectItem
- Documents
  - Document
    - GridView
    - AuthenticView
      - AuthenticRange
      - AuthenticDataTransfer (previously DocEditDataTransfer)
      - AuthenticDataTransfer (previously DocEditDataTransfer)
    - TextView
    - XMLData
- Dialogs
  - CodeGeneratorDlg
  - FileSelectionDlg
  - SchemaDocumentationDlg
  - GenerateSampleXMLDlg
  - DTDSchemaGeneratorDlg
  - FindInFilesDlg
- DatabaseConnection
- ExportSettings
- TextImportExportSettings
- ElementList
  - ElementListItem

Enumerations

Description

This chapter contains the reference of the Authentic Desktop 1.5 Type Library.

Most of the given examples are written in VisualBasic. These code snippets assume that there is a variable defined and set, called **objSpy of type Application**. There are also some code samples written in JavaScript.

14.3.2.1 Application

Methods

GetDatabaseImportElementList
GetDatabaseSettings
GetDatabaseTables
ImportFromDatabase
CreateXMLSchemaFromDBStructure

GetTextImportElementList
GetTextImportExportSettings
ImportFromText

© 2016-2022 Altova GmbH

ImportFromWord

ImportFromSchema

GetExportSettings

NewProject
OpenProject

AddMacroMenuItem
ClearMacroMenu

ShowForm

ShowApplication

URLDelete
URLMakeDirectory

AddXSLT_XQParameter
GetXSLT_XQParameterCount
GetXSLT_XQParameterName
GetXSLT_XQParameterXPath
RemoveXSLT_XQParameter

FindInFiles

Quit

Properties
Application
Parent

ActiveDocument
Documents

CurrentProject

Dialogs

WarningNumber
WarningText

Status
MajorVersion
MinorVersion
Edition
IsAPISupported
ServicePackVersion

Description

Application is the root for all other objects. It is the only object you can create by CreateObject (VisualBasic) or other similar COM related functions.

Example

```
Dim objSpy As Application
Set objSpy = CreateObject("XMLSpy.Application")
```

14.3.2.1.1 Events

14.3.2.1.1.1 OnBeforeOpenDocument

Event: OnBeforeOpenDocument(*objDialog* as FileSelectionDlg)

Description
This event gets fired whenever a document gets opened via the OpenFile or OpenURL menu command. It is sent after a document file has been selected but before the document gets opened. The file selection dialog object is initialized with the name of the selected document file. You can modify this selection. To continue the opening of the document leave the FileSelectionDlg.DialogAction property of *io_objDialog* at its default value spyDialogOK. To abort the opening of the document set this property to spyDialogCancel.

Examples
Given below are examples of how this event can be scripted.

XMLSpy scripting environment - VBScript:

```
Function On_BeforeOpenDocument(objDialog)
End Function
```

XMLSpy scripting environment - JScript:

```
function On_BeforeOpenDocument(objDialog)
{
}
```

XMLSpy IDE Plugin:

```
IXMLSpyPlugIn.OnEvent (26, ...) // nEventId = 26
```

14.3.2.1.1.2 OnBeforeOpenProject

Event: OnBeforeOpenProject(*objDialog* as FileSelectionDlg)

Description
This event gets fired after a project file has been selected but before the project gets opened. The file selection dialog object is initialized with the name of the selected project file. You can modify this selection. To continue the opening of the project leave the FileSelectionDlg.DialogAction property of *io_objDialog* at its default value spyDialogOK. To abort the opening of the project set this property to spyDialogCancel.

Examples

© 2016-2022 Altova GmbH

Given below are examples of how this event can be scripted.

XMLSpy scripting environment - VBScript:
Function On_BeforeOpenProject(*objDialog*)
End Function

XMLSpy scripting environment - JScript:
function On_BeforeOpenProject(*objDialog*)
{
}

XMLSpy IDE Plugin:
IXMLSpyPlugIn.OnEvent (25, ...) // nEventId = 25

14.3.2.1.1.3 OnDocumentOpened

Event: OnDocumentOpened(*objDocument* as Document)

Description
This event gets fired whenever a document opens in Authentic Desktop. This can happen due to opening a file with the OpenFile or OpenURL dialog, creating a new file or dropping a file onto Authentic Desktop. The new document gets passed as parameter. The operation cannot be canceled.

Examples
Given below are examples of how this event can be scripted.

XMLSpy scripting environment - VBScript:
Function On_OpenDocument(*objDocument*)
End Function

XMLSpy scripting environment - JScript:
function On_OpenDocument(*objDocument*)
{
}

XMLSpy IDE Plugin:
IXMLSpyPlugIn.OnEvent (7, ...) // nEventId = 7

14.3.2.1.1.4 OnProjectOpened

Event: OnProjectOpened(*objProject* as SpyProject)

Description
This event gets fired whenever a project gets opened in Authentic Desktop. The new project gets passed as parameter.

Examples
Given below are examples of how this event can be scripted.

XMLSpy scripting environment - VBScript:
Function On_OpenProject(*objProject*)
End Function

XMLSpy scripting environment - JScript:
function On_OpenProject(*objProject*)
{
}

XMLSpy IDE Plugin:
IXMLSpyPlugIn.OnEvent (6, ...) // nEventId = 6

14.3.2.1.2 ActiveDocument

Property: ActiveDocument as Document

Description
Reference to the active document. If no document is open, ActiveDocument is null (nothing).

Errors
1111 The application object is no longer valid.
1100 Invalid address for the return parameter was specified.

14.3.2.1.3 AddMacroMenuItem

Method: AddMacroMenuItem(*strMacro* as String,*strDisplayText* as String)

Description
Adds a menu item to the **Tools** menu. This new menu item invokes the macro defined by strMacro. See also Example Scripting Project.

Errors
1111 The application object is no longer valid.
1100 Invalid parameter or invalid address for the return parameter was specified.
1108 Number of macro items is limited to 16 items.

14.3.2.1.4 AddXSLT_XQParameter

Method: AddXSLT_XQParameter(name as String, XPath as String)

Description
Adds an XSLT or XQuery parameter. The parameter's name and value are the two arguments of the method.

© 2016-2022 Altova GmbH

Errors

1111 The application object is no longer valid.
1100 Invalid address for the return parameter was specified.
1124 The XPath expression is not set.
1125 Not a QName.
1126 The specified XPath is not valid. Reason for invalidity appended.
1127 A parameter with the submitted name already exists.

14.3.2.1.5 Application

Property: Application as Application (read-only)

Description
Accesses the Authentic Desktop application object.

Errors

1111 The application object is no longer valid.
1100 Invalid address for the return parameter was specified.

14.3.2.1.6 ClearMacroMenu

Method: ClearMacroMenu()

Return Value
None

Description
Removes from the **Tools** menu those menu items that were added by calling `AddMacroMenuItem`. See also Example Scripting Project.

Errors

1111 The application object is no longer valid.

14.3.2.1.7 CreateXMLSchemaFromDBStructure

Method: CreateXMLSchemaFromDBStructure(pImportSettings as DatabaseConnection, pTables as ElementList)

Description
CreateXMLSchemaFromDBStructure creates from a database specified in pImportSettings for the defined tables in pTables new XML Schema document(s) describing the database tables structure.

The parameter pTables specifies which table structures the XML Schema document should contain. This parameter can be NULL, specifying that all table structures will be exported.

See also GetDataBaseTables.

Errors
1112 Invalid database specified.
1120 Database import failed.

14.3.2.1.8 CurrentProject

Property: CurrentProject as SpyProject

Description
Reference to the active document. If no project is open, CurrentProject is null (nothing).

Errors
1111 The application object is no longer valid.
1100 Invalid address for the return parameter was specified.

14.3.2.1.9 Dialogs

Property: Dialogs as Dialogs (read-only)

Description
Access the built-in dialogs of Authentic Desktop.

Errors
1111 The application object is no longer valid.
1100 Invalid address for the return parameter was specified.

14.3.2.1.10 Documents

Property: Documents as Documents

Description
Collection of all open documents.

Errors
1111 The application object is no longer valid.
1100 Invalid address for the return parameter was specified.

14.3.2.1.11 Edition

Property: Edition as String

Description

© 2016-2022 Altova GmbH

Returns the edition of the application, for example `Altova Authentic Desktop Enterprise Edition` for the Enterprise edition.

Errors

1111 The application object is no longer valid.
1100 Invalid address for the return parameter was specified.

14.3.2.1.12 FindInFiles

Method: FindInFiles(*pSettings* as FindInFilesDlg) as FindInFilesResults

Description

Returns a FindInFilesResults object containing information about the files that matched the specified settings.

Errors

1111 The application object is no longer valid.
1100 Invalid address for the return parameter was specified.

14.3.2.1.13 GetDatabaseImportElementList

Method: GetDatabaseImportElementList(*pImportSettings* as DatabaseConnection) as ElementList

Description

The function returns a collection of ElementListItems where the properties ElementListItem.Name contain the names of the fields that can be selected for import and the properties ElementListItem.ElementKind are initialized either to *spyXMLDataAttr* or *spyXMLDataElement*, depending on the value passed in DatabaseConnection.AsAttributes. This list serves as a filter to what finally gets imported by a future call to ImportFromDatabase. Use ElementList.RemoveElement to exclude fields from import.

Properties mandatory to be filled out for the database connection are one of DatabaseConnection.File, DatabaseConnection.ADOConnection and DatabaseConnection.ODBCConnection, as well as DatabaseConnection.SQLSelect. Use the property DatabaseConnection.AsAttributes to initialize ElementListItem.ElementKind of the resulting element list to either *spyXMLDataAttr* or *spyXMLDataElement*, respectively.

Example

See example at ImportFromDatabase.

Errors

1111 The application object is no longer valid.
1100 Invalid parameter or invalid address for the return parameter was specified.
1107 Import from database failed.
1112 Invalid database specified.
1114 Select statement is missing.
1119 database element list import failed.

14.3.2.1.14 GetDatabaseSettings

Method: GetDatabaseSettings() as DatabaseConnection

Description
GetDatabaseSettings creates a new object of database settings. The object is used to specify database connection parameters for the methods GetDatabaseTables, GetDatabaseImportElementList, ImportFromDatabase, ImportFromSchema and ExportToDatabase.

Example
See example of ImportFromDatabase.

Errors
1111 The application object is no longer valid.
1100 Invalid address for the return parameter was specified.

14.3.2.1.15 GetDatabaseTables

Method: GetDatabaseTables(*pImportSettings* as DatabaseConnection) as ElementList

Description
GetDatabaseTables reads the table names from the database specified in *pImportSettings*. Properties mandatory to be filled out for the database connection are one of DatabaseConnection.File, DatabaseConnection.ADOConnection and DatabaseConnection.ODBCConnection. All other properties are ignored.
The function returns a collection of ElementListItems where the properties ElementListItem.Name contain the names of tables stored in the specified database. The remaining properties of ElementListItem are unused.

Errors
1111 The application object is no longer valid.
1100 Invalid parameter or invalid address for the return parameter was specified.
1112 Invalid database specified.
1113 Error while reading database table information.
1118 Database table query failed.

Example

```
        Dim objImpSettings As DatabaseConnection
Set objImpSettings = objSpy.GetDatabaseSettings
objImpSettings.ADOConnection = TxtADO.Text

'store table names in list box
ListTables.Clear

Dim objList As ElementList
Dim objItem As ElementListItem
On Error GoTo ErrorHandler
Set objList = objSpy.GetDatabaseTables(objImpSettings)

        For Each objItem In objList
```


© 2016-2022 Altova GmbH

```
            ListTables.AddItem objItem.Name
    Next
```

14.3.2.1.16 GetExportSettings

Method: GetExportSettings()as ExportSettings (read-only)

Description
GetExportSettings creates a new object of common export settings. This object is used to pass the parameters to the export functions and defines the behaviour of the export calls. See also the export functions from Document.

Errors
1111 The application object is no longer valid.
1100 Invalid address for the return parameter was specified.

14.3.2.1.17 GetTextImportElementList

Method: GetTextImportElementList(pImportSettings as TextImportExportSettings) as ElementList

Description
GetTextImportElementList retrieves importing information about the text-file as specified in pImportSettings. The function returns a collection of ElementListItems where the properties ElementListItem.Name contain the names of the fields found in the file. The values of remaining properties are undefined.

If the text-file does not contain a column header, set `pImportSettings.HeaderRow` to `false`. The resulting element list will contain general column names like 'Field1' and so on.

Errors
1111 The application object is no longer valid.
1100 Invalid parameter or invalid address for the return parameter was specified.
1107 Import from database failed.
1115 Error during text element list import. Cannot create parser for import file.
1116 Error during text element list import.

Example

```
' ------------------------------------------------------
' VBA client code fragment - import selected fields from text file
' ------------------------------------------------------
        Dim objImpSettings As TextImportExportSettings
        Set objImpSettings = objSpy.GetTextImportExportSettings

        objImpSettings.ImportFile = "C:\ImportMe.txt"
        objImpSettings.HeaderRow = False

        Dim objList As ElementList
        Set objList = objSpy.GetTextImportElementList(objImpSettings)

        'exclude first column
```

```
objList.RemoveItem 1

Dim objImpDoc As Document
On Error Resume Next
Set objImpDoc = objSpy.ImportFromText(objImpSettings, objList)
CheckForError
```

14.3.2.1.18 GetTextImportExportSettings

Method: GetTextImportExportSettings() as TextImportExportSettings (read-only)

Description
GetTextImportExportSettings creates a new object of common import and export settings for text files. See also the example for Application.GetTextImportElementList.

Errors
1111 The application object is no longer valid.
1100 Invalid address for the return parameter was specified.

14.3.2.1.19 GetXSLT_XQParameterCount

Method: GetXSLT_XQParameterCount() as Long

Description
Returns the number of XSLT and XQuery parameters.

Errors
1111 The application object is no longer valid.
1100 Invalid address for the return parameter was specified.

14.3.2.1.20 GetXSLT_XQParameterName

Method: GetXSLT_XQParameterName(index as Long) as String

Description
Returns the name of the XSLT or XQuery parameter identified by the supplied index.

Errors
1111 The application object is no longer valid.
1100 Invalid address for the return parameter was specified.

© 2016-2022 Altova GmbH

14.3.2.1.21 GetXSLT_XQParameterXPath

Method: GetXSLT_XQParameterXPath(index as Long) as String

Description
Returns the XPath expression of the XSLT or XQuery parameter identified by the supplied index.

Errors
1111 The application object is no longer valid.
1100 Invalid address for the return parameter was specified.

14.3.2.1.22 ImportFromDatabase

Method: ImportFromDatabase(pImportSettings as DatabaseConnection,pElementList as ElementList) as Document

Return Value
Creates a new document containing the data imported from the database.

Description
ImportFromDatabase imports data from a database as specified in pImportSettings and creates a new document containing the data imported from the database. Properties mandatory to be filled out are one of DatabaseConnection.File, DatabaseConnection.ADOConnection or DatabaseConnection.ODBCConnection and DatabaseConnection.SQLSelect. Additionally, you can use DatabaseConnection.AsAttributes, DatabaseConnection.ExcludeKeys, DatabaseConnection.IncludeEmptyElements and NumberDateTimeFormat to further parameterize import.

The parameter pElementList specifies which fields of the selected data gets written into the newly created document, and which are created as elements and which as attributes. This parameter can be NULL, specifying that all selected fields will be imported as XML elements.

See GetDatabaseSettings and GetDatabaseImportElementList for necessary steps preceding any import of data from a database.

Errors
1111 The application object is no longer valid.
1100 Invalid parameter or invalid address for the return parameter was specified.
1107 Import from database failed.
1112 Invalid database specified.
1114 Select statement is missing.
1117 Transformation to XML failed.
1120 Database import failed.

Example

```
Dim objImpSettings As DatabaseConnection
Set objImpSettings = objSpy.GetDatabaseSettings
```

```
        objImpSettings.ADOConnection = strADOConnection
        objImpSettings.SQLSelect = "SELECT * FROM MyTable"

        Dim objDoc As Document
        On Error Resume Next
        Set objDoc = objSpy.ImportFromDatabase(objImpSettings,
objSpy.GetDatabaseImportElementList(objImpSettings))
        ' CheckForError here
```

14.3.2.1.23 ImportFromSchema

Method: ImportFromSchema(*pImportSettings* as DatabaseConnection,*strTable* as String,*pSchemaDoc* as Document) as Document

Return Value
Creates a new document filled with data from the specified database as specified by the schema definition in *pSchemaDoc*.

Description
ImportFromSchema imports data from a database specified in pImportSettings. Properties mandatory to be filled out are one of DatabaseConnection.File, DatabaseConnection.ADOConnection or DatabaseConnection.ODBCConnection. Additionally, you can use DatabaseConnection.AsAttributes, DatabaseConnection.ExcludeKeys and NumberDateTimeFormat to further parameterize import. All other properties get ignored.

ImportFromSchema does not use an explicit SQL statement to select the data. Instead, it expects a structure definition of the document to create in form of an XML schema document in *pSchemaDoc*. From this definition the database select statement is automatically deduced. Specify in *strTable* the table name of the import root that will become the root node in the new document.

See GetDatabaseSettings and GetDatabaseTables for necessary steps preceding an import from a database based on a schema definition. To create the schema definition file use command 'create database schema' from the 'convert' menu of Authentic Desktop.

Errors

1111	The application object is no longer valid.
1100	Invalid parameter or invalid address for the return parameter was specified.
1107	Import from database failed.
1112	Invalid database specified.
1120	Database import failed.
1121	Could not create validator for the specified schema.
1122	Failed parsing schema for database import.

14.3.2.1.24 ImportFromText

Method: ImportFromText(*pImportSettings* as TextImportExportSettings,*pElementList* as ElementList) as Document

Description

© 2016-2022 Altova GmbH

ImportFromText imports the text file as specified in pImportSettings. The parameter pElementList can be used as import filter. Either pass the list returned by a previous call to GetTextImportElementList or null to import all columns. To avoid import of unnecessary columns use ElementList.RemoveElement to remove the corresponding field names from `pElementList` before calling `ImportFromText`.
The method returns the newly created document containing the imported data. This document is the same as the active document of Authentic Desktop.

Errors

1111 The application object is no longer valid.
1100 Invalid parameter or invalid address for the return parameter was specified.
1107 Import from text file failed.
1117 Transformation to XML failed.

Example

```
' ------------------------------------------------------
' VBA client code fragment - import from text file
' ------------------------------------------------------
        Dim objImpSettings As TextImportExportSettings
        Set objImpSettings = objSpy.GetTextImportExportSettings

        objImpSettings.ImportFile = strFileName
        objImpSettings.HeaderRow = False

        Dim objImpDoc As Document
        On Error Resume Next
        Set objImpDoc = objSpy.ImportFromText(objImpSettings,
objSpy.GetTextImportElementList(objImpSettings))

        CheckForError
```

14.3.2.1.25 ImportFromWord

Method: ImportFromWord(*strFile* as String) as Document

Description
ImportFromWord imports the MS-Word Document strFile into a new XML document.

Errors

1111 The application object is no longer valid.
1100 Invalid parameter or invalid address for the return parameter was specified.
Import from document failed.

14.3.2.1.26 IsAPISupported

Property: IsAPISupported as Boolean

Description
Returns whether the API is supported in this version or not.

Errors
1111 The application object is no longer valid.
1100 Invalid address for the return parameter was specified.

14.3.2.1.27 MajorVersion

Property: MajorVersion as Integer

Description
Returns the application version's major number, for example `15` for 2013 versions, and `16` for 2014 versions..

Errors
1111 The application object is no longer valid.
1100 Invalid address for the return parameter was specified.

14.3.2.1.28 MinorVersion

Property: MinorVersion as Integer

Description
Returns the application version's minor number.

Errors
1111 The application object is no longer valid.
1100 Invalid address for the return parameter was specified.

14.3.2.1.29 NewProject

Method: NewProject(*strPath* as String,*bDiscardCurrent* as Boolean)

Description
NewProject creates a new project.

If there is already a project open that has been modified and bDiscardCurrent is false, then NewProject() fails.

Errors
1111 The application object is no longer valid.
1102 A project is already open but *bDiscardCurrent* is *true*.
1103 Creation of new project failed.

14.3.2.1.30 OpenProject

Method: OpenProject(*strPath* as String,*bDiscardCurrent* as Boolean,*bDialog* as Boolean)

Parameters

© 2016-2022 Altova GmbH

strPath
Path and file name of the project to open. Can be empty if bDialog is true.

bDiscardCurrent
Discard currently open project and possibly lose changes.

bDialog
Show dialogs for user input.

Return Value
None

Description
OpenProject opens an existing project. If there is already a project open that has been modified and bDiscardCurrent is false, then OpenProject() fails.

Errors
1111 The application object is no longer valid.
1100 Invalid parameter or invalid address for the return parameter was specified.
1101 Cannot open specified project.
1102 A project is already open but *bDiscardCurrent* is *true*.

14.3.2.1.31 Parent

Property: Parent as Application (read-only)

Description
Accesses the Authentic Desktop application object.

Errors
1111 The application object is no longer valid.
1100 Invalid address for the return parameter was specified.

14.3.2.1.32 Quit

Method: Quit()

Return Value
None

Description
This method terminates Authentic Desktop. All modified documents will be closed without saving the changes. This is also true for an open project.

If Authentic Desktop was automatically started as an automation server by a client program, the application will not shut down automatically when your client program shuts down if a project or any document is still open. Use the Quit method to ensure automatic shut-down.

Errors

1111 The application object is no longer valid.

14.3.2.1.33 ReloadSettings

Method: ReloadSettings

Return Value

Description
The application settings are reloaded from the registry.

Available with TypeLibrary version 1.5

Errors
1111 The application object is no longer valid.

14.3.2.1.34 RemoveXSLT_XQParameter

Method: RemoveXSLT_XQParameter(index as Long)

Description
Removes the XSLT or XQuery parameter identified by the supplied index.

Errors
1111 The application object is no longer valid.
1100 Invalid address for the return parameter was specified.

14.3.2.1.35 RunMacro

Method: RunMacro(*strMacro* as String)

Return Value

Description
Calls the specified macro either from the project scripts (if present) or from the global scripts.

Available with TypeLibrary version 1.5

Errors
1111 The application object is no longer valid.

14.3.2.1.36 ScriptingEnvironment

Property: ScriptingEnvironment as IUnknown (read-only)

© 2016-2022 Altova GmbH

Description
Reference to any active scripting environment. This property makes it possible to access the TypeLibrary of the XMLSpyFormEditor.exe application which is used as the current scripting environment.

Available with TypeLibrary version 1.5

Errors
1111 The application object is no longer valid.
1100 Invalid address for the return parameter was specified.

14.3.2.1.37 ServicePackVersion

Property: ServicePackVersion as Long

Description
Returns the Service Pack version number of the application. Eg: 1 for 2010 R2 SP1

Errors
1111 The application object is no longer valid.
1100 Invalid address for the return parameter was specified.

14.3.2.1.38 ShowApplication

Method: ShowApplication(*bShow* as Boolean)

Return Value
None

Description
The method shows (bShow = True) or hides (bShow = False) Authentic Desktop.

Errors
1110 The application object is no longer valid.

14.3.2.1.39 ShowFindInFiles

Method: ShowFindInFiles(pSettings as FindInFilesDlg) as Boolean

Return Value
Returns false if the user pressed the Cancel button, true otherwise.

Description
Displays the FindInFiles dialog preset with the given settings. The user modifications of the settings are stored in the passed dialog object.

Errors
1111 The application object is no longer valid.
1100 Invalid parameter or invalid address for the return parameter was specified.

14.3.2.1.40 ShowForm

Method: ShowForm(*strFormName* as String) as Long

Return Value
Returns zero if the user pressed a Cancel button or the form calls TheView.Cancel().

Description
Displays the form strFormName.

Forms, event handlers and macros can be created with the Scripting Environment. Select "Switch to scripting environment" from the **Tools** menu to invoke the Scripting Environment.

Errors
1111 The application object is no longer valid.
1100 Invalid parameter or invalid address for the return parameter was specified.

14.3.2.1.41 Status

Property: Status as ENUMApplicationStatus

Description
Returns the current status of the running application.

Errors
1111 The application object is no longer valid.
1100 Invalid address for the return parameter was specified.

14.3.2.1.42 URLDelete

Method: URLDelete(*strURL* as String,*strUser* as String,*strPassword* as String)

Return Value
None

Description
The method deletes the file at the URL strURL.

Errors
1111 The application object is no longer valid.
1109 Error deleting file at specified URL.

14.3.2.1.43 URLMakeDirectory

Method: URLMakeDirectory(*strURL* as String,*strUser* as String,*strPassword* as String)

Return Value

© 2016-2022 Altova GmbH

None

Description
The method creates a new directory at the URL strURL.

Errors
1111 The application object is no longer valid.
1100 Invalid parameter specified.

14.3.2.1.44 Visible

Property: Visible as VARIANT_BOOL

Description
Sets or gets the visibility attribute of Authentic Desktop. This standard automation property makes usage of ShowApplication obsolete.

Errors
1110 The application object is no longer valid.
1100 Invalid address for the return parameter was specified.

14.3.2.1.45 WarningNumber

Property: WarningNumber as integer

Description
Some methods fill the property WarningNumber with additional information if an error occurs.

Currently just Documents.OpenFile fills this property.

Errors
1111 The application object is no longer valid.
1100 Invalid address for the return parameter was specified.

14.3.2.1.46 WarningText

Property: WarningText as String

Description
Some methods fill the property WarningText with additional information if an error occurs.

Currently just Documents.OpenFile fills this property.

Errors
1111 The application object is no longer valid.
1100 Invalid address for the return parameter was specified.

14.3.2.2 AuthenticContextMenu

The context menu interface provides the means for the user to customize the context menus shown in Authentic. The interface has the methods listed in this section.

14.3.2.2.1 CountItems

Method: CountItems `()` nItems as long

Return Value
Returns the number of menu items.

Errors
2501 Invalid object.

14.3.2.2.2 DeleteItem

Method: DeleteItem `(IndexPosition as long)`

Return Value
Deletes the menu item that has the index position submitted in the first parameter.

Errors
2501 Invalid object
2502 Invalid index

14.3.2.2.3 GetItemText

Method: GetItemText `(IndexPosition as long)` MenuItemName as string

Return Value
Gets the name of the menu item located at the index position submitted in the first parameter.

Errors
2501 Invalid object
2502 Invalid index

© 2016-2022 Altova GmbH

14.3.2.2.4 InsertItem

Method: InsertItem `(IndexPosition as long, MenuItemName as string, MacroName as string)`

Return Value
Inserts a user-defined menu item at the position in the menu specified in the first parameter and having the name submitted in the second parameter. The menu item will start a macro, so a valid macro name must be submitted.

Errors
2501 Invalid object
2502 Invalid index
2503 No such macro
2504 Internal error

14.3.2.2.5 SetItemText

Method: SetItemText `(IndexPosition as long, MenuItemName as string)`

Return Value
Sets the name of the menu item located at the index position submitted in the first parameter.

Errors
2501 Invalid object
2502 Invalid index

14.3.2.3 AuthenticDataTransfer

Renamed from DocEditDataTransfer to AuthenticDataTransfer

The DocEditView object is renamed to OldAuthenticView.
DocEditSelection is renamed to AuthenticSelection.
DocEditEvent is renamed to AuthenticEvent.
DocEditDataTransfer is renamed to AuthenticDataTransfer.

Their usage—except for AuthenticDataTransfer—is no longer recommended. We will continue to support existing functionality for a yet undefined period of time but no new features will be added to these interfaces.

For examples on migrating from DocEdit to Authentic see the description of the different methods and properties of the different DocEdit objects.

Methods

getData

Properties

dropEffect
ownDrag
type

Description

The events OnDragOver and OnBeforeDrop provide information about the object being dragged with an instance of type AuthenticDataTransfer. It contains a description of the dragged object and its content. The latter is available either as string or a pointer to a COM object supporting the IUnkown interface.

14.3.2.3.1 dropEffect

Property: dropEffect as long

Description
The property stores the drop effect from the default event handler. You can set the drop effect if you change this value and return TRUE for the event handler.

Errors
2101 Invalid address for the return parameter was specified.

14.3.2.3.2 getData

Method: getData() as Variant

Description
Retrieve the data associated with the dragged object. Depending on AuthenticDataTransfer.type, that data is either a string or a COM interface pointer of type IUnknown.

Errors
2101 Invalid address for the return parameter was specified.

14.3.2.3.3 ownDrag

Property: ownDrag as Boolean (read-only)

Description
The property is TRUE if the current dragging source comes from inside Authentic View.

Errors
2101 Invalid address for the return parameter was specified.

© 2016-2022 Altova GmbH

14.3.2.3.4 type

Property: type as String (read-only)

Description
Holds the type of data you get with the DocEditDataTransfer.getData method.

Currently supported data types are:

OWN	data from Authentic View itself
TEXT	plain text
UNICODETEXT	plain text as UNICODE

Errors
2101 Invalid address for the return parameter was specified.

14.3.2.4 AuthenticEventContext

The `EventContext` interface gives access to many properties of the context in which a macro is executed.

14.3.2.4.1 EvaluateXPath

Method: EvaluateXPath (strExpression as string) as strValue as string

Return Value
The method evaluates the XPath expression in the context of the node within which the event was triggered and returns a string.

Description
`EvaluateXPath()` executes an XPath expression with the given event context. The result is returned as a string, in the case of a sequence it is a space-separated string.

Errors
2201 Invalid object.
2202 No context.
2209 Invalid parameter.
2210 Internal error.
2211 XPath error.

14.3.2.4.2 GetEventContextType

Method: GetEventContextType `()` Type as `AuthenticEventContextType` enumeration

Return Value
Returns the context node type.

Description
`GetEventContextType` allows the user to determine whether the macro is in an XML node or in an XPath atomic item context. The enumeration `AuthenticEventContextType` is defined as follows:

authenticEventContextXML,
authenticEventContextAtomicItem,
authenticEventContextOther

If the context is a normal XML node, the `GetXMLNode()` function gives access to it (returns `NULL` if not).

Errors
2201 Invalid object.
2202 No context.
2209 Invalid parameter.

14.3.2.4.3 GetNormalizedTextValue

Method: GetNormalizedTextValue`()` strValue as string

Return Value
Returns the value of the current node as string

Errors
2201 Invalid object.
2202 No context.
2203 Invalid context
2209 Invalid parameter.

14.3.2.4.4 GetVariableValue

Method: GetVariableValue`(strName as string) strValue as string`

Return Value
Gets the value of the variable submitted as the parameter.

Description
`GetVariableValue` gets the variable's value in the scope of the context.

```
nZoom = parseInt( AuthenticView.EventContext.GetVariableValue( 'Zoom' ) );
if ( nZoom > 1 )
{
    AuthenticView.EventContext.SetVariableValue( 'Zoom', nZoom - 1 );
}
```

Errors
2201 Invalid object.
2202 No context.

© 2016-2022 Altova GmbH

2204 No such variable in scope
2205 Variable cannot be evaluated
2206 Variable returns sequence
2209 Invalid parameter

14.3.2.4.5 GetXMLNode

Method: GetXMLNode`()` Node as XMLData object

Return Value
Returns the context XML node or `NULL`

Errors
2201 Invalid object.
2202 No context.
2203 Invalid context
2209 Invalid parameter.

14.3.2.4.6 IsAvailable

Method: IsAvailable`()` as Boolean

Return Value
Returns true if `EventContext` is set, false otherwise.

Errors
2201 Invalid object.

14.3.2.4.7 SetVariableValue

Method: SetVariableValue`(strName as string, strValue as string)`

Return Value
Sets the value (second parameter) of the variable submitted in the first parameter.

Description
`SetVariableValue` sets the variable's value in the scope of the context.

```
nZoom = parseInt( AuthenticView.EventContext.GetVariableValue( 'Zoom' ) );
if ( nZoom > 1 )
{
    AuthenticView.EventContext.SetVariableValue( 'Zoom', nZoom - 1 );
```

}

Errors

2201	Invalid object.
2202	No context.
2204	No such variable in scope
2205	Variable cannot be evaluated
2206	Variable returns sequence
2207	Variable read-only
2208	No modification allowed

14.3.2.5 AuthenticRange

The first table lists the properties and methods of AuthenticRange that can be used to navigate through the document and select specific portions.

Properties	Methods	
Application	Clone	MoveBegin
FirstTextPosition	CollapsToBegin	MoveEnd
FirstXMLData	CollapsToEnd	NextCursorPosition
FirstXMLDataOffset	ExpandTo	PreviousCursorPosition
LastTextPosition	Goto	Select
LastXMLData	GotoNext	SelectNext
LastXMLDataOffset	GotoPrevious	SelectPrevious
Parent	IsEmpty	SetFromRange
	IsEqual	

The following table lists the content modification methods, most of which can be found on the right/button mouse menu.

Properties	Edit operations	Dynamic table operations
Text	Copy	AppendRow
	Cut	DeleteRow
	Delete	DuplicateRow
	IsCopyEnabled	InsertRow
	IsCutEnabled	IsFirstRow
	IsDeleteEnabled	IsInDynamicTable
	IsPasteEnabled	IsLastRow
	Paste	MoveRowDown
		MoveRowUp

The following methods provide the functionality of the Authentic entry helper windows for range objects.

Operations of the entry helper windows

Elements	Attributes	Entities

© 2016-2022 Altova GmbH

CanPerformActionWith
CanPerformAction
PerformAction
GetElementAttributeValue
GetElementAttributeNames
GetElementHierarchy
HasElementAttribute
IsTextStateApplied
SetElementAttributeValue
GetEntityNames
InsertEntity

Description
AuthenticRange objects are the 'cursor' selections of the automation interface. You can use them to point to any cursor position in the Authentic view, or select a portion of the document. The operations available for AuthenticRange objects then work on this selection in the same way, as the corresponding operations of the user interface do with the current user interface selection. The main difference is that you can use an arbitrary number of AuthenticRange objects at the same time, whereas there is exactly one cursor selection in the user interface.

To get to an initial range object use AuthenticView.Selection, to obtain a range corresponding with the current cursor selection in the user interface. Alternatively, some trivial ranges are accessible via the read/only properties AuthenticView.DocumentBegin, AuthenticView.DocumentEnd, and AuthenticView.WholeDocument. The most flexible method is AuthenticView.Goto, which allows navigation to a specific portion of the document within one call. For more complex selections, combine the above with the various navigation methods on range objects listed in the first table on this page.

Another method to select a portion of the document is to use the position properties of the range object. Two positioning systems are available and can be combined arbitrarily:

- **Absolute** text cursor positions, starting with position 0 at the document beginning, can be set and retrieved for the beginning and end of a range. For more information see FirstTextPosition and LastTextPosition. This method requires complex internal calculations and should be used with care.

- The **XMLData** element and a text position inside this element, can be set and retrieved for the beginning and end of a range. For more information see FirstXMLData, FirstXMLDataOffset, LastXMLData, and LastXMLDataOffset. This method is very efficient but requires knowledge of the underlying document structure. It can be used to locate XMLData objects and perform operations on them otherwise not accessible through the user interface.

Modifications to the document content can be achieved by various methods:

- The Text property allows you to retrieve the document text selected by the range object. If set, the selected document text gets replaced with the new text.
- The standard document edit functions Cut, Copy, Paste and Delete.
- Table operations for tables that can grow dynamically.
- Methods that map the functionality of the Authentic entry helper windows.
- Access to the XMLData objects of the underlying document to modify them directly.

14.3.2.5.1 AppendRow

Method: AppendRow()as Boolean

Description

If the beginning of the range is inside a dynamic table, this method inserts a new row at the end of the selected table. The selection of the range is modified to point to the beginning of the new row. The function returns *true* if the append operation was successful, otherwise *false*.

Errors

2001 The authentic range object or its related view object is no longer valid.
2005 Invalid address for the return parameter was specified.

Examples

```
' ----------------------------------------------------
' Scripting environment - VBScript
' Append row at end of current dynamically growable table
' ----------------------------------------------------
Dim objRange
' we assume that the active document is open in authentic view mode
Set objRange = Application.ActiveDocument.AuthenticView.Selection

' check if we can insert something
If objRange.IsInDynamicTable Then
        objRange.AppendRow
        ' objRange points to beginning of new row
        objRange.Select
End If
```

14.3.2.5.2 Application

Property: Application as Application (read-only)

Description

Accesses the Authentic Desktop application object.

Errors

2001 The authentic range object or its related view object is no longer valid.
2005 Invalid address for the return parameter was specified.

14.3.2.5.3 CanPerformAction

Method: CanPerformAction (*eAction* as SPYAuthenticActions, *strElementName* as String) as Boolean

Description

CanPerformAction and its related methods enable access to the entry-helper functions of Authentic. This function allows easy and consistent modification of the document content, without having to know exactly where the modification will take place. The beginning of the range object is used to locate the next valid location where the specified action can be performed. If the location can be found, the method returns *True*, otherwise it returns *False*.

HINT: To find out all valid element names for a given action, use CanPerformActionWith.

Errors

2001 The authentic range object or its related view object is no longer valid.

© 2016-2022 Altova GmbH

2005 Invalid address for the return parameter was specified.
2007 Invalid action was specified.

Examples
See PerformAction.

14.3.2.5.4 CanPerformActionWith

Method: CanPerformActionWith (*eAction* as SPYAuthenticActions, *out_arrElementNames* as Variant)

Description
PerformActionWith and its related methods, enable access to the entry-helper functions of Authentic. This function allows easy and consistent modification of the document content without having to know exactly where the modification will take place.

This method returns an array of those element names that the specified action can be performed with.

HINT: To apply the action use CanPerformActionWith.

Errors
2001 The authentic range object, or its related view object is no longer valid.
2005 Invalid address for the return parameter was specified.
2007 Invalid action was specified.

Examples
See PerformAction.

14.3.2.5.5 Clone

Method: Clone() as AuthenticRange

Description
Returns a copy of the range object.

Errors
2001 The authentic range object, or its related view object is no longer valid.
2005 Invalid address for the return parameter was specified.

14.3.2.5.6 CollapsToBegin

Method: CollapsToBegin() as AuthenticRange

Description
Sets the end of the range object to its begin. The method returns the modified range object.

Errors
2001 The authentic range object, or its related view object is no longer valid.

2005 Invalid address for the return parameter was specified.

14.3.2.5.7 CollapsToEnd

Method: CollapsToEnd() as AuthenticRange

Description
Sets the beginning of the range object to its end. The method returns the modified range object.

Errors
2001 The authentic range object, or its related view object is no longer valid.
2005 Invalid address for the return parameter was specified.

14.3.2.5.8 Copy

Method: Copy() as Boolean

Description
Returns *False* if the range contains no portions of the document that may be copied.
Returns *True* if text, and in case of fully selected XML elements the elements as well, has been copied to the copy/paste buffer.

Errors
2001 The authentic range object or its related view object is no longer valid.
2005 Invalid address for the return parameter was specified.

14.3.2.5.9 Cut

Method: Cut() as Boolean

Description
Returns *False* if the range contains portions of the document that may not be deleted.
Returns *True* after text, and in case of fully selected XML elements the elements as well, has been deleted from the document and saved in the copy/paste buffer.

Errors
2001 The authentic range object, or its related view object is no longer valid.
2005 Invalid address for the return parameter was specified.

14.3.2.5.10 Delete

Method: Delete() as Boolean

Description
Returns *False* if the range contains portions of the document that may not be deleted.

© 2016-2022 Altova GmbH

Returns *True* after text, and in case of fully selected XML elements the elements as well, has been deleted from the document.

Errors
2001 The authentic range object or its related view object is no longer valid.
2005 Invalid address for the return parameter was specified.

14.3.2.5.11 DeleteRow

Method: DeleteRow() as Boolean

Description
If the beginning of the range is inside a dynamic table, this method deletes the selected row. The selection of the range gets modified to point to the next element after the deleted row. The function returns *true,* if the delete operation was successful, otherwise *false*.

Errors
2001 The authentic range object, or its related view object is no longer valid.
2005 Invalid address for the return parameter was specified.

Examples
```
' ----------------------------------------------------------
' Scripting environment - VBScript
' Delete selected row from dynamically growing table
' ----------------------------------------------------------
Dim objRange
' we assume that the active document is open in authentic view mode
Set objRange = Application.ActiveDocument.AuthenticView.Selection

' check if we are in a table
If objRange.IsInDynamicTable Then
        objRange.DeleteRow
End If
```

14.3.2.5.12 DuplicateRow

Method: DuplicateRow() as Boolean

Description
If the beginning of the range is inside a dynamic table, this method inserts a duplicate of the current row after the selected one. The selection of the range gets modified to point to the beginning of the new row. The function returns *true* if the duplicate operation was successful, otherwise *false*.

Errors
2001 The authentic range object, or its related view object is no longer valid.
2005 Invalid address for the return parameter was specified.

Examples
```
' ----------------------------------------------------------
' Scripting environment - VBScript
```

```
' duplicate row in current dynamically growable table
' -------------------------------------------------------
Dim objRange
' we assume that the active document is open in authentic view mode
Set objRange = Application.ActiveDocument.AuthenticView.Selection

' check if we can insert something
If objRange.IsInDynamicTable Then
        objRange.DuplicateRow
        ' objRange points to beginning of new row
        objRange.Select
End If
```

14.3.2.5.13 EvaluateXPath

Method: EvaluateXPath (strExpression as string) strValue as string

Return Value
The method returns a string

Description
`EvaluateXPath()` executes an XPath expression with the context node being the beginning of the range selection. The result is returned as a string, in the case of a sequence it is a space-separated string. If XML context node is irrelevant, the user may provide any node, like `AuthenticView.XMLDataRoot`.

Errors

2001	Invalid object
2005	Invalid parameter
2008	Internal error
2202	Missing context node
2211	XPath error

14.3.2.5.14 ExpandTo

Method: ExpandTo (*eKind* as SPYAuthenticElementKind), as AuthenticRange

Description
Selects the whole element of type eKind, that starts at, or contains, the first cursor position of the range. The method returns the modified range object.

Errors

2001	The authentic range object, or its related view object is no longer valid.
2003	Range expansion would be beyond end of document.
2005	Invalid address for the return parameter was specified.

© 2016-2022 Altova GmbH

14.3.2.5.15 FirstTextPosition

Property: FirstTextPosition as Long

Description
Set or get the left-most text position index of the range object. This index is always less or equal to LastTextPosition. Indexing starts with 0 at document beginning, and increments with every different position that the text cursor can occupy. Incrementing the test position by 1, has the same effect as the cursor-right key. Decrementing the test position by 1 has the same effect as the cursor-left key.

If you set FirstTextPosition to a value greater than the current LastTextPosition, LastTextPosition gets set to the new FirstTextPosition.

HINT: Use text cursor positions with care, since this is a costly operation compared to XMLData based cursor positioning.

Errors

2001	The authentic range object, or its related view object is not valid.
2005	Invalid address for the return parameter was specified.
2006	A text position outside the document was specified.

Examples

```
' ---------------------------------------
' Scripting environment - VBScript
' ---------------------------------------
Dim objAuthenticView
' we assume that the active document is open in authentic view mode
Set objAuthenticView = Application.ActiveDocument.AuthenticView

nDocStartPosition = objAuthenticView.DocumentBegin.FirstTextPosition
nDocEndPosition = objAuthenticView.DocumentEnd.FirstTextPosition

' let's create a range that selects the whole document
' in an inefficient way
Dim objRange
' we need to get a (any) range object first
Set objRange = objAuthenticView.DocumentBegin
objRange.FirstTextPosition = nDocStartPosition
objRange.LastTextPosition = nDocEndPosition

' let's check if we got it right
If objRange.isEqual(objAuthenticView.WholeDocument) Then
        MsgBox "Test using direct text cursor positioning was ok"
Else
        MsgBox "Ooops!"
End If
```

14.3.2.5.16 FirstXMLData

Property: FirstXMLData as XMLData

Description
Set or get the first XMLData element in the underlying document that is partially, or completely selected by the range. The exact beginning of the selection is defined by the FirstXMLDataOffset attribute.

Whenever you set FirstXMLData to a new data object, FirstXMLDataOffset gets set to the first cursor position inside this element. Only XMLData objects that have a cursor position may be used. If you set FirstXMLData / FirstXMLDataOffset selects a position greater then the current LastXMLData / LastXMLDataOffset, the latter gets moved to the new start position.

HINT: You can use the FirstXMLData and LastXMLData properties to directly access and manipulate the underlying XML document in those cases where the methods available with the AuthenticRange object are not sufficient.

Errors

2001	The authentic range object, or its related view object is not valid.
2005	Invalid address for the return parameter was specified.
2008	Internal error
2009	The XMLData object cannot be accessed.

Examples

```
' ---------------------------------------------
' Scripting environment - VBScript
' show name of currently selected XMLData element
' ---------------------------------------------
Dim objAuthenticView
' we assume that the active document is open in authentic view mode
Set objAuthenticView = Application.ActiveDocument.AuthenticView

Dim objXmlData
Set objXMLData = objAuthenticView.Selection.FirstXMLData
' authentic view adds a 'text' child element to elements
' of the document which have content. So we have to go one
' element up.
Set objXMLData = objXMLData.Parent
MsgBox "Current selection selects element " & objXMLData.Name
```

14.3.2.5.17 FirstXMLDataOffset

Property: FirstXMLDataOffset as Long

Description
Set or get the cursor position offset inside FirstXMLData element for the beginning of the range. Offset positions are based on the characters returned by the Text property, and start with 0. When setting a new offset, use -1 to set the offset to the last possible position in the element. The following cases require specific attention:

© 2016-2022 Altova GmbH

- The textual form of entries in Combo Boxes, Check Boxes and similar controls can be different from what you see on screen. Although the data offset is based on this text, there only two valid offset positions, one at the beginning and one at the end of the entry. An attempt to set the offset to somewhere in the middle of the entry, will result in the offset being set to the end.

- The textual form of XML Entities might differ in length from their representation on the screen. The offset is based on this textual form.

If FirstXMLData / FirstXMLDataOffset selects a position after the current LastXMLData / LastXMLDataOffset, the latter gets moved to the new start position.

Errors

2001 The authentic range object, or its related view object is not valid.
2005 Invalid offset was specified.
Invalid address for the return parameter was specified.

Examples

```
' ---------------------------------------------
' Scripting environment - VBScript
' Select the complete text of an XMLData element
' using XMLData based selection and ExpandTo
' ---------------------------------------------
Dim objAuthenticView
' we assume that the active document is open in authentic view mode
Set objAuthenticView = Application.ActiveDocument.AuthenticView

' first we use the XMLData based range properties
' to select all text of the first XMLData element
' in the current selection
Dim objRange
Set objRange = objAuthenticView.Selection
objRange.FirstXMLDataOffset = 0  ' start at beginning of element text
objRange.LastXMLData = objRange.FirstXMLData  ' select only one element
objRange.LastXMLDataOffset = -1  ' select till its end

' the same can be achieved with the ExpandTo method
Dim objRange2
Set objRange2 = objAuthenticView.Selection.ExpandTo(spyAuthenticTag)

' were we successful?
If objRange.IsEqual(objRange2) Then
        objRange.Select()
Else
        MsgBox "Oops"
End If
```

14.3.2.5.18 GetElementAttributeNames

Method: GetElementAttributeNames (*strElementName* as String, *out_arrAttributeNames* as Variant)

Description

Retrieve the names of all attributes for the enclosing element with the specified name. Use the element/attribute pairs, to set or get the attribute value with the methods GetElementAttributeValue and SetElementAttributeValue.

Errors

2001 The authentic range object, or its related view object is no longer valid.
2005 Invalid element name was specified.
Invalid address for the return parameter was specified.

Examples

See SetElementAttributeValue.

14.3.2.5.19 GetElementAttributeValue

Method: GetElementAttributeValue (*strElementName* as String, *strAttributeName* as String) as String

Description

Retrieve the value of the attribute specified in strAttributeName, for the element identified with strElementName. If the attribute is supported but has no value assigned, the empty string is returned. To find out the names of attributes supported by an element, use GetElementAttributeNames, or HasElementAttribute.

Errors

2001 The authentic range object, or its related view object is no longer valid.
2005 Invalid element name was specified.
Invalid attribute name was specified.
Invalid address for the return parameter was specified.

Examples

See SetElementAttributeValue.

14.3.2.5.20 GetElementHierarchy

Method: GetElementHierarchy (*out_arrElementNames* as Variant)

Description

Retrieve the names of all XML elements that are parents of the current selection. Inner elements get listed before enclosing elements. An empty list is returned whenever the current selection is not inside a single XMLData element.

The names of the element hierarchy, together with the range object uniquely identify XMLData elements in the document. The attributes of these elements can be directly accessed by GetElementAttributeNames, and related methods.

Errors

2001 The authentic range object, or its related view object is no longer valid.
2005 Invalid address for the return parameter was specified.

© 2016-2022 Altova GmbH

14.3.2.5.21 GetEntityNames

Method: GetEntityNames (*out_arrEntityNames* as Variant)

Description
Retrieve the names of all defined entities. The list of retrieved entities is independent of the current selection, or location. Use one of these names with the InsertEntity function.

Errors
2001 The authentic range object, or its related view object is no longer valid.
2005 Invalid address for the return parameter was specified.

Examples
See: GetElementHierarchy and InsertEntity.

14.3.2.5.22 GetVariableValue

Method: GetVariableValue `(strName as string) strVal as string`

Return Value
Gets the value of the variable named as the method's parameter.

Errors
2001 Invalid object.
2202 No context.
2204 No such variable in scope
2205 Variable cannot be evaluated
2206 Variable returns sequence
2209 Invalid parameter

14.3.2.5.23 Goto

Method: Goto (*eKind* as SPYAuthenticElementKind, *nCount* as Long, *eFrom* as SPYAuthenticDocumentPosition) as AuthenticRange

Description
Sets the range to point to the beginning of the nCount element of type eKind. The start position is defined by the parameter eFrom.

Use positive values for nCount to navigate to the document end. Use negative values to navigate to the beginning of the document. The method returns the modified range object.

Errors
2001 The authentic range object, or its related view object is no longer valid.
2003 Target lies after end of document.

2004 Target lies before begin of document.
2005 Invalid element kind specified.
Invalid start position specified.
Invalid address for the return parameter was specified.

14.3.2.5.24 GotoNext

Method: GotoNext (*eKind* as SPYAuthenticElementKind) as AuthenticRange

Description
Sets the range to the beginning of the next element of type eKind. The method returns the modified range object.

Errors
2001 The authentic range object, or its related view object is no longer valid.
2003 Target lies after end of document.
2005 Invalid element kind specified.
Invalid address for the return parameter was specified.

Examples

```
' ----------------------------------------
' Scripting environment - VBScript
' Scan through the whole document word-by-word
' ----------------------------------------
Dim objAuthenticView
' we assume that the active document is open in authentic view mode
Set objAuthenticView = Application.ActiveDocument.AuthenticView

Dim objRange
Set objRange = objAuthenticView.DocumentBegin
Dim bEndOfDocument
bEndOfDocument = False

On Error Resume Next
While Not bEndOfDocument
	objRange.GotoNext(spyAuthenticWord).Select
	If ((Err.number - vbObjecterror) = 2003) Then
		bEndOfDocument = True
		Err.Clear
	ElseIf (Err.number <> 0) Then
		Err.Raise  ' forward error
	End If
Wend
```

14.3.2.5.25 GotoNextCursorPosition

Method: GotoNextCursorPosition() as AuthenticRange

Description
Sets the range to the next cursor position after its current end position. Returns the modified object.

© 2016-2022 Altova GmbH

Errors

2001 The authentic range object, or its related view object is no longer valid.
2003 Target lies after end of document.
2005 Invalid address for the return parameter was specified.

14.3.2.5.26 GotoPrevious

Method: GotoPrevious (*eKind* as SPYAuthenticElementKind) as AuthenticRange

Description

Sets the range to the beginning of the element of type eKind which is before the beginning of the current range. The method returns the modified range object.

Errors

2001 The authentic range object, or its related view object is no longer valid.
2004 Target lies before beginning of document.
2005 Invalid element kind specified.
Invalid address for the return parameter was specified.

Examples

```
' ----------------------------------------
' Scripting environment - VBScript
' Scan through the whole document tag-by-tag
' ----------------------------------------
Dim objAuthenticView
' we assume that the active document is open in authentic view mode
Set objAuthenticView = Application.ActiveDocument.AuthenticView

Dim objRange
Set objRange = objAuthenticView.DocumentEnd
Dim bEndOfDocument
bBeginOfDocument = False

On Error Resume Next
While Not bBeginOfDocument
        objRange.GotoPrevious(spyAuthenticTag).Select
        If ((Err.number - vbObjecterror) = 2004) Then
                bBeginOfDocument = True
                Err.Clear
        ElseIf (Err.number <> 0) Then
                Err.Raise  ' forward error
        End If
Wend
```

14.3.2.5.27 GotoPreviousCursorPosition

Method: GotoPreviousCursorPosition() as AuthenticRange

Description

Set the range to the cursor position immediately before the current position. Returns the modified object.

Errors

2001 The authentic range object, or its related view object is no longer valid.
2004 Target lies before begin of document.
2005 Invalid address for the return parameter was specified.

14.3.2.5.28 HasElementAttribute

Method: HasElementAttribute (*strElementName* as String, *strAttributeName* as String) as Boolean

Description

Tests if the enclosing element with name strElementName, supports the attribute specified in strAttributeName.

Errors

2001 The authentic range object, or its related view object is no longer valid.
2005 Invalid element name was specified.
Invalid address for the return parameter was specified.

14.3.2.5.29 InsertEntity

Method: InsertEntity (*strEntityName* as String)

Description

Replace the ranges selection with the specified entity. The specified entity must be one of the entity names returned by GetEntityNames.

Errors

2001 The authentic range object, or its related view object is no longer valid.
2005 Unknown entry name was specified.

Examples

```
' ----------------------------------------------------------
' Scripting environment - VBScript
' Insert the first entity in the list of available entities
' ----------------------------------------------------------
Dim objRange
' we assume that the active document is open in authentic view mode
Set objRange = Application.ActiveDocument.AuthenticView.Selection

' first we get the names of all available entities as they
' are shown in the entry helper of XMLSpy
Dim arrEntities
objRange.GetEntityNames arrEntities

' we insert the first one of the list
If UBound(arrEntities) >= 0 Then
        objRange.InsertEntity arrEntities(0)
Else
```


© 2016-2022 Altova GmbH

```
        MsgBox "Sorry, no entities are available for this document"
End If
```

14.3.2.5.30 InsertRow

Method: InsertRow() as Boolean

Description
If the beginning of the range is inside a dynamic table, this method inserts a new row before the current one. The selection of the range gets modified to point to the beginning of the newly inserted row. The function returns *true* if the insert operation was successful, otherwise *false*.

Errors
2001 The authentic range object, or its related view object is no longer valid.
2005 Invalid address for the return parameter was specified.

Examples

```
' ----------------------------------------------------------
' Scripting environment - VBScript
' Insert row at beginning of current dynamically growing table
' ----------------------------------------------------------
Dim objRange
' we assume that the active document is open in authentic view mode
Set objRange = Application.ActiveDocument.AuthenticView.Selection

' check if we can insert something
If objRange.IsInDynamicTable Then
        objRange.InsertRow
        ' objRange points to beginning of new row
        objRange.Select
  End If
```

14.3.2.5.31 IsCopyEnabled

Property: IsCopyEnabled as Boolean (read-only)

Description
Checks if the copy operation is supported for this range.

Errors
2001 The authentic range object, or its related view object is no longer valid.
2005 Invalid address for the return parameter was specified.

14.3.2.5.32 IsCutEnabled

Property: IsCutEnabled as Boolean (read-only)

Description

Checks if the cut operation is supported for this range.

Errors
2001 The authentic range object, or its related view object is no longer valid.
2005 Invalid address for the return parameter was specified.

14.3.2.5.33 IsDeleteEnabled

Property: IsDeleteEnabled as Boolean (read-only)

Description
Checks if the delete operation is supported for this range.

Errors
2001 The authentic range object, or its related view object is no longer valid.
2005 Invalid address for the return parameter was specified.

14.3.2.5.34 IsEmpty

Method: IsEmpty() as Boolean

Description
Tests if the first and last position of the range are equal.

Errors
2001 The authentic range object, or its related view object is no longer valid.
2005 Invalid address for the return parameter was specified.

14.3.2.5.35 IsEqual

Method: IsEqual (*objCmpRange* as AuthenticRange) as Boolean

Description
Tests if the start and end of both ranges are the same.

Errors
2001 One of the two range objects being compared, is invalid.
2005 Invalid address for a return parameter was specified.

14.3.2.5.36 IsFirstRow

Property: IsFirstRow as Boolean (read-only)

Description
Test if the range is in the first row of a table. Which table is taken into consideration depends on the extend of the range. If the selection exceeds a single row of a table, the check is if this table is the first element in an embedding table. See the entry helpers of the user manual for more information.

© 2016-2022 Altova GmbH

Errors
2001 The authentic range object, or its related view object is no longer valid.
2005 Invalid address for the return parameter was specified.

14.3.2.5.37 IsInDynamicTable

Method: IsInDynamicTable() as Boolean

Description
Test if the whole range is inside a table that supports the different row operations like 'insert', 'append', duplicate, etc.

Errors
2001 The authentic range object, or its related view object is no longer valid.
2005 Invalid address for the return parameter was specified.

14.3.2.5.38 IsLastRow

Property: IsLastRow as Boolean (read-only)

Description
Test if the range is in the last row of a table. Which table is taken into consideration depends on the extend of the range. If the selection exceeds a single row of a table, the check is if this table is the last element in an embedding table. See the entry helpers of the user manual for more information.

Errors
2001 The authentic range object, or its related view object is no longer valid.
2005 Invalid address for the return parameter was specified.

14.3.2.5.39 IsPasteEnabled

Property: IsPasteEnabled as Boolean (read-only)

Description
Checks if the paste operation is supported for this range.

Errors
2001 The authentic range object, or its related view object is no longer valid.
2005 Invalid address for the return parameter was specified.

14.3.2.5.40 IsSelected

Property: IsSelected as Boolean

Description

Returns true() if selection is present. The selection range still can be empty: that happens when e.g. only the cursor is set.

14.3.2.5.41 IsTextStateApplied

Method: IsTextStateApplied (*i_strElementName* as String) as Boolean

Description
Checks if all the selected text is embedded into an XML Element with name i_strElementName. Common examples for the parameter i_strElementName are "strong", "bold" or "italic".

Errors
2001 The authentic range object, or its related view object is no longer valid.
2005 Invalid address for the return parameter was specified.

14.3.2.5.42 LastTextPosition

Property: LastTextPosition as Long

Description
Set or get the rightmost text position index of the range object. This index is always greater or equal to FirstTextPosition. Indexing starts with 0 at the document beginning, and increments with every different position that the text cursor can occupy. Incrementing the test position by 1, has the same effect as the cursor-right key. Decreasing the test position by 1 has the same effect as the cursor-left key.

If you set LastTextPosition to a value less then the current FirstTextPosition, FirstTextPosition gets set to the new LastTextPosition.

HINT: Use text cursor positions with care, since this is a costly operation compared to XMLData based cursor positioning.

Errors
2001 The authentic range object, or its related view object is not valid.
2005 Invalid address for the return parameter was specified.
2006 A text position outside the document was specified.

Examples

```
' ------------------------------------
' Scripting environment - VBScript
' ------------------------------------
Dim objAuthenticView
' we assume that the active document is open in authentic view mode
Set objAuthenticView = Application.ActiveDocument.AuthenticView

nDocStartPosition = objAuthenticView.DocumentBegin.FirstTextPosition
nDocEndPosition = objAuthenticView.DocumentEnd.FirstTextPosition

' let's create a range that selects the whole document
' in an inefficient way
```

 © 2016-2022 Altova GmbH

```
Dim objRange
' we need to get a (any) range object first
Set objRange = objAuthenticView.DocumentBegin
objRange.FirstTextPosition = nDocStartPosition
objRange.LastTextPosition = nDocEndPosition

' let's check if we got it right
If objRange.isEqual(objAuthenticView.WholeDocument) Then
        MsgBox "Test using direct text cursor positioning was ok"
Else
        MsgBox "Oops!"
End If
```

14.3.2.5.43 LastXMLData

Property: LastXMLData as XMLData

Description
Set or get the last XMLData element in the underlying document that is partially or completely selected by the range. The exact end of the selection is defined by the LastXMLDataOffset attribute.

Whenever you set LastXMLData to a new data object, LastXMLDataOffset gets set to the last cursor position inside this element. Only XMLData objects that have a cursor position may be used. If you set LastXMLData / LastXMLDataOffset, select a position less then the current FirstXMLData / FirstXMLDataOffset, the latter gets moved to the new end position.

HINT: You can use the FirstXMLData and LastXMLData properties to directly access and manipulate the underlying XML document in those cases, where the methods available with the AuthenticRange object are not sufficient.

Errors

2001	The authentic range object, or its related view object is not valid.
2005	Invalid address for the return parameter was specified.
2008	Internal error
2009	The XMLData object cannot be accessed.

14.3.2.5.44 LastXMLDataOffset

Property: LastXMLDataOffset as Long

Description
Set or get the cursor position inside LastXMLData element for the end of the range.

Offset positions are based on the characters returned by the Text property and start with 0. When setting a new offset, use -1 to set the offset to the last possible position in the element. The following cases require specific attention:

- The textual form of entries in Combo Boxes, Check Boxes and similar controls can be different from what you see on the screen. Although, the data offset is based on this text, there only two valid offset positions,

one at the beginning and one at the end of the entry. An attempt to set the offset to somewhere in the middle of the entry, will result in the offset being set to the end.
- The textual form of XML Entities might differ in length from their representation on the screen. The offset is based on this textual form.

If LastXMLData / LastXMLDataOffset selects a position before FirstXMLData / FirstXMLDataOffset, the latter gets moved to the new end position.

Errors

2001	The authentic range object, or its related view object is not valid.
2005	Invalid offset was specified.
	Invalid address for the return parameter was specified.

Examples

```
' ----------------------------------------------
' Scripting environment - VBScript
' Select the complete text of an XMLData element
' using XMLData based selection and ExpandTo
' ----------------------------------------------
Dim objAuthenticView
' we assume that the active document is open in authentic view mode
Set objAuthenticView = Application.ActiveDocument.AuthenticView

' first we use the XMLData based range properties
' to select all text of the first XMLData element
' in the current selection
Dim objRange
Set objRange = objAuthenticView.Selection
objRange.FirstXMLDataOffset = 0  ' start at beginning of element text
objRange.LastXMLData = objRange.FirstXMLData  ' select only one element
objRange.LastXMLDataOffset = -1  ' select till its end

' the same can be achieved with the ExpandTo method
Dim objRange2
Set objRange2 = objAuthenticView.Selection.ExpandTo(spyAuthenticTag)

' were we successful?
If objRange.IsEqual(objRange2) Then
        objRange.Select()
Else
        MsgBox "Ooops"
End If
```

14.3.2.5.45 MoveBegin

Method: MoveBegin (*eKind* as SPYAuthenticElementKind, *nCount* as Long) as AuthenticRange

Description

Move the beginning of the range to the beginning of the nCount element of type eKind. Counting starts at the current beginning of the range object.

© 2016-2022 Altova GmbH

Use positive numbers for nCount to move towards the document end, use negative numbers to move towards document beginning. The end of the range stays unmoved, unless the new beginning would be larger than it. In this case, the end is moved to the new beginning. The method returns the modified range object.

Errors

2001	The authentic range object, or its related view object is no longer valid.
2003	Target lies after end of document.
2004	Target lies before beginning of document.
2005	Invalid element kind specified.
	Invalid address for the return parameter was specified.

14.3.2.5.46 MoveEnd

Method: MoveEnd (*eKind* as SPYAuthenticElementKind, *nCount* as Long) as AuthenticRange

Description
Move the end of the range to the begin of the nCount element of type eKind. Counting starts at the current end of the range object.

Use positive numbers for nCount to move towards the document end, use negative numbers to move towards document beginning. The beginning of the range stays unmoved, unless the new end would be less than it. In this case, the beginning gets moved to the new end. The method returns the modified range object.

Errors

2001	The authentic range object, or its related view object is no longer valid.
2003	Target lies after end of document.
2004	Target lies before begin of document.
2005	Invalid element kind specified.
	Invalid address for the return parameter was specified.

14.3.2.5.47 MoveRowDown

Method: MoveRowDown() as Boolean

Description
If the beginning of the range is inside a dynamic table and selects a row which is not the last row in this table, this method swaps this row with the row immediately below. The selection of the range moves with the row, but does not otherwise change. The function returns *true* if the move operation was successful, otherwise *false*.

Errors

2001	The authentic range object or its related view object is no longer valid.
2005	Invalid address for the return parameter was specified.

14.3.2.5.48 MoveRowUp

Method: MoveRowUp() as Boolean

Description

If the beginning of the range is inside a dynamic table and selects a row which is not the first row in this table, this method swaps this row with the row above. The selection of the range moves with the row, but does not change otherwise. The function returns *true* if the move operation was successful, otherwise *false*.

Errors
2001 The authentic range object, or its related view object is no longer valid.
2005 Invalid address for the return parameter was specified.

14.3.2.5.49 Parent

Property: Parent as AuthenticView (read-only)

Description
Access the view that owns this range object.

Errors
2001 The authentic range object, or its related view object is no longer valid.
2005 Invalid address for the return parameter was specified.

14.3.2.5.50 Paste

Method: Paste() as Boolean

Description
Returns *False* if the copy/paste buffer is empty, or its content cannot replace the current selection.

Otherwise, deletes the current selection, inserts the content of the copy/paste buffer, and returns *True*.

Errors
2001 The authentic range object, or its related view object is no longer valid.
2005 Invalid address for the return parameter was specified.

14.3.2.5.51 PerformAction

Method: PerformAction (*eAction* as SPYAuthenticActions, *strElementName* as String) as Boolean

Description
PerformAction and its related methods, give access to the entry-helper functions of Authentic. This function allows easy and consistent modification of the document content without a need to know exactly where the modification will take place. The beginning of the range object is used to locate the next valid location where the specified action can be performed. If no such location can be found, the method returns *False*. Otherwise, the document gets modified and the range points to the beginning of the modification.

HINT: To find out element names that can be passed as the second parameter use CanPerformActionWith.

Errors
2001 The authentic range object, or its related view object is no longer valid.
2005 Invalid address for the return parameter was specified.

© 2016-2022 Altova GmbH

2007 Invalid action was specified.

Examples

```
' ----------------------------------------
' Scripting environment - VBScript
' Insert the innermost element
' ----------------------------------------
Dim objRange
' we assume that the active document is open in authentic view mode
Set objRange = Application.ActiveDocument.AuthenticView.Selection

' we determine the elements that can be inserted at the current position
Dim arrElements()
objRange.CanPerformActionWith spyAuthenticInsertBefore, arrElements

' we insert the first (innermost) element
If UBound(arrElements) >= 0 Then
        objRange.PerformAction spyAuthenticInsertBefore, arrElements(0)
        ' objRange now points to the beginning of the inserted element
        ' we set a default value and position at its end
        objRange.Text = "Hello"
        objRange.ExpandTo(spyAuthenticTag).CollapsToEnd().Select
Else
        MsgBox "Can't insert any elements at current position"
End If
```

14.3.2.5.52 Select

Method: Select()

Description

Makes this range the current user interface selection. You can achieve the same result using: '*objRange.Parent.Selection = objRange*'

Errors

2001 The authentic range object or its related view object is no longer valid.

Examples

```
' ----------------------------------------
' Scripting environment - VBScript
' ----------------------------------------
Dim objAuthenticView
' we assume that the active document is open in authentic view mode
Set objAuthenticView = Application.ActiveDocument.AuthenticView

' set current selection to end of document
objAuthenticView.DocumentEnd.Select()
```

14.3.2.5.53 SelectNext

Method: SelectNext (*eKind* as SPYAuthenticElementKind) as AuthenticRange

Description
Selects the element of type eKind after the current end of the range. The method returns the modified range object.

Errors
2001 The authentic range object, or its related view object is no longer valid.
2003 Target lies after end of document.
2005 Invalid element kind specified.
Invalid address for the return parameter was specified.

Examples

```
' ----------------------------------------
' Scripting environment - VBScript
' Scan through the whole document word-by-word
' ----------------------------------------
Dim objAuthenticView
' we assume that the active document is open in authentic view mode
Set objAuthenticView = Application.ActiveDocument.AuthenticView

Dim objRange
Set objRange = objAuthenticView.DocumentBegin
Dim bEndOfDocument
bEndOfDocument = False

On Error Resume Next
While Not bEndOfDocument
        objRange.SelectNext(spyAuthenticWord).Select
        If ((Err.number - vbObjecterror) = 2003) Then
                bEndOfDocument = True
                Err.Clear
        ElseIf (Err.number <> 0) Then
                Err.Raise  ' forward error
        End If
Wend
```

14.3.2.5.54 SelectPrevious

Method: GotoPrevious (*eKind* as SPYAuthenticElementKind) as AuthenticRange

Description
Selects the element of type eKind before the current beginning of the range. The method returns the modified range object.

Errors
2001 The authentic range object, or its related view object is no longer valid.

© 2016-2022 Altova GmbH

2004 Target lies before begin of document.
2005 Invalid element kind specified.
Invalid address for the return parameter was specified.

Examples

```
' ----------------------------------------
' Scripting environment - VBScript
' Scan through the whole document tag-by-tag
' ----------------------------------------
Dim objAuthenticView
' we assume that the active document is open in authentic view mode
Set objAuthenticView = Application.ActiveDocument.AuthenticView

Dim objRange
Set objRange = objAuthenticView.DocumentEnd
Dim bEndOfDocument
bBeginOfDocument = False

On Error Resume Next
While Not bBeginOfDocument
        objRange.SelectPrevious(spyAuthenticTag).Select
        If ((Err.number - vbObjecterror) = 2004) Then
                bBeginOfDocument = True
                Err.Clear
        ElseIf (Err.number <> 0) Then
                Err.Raise  ' forward error
        End If
Wend
```

14.3.2.5.55 SetElementAttributeValue

Method: SetElementAttributeValue (*strElementName* as String, *strAttributeName* as String, *strAttributeValue* as String)

Description

Set the value of the attribute specified in strAttributeName for the element identified with strElementName. If the attribute is supported but has no value assigned, the empty string is returned. To find out the names of attributes supported by an element, use GetElementAttributeNames, or HasElementAttribute.

Errors

2001 The authentic range object or its related view object is no longer valid.
2005 Invalid element name was specified.
Invalid attribute name was specified.
Invalid attribute value was specified.

Examples

```
' ----------------------------------------
' Scripting environment - VBScript
' Get and set element attributes
' ----------------------------------------
```

```
Dim objRange
' we assume that the active document is open in authentic view mode
Set objRange = Application.ActiveDocument.AuthenticView.Selection

' first we find out all the elements below the beginning of the range
Dim arrElements
objRange.GetElementHierarchy arrElements

If IsArray(arrElements) Then
	If UBound(arrElements) >= 0 Then
		' we use the top level element and find out its valid attributes
		Dim arrAttrs()
		objRange.GetElementAttributeNames arrElements(0), arrAttrs

		If UBound(arrAttrs) >= 0 Then
			' we retrieve the current value of the first valid attribute
			Dim strAttrVal
			strAttrVal = objRange.GetElementAttributeValue (arrElements(0), arrAttrs(0))
			msgbox "current value of " & arrElements(0) & "//" & arrAttrs(0) & " is: " & strAttrVal

			' we change this value and read it again
			strAttrVal = "Hello"
			objRange.SetElementAttributeValue arrElements(0), arrAttrs(0), strAttrVal
			strAttrVal = objRange.GetElementAttributeValue (arrElements(0), arrAttrs(0))
			msgbox "new value of " & arrElements(0) & "//" & arrAttrs(0) & " is: " & strAttrVal
		End If
	End If
End If
```

14.3.2.5.56 SetFromRange

Method: SetFromRange (*objSrcRange* as AuthenticRange)

Description
Sets the range object to the same beginning and end positions as objSrcRange.

Errors

2001	One of the two range objects, is invalid.
2005	Null object was specified as source object.

14.3.2.5.57 SetVariableValue

Method: SetVariableValue`(strName as string, strValue as string)`

Return Value
Sets the value (second parameter) of the variable named in the first parameter.

© 2016-2022 Altova GmbH

Errors

2201	Invalid object.
2202	No context.
2204	No such variable in scope
2205	Variable cannot be evaluated
2206	Variable returns sequence
2207	Variable read-only
2208	No modification allowed

14.3.2.5.58 Text

Property: Text as String

Description
Set or get the textual content selected by the range object.

The number of characters retrieved are not necessarily identical, as there are text cursor positions between the beginning and end of the selected range. Most document elements support an end cursor position different to the beginning cursor position of the following element. Drop-down lists maintain only one cursor position, but can select strings of any length. In the case of radio buttons and check boxes, the text property value holds the string of the corresponding XML element.

If the range selects more then one element, the text is the concatenation of the single texts. XML entities are expanded so that '&' is expected as '&'.

Setting the text to the empty string, does not delete any XML elements. Use Cut, Delete or PerformAction instead.

Errors

2001	The authentic range object or its related view object is no longer valid.
2005	Invalid address for a return parameter was specified.

14.3.2.6 AuthenticView

Properties
Application
AsXMLString
DocumentBegin
DocumentEnd
Event
MarkupVisibility
Parent
Selection
XMLDataRoot
WholeDocument

Methods
Goto
IsRedoEnabled
IsUndoEnabled
Print
Redo
Undo
UpdateXMLInstanceEntities

Events
OnBeforeCopy
OnBeforeCut
OnBeforeDelete
OnBeforeDrop
OnBeforePaste
OnDragOver
OnKeyBoardEvent
OnMouseEvent
OnSelectionChanged

Description
AuthenticView and its child objects AuthenticRange and AuthenticDataTransfer provide you with an interface for Authentic View, which allow easy and consistent modification of document contents. These interfaces replace the following interfaces which are marked now as **obsolete:**

OldAuthenticView (old name was DocEditView)
AuthenticSelection (old name was DocEditSelection, superseded by AuthenticRange)
AuthenticEvent (old name was DocEditEvent)

AuthenticView gives you easy access to specific features such as printing, the multi-level undo buffer, and the current cursor selection, or position.

AuthenticView uses objects of type AuthenticRange to make navigation inside the document straight-forward, and to allow for the flexible selection of logical text elements. Use the properties DocumentBegin, DocumentEnd, or WholeDocument for simple selections, while using the Goto method for more complex selections. To navigate relative to a given document range, see the methods and properties of the AuthenticRange object.

14.3.2.6.1 Events

14.3.2.6.1.1 OnBeforeCopy

Event: OnBeforeCopy() as Boolean

Scripting environment - VBScript:
```
Function On_AuthenticBeforeCopy()
        ' On_AuthenticBeforeCopy = False  ' to disable operation
End Function
```

Scripting environment - JScript:
```
function On_AuthenticBeforeCopy()
{
        // return false;  /* to disable operation */
}
```

IDE Plugin:
IXMLSpyPlugIn.OnEvent (21, ...) // nEventId = 21

Description
This event gets triggered before a copy operation gets performed on the document. Return *True* (or nothing) to allow copy operation. Return *False* to disable copying.

14.3.2.6.1.2 OnBeforeCut

Event: OnBeforeCut() as Boolean

Scripting environment - VBScript:

 © 2016-2022 Altova GmbH

```
Function On_AuthenticBeforeCut()
	' On_AuthenticBeforeCut = False  ' to disable operation
End Function
```

Scripting environment - JScript:

```
function On_AuthenticBeforeCut()
{
	// return false;  /* to disable operation */
}
```

IDE Plugin:

IXMLSpyPlugIn.OnEvent (20, ...) // nEventId = 20

Description

This event gets triggered before a cut operation gets performed on the document. Return *True* (or nothing) to allow cut operation. Return *False* to disable operation.

14.3.2.6.1.3 OnBeforeDelete

Event: OnBeforeDelete() as Boolean

Scripting environment - VBScript:

```
Function On_AuthenticBeforeDelete()
	' On_AuthenticBeforeDelete = False  ' to disable operation
End Function
```

Scripting environment - JScript:

```
function On_AuthenticBeforeDelete()
{
	// return false;  /* to disable operation */
}
```

IDE Plugin:

IXMLSpyPlugIn.OnEvent (22, ...) // nEventId = 22

Description

This event gets triggered before a delete operation gets performed on the document. Return *True* (or nothing) to allow delete operation. Return *False* to disable operation.

14.3.2.6.1.4 OnBeforeDrop

Event: OnBeforeDrop (*i_nXPos* as Long, *i_nYPos* as Long, *i_ipRange* as AuthenticRange, *i_ipData* as cancelBoolean

Scripting environment - VBScript:

```
Function On_AuthenticBeforeDrop(nXPos, nYPos, objRange, objData)
	' On_AuthenticBeforeDrop = False  ' to disable operation
End Function
```

Scripting environment - JScript:

```
function On_AuthenticBeforeDrop(nXPos, nYPos, objRange, objData)
{
        // return false;  /* to disable operation */
}
```

IDE Plugin:
IXMLSpyPlugIn.OnEvent (11, ...) // nEventId = 11

Description
This event gets triggered whenever a previously dragged object gets dropped inside the application window. All event related information gets passed as parameters.

The first two parameters specify the mouse position at the time when the event occurred. The parameter *objRange* passes a range object that selects the XML element below the mouse position. The value of this parameter might be *NULL*. Be sure to check before you access the range object. The parameter *objData* allows to access information about the object being dragged.

Return *False* to cancle the drop operation. Return *True* (or nothing) to continue normal operation.

14.3.2.6.1.5 OnBeforePaste

Event: OnBeforePaste (*objData* as Variant, *strType* as String) as Boolean

Scripting environment - VBScript:

```
Function On_AuthenticBeforePaste(objData, strType)
        ' On_AuthenticBeforePaste = False  ' to disable operation
End Function
```

Scripting environment - JScript:

```
function On_AuthenticBeforePaste(objData, strType)
{
        // return false;  /* to disable operation */
}
```

IDE Plugin:
IXMLSpyPlugIn.OnEvent (19, ...) // nEventId = 19

Description
This event gets triggered before a paste operation gets performed on the document. The parameter *strType* is one of "TEXT", "UNICODETEXT" or "IUNKNOWN". In the first two cases *objData* contains a string representation of the object that will be pasted. In the later case, *objData* contains a pointer to an IUnknown COM interface.

Return *True* (or nothing) to allow paste operation. Return *False* to disable operation.

14.3.2.6.1.6 OnBeforeSave

Event: OnBeforeSave (SaveAs flag) as Boolean

 © 2016-2022 Altova GmbH

Description: `OnBeforeSave` gives the opportunity to e.g. warn the user about overwriting the existing XML document, or to make the document read-only when specific circumstances are not met. The event will be fired before the file dialog is shown.

14.3.2.6.1.7 OnDragOver

Event: OnDragOver (*nXPos* as Long, *nYPos* as Long, *eMouseEvent* as SPYMouseEvent, *objRange* as AuthenticRange, *objData* as AuthenticDataTransfer) as Boolean

Scripting environment - VBScript:
```
Function On_AuthenticDragOver(nXPos, nYPos, eMouseEvent, objRange, objData)
	' On_AuthenticDragOver = False  ' to disable operation
End Function
```

Scripting environment - JScript:
```
function On_AuthenticDragOver(nXPos, nYPos, eMouseEvent, objRange, objData)
{
	// return false;  /* to disable operation */
}
```

IDE Plugin:
IXMLSpyPlugIn.OnEvent (10, ...) // nEventId = 10

Description
This event gets triggered whenever an object from within or outside of Authentic View gets dragged with the mouse over the application window. All event related information gets passed as parameters.

The first three parameters specify the mouse position, the mouse button status and the status of the virtual keys at the time when the event occurred. The parameter *objRange* passes a range object that selects the XML element below the mouse position. The value of this parameter might be *NULL*. Be sure to check before you access the range object. The parameter *objData* allows to access information about the object being dragged.

Return *False* to cancel the drag operation. Return *True* (or nothing) to continue normal operation.

14.3.2.6.1.8 OnKeyboardEvent

Event: OnKeyboardEvent (*eKeyEvent* as SPYKeyEvent, *nKeyCode* as Long, *nVirtualKeyStatus* as Long) as Boolean

Scripting environment - VBScript:
```
Function On_AuthenticKeyboardEvent(eKeyEvent, nKeyCode, nVirtualKeyStatus)
	' On_AuthenticKeyboardEvent = True  ' to cancel bubbling of event
End Function
```

Scripting environment - JScript:
```
function On_AuthenticKeyboardEvent(eKeyEvent, nKeyCode, nVirtualKeyStatus)
{
	// return true;  /* to cancel bubbling of event */
```

```
}
```

IDE Plugin:
IXMLSpyPlugIn.OnEvent (30, ...) // nEventId = 30

Description
This event gets triggered for *WM_KEYDOWN*, *WM_KEYUP* and *WM_CHAR* Windows messages.

The actual message type is available in the *eKeyEvent* parameter. The status of virtual keys is combined in the parameter *nVirtualKeyStatus*. Use the bit-masks defined in the enumeration datatype SPYVirtualKeyMask, to test for the different keys or their combinations.

14.3.2.6.1.9 OnLoad

Event: OnLoad ()

Description: `OnLoad` can be used e.g. to restrict some AuthenticView functionality, as shown in the example below:

```
function On_AuthenticLoad(  )
{
    // We are disabling all entry helpers in order to prevent user from manipulating XML tree
    AuthenticView.DisableElementEntryHelper();
    AuthenticView.DisableAttributeEntryHelper();

    // We are also disabling the markup buttons for the same purpose
    AuthenticView.SetToolbarButtonState( 'AuthenticMarkupSmall', authenticToolbarButtonDisabled );
    AuthenticView.SetToolbarButtonState( 'AuthenticMarkupLarge', authenticToolbarButtonDisabled );
    AuthenticView.SetToolbarButtonState( 'AuthenticMarkupMixed', authenticToolbarButtonDisabled );
}
```

In the example the status of the Markup Small, Markup Large, Markup Mixed toolbar buttons are manipulated with the help of button identifiers. See complete list.

14.3.2.6.1.10 OnMouseEvent

Event: OnMouseEvent (*nXPos* as Long, *nYPos* as Long, *eMouseEvent* as SPYMouseEvent, *objRange* as AuthenticRange) as Boolean

Scripting environment - VBScript:

```
Function On_AuthenticMouseEvent(nXPos, nYPos, eMouseEvent, objRange)
        ' On_AuthenticMouseEvent = True  ' to cancel bubbling of event
End Function
```

Scripting environment - JScript:

```
function On_AuthenticMouseEvent(nXPos, nYPos, eMouseEvent, objRange)
{
        // return true;  /* to cancel bubbling of event */
}
```


© 2016-2022 Altova GmbH

IDE Plugin:
IXMLSpyPlugIn.OnEvent (31, ...) // nEventId = 31

Description
This event gets triggered for every mouse movement and mouse button Windows message.

The actual message type and the mouse buttons status, is available in the *eMouseEvent* parameter. Use the bit-masks defined in the enumeration datatype SPYMouseEvent to test for the different messages, button status, and their combinations.

The parameter *objRange* identifies the part of the document found at the current mouse cursor position. The range objects always selects a complete tag of the document. (This might change in future versions, when a more precise positioning mechanism becomes available). If no selectable part of the document is found at the current position, the range object is *null*.

14.3.2.6.1.11 OnSelectionChanged

Event: OnSelectionChanged (*objNewSelection* as AuthenticRange)

Scripting environment - VBScript:
Function On_AuthenticSelectionChanged (*objNewSelection*)
End Function

Scripting environment - JScript:
function On_AuthenticSelectionChanged (*objNewSelection*)
{
}

IDE Plugin:
IXMLSpyPlugIn.OnEvent (23, ...) // nEventId = 23

Description
This event gets triggered whenever the selection in the user interface changes.

14.3.2.6.1.12 OnToolbarButtonClicked

Event: OnToolbarButtonClicked (Button identifier)

Description: `OnToolbarButtonClicked` is fired when a toolbar button was clicked by user. The parameter button identifier helps to determine which button was clicked. The list of predefined button identifiers is below:

- AuthenticPrint
- AuthenticPrintPreview
- AuthenticUndo
- AuthenticRedo
- AuthenticCut
- AuthenticCopy
- AuthenticPaste
- AuthenticClear
- AuthenticMarkupHide

- AuthenticMarkupLarge
- AuthenticMarkupMixed
- AuthenticMarkupSmall
- AuthenticValidate
- AuthenticChangeWorkingDBXMLCell
- AuthenticSave
- AuthenticSaveAs
- AuthenticReload
- AuthenticTableInsertRow
- AuthenticTableAppendRow
- AuthenticTableDeleteRow
- AuthenticTableInsertCol
- AuthenticTableAppendCol
- AuthenticTableDeleteCol
- AuthenticTableJoinCellRight
- AuthenticTableJoinCellLeft
- AuthenticTableJoinCellAbove
- AuthenticTableJoinCellBelow
- AuthenticTableSplitCellHorizontally
- AuthenticTableSplitCellVertically
- AuthenticTableAlignCellContentTop
- AuthenticTableCenterCellVertically
- AuthenticTableAlignCellContentBottom
- AuthenticTableAlignCellContentLeft
- AuthenticTableCenterCellContent
- AuthenticTableAlignCellContentRight
- AuthenticTableJustifyCellContent
- AuthenticTableInsertTable
- AuthenticTableDeleteTable
- AuthenticTableProperties
- AuthenticAppendRow
- AuthenticInsertRow
- AuthenticDuplicateRow
- AuthenticMoveRowUp
- AuthenticMoveRowDown
- AuthenticDeleteRow
- AuthenticDefineEntities
- AuthenticXMLSignature

For custom buttons the user might add his own identifiers. Please, note that the user must take care, as the identifiers are not checked for uniqueness. The same identifiers can be used to identify buttons in the `Set/GetToolbarState()` COM API calls. By adding code for different buttons, the user is in the position to completely redefine the AuthenticView toolbar behavior, adding own methods for table manipulation, etc.

14.3.2.6.1.13 OnToolbarButtonExecuted

Event: OnToolbarButtonExecuted (Button identifier)

Description: `OnToolbarButtonClicked` is fired when a toolbar button was clicked by user. The parameter button identifier helps to determine which button was clicked. See the list of predefined button identifiers.

© 2016-2022 Altova GmbH

`OnToolbarButtonExecuted` is fired after the toolbar action was executed. It is useful e.g. to add update code, as shown in the example below:

```
//event fired when a toolbar button action was executed
function On_AuthenticToolbarButtonExecuted( varBtnIdentifier )
{
    // After whatever command user has executed - make sure to update toolbar button states
    UpdateOwnToolbarButtonStates();
}
```

In this case `UpdateOwnToolbarButtonStates` is a user function defined in the Global Declarations.

14.3.2.6.1.14 OnUserAddedXMLNode

Event: OnUserAddedXMLNode (XML node)

Description: `OnUserAddedXMLNode` will be fired when the user adds an XML node as a primary action. This happens in the situations, where the user clicks on

- auto-add hyperlinks (see example OnUserAddedXMLNode.sps)
- the Insert…, Insert After…, Insert Before… context menu items
- Append row, Insert row toolbar buttons
- Insert After…, Insert Before… actions in element entry helper (outside StyleVision)

The event doesn't get fired on Duplicate row, or when the node was added externally (e.g. via COM API), or on Apply (e.g. Text State Icons), or when in XML table operations or in DB operations.

The event parameter is the XML node object, which was added giving the user an opportunity to manipulate the XML node added. An elaborate example for an event handler can be found in the `OnUserAddedXMLNode.sps` file.

14.3.2.6.2 Application

Property: Application as Application (read-only)

Description
Accesses the Authentic Desktop application object.

Errors

2000	The authentic view object is no longer valid.
2005	Invalid address for the return parameter was specified.

14.3.2.6.3 AsXMLString

Property: AsXMLString as String

Description

Returns or sets the document content as an XML string. Setting the content to a new value does not change the schema file or sps file in use. If the new XMLString does not match the actual schema file error 2011 gets returned.

Errors

2000 The authentic view object is no longer valid.
2011 AsXMLString was set to a value which is no valid XML for the current schema file.

14.3.2.6.4 ContextMenu

Property: ContextMenu() as ContextMenu

Description

The property `ContextMenu` gives access to customize the context menu. The best place to do it is in the event handler `OnContextMenuActivated`.

Errors

2000 Invalid object.
2005 Invalid parameter.

14.3.2.6.5 CreateXMLNode

Method: CreateXMLNode (*nKind* as SPYXMLDataKind) as XMLData

Return Value

The method returns the new XMLData object.

Description

To create a new XMLData object use the CreateXMLNode() method.

Errors

2000 Invalid object.
2012 Cannot create XML node.

14.3.2.6.6 DisableAttributeEntryHelper

Method: DisableAttributeEntryHelper`()`

Description

DisableAttributeEntryHelper`()` disables the attribute entry helper in XMLSpy, Authentic Desktop and Authentic Browser plug-in.

Errors

2000 Invalid object.

© 2016-2022 Altova GmbH

14.3.2.6.7 DisableElementEntryHelper

Method: DisableElementEntryHelper()

Description
DisableElementEntryHelper() disables the element entry helper in XMLSpy, Authentic Desktop and Authentic Browser plug-in.

Errors
2000 Invalid object.

14.3.2.6.8 DisableEntityEntryHelper

Method: DisableEntityEntryHelper()

Description
DisableEntityEntryHelper() disables the entity entry helper in XMLSpy, Authentic Desktop and Authentic Browser plug-in.

Errors
2000 Invalid object.

14.3.2.6.9 DocumentBegin

Property: DocumentBegin as AuthenticRange (read-only)

Description
Retrieve a range object that points to the beginning of the document.

Errors
2000 The authentic view object is no longer valid.
2005 Invalid address for the return parameter was specified.

14.3.2.6.10 DocumentEnd

Property: DocumentEnd as AuthenticRange (read-only)

Description
Retrieve a range object that points to the end of the document.

Errors
2000 The authentic view object is no longer valid.
2005 Invalid address for the return parameter was specified.

14.3.2.6.11 DoNotPerformStandardAction

Method: DoNotPerformStandardAction `()`

Description
`DoNotPerformStandardAction()` serves as cancel bubble for macros, and stops further execution after macro has finished.

Errors
2000 Invalid object.

14.3.2.6.12 EvaluateXPath

Method: EvaluateXPath (XMLData as XMLData, strExpression as string) strValue as string

Return Value
The method returns a string

Description
`EvaluateXPath()` executes an XPath expression with the given XML context node. The result is returned as a string, in the case of a sequence it is a space-separated string.

Errors
2000 Invalid object.
2005 Invalid parameter.
2008 Internal error.
2013 XPath error.

14.3.2.6.13 Event

Property: Event as AuthenticEvent (read-only)

Description
This property gives access to parameters of the last event in the same way as OldAuthenticView.event does. Since all events for the scripting environment and external clients are now available with parameters this Event property should only be used from within IDE-Plugins.

Errors
2000 The authentic view object is no longer valid.
2005 Invalid address for the return parameter was specified.

© 2016-2022 Altova GmbH

14.3.2.6.14 EventContext

Property: EventContext() as EventContext

Description
`EventContext` property gives access to the running macros context. See the `EventContext` interface description for more details.

Errors
2000 Invalid object.

14.3.2.6.15 GetToolbarButtonState

Method: GetToolbarButtonState (`ButtonIdentifier` as string) as `AuthenticToolbarButtonState`

Return Value
The method returns `AuthenticToolbarButtonState`

Description
`Get/SetToolbarButtonState` queries the status of a toolbar button, and lets the user disable or enable the button, identified via its button identifier (see list above). One usage is to disable toolbar buttons permanently. Another usage is to put `SetToolbarButtonState` in the `OnSelectionChanged` event handler, as toolbar buttons are updated regularly when the selection changes in the document.

Toolbar button states are given by the listed enumerations.

The default state means that the enable/disable of the button is governed by AuthenticView. When the user sets the button state to enable or disable, the button remains in that state as long as the user does not change it.

Errors
2000 Invalid object.
2005 Invalid parameter.
2008 Internal error.
2014 Invalid button identifier.

14.3.2.6.16 Goto

Method: Goto (*eKind* as SPYAuthenticElementKind, *nCount* as Long, *eFrom* as *SPYAuthenticDocumentPosition*) as *AuthenticRange*

Description
Retrieve a range object that points to the beginning of the *nCount* element of type *eKind*. The start position is defined by the parameter *eFrom*. Use positive values for *nCount* to navigate to the document end. Use negative values to navigate towards the beginning of the document.

Errors

2000	The authentic view object is no longer valid.
2003	Target lies after end of document.
2004	Target lies before beginning of document.
2005	Invalid element kind specified. The document position to start from is not one of *spyAuthenticDocumentBegin* or *spyAuthenticDocumentEnd*. Invalid address for the return parameter was specified.

Examples

```
' ---------------------------------------
' Scripting environment - VBScript
' ---------------------------------------
Dim objAuthenticView
' we assume that the active document is open in authentic view mode
Set objAuthenticView = Application.ActiveDocument.AuthenticView

On Error Resume Next
Dim objRange
' goto beginning of first table in document
Set objRange = objAuthenticView.Goto (spyAuthenticTable, 1, spyAuthenticDocumentBegin)
If (Err.number = 0) Then
        objRange.Select()
Else
        MsgBox "No table found in document"
End If
```

14.3.2.6.17 IsRedoEnabled

Property: IsRedoEnabled as Boolean (read-only)

Description

True if redo steps are available and Redo is possible.

Errors

2000	The authentic view object is no longer valid.
2005	Invalid address for the return parameter was specified.

14.3.2.6.18 IsUndoEnabled

Property: IsUndoEnabled as Boolean (read-only)

Description

True if undo steps are available and Undo is possible.

Errors

2000	The authentic view object is no longer valid.
2005	Invalid address for the return parameter was specified.

© 2016-2022 Altova GmbH

14.3.2.6.19 MarkupVisibility

Property: MarkupVisibility as SPYAuthenticMarkupVisibility

Description
Set or get current visibility of markup.

Errors
2000 The authentic view object is no longer valid.
2005 Invalid enumeration value was specified.
Invalid address for the return parameter was specified.

14.3.2.6.20 Parent

Property: Parent as Document (read-only)

Description
Access the document shown in this view.

Errors
2000 The authentic view object is no longer valid.
2005 Invalid address for the return parameter was specified.

14.3.2.6.21 Print

Method: Print (*bWithPreview* as Boolean, *bPromptUser* as Boolean)

Description
Print the document shown in this view. If *bWithPreview* is set to *True*, the print preview dialog pops up. If *bPromptUser* is set to *True*, the print dialog pops up. If both parameters are set to *False*, the document gets printed without further user interaction.

Errors
2000 The authentic view object is no longer valid.

14.3.2.6.22 Redo

Method: Redo() as Boolean

Description
Redo the modification undone by the last undo command.

Errors

2000 The authentic view object is no longer valid.
2005 Invalid address for the return parameter was specified.

14.3.2.6.23 Selection

Property: Selection as AuthenticRange

Description

Set or get current text selection in user interface.

Errors

2000 The authentic view object is no longer valid.
2002 No cursor selection is active.
2005 Invalid address for the return parameter was specified.

Examples

```
' ---------------------------------------
' Scripting environment - VBScript
' ---------------------------------------
Dim objAuthenticView
' we assume that the active document is open in authentic view mode
Set objAuthenticView = Application.ActiveDocument.AuthenticView

' if we are the end of the document, re-start at the beginning
If (objAuthenticView.Selection.IsEqual(objAuthenticView.DocumentEnd)) Then
        objAuthenticView.Selection = objAuthenticView.DocumentBegin
Else
        ' objAuthenticView.Selection = objAuthenticView.Selection.GotoNextCursorPosition()
        ' or shorter:
         objAuthenticView.Selection.GotoNextCursorPosition().Select
End If
```

14.3.2.6.24 SetToolbarButtonState

Method: SetToolbarButtonState (`ButtonIdentifier` as string, `AuthenticToolbarButtonState` state)

Description

`Get/SetToolbarButtonState` queries the status of a toolbar button, and lets the user disable or enable the button, identified via its button identifier (see list above). One usage is to disable toolbar buttons permanently. Another usage is to put `SetToolbarButtonState` in the `OnSelectionChanged` event handler, as toolbar buttons are updated regularly when the selection changes in the document.

Toolbar button states are given by the listed enumerations.

The default state means that the enable/disable of the button is governed by AuthenticView. When the user sets the button state to enable or disable, the button remains in that state as long as the user does not change it.

Errors

© 2016-2022 Altova GmbH

2000 Invalid object.
2008 Internal error.
2014 Invalid button identifier.

14.3.2.6.25 Undo

Method: Undo() as Boolean

Description
Undo the last modification of the document from within this view.

Errors
2000 The authentic view object is no longer valid.
2005 Invalid address for the return parameter was specified.

14.3.2.6.26 UpdateXMLInstanceEntities

Method: UpdateXMLInstanceEntities()

Description
Updates the internal representation of the declared entities, and refills the entry helper. In addition, the validator is reloaded, allowing the XML file to validate correctly. Please note that this may also cause schema files to be reloaded.

Errors
The method never returns an error.

Example

```
// ----------------------------------------
// Scripting environment - JavaScript
// ----------------------------------------
if(Application.ActiveDocument && (Application.ActiveDocument.CurrentViewMode == 4))
{
	var objDocType;
	objDocType = Application.ActiveDocument.DocEditView.XMLRoot.GetFirstChild(10);

	if(objDocType)
	{
		var objEntity = Application.ActiveDocument.CreateChild(14);
		objEntity.Name = "child";
		objEntity.TextValue = "SYSTEM \"child.xml\"";
		objDocType.AppendChild(objEntity);

		Application.ActiveDocument.AuthenticView.UpdateXMLInstanceEntities();
	}
}
```

14.3.2.6.27 WholeDocument

Property: WholeDocument as AuthenticRange (read-only)

Description
Retrieve a range object that selects the whole document.

Errors
2000 The authentic view object is no longer valid.
2005 Invalid address for the return parameter was specified.

14.3.2.6.28 XMLDataRoot

Property: XMLDataRoot as XMLData (read-only)

Description
Returns or sets the top-level XMLData element of the current document. This element typically describes the document structure and would be of kind spyXMLDataXMLDocStruct, spyXMLDataXMLEntityDocStruct or spyXMLDataDTDDocStruct..

Errors
2000 The authentic view object is no longer valid.
2005 Invalid address for the return parameter was specified.

14.3.2.7 CodeGeneratorDlg

Only available/enabled in the Enterprise edition. An error is returned, if accessed by any other version.

Properties and Methods

Standard automation properties
Application
Parent

Programming language selection properties
ProgrammingLanguage
TemplateFileName

Settings for C++ code
CPPSettings_DOMType
CPPSettings_LibraryType
CPPSettings_UseMFC
CPPSettings_GenerateVC6ProjectFile
CPPSettings_GenerateVSProjectFile

Settings for C# code
CSharpSettings_ProjectType

© 2016-2022 Altova GmbH

Dialog handling for above code generation properties
PropertySheetDialogAction

Output path selection properties
OutputPath
OutputPathDialogAction

Presentation of result
OutputResultDialogAction

Description
Use this object to configure the generation of program code for schema files. The method GenerateProgramCode expects a CodeGeneratorDlg as parameter to configure code generation as well as the associated user interactions.

14.3.2.7.1 Application

Only available/enabled in the Enterprise edition. An error is returned, if accessed by any other version.

Property: Application as Application (read-only)

Description
Access the Authentic Desktop application object.

Errors
2200 The object is no longer valid.
2201 Invalid address for the return parameter was specified.

14.3.2.7.2 CPPSettings_DOMType

Property: CPPSettings_DOMType as SPYDOMType
Only available/enabled in the Enterprise edition. An error is returned, if accessed by any other version.

Description
Defines one of the settings that configure generation of C++ code.

Errors
2200 The object is no longer valid.
2201 Invalid action passed as parameter or an invalid address was specified for the return parameter.

14.3.2.7.3 CPPSettings_GenerateVC6ProjectFile

Property: CPPSettings_GenerateVC6ProjectFile as Boolean
Only available/enabled in the Enterprise edition. An error is returned, if accessed by any other version.

Description
Defines one of the settings that configure generation of C++ code.

Errors
2200 The object is no longer valid.
2201 Invalid action passed as parameter or an invalid address was specified for the return parameter.

14.3.2.7.4 CPPSettings_GenerateGCCMakefile

Property: CPPSettings_GenerateGCCMakefile as Boolean
Only available/enabled in the Enterprise edition. An error is returned, if accessed by any other version.

Description
Creates makefiles to compile the generated code under Linux with GCC.

Errors
2200 The object is no longer valid.
2201 Invalid action passed as parameter or an invalid address was specified for the return parameter.

14.3.2.7.5 CPPSettings_GenerateVSProjectFile

Property: CSharpSettings_GenerateVSProjectFile as SPYProjectType
Only available/enabled in the Enterprise edition. An error is returned, if accessed by any other version.

Description
Defines one of the settings that configure generation of C++ code. Only `spyVisualStudio2005Project (=4)` and `spyVisualStudio2008Project (=5)` and `spyVisualStudio2010Project (=6)` are valid project types.

Errors
2200 The object is no longer valid.
2201 Invalid action passed as parameter or an invalid address was specified for the return parameter.

14.3.2.7.6 CPPSettings_LibraryType

Property: CPPSettings_LibraryType as SPYLibType
Only available/enabled in the Enterprise edition. An error is returned, if accessed by any other version.

Description
Defines one of the settings that configure generation of C++ code.

© 2016-2022 Altova GmbH

Errors

2200 The object is no longer valid.
2201 Invalid action passed as parameter or an invalid address was specified for the return parameter.

14.3.2.7.7 CPPSettings_UseMFC

Property: CPPSettings_UseMFC as Boolean
Only available/enabled in the Enterprise edition. An error is returned, if accessed by any other version.

Description
Defines one of the settings that configure generation of C++ code.

Errors

2200 The object is no longer valid.
2201 Invalid action passed as parameter or an invalid address was specified for the return parameter.

14.3.2.7.8 CSharpSettings_ProjectType

Property: CSharpSettings_ProjectType as SPYProjectType
Only available/enabled in the Enterprise edition. An error is returned, if accessed by any other version.

Description
Defines the only setting to configure generation of C# code.

Errors

2200 The object is no longer valid.
2201 Invalid action passed as parameter or an invalid address was specified for the return parameter.

14.3.2.7.9 OutputPath

Property: OutputPath as String
Only available/enabled in the Enterprise edition. An error is returned, if accessed by any other version.

Description
Selects the base directory for all generated code.

Errors

2200 The object is no longer valid.
2201 Invalid address for the return parameter was specified.

14.3.2.7.10 OutputPathDialogAction

Property: OutputPathDialogAction as SPYDialogAction
Only available/enabled in the Enterprise edition. An error is returned, if accessed by any other version.

Description
Defines how the sub-dialog for selecting the code generation output path gets handled. Set this value to *spyDialogUserInput(2)* to show the dialog with the current value of the OutputPath property as default. Use *spyDialogOK(0)* to hide the dialog from the user.

Errors
2200 The object is no longer valid.
2201 Invalid action passed as parameter or an invalid address was specified for the return parameter.

14.3.2.7.11 OutputResultDialogAction

Property: OutputResultDialogAction as SPYDialogAction
Only available/enabled in the Enterprise edition. An error is returned, if accessed by any other version.

Description
Defines how the sub-dialog that asks to show the result of the code generation process gets handled. Set this value to *spyDialogUserInput(2)* to show the dialog. Use *spyDialogOK(0)* to hide the dialog from the user.

Errors
2200 The object is no longer valid.
2201 Invalid action passed as parameter or an invalid address was specified for the return parameter.

14.3.2.7.12 Parent

Only available/enabled in the Enterprise edition. An error is returned, if accessed by any other version.

Property: Parent as Dialogs (read-only)

Description
Access the parent of the object.

Errors
2200 The object is no longer valid.
2201 Invalid address for the return parameter was specified.

14.3.2.7.13 ProgrammingLanguage

Property: ProgrammingLanguage as ProgrammingLanguage
Only available/enabled in the Enterprise edition. An error is returned, if accessed by any other version.

© 2016-2022 Altova GmbH

Description
Selects the output language for the code to be generated.

CAUTION: Setting this property to one of C++, C# or Java, changes the property TemplateFileName to the appropriate template file delivered with Authentic Desktop as well. If you want to generate C++, C# or Java code based on your own templates, set first the programming language and then select your template file.

Errors
2200 The object is no longer valid.
2201 Invalid address for the return parameter was specified.

14.3.2.7.14 PropertySheetDialogAction

Property: PropertySheetDialogAction as SPYDialogAction
Only available/enabled in the Enterprise edition. An error is returned, if accessed by any other version.

Description
Defines how the sub-dialog that configures the code generation process gets handled. Set this value to *spyDialogUserInput(2)* to show the dialog with the current values as defaults. Use *spyDialogOK(0)* to hide the dialog from the user.

Errors
2200 The object is no longer valid.
2201 Invalid action passed as parameter or an invalid address was specified for the return parameter.

14.3.2.7.15 TemplateFileName

Property: TemplateFileName as String
Only available/enabled in the Enterprise edition. An error is returned, if accessed by any other version.

Description
Selects the code generation template file. Authentic Desktop comes with template files for C++, C# or Java in the SPL folder of your installation directory.

Setting this property to one of the code generation template files of your Authentic Desktop installation automatically sets the ProgrammingLanguage property to its appropriate value.

Errors
2200 The object is no longer valid.
2201 Invalid address for the return parameter was specified.

14.3.2.8 DatabaseConnection

Properties for import and export
File or

ADOConnection or
ODBCConnection

Properties for import only
DatabaseKind
SQLSelect
AsAttributes
ExcludeKeys
IncludeEmptyElements
NumberDateTimeFormat
NullReplacement
CommentIncluded

Properties for export only
CreateMissingTables
CreateNew
TextFieldLen
DatabaseSchema

Properties for XML Schema from DB Structure generation
PrimaryKeys
ForeignKeys
UniqueKeys
SchemaExtensionType
SchemaFormat
ImportColumnsType

Description
DatabaseConnection specifies the parameters for the database connection.

Please note that the properties of the DatabaseConnection interface are referring to the settings of the import and export dialogs of Authentic Desktop.

14.3.2.8.1 ADOConnection

Property: ADOConnection as String

Description
The property ADOConnection contains a connection string. Either use this property or ODBCConnection or File to refer to a database.

Errors
No error codes are returned.

Example

```
Dim objSpyConn As DatabaseConnection
Set objSpyConn = objSpy.GetDatabaseSettings

Dim objADO As DataLinks
Set objADO = CreateObject("DataLinks")
```


© 2016-2022 Altova GmbH

```
If Not (objADO Is Nothing) Then
        Dim objConn As Connection
        Set objConn = objADO.PromptNew
        objSpyConn.ADOConnection = objConn.ConnectionString
End If
```

14.3.2.8.2 AsAttributes

Property: AsAttributes as Boolean

Description
Set AsAttributes to true if you want to initialize all import fields to be imported as attributes. Default is false and will initialize all fields to be imported as elements. This property is used only in calls to Application.GetDatabaseImportElementList.

Errors
No error codes are returned.

14.3.2.8.3 CommentIncluded

Property: CommentIncluded as Boolean

Description
This property tells whether additional comments are added to the generated XML. Default is true. This property is used only when importing from databases.

Errors
No error codes are returned.

14.3.2.8.4 CreateMissingTables

Property: CreateMissingTables as Boolean

Description
If CreateMissingTables is true, tables which are not already defined in the export database will be created during export. Default is true. This property is used only when exporting to databases.

Errors
No error codes are returned.

14.3.2.8.5 CreateNew

Property: CreateNew as Boolean

Description

Set CreateNew true if you want to create a new database on export. Any existing database will be overwritten. See also DatabaseConnection.File. Default is false. This property is used only when exporting to databases.

Errors
No error codes are returned.

14.3.2.8.6 DatabaseKind

Property: DatabaseKind as SPYDatabaseKind

Description
Select the kind of database that gets access. The default value is spyDB_Unspecified(7) and is sufficient in most cases. This property is used only when importing from databases.

Errors
No error codes are returned.

14.3.2.8.7 DatabaseSchema

Property: DatabaseSchema as String

Description
This property specifies the Schema used for export in Schema aware databases. Default is "". This property is used only when exporting to databases.

Errors
No error codes are returned.

14.3.2.8.8 ExcludeKeys

Property: ExcludeKeys as Boolean

Description
Set ExcludeKeys to true if you want to exclude all key columns from the import data. Default is false. This property is used only when importing from databases.

Errors
No error codes are returned.

14.3.2.8.9 File

Property: File as String

Description

© 2016-2022 Altova GmbH

The property File sets the path for the database during export or import. This property can only be used in conjunction with a Microsoft Access database. Either use this property or ODBCConnection or ADOConnection to refer to the database.

Errors
No error codes are returned.

14.3.2.8.10 ForeignKeys

Property: ForeignKeys as Boolean

Description
Specifies whether the Foreign Keys constraint is created or not. Default is true. This property is used only when creating a XML Schema from a DB structure.

Errors
No error codes are returned.

14.3.2.8.11 ImportColumnsType

Property: ImportColumnsType as SPYImportColumnsType

Description
Defines if column information from the DB is saved as element or attribute in the XML Schema. Default is as element. This property is used only when creating a XML Schema from a DB structure.

Errors
No error codes are returned.

14.3.2.8.12 IncludeEmptyElements

Property: IncludeEmptyElements as Boolean

Description
Set IncludeEmptyElements to false if you want to exclude all empty elements. Default is true. This property is used only when importing from databases.

Errors
No error codes are returned.

14.3.2.8.13 NullReplacement

Property: NullReplacement as String

Description
This property contains the text value that is used during import for empty elements (null values). Default is "". This property is used only when importing from databases.

Errors
No error codes are returned.

14.3.2.8.14 NumberDateTimeFormat

Property: NumberDateTimeFormat as SPYNumberDateTimeFormat

Description
The property NumberDateTimeFormat sets the format of numbers and date- and time-values. Default is spySystemLocale. This property is used only when importing from databases.

Errors
No error codes are returned.

14.3.2.8.15 ODBCConnection

Property: ODBCConnection as String

Description
The property ODBCConnection contains a ODBC connection string. Either use this property or ADOConnection or File to refer to a database.

Errors
No error codes are returned.

14.3.2.8.16 PrimaryKeys

Property: PrimaryKeys as Boolean

Description
Specifies whether the Primary Keys constraint is created or not. Default is true. This property is used only when creating a XML Schema from a DB structure.

Errors
No error codes are returned.

14.3.2.8.17 SchemaExtensionType

Property: SchemaExtensionType as SPYSchemaExtensionType

Description
Defines the Schema extension type used during the Schema generation. This property is used only when creating a XML Schema from a DB structure.

Errors
No error codes are returned.

© 2016-2022 Altova GmbH

14.3.2.8.18 SchemaFormat

Property: SchemaFormat as SPYSchemaFormat

Description
Defines the Schema format used during the Schema generation. This property is used only when creating a XML Schema from a DB structure.

Errors
No error codes are returned.

14.3.2.8.19 SQLSelect

Property: SQLSelect as String

Description
The SQL query for the import is stored in the property SQLSelect. This property is used only when importing from databases.

Errors
No error codes are returned.

14.3.2.8.20 TextFieldLen

Property: TextFieldLen as long

Description
The property TextFieldLen sets the length for created text fields during the export. Default is 255. This property is used only when exporting to databases.

Errors
No error codes are returned.

14.3.2.8.21 UniqueKeys

Property: UniqueKeys as Boolean

Description
Specifies whether the Unique Keys constraint is created or not. Default is true. This property is used only when creating a XML Schema from a DB structure.

Errors
No error codes are returned.

14.3.2.9 Dialogs

Properties and Methods

Standard automation properties
Application
Parent

Various dialog objects
CodeGeneratorDlg
FileSelectionDlg
SchemaDocumentationDlg
GenerateSampleXMLDlg
DTDSchemaGeneratorDlg
FindInFilesDlg

Description
The Dialogs object provides access to different built-in dialogs of Authentic Desktop. These dialog objects allow to initialize the fields of user dialogs before they get presented to the user or allow to simulate complete user input by your program.

14.3.2.9.1 Application

Property: Application as Application (read-only)

Description
Access the Authentic Desktop application object.

Errors
2300 The object is no longer valid.
2301 Invalid address for the return parameter was specified.

14.3.2.9.2 CodeGeneratorDlg

Only available/enabled in the Enterprise edition. An error is returned, if accessed by any other version.

Property: CodeGeneratorDlg as CodeGeneratorDlg (read-only)

Description
Get a new instance of a code generation dialog object. You will need this object to pass the necessary parameters to the code generation methods. Initial values are taken from last usage of the code generation dialog.

Errors
2300 The Dialogs object or one of its parents is no longer valid.
2301 Invalid address for the return parameter was specified.

© 2016-2022 Altova GmbH

14.3.2.9.3 FileSelectionDlg

Property: FileSelectionDlg as FileSelectionDlg (read-only)

Description
Get a new instance of a file selection dialog object.

File selection dialog objects are passed to you with the some events that signal opening or saving of documents and projects.

Errors
2300 The Dialogs object or one of its parents is no longer valid.
2301 Invalid address for the return parameter was specified.

14.3.2.9.4 Parent

Property: Parent as Application (read-only)

Description
Access the Authentic Desktop application object.

Errors
2300 The object is no longer valid.
2301 Invalid address for the return parameter was specified.

14.3.2.9.5 SchemaDocumentationDlg

Property: SchemaDocumentationDlg as SchemaDocumentationDlg (read-only)

Description
Get a new instance of a dialog object that parameterizes generation of schema documentation. See Document.GenerateSchemaDocumentation for its usage.

Errors
2300 The Dialogs object or one of its parents is no longer valid.
2301 Invalid address for the return parameter was specified.

14.3.2.9.6 GenerateSampleXMLDlg

Property: GenerateSampleXMLDlg as GenerateSampleXMLDlg (read-only)

Description
Get a new instance of a dialog object that parameterizes generation of a sample XML based on a W3C schema or DTD. See GenerateSampleXML for its usage.

Errors

2300 The Dialogs object or one of its parents is no longer valid.
2301 Invalid address for the return parameter was specified.

14.3.2.9.7 DTDSchemaGeneratorDlg

Property: DTDSchemaGeneratorDlg as DTDSchemaGeneratorDlg (read-only)

Description
Get a new instance of a dialog object that parameterizes generation of a schema or DTD. See Document.GenerateDTDOrSchemaEx for its usage.

Errors
2300 The Dialogs object or one of its parents is no longer valid.
2301 Invalid address for the return parameter was specified.

14.3.2.9.8 FindInFilesDlg

Property: FindInFilesDlg as FindInFilesDlg (read-only)

Description
Get a new instance of a dialog object that parameterizes the search (or replacement) of strings in files. See Application.FindInFiles for its usage.

Errors
2300 The Dialogs object or one of its parents is no longer valid.
2301 Invalid address for the return parameter was specified.

14.3.2.9.9 WSDLDocumentationDlg

Property: WSDLDocumentationDlg as WSDLDocumentationDlg (read-only)

Description
Get a new instance of a dialog object that parameterizes generation of WSDL documentation. See Document.GenerateWSDLDocumentation for its usage.

Errors
2300 The Dialogs object or one of its parents is no longer valid.
2301 Invalid address for the return parameter was specified.

14.3.2.9.10 WSDL20DocumentationDlg

Property: WSDL20DocumentationDlg as WSDL20DocumentationDlg (read-only)

Description
Get a new instance of a dialog object that parameterizes generation of WSDL 2.0 documentation. See Document.GenerateWSD20LDocumentation for its usage.

© 2016-2022 Altova GmbH

Errors

2300 The Dialogs object or one of its parents is no longer valid.
2301 Invalid address for the return parameter was specified.

14.3.2.9.11 XBRLDocumentationDlg

Property: XBRL20DocumentationDlg as XBRL20DocumentationDlg (read-only)

Description

Get a new instance of a dialog object that parameterizes generation of WSDL 2.0 documentation. See Document.GenerateXBRLDocumentation for its usage.

Errors

2300 The Dialogs object or one of its parents is no longer valid.
2301 Invalid address for the return parameter was specified.

14.3.2.10 Document

Properties and Methods

Standard automation properties
Application
Parent

Various document properties and methods
SetActiveDocument
Encoding
SetEncoding (obsolete)
Suggestions

XML validation
IsValid
SetExternalIsValid

Document conversion and transformation
AssignDTD
AssignSchema
AssignXSL
AssignXSLFO
ConvertDTDOrSchema
ConvertDTDOrSchemaEx
GenerateDTDOrSchema
GenerateDTDOrSchemaEx
FlattenDTDOrSchema
CreateSchemaDiagram
ExecuteXQuery
TransformXSL
TransformXSLEx
TransformXSLFO
GenerateProgramCode (Enterprise Edition only)

GenerateSchemaDocumentation
GenerateSampleXML
ConvertToWSDL20

Document export
GetExportElementList
ExportToText
ExportToDatabase
CreateDBStructureFromXMLSchema
GetDBStructureList

File saving and naming
FullName
Name
Path
GetPathName (obsolete)
SetPathName (obsolete)
Title
IsModified
Saved
SaveAs
Save
SaveInString
SaveToURL
Close

View access
CurrentViewMode
SwitchViewMode
AuthenticView
GridView
DocEditView (obsolete)

Access to XMLData
RootElement
DataRoot
CreateChild
UpdateViews
StartChanges
EndChanges
UpdateXMLData

Description
Document objects represent XML documents opened in Authentic Desktop.

Use one of the following properties to access documents that are already open Authentic Desktop:
Application.ActiveDocument
Application.Documents

Use one of the following methods to open a new document in Authentic Desktop:
Documents.OpenFile
Documents.OpenURL
Documents.OpenURLDialog

© 2016-2022 Altova GmbH

Documents.NewFile
Documents.NewFileFromText
SpyProjectItem.Open
Application.ImportFromDatabase
Application.ImportFromSchema
Application.ImportFromText
Application.ImportFromWord
Document.ConvertDTDOrSchema
Document.GenerateDTDOrSchema

14.3.2.10.1 Events

14.3.2.10.1.1 OnBeforeSaveDocument

Event: OnBeforeSaveDocument(*objDocument* as Document, *objDialog* as FileSelectionDlg)

XMLSpy scripting environment - VBScript:

```
Function On_BeforeSaveDocument(objDocument, objDialog)
End Function

' old handler - now obsolete
' return string to save to new file name
' return empty string to cancel save operation
' return nothing to save to original name
Function On_SaveDocument(objDocument, strFilePath)
End Function
```

XMLSpy scripting environment - JScript:

```
function On_BeforeSaveDocument(objDocument, objDialog)
{
}

// old handler - now obsolete
// return string to save to new file name
// return empty string to cancel save operation
// return nothing to save to original name
function On_SaveDocument(objDocument, strFilePath)
{
}
```

XMLSpy IDE Plugin:

IXMLSpyPlugIn.OnEvent (27, ...) // nEventId = 27

Description

This event gets fired on any attempt to save a document. The file selection dialog object is initialized with the name chosen for the document file. You can modify this selection. To continue saving the document leave the FileSelectionDlg.DialogAction property of *io_objDialog* at its default value spyDialogOK. To abort saving of the document set this property to spyDialogCancel.

14.3.2.10.1.2 OnBeforeCloseDocument

Event: OnBeforeCloseDocument(*objDocument* as Document)as Boolean

XMLSpy scripting environment - VBScript:
```
Function On_BeforeCloseDocument(objDocument)
	' On_BeforeCloseDocument = False  ' to prohibit closing of document
End Function
```

XMLSpy scripting environment - JScript:
```
function On_BeforeCloseDocument(objDocument)
{
	// return false;  /* to prohibit closing of document */
}
```

XMLSpy IDE Plugin:
IXMLSpyPlugIn.OnEvent (28, ...) // nEventId = 28

Description
This event gets fired on any attempt to close a document. To prevent the document from being closed return false.

14.3.2.10.1.3 OnBeforeValidate

Event: OnBeforeValidate(*objDocument* as Document, *bOnLoading* as Boolean, *bOnCommand* as Boolean) as Boolean

XMLSpy scripting environment - VBScript:
```
Function On_BeforeValidate(objDocument, bOnLoading, bOnCommand)
	On_BeforeValidate = bCancelDefaultValidation     'set by the script if necessary
End Function
```

XMLSpy scripting environment - JScript:
```
function On_BeforeValidate(objDocument, bOnLoading, bOnCommand)
{
	return bCancelDefaultValidation   //set by the script if necessary
}
```

XMLSpy IDE Plugin:
IXMLSpyPlugIn.OnEvent (32, ...) // nEventId = 32

Description
This event gets fired before the document is validated. It is possible to suppress the default validation by returning false from the event handler. In this case the script should also set the validation result using the SetExternalIsValid method.

bOnLoading is true if the event is raised on the initial validation on loading the document.

bOnCommand is true whenever the user selected the Validate command from the Toolbar or menu.

© 2016-2022 Altova GmbH

Available with TypeLibrary version 1.5

14.3.2.10.1.4 OnCloseDocument

Event: OnCloseDocument(*objDocument* as Document)

XMLSpy scripting environment - VBScript:
Function On_Close Document(*objDocument*)
End Function

XMLSpy scripting environment - JScript:
function On_Close Document(*objDocument*)
{
}

XMLSpy IDE Plugin:
IXMLSpyPlugIn.OnEvent (8, ...) // nEventId = 8

Description
This event gets fired as a result of closing a document. Do not modify the document from within this event.

14.3.2.10.1.5 OnViewActivation

Event: OnViewActivation(*objDocument* as Document, *eViewMode* as SPYViewModes, *bActivated* as Boolean)

XMLSpy scripting environment - VBScript:
Function On_ViewActivation(*objDocument*, *eViewMode*, *bActivated*)
End Function

XMLSpy scripting environment - JScript:
function On_ViewActivation(*objDocument*, *eViewMode*, *bActivated*)
{
}

XMLSpy IDE Plugin:
IXMLSpyPlugIn.OnEvent (29, ...) // nEventId = 29

Description
This event gets fired whenever a view of a document becomes visible (i.e. becomes the active view) or invisible (i.e. another view becomes the active view or the document gets closed). However, the first view activation event after a document gets opened cannot be received, since there is no document object to get the event from. Use the Application.OnDocumentOpened event instead.

14.3.2.10.2 Application

Property: Application as Application (read-only)

Description
Accesses the Authentic Desktop application object.

Errors
1400 The object is no longer valid.
1407 Invalid address for the return parameter was specified.

14.3.2.10.3 AssignDTD

Method: AssignDTD(*strDTDFile* as String, *bDialog* as Boolean)

Description
The method places a reference to the DTD file "strDTDFile" into the document. Note that no error occurs if the file does not exist, or is not accessible. If bDialog is true Authentic Desktop presents a dialog to set the file.

Errors
1400 The object is no longer valid.
1409 You are not allowed to assign a DTD to the document.

14.3.2.10.4 AssignSchema

Method: AssignSchema (*strSchemaFile* as String, *bDialog* as Boolean)

Description
The method places a reference to the schema file "strSchemaFile" into the document. Note that no error occurs if the file does not exist or is not accessible. If bDialog is true Authentic Desktop presents a dialog to set the file.

Errors
1400 The object is no longer valid.
1409 You are not allowed to assign a schema file to the document.

14.3.2.10.5 AssignXSL

Method: AssignXSL (*strXSLFile* as String, *bDialog* as Boolean)

Description
The method places a reference to the XSL file "strXSLFile" into the document. Note that no error occurs if the file does not exist or is not accessible. If bDialog is true Authentic Desktop presents a dialog to set the file.

Errors
1400 The object is no longer valid.
1409 You are not allowed to assign an XSL file to the document.

© 2016-2022 Altova GmbH

14.3.2.10.6 AssignXSLFO

Method: AssignXSLFO (*strXSLFOFile* as String, *bDialog* as Boolean)

Description
The method places a reference to the XSLFO file "strXSLFile" into the document. Note that no error occurs if the file does not exist or is not accessible. If bDialog is true Authentic Desktop presents a dialog to set the file.

Errors
1400 The object is no longer valid.
1409 You are not allowed to assign an XSL file to the document.

14.3.2.10.7 AsXMLString

Property: AsXMLString as String

Description
This property can be used to get or set the document content.

Errors
1400 The document object is no longer valid.
1404 Cannot create XMLData object.
1407 View mode cannot be switched.

14.3.2.10.8 AuthenticView

Method: AuthenticView as AuthenticView (read-only)

Description
Returns an object that gives access to properties and methods specific to Authentic view. The object returned is only valid if the current document is opened in Authentic view mode. The lifetime of an object ends with the next view switch. Any attempt to access objects or any of its children afterwards will result in an error indicating that the object is invalid.

Errors
1400 The object is no longer valid.
1417 Document needs to be open in authentic view mode.

Examples

```
' ---------------------------------------
' XMLSpy scripting environment - VBScript
' secure access to authentic view object
' ---------------------------------------
Dim objDocument
Set objDocument = Application.ActiveDocument
If (Not objDocument Is Nothing) Then
        ' we have an active document, now check for view mode
        If (objDocument.CurrentViewMode <> spyViewAuthentic) Then
```

```
            If (Not objDocument.SwitchViewMode (spyViewAuthentic)) Then
                MsgBox "Active document does not support authentic view mode"
            Else
                ' now it is safe to access the authentic view object
                Dim objAuthenticView
                Set objAuthenticView = objDocument.AuthenticView
                ' now use the authentic view object

            End If
        End If
Else
        MsgBox "No document is open"
End If
```

14.3.2.10.9 Close

Method: Close (*bDiscardChanges* as Boolean)

Description
To close the document call this method. If bDiscardChanges is true and the document is modified, the document will be closed but not saved.

Errors
1400 The object is no longer valid.
1401 Document needs to be saved first.

14.3.2.10.10 ConvertDTDOrSchema

Method: ConvertDTDOrSchema (*nFormat* as SPYDTDSchemaFormat, *nFrequentElements* as SPYFrequentElements)

Parameters
nFormat
Sets the schema output format to DTD or W3C.

nFrequentElements
Create complex elements as elements or complex types.

Description
ConvertDTDOrSchema takes an existing schema format and converts it into a different format.
For a finer tuning of DTD/XSD conversion, use ConvertDTDOrSchemaEx.

Errors
1400 The object is no longer valid.
1412 Error during conversion. In the case of DTD to DTD or XSD to XSD conversion, the following errors are returned: *DTD to DTD conversion is not supported. Please use function FlattenDTDOrSchema instead* and

© 2016-2022 Altova GmbH

Schema to schema conversion is not supported. Please use function FlattenDTDOrSchema instead.

14.3.2.10.11 ConvertDTDOrSchemaEx

Method: ConvertDTDOrSchemaEx (*nFormat* as SPYDTDSchemaFormat, *nFrequentElements* as SPYFrequentElements, sOutputPath as String, nOutputPathDialogAction as SPYDialogAction)

Parameters

nFormat
Sets the schema output format to DTD, or W3C.

nFrequentElements
Create complex elements as elements or complex types.

`sOutputPath`
The file path for the newly generated file.

`nOutputPathDialogAction`
Defines the dialog interaction for this call.

Description

`ConvertDTDOrSchemaEx` takes an existing schema format and converts it into a different format.

Errors

1400 The object is no longer valid.
1412 Error during conversion. In the case of DTD to DTD or XSD to XSD conversion, the following errors are returned: *DTD to DTD conversion is not supported. Please use function FlattenDTDOrSchema instead* and *Schema to schema conversion is not supported. Please use function FlattenDTDOrSchema instead.*

14.3.2.10.12 ConvertToWSDL20

Method: ConvertToWSDL20 (*sFilePath* as String, *bShowDialogs* as Boolean)

Parameters

sFilePath
This specifies the file name of the converted WSDL. In case the source WSDL includes files which also must be converted, then only the directory part of the given path is used and the file names are generated automatically.

bShowDialogs
Defines whether file/folder selection dialogs are shown.

Description
Converts the WSDL 1.1 document to a WSDL 2.0 file. It will also convert any referenced WSDL files that are referenced from within this document. Note that this functionality is limited to WSDL View only. See Document.CurrentViewMode. and SPYViewModes.

Errors
1400 The document object is no longer valid.
1407 Invalid parameters have been passed or an empty file name has been specified as output target.
1417 The document is not opened in WSDL view, maybe it is not an '.wsdl' file.
1421 Feature is not available in this edition.
1433 WSDL 1.1 to WSDL 2.0 conversion failed.

14.3.2.10.13 CreateChild

Method: CreateChild (*nKind* as SPYXMLDataKind) as XMLData

Return Value
The method returns the new XMLData object.

Description
To create a new XMLData object use the CreateChild() method.

Errors
1400 The object is no longer valid.
1404 Cannot create XMLData object.
1407 Invalid address for the return parameter was specified.

14.3.2.10.14 CreateDBStructureFromXMLSchema

Method: CreateDBStructureFromXMLSchema (*pDatabase* as DatabaseConnection, *pTables* as ElementList, *bDropTableWithExistingName* as Boolean) as String

Description
CreateDBStructureFromXMLSchema exports the given tables to the specified database. The function returns the SQL statements that were necessary to perform the changes.

See also GetDBStructureList.

Errors
1429 Database selection missing.
1430 Document export failed.

© 2016-2022 Altova GmbH

14.3.2.10.15 CreateSchemaDiagram

Method: CreateSchemaDiagram (*nKind* as SPYSchemaDefKind, *strName* as String, *strFile* as String)

Return Value
None.

Description
The method creates a diagram of the schema type strName of kind nKind and saves the output file into strFile. Note that this functionality is limited to Schema View only. See Document.CurrentViewMode. and SPYViewModes.

Errors
1400 The object is no longer valid.
1414 Failed to save diagram.
1415 Invalid schema definition type specified.

14.3.2.10.16 CurrentViewMode

Method: CurrentViewMode as SPYViewModes

Description
The property holds the current view mode of the document. See also Document.SwitchViewMode.

Errors
1400 The object is no longer valid.
1407 Invalid address for the return parameter was specified.

14.3.2.10.17 DataRoot

Property: DataRoot as XMLData (read-only)

Description
This property provides access to the document's first XMLData object of type *spyXMLDataElement*. This is typically the root element for all document content data. See XMLSpyDocument.RootElement to get the root element of the whole document including XML prolog data. If the CurrentViewMode is not *spyViewGrid* or *spyViewAuthentic* an UpdateXMLData may be necessary to get access to the latest XMLData.

Errors
1400 The document object is no longer valid.
1407 Invalid address for the return parameter was specified.

14.3.2.10.18 DocEditView

Method: DocEditView as DocEditView

Description

Holds a reference to the current Authentic View object.

Errors
1400 The object is no longer valid.
1407 Invalid address for the return parameter was specified.
1417 Document needs to be open in authentic view mode.

14.3.2.10.19 Encoding

Property: Encoding as String

Description
This property provides access to the document's encoding value. However, this property can only be accessed when the document is opened in *spyViewGrid*, *spyViewText* or *spyViewAuthentic*. See CurrentViewMode on how to detect that a document's actual view mode.

This property makes the method SetEncoding obsolete.

Possible values are, for example:

8859-1,
8859-2,
ASCII, ISO-646,
850,
1252,
1255,
SHIFT-JIS, MS-KANJI,
BIG5, FIVE,
UTF-7,
UTF-8,
UTF-16

Errors
1400 The document object is no longer valid.
1407 Invalid address for the return parameter was specified.
1416 Operation not supported in current view mode.

14.3.2.10.20 EndChanges

Method: EndChanges()

Description
Use the method EndChanges to display all changes since the call to Document.StartChanges.

Errors
1400 The object is no longer valid.

© 2016-2022 Altova GmbH

14.3.2.10.21 ExecuteXQuery

Method: ExecuteXQuery (*strXMLFileName* as String)

Description
Execute the XQuery statements contained in the document of the document object. Either an XQuery execution or an XQuery Update is performed depending on the file extension of the document. Use the XML file specified in the argument as the XML target document that the XQuery document processes.

- If the document has an XQuery file extension as defined in the Options dialog of Authentic Desktop, then an XQuery execution is performed. By default: `.xq`, `.xql`, and `.xquery` are set as XQuery file extensions in Authentic Desktop.
- If the document has an XQuery Update file extension as defined in the Options dialog of Authentic Desktop, then an XQuery Update action is performed. By default: `.xqu` is set as an XQuery Update file extension in Authentic Desktop.

If your XQuery script does not use an XML source, set the parameter `strXMLFileName` to an empty string.

Errors

1400	The document object is no longer valid.
1423	XQuery transformation error.
1424	Not all files required for operation could be loaded. Most likely, the file specified in `strXMLFileName` does not exist or is not valid.

14.3.2.10.22 ExportToDatabase

Method: ExportToDatabase (*pFromChild* as XMLData, *pExportSettings* as ExportSettings, *pDatabase* as DatabaseConnection)

Description
ExportToDatabase exports the XML document starting with the element pFromChild. The parameter pExportSettings defines the behaviour of the export (see Application.GetExportSettings). The parameter pDatabase specifies the destination of the export (see Application.GetDatabaseSettings). `UpdateXMLData()` might be indirectly needed as you have to pass the `XMLData` as parameter to this function.

Errors

1400	The object is no longer valid.
1407	Invalid parameter or invalid address for the return parameter was specified.
1416	Error during export.
1429	Database selection missing.
1430	Document export failed.

Example

```
Dim objDoc As Document
Set objDoc = objSpy.ActiveDocument
```

```
'set the behaviour of the export with ExportSettings
Dim objExpSettings As ExportSettings
Set objExpSettings = objSpy.GetExportSettings

'set the destination with DatabaseConnection
Dim objDB As DatabaseConnection
Set objDB = objSpy.GetDatabaseSettings

objDB.CreateMissingTables = True
objDB.CreateNew = True
objDB.File = "C:\Export.mdb"

objDoc.ExportToDatabase objDoc.RootElement, objExpSettings, objDB
If Err.Number <> 0 Then
        a = MsgBox("Error: " & (Err.Number - vbObjectError) & Chr(13) &
                        "Description: " & Err.Description)
End If
```

14.3.2.10.23 ExportToText

Method: ExportToText (*pFromChild* as XMLData, *pExportSettings* as ExportSettings, *pTextSettings* as TextImportExportSettings)

Description
ExportToText exports tabular information from the document starting at pFromChild into one or many text files. Columns of the resulting tables are generated in alphabetical order of the column header names. Use GetExportElementList to learn about the data that will be exported. The parameter pExportSettings defines the specifics for the export. Set the property ExportSettings.ElementList to the - possibly modified - list returned by GetExportElementList to avoid exporting all contained tables. The parameter pTextSettings defines the options specific to text export and import. You need to set the property TextImportExportSettings.DestinationFolder before you call ExportToText. `UpdateXMLData()` might be indirectly needed as you have to pass the `XMLData` as parameter to this function.

Errors

1400	The object is no longer valid.
1407	Invalid parameter or invalid address for the return parameter was specified.
1416	Error during export.
1430	Document export failed.

Example

```
' ----------------------------------------------------
' VBA client code fragment - export document to text files
' ----------------------------------------------------
        Dim objDoc As Document
        Set objDoc = objSpy.ActiveDocument

        Dim objExpSettings As ExportSettings
        Set objExpSettings = objSpy.GetExportSettings
        objExpSettings.ElementList = objDoc.GetExportElementList(
```


© 2016-2022 Altova GmbH

```
                                                    objDoc.RootElement,
                                                    objExpSettings)

    Dim objTextExp As TextImportExportSettings
    Set objTextExp = objSpy.GetTextExportSettings
    objTextExp.HeaderRow = True
    objTextExp.DestinationFolder = "C:\Exports"

    On Error Resume Next
    objDoc.ExportToText objDoc.RootElement, objExpSettings, objTextExp

    If Err.Number <> 0 Then
            a = MsgBox("Error: " & (Err.Number - vbObjectError) & Chr(13) & "Description: "
            & Err.Description)
    End If
```

14.3.2.10.24 FlattenDTDOrSchema

Method: FlattenDTDOrSchema (sOutputPath as String, nOutputPathDialogAction as SPYDialogAction)

Parameters

`sOutputPath`
The file path for the newly generated file.

`nOutputPathDialogAction`
Defines the dialog interaction for this call.

Description

`FlattenDTDOrSchema` takes an existing DTD or schema, generates a flattened file, and saves the generated file at the specified location. In the case of DTDs, flattening removes parameter entities and produces a single DTD from a collection of modules; sections marked `IGNORE` are suppressed and unused parameter entities are deleted. When an XML Schema is flattened, (i) the components of all included schemas are added as global components of the active schema, and (ii) included schemas are deleted.

Errors

1400 The object is no longer valid.
1412 Error during conversion.

14.3.2.10.25 FullName

Property: FullName as String

Description

This property can be used to get or set the full file name - including the path - to where the document gets saved. The validity of the name is not verified before the next save operation.

This property makes the methods GetPathName and SetPathName obsolete.

Errors
1400 The document object is no longer valid.
1402 Empty string has been specified as full file name.

14.3.2.10.26 GenerateDTDOrSchema

Method: GenerateDTDOrSchema (*nFormat* as SPYDTDSchemaFormat, *nValuesList* as integer, *nDetection* as SPYTypeDetection, *nFrequentElements* as SPYFrequentElements)

Parameters
nFormat
Sets the schema output format to DTD, or W3C.

nValuesList
Generate not more than this amount of enumeration-facets per type. Set to -1 for unlimited.

nDetection
Specifies granularity of simple type detection.

nFrequentElements
Shall the types for all elements be defined as global? Use that value *spyGlobalComplexType* to define them on global scope. Otherwise, use the value *spyGlobalElements*.

Description
Use this method to automatically generate a DTD or schema for the current XML document.
For a finer tuning of DTD / schema generation, use GenerateDTDOrSchemaEx.
Note that this functionality is not available in ZIP View only. See Document.CurrentViewMode. and SPYViewModes.

Errors
1400 The object is no longer valid.
1407 Invalid parameter or invalid address for the return parameter was specified.

14.3.2.10.27 GenerateDTDOrSchemaEx

Method: GenerateDTDOrSchemaEx (*objDlg* as DTDSchemaGeneratorDlg) as Document

Description
Use this method to automatically generate a DTD or schema for the current XML document. A DTDSchemaGeneratorDlg object is used to pass information to the schema/DTD generator. The generation process can be configured to allow user interaction or run without further user input.
Note that this functionality is not available in ZIP View only. See Document.CurrentViewMode. and SPYViewModes.

Errors
1400 The object is no longer valid.
1407 Invalid parameter or invalid address for the return parameter was specified.

© 2016-2022 Altova GmbH

14.3.2.10.28 GenerateProgramCode

Method: GenerateProgramCode (*objDlg* as CodeGeneratorDlg)
Only available/enabled in the Enterprise edition. An error is returned, if accessed by any other version.

Description
Generate Java, C++ or C# class files from the XML Schema definitions in your document. A CodeGeneratorDlg object is used to pass information to the code generator. The generation process can be configured to allow user interaction or run without further user input.

Errors
1400 The document object is no longer valid.
1407 An empty file name has been specified.
1421 Feature not available in this edition

14.3.2.10.29 GenerateSampleXML

Method: GenerateSampleXML (*objDlg* as GenerateSampleXMLDlg) as Document

Description
Generates a sample XML if the document is a schema or DTD. Use Dialogs.GenerateSampleXMLDlg to get an initialized set of options.

Available with TypeLibrary version 1.5

Errors
1400 The document object is no longer valid.

14.3.2.10.30 GenerateSchemaDocumentation

Method: GenerateSchemaDocumentation (*objDlg* as SchemaDocumentationDlg)

Description
Generate documentation for a schema definition file in HTML, MS-Word, or RTF format. The parameter objDlg is used to parameterize the generation process. Use Dialogs.SchemaDocumentationDlg to get an initialized set of options. As a minimum, you will need to set the property SchemaDocumentationDlg.OutputFile before starting the generation process. Note that this functionality is limited to Schema View only. See Document.CurrentViewMode and SPYViewModes.

Errors
1400 The document object is no longer valid.
1407 Invalid parameters have been passed or an empty file name has been specified as output target.
1417 The document is not opened in schema view, maybe it is not an '.xsd' file.
1421 Feature is not available in this edition.
1422 Error during generation

14.3.2.10.31 GenerateWSDL20Documentation

Method: GenerateWSDL20Documentation (*objDlg* as WSDL20DocumentationDlg)

Description
Generate documentation for a WSDL definition file in HTML, MS-Word, or RTF format. The parameter objDlg is used to parameterize the generation process. Use Dialogs.WSDL20DocumentationDlg to get an initialized set of options. As a minimum, you will need to set the property WSDL20DocumentationDlg.OutputFile before starting the generation process. Note that this functionality is limited to WSDL View only. See Document.CurrentViewMode and SPYViewModes.

Errors

1400	The document object is no longer valid.
1407	Invalid parameters have been passed or an empty file name has been specified as output target.
1417	The document is not opened in schema view, maybe it is not an '.xsd' file.
1421	Feature is not available in this edition.
1422	Error during generation

14.3.2.10.32 GenerateWSDLDocumentation

Method: GenerateWSDLDocumentation (*objDlg* as WSDLDocumentationDlg)

Description
Generate documentation for a WSDL definition file in HTML, MS-Word, or RTF format. The parameter objDlg is used to parameterize the generation process. Use Dialogs.WSDLDocumentationDlg to get an initialized set of options. As a minimum, you will need to set the property WSDLDocumentationDlg.OutputFile before starting the generation process. Note that this functionality is limited to WSDL View only. See Document.CurrentViewMode and SPYViewModes.

Errors

1400	The document object is no longer valid.
1407	Invalid parameters have been passed or an empty file name has been specified as output target.
1417	The document is not opened in schema view, maybe it is not an '.xsd' file.
1421	Feature is not available in this edition.
1422	Error during generation

14.3.2.10.33 GenerateXBRLDocumentation

Method: GenerateXBRLDocumentation (*objDlg* as XBRLDocumentationDlg)

Description

© 2016-2022 Altova GmbH

Generate documentation for a WSDL definition file in HTML, MS-Word, or RTF format. The parameter objDlg is used to parameterize the generation process. Use Dialogs.XBRLDocumentationDlg to get an initialized set of options. As a minimum, you will need to set the property XBRLDocumentationDlg.OutputFile before starting the generation process. Note that this functionality is limited to XBRL View only. See Document.CurrentViewMode and SPYViewModes.

Errors

1400 The document object is no longer valid.
1407 Invalid parameters have been passed or an empty file name has been specified as output target.
1417 The document is not opened in schema view, maybe it is not an '.xsd' file.
1421 Feature is not available in this edition.
1422 Error during generation

14.3.2.10.34 GetDBStructureList

Method: GetDBStructureList (*pDatabase* as DatabaseConnection) as ElementList

Description

GetDBStructureList creates a collection of elements from the Schema document for which tables in the specified database are created. The function returns a collection of ElementListItems where the properties ElementListItem.Name contain the names of the tables.

See also CreateDBStructureFromXMLSchema.

Errors

1400 The object is no longer valid.
1427 Failed creating parser for the specified XML.
1428 Export of element list failed.
1429 Database selection missing.

14.3.2.10.35 GetExportElementList

Method: GetExportElementList (*pFromChild* as XMLData, *pExportSettings* as ExportSettings) as ElementList

Description

GetExportElementList creates a collection of elements to export from the document, depending on the settings in pExportSettings and starting from the element pFromChild. The function returns a collection of ElementListItems where the properties ElementListItem.Name contain the names of the tables that can be exported from the document. The property ElementListItem.FieldCount contains the number of columns in the table. The property ElementListItem.RecordCount contains the number of records in the table. The property ElementListItem.ElementKind is unused. `UpdateXMLData()` might be indirectly needed as you have to pass the `XMLData` as parameter to this function.

Errors

1400 The object is no longer valid.
1407 Invalid parameter or invalid address for the return parameter was specified.

1427 Failed creating parser for the specified XML.
1428 Export of element list failed.

14.3.2.10.36 GetPathName (obsolete)

Superseded by Document.FullName

```
// ----- javascript sample -----
// instead of:
// strPathName = Application.ActiveDocument.GetPathName();
// use now:
strPathName = Application.ActiveDocument.FullName;
```

Method: GetPathName() as String

Description
The method GetPathName gets the path of the active document.

See also Document.SetPathName (obsolete).

14.3.2.10.37 GridView

Property: GridView as GridView

Description
This property provides access to the grid view functionality of the document.

Errors
1400 The object is no longer valid.
1407 Invalid address for the return parameter was specified.
1417 Document needs to be open in enhanced grid view mode.

14.3.2.10.38 IsModified

Property: IsModified as Boolean

Description
True if the document is modified.

Errors
1400 The object is no longer valid.
1407 Invalid address for the return parameter was specified.

© 2016-2022 Altova GmbH

14.3.2.10.39 IsValid

Method: IsValid (*strError* as Variant) as Boolean

Return Value
True if the document is valid, false if not. To call `IsValid()`, the application GUI must be visible. (If you wish to validate without the GUI being visible, please use Altova RaptorXML Server.)

Description
`IsValid` validates the document against its associated schema or DTD. `strError` gives you the same error message as when you validate the file within the GUI.

Errors
1400 The object is no longer valid.
1407 Invalid parameter or invalid address for the return parameter was specified.
1408 Unable to validate file.

14.3.2.10.40 IsValidEx

Method: IsValidEx (*nXSDVersion* as `SPYValidateXSDVersion`, *nErrorLimit* as `int`, *nErrorFormat* as `SPYValidateErrorFormat`, out *strError* as Variant) as Boolean

Return Value
True if the document is valid, false if not.

Description
`IsValidEx` validates the document against its associated schema or DTD.

In parameters:
`nXSDVersion` which is an enumeration value of `SPYValidateXSDVersion` that selects the XSD version to validate against.
`nErrorLimit` which is an integer. Values must be `1` to `999`.
`nErrorFormat` which is an enumeration value of `SPYValidateErrorFormat` that selects the XSD version to validate against.

Out parameter:
`strError` is the error message, and is the same as that received when validating the file within the GUI.

Errors
1400 The object is no longer valid.
1407 Invalid parameter or invalid address for the return parameter was specified.
1408 Unable to validate file.

14.3.2.10.41 IsWellFormed

Method: IsWellFormed (*pData* as XMLData, *bWithChildren* as Boolean, *strError* as Variant, *nErrorPos* as Variant, *pBadXMLData* as Variant) as Boolean

Return Value
True if the document is well formed.

Description
IsWellFormed checks the document for well-formedness starting at the element pData.

If the document is not well formed, strError contains an error message, nErrorPos the position in the file and pBadXMLData holds a reference to the element which breaks the well-formedness. These out-parameters are defined as VARIANTs to support scripting languages like VBScript.

Errors
1400 The object is no longer valid.
1407 Invalid parameter or invalid address for the return parameter was specified.

Example

See IsValid.

14.3.2.10.42 Name

Property: Name as String (read-only)

Description
Use this property to retrieve the name - not including the path - of the document file. To change the file name for a document use the property FullName.

Errors
1400 The document object is no longer valid.
1407 Invalid address for the return parameter was specified.

14.3.2.10.43 Parent

Property: Parent as Documents (read-only)

Description
Access the parent of the document object.

Errors
1400 The document object is no longer valid.
1407 Invalid address for the return parameter was specified.

Property: Parent as Application (read-only)

© 2016-2022 Altova GmbH

14.3.2.10.44 Path

Property: Path as String (read-only)

Description
Use this property to retrieve the path - not including the file name - of the document file. To change the file name and path for a document use the property FullName.

Errors
1400 The document object is no longer valid.
1407 Invalid address for the return parameter was specified.

14.3.2.10.45 RootElement

Property: RootElement as XMLData (read-only)

Description
The property RootElement provides access to the root element of the XML structure of the document including the XML prolog data. To access the first element of a document's content navigate to the first child of kind *spyXMLDataElement* or use the Document.DataRoot property. If the CurrentViewMode is not *spyViewGrid* or *spyViewAuthentic* an UpdateXMLData may be necessary to get access to the latest XMLData.

Errors
1400 The document object is no longer valid.
1407 Invalid address for the return parameter was specified.

14.3.2.10.46 Save

Method: Save()

Description
The method writes any modifications of the document to the associated file. See also Document.FullName.

Errors
1400 The document object is no longer valid.
1407 An empty file name has been specified.
1403 Error when saving file, probably the file name is invalid.

14.3.2.10.47 SaveAs

Method: SaveAs (*strFileName* as String)

Description
Save the document to the file specified. If saving was successful, the FullName property gets set to the specified file name.

Errors
1400 The document object is no longer valid.
1407 An empty file name has been specified.
1403 Error when saving file, probably the file name is invalid.

14.3.2.10.48 Saved

Property: Saved as Boolean (read-only)

Description
This property can be used to check if the document has been saved after the last modifications. It returns the negation of IsModified.

Errors
1400 The document object is no longer valid.
1407 Invalid address for the return parameter was specified.

14.3.2.10.49 SaveInString

Method: SaveInString (*pData* as XMLData, *bMarked* as Boolean) as String

Parameters
pData
XMLData element to start. Set pData to Document.RootElement if you want to copy the complete file.

bMarked
If bMarked is true, only the elements selected in the grid view are copied.

Return Value
Returns a string with the XML data.

Description
SaveInString starts at the element pData and converts the XMLData objects to a string representation. `UpdateXMLData()` might be indirectly needed as you have to pass the `XMLData` as parameter to this function.

Errors
1400 The object is no longer valid.
1407 Invalid parameter or invalid address for the return parameter was specified.

14.3.2.10.50 SaveToURL

Method: SaveToURL (*strURL* as String, *strUser* as String, *strPassword* as String)

Return Value

Description

© 2016-2022 Altova GmbH

SaveToURL() writes the document to the URL strURL. This method does not set the permanent file path of the document.

Errors

1400 The object is no longer valid.
1402 Invalid URL specified.
1403 Error while saving to URL.

14.3.2.10.51 SetActiveDocument

Method: SetActiveDocument()

Description

The method sets the document as the active and brings it to the front.

Errors

1400 The object is no longer valid.

14.3.2.10.52 SetEncoding (obsolete)

Superseded by Document.Encoding

```
// ----- javascript sample -----
// instead of:
// Application.ActiveDocument.SetEncoding("UTF-16");
// use now:
Application.ActiveDocument.Encoding = "UTF-16";
```

Method: SetEncoding (*strEncoding* as String)

Description

SetEncoding sets the encoding of the document like the menu item "File/Encoding..." in Authentic Desktop. Possible values for strEncoding are, for example:

8859-1,
8859-2,
ASCII, ISO-646,
850,
1252,
1255,
SHIFT-JIS, MS-KANJI,
BIG5, FIVE,
UTF-7,
UTF-8,
UTF-16

14.3.2.10.53 SetExternalIsValid

Method: SetExternalIsValid (*bValid* as Boolean)

Parameters

bValid
Sets the result of an external validation process.

Description
The internal information set by this method is only queried on cancelling the default validation in any OnBeforeValidate handler.

Available with TypeLibrary version 1.5

Errors
1400 The object is no longer valid.

14.3.2.10.54 SetPathName (obsolete)

Superseded by Document.FullName

```
// ----- javascript sample -----
// instead of:
// Application.ActiveDocument.SetPathName("C:\\myXMLFiles\\test.xml");
// use now:
Application.ActiveDocument.FullName = "C:\\myXMLFiles\\test.xml";
```

Method: SetPathName (*strPath* as String)

Description
The method SetPathName sets the path of the active document. SetPathName only copies the string and does not check if the path is valid. All succeeding save operations are done into this file.

14.3.2.10.55 StartChanges

Method: StartChanges()

Description
After StartChanges is executed Authentic Desktop will not update its editor windows until Document.EndChanges is called. This increases performance of complex tasks to the XML structure.

Errors
1400 The object is no longer valid.

© 2016-2022 Altova GmbH

14.3.2.10.56 Suggestions

Property: Suggestions as Array

Description
This property contains the last valid user suggestions for this document. The XMLSpy generated suggestions can be modified before they are shown to the user in the OnBeforeShowSuggestions event.

Errors
1400 The object is no longer valid.
1407 Invalid parameter or invalid address for the return parameter was specified.

14.3.2.10.57 SwitchViewMode

Method: SwitchViewMode (*nMode* as SPYViewModes) as Boolean

Return value
Returns true if view mode is switched.

Description
The method sets the current view mode of the document in Authentic Desktop. See also Document.CurrentViewMode.

Errors
1400 The object is no longer valid.
1407 Invalid address for the return parameter was specified.
1417 Invalid view mode specified.

14.3.2.10.58 TextView

Property: TextView as TextView

Description
This property provides access to the text view functionality of the document.

Errors
1400 The object is no longer valid.
1407 Invalid address for the return parameter was specified.

14.3.2.10.59 Title

Property: Title as String (read-only)

Description
Title contains the file name of the document. To get the path and filename of the file use FullName.

Errors

1400 The document object is no longer valid.
1407 Invalid address for the return parameter was specified.

14.3.2.10.60 TransformXSL

Method: TransformXSL()

Description
TransformXSL processes the XML document via the associated XSL file. See Document.AssignXSL on how to place a reference to a XSL file into the document.

Errors
1400 The document object is no longer valid.
1411 Error during transformation process.

14.3.2.10.61 TransformXSLEx

Method: TransformXSLEx(*nAction* as SPYDialogAction)

Description
TransformXSLEx processes the XML document via the associated XSL file. The parameter specifies whether a dialog asking for the result document name should pop up or not. See Document.AssignXSL on how to place a reference to a XSL file into the document.

Errors
1400 The document object is no longer valid.
1411 Error during transformation process.

14.3.2.10.62 TransformXSLFO

Method: TransformXSLFO()

Description
TransformXSLFO processes the XML document via the associated XSLFO file. See AssignXSLFO on how to place a reference to a XSLFO file into the document. You need to assign a FOP processor to Authentic Desktop before you can use this method.

Errors
1400 The document object is no longer valid.
1411 Error during transformation process.

14.3.2.10.63 TreatXBRLInconsistenciesAsErrors

Property: TreatXBRLInconsistenciesAsErrors as Boolean

Description

© 2016-2022 Altova GmbH

If this is set to `true` the `Document.IsValid()` method will return `false` for XBRL instances containing inconsistencies as defined by the XBRL Specification. The default value of this property is `false`.

Errors
1400 The document object is no longer valid.
1407 Invalid address for the return parameter was specified.

14.3.2.10.64 UpdateViews

Method: UpdateViews()

Description
To redraw the Enhanced Grid View and the Tree View call UpdateViews. This can be important after you changed the XMLData structure of a document. This method does not redraw the text view of Authentic Desktop.

Errors
1400 The document object is no longer valid.

14.3.2.10.65 UpdateXMLData

Method: UpdateXMLData() as Boolean

Description
The XMLData tree is updated from the current view. Please note that this can fail in case of the TextView if the current XML text is not well-formed. This is not necessary if CurrentViewMode is *spyViewGrid* or *spyViewAuthentic* because these views keep the XMLData updated.

Available with TypeLibrary version 1.5

Errors
1400 The document object is no longer valid.

14.3.2.11 Documents

Properties
Count
Item

Methods
NewAuthenticFile
NewFile
NewFileFromText

OpenAuthenticFile
OpenFile
OpenURL
OpenURLDialog

Description
This object represents the set of documents currently open in Authentic Desktop. Use this object to open further documents or iterate through already opened documents.

Examples

```
' ---------------------------------------
' XMLSpy scripting environment - VBScript
' iterate through open documents
' ---------------------------------------
Dim objDocuments
Set objDocuments = Application.Documents

For Each objDoc In objDocuments
        'do something useful with your document
        objDoc.SetActiveDocument()
Next
```

```
// ---------------------------------------
// XMLSpy scripting environment - JScript
// close all open documents
// ---------------------------------------
for (var iter = new Enumerator (Application.Documents);
    ! iter.atEnd();
    iter.moveNext())
{
        // MsgBox ("Closing file " + iter.item().Name);
        iter.item().Close (true);
}
```

14.3.2.11.1 Count

Property: Count as long

Description
Count of open documents.

Errors
1600 Invalid `Documents` object
1601 Invalid input parameter

© 2016-2022 Altova GmbH

14.3.2.11.2 Item

Method: Item (*n* as long) as Document

Description
Gets the document with the index n in this collection. Index is 1-based.

Errors
1600 Invalid `Documents` object
1601 Invalid input parameter

14.3.2.11.3 NewAuthenticFile

Method: NewAuthenticFile (*strSPSPath* as String, *strXMLPath* as String) as Document

Parameters
strSPSPath
The path to the SPS document.

strXMLPath
The new XML document name.

Return Value
The method returns the new document.

Description
NewAuthenticFile creates a new XML file and opens it in Authentic View using SPS design strSPSPath.

14.3.2.11.4 NewFile

Method: NewFile (*strFile* as String, *strType* as String) as Document

Parameters
strFile
Full path of new file.

strType
Type of new file as string (i.e. "xml", "xsd", ...)

Return Value
Returns the new file.

Description
NewFile creates a new file of type strType (i.e. "xml"). The newly created file is also the ActiveDocument.

14.3.2.11.5 NewFileFromText

Method: NewFileFromText (*strText* as String, *strType* as String) as Document

Parameters
strText
The content of the new document in plain text.

strType
Type of the document to create (i.e. "xml").

Return Value
The method returns the new document.

Description
NewFileFromText creates a new document with strText as its content.

14.3.2.11.6 OpenAuthenticFile

Method: OpenAuthenticFile (*strSPSPath* as String, *strXMLPath* as String) as Document

Parameters
strSPSPath
The path to the SPS document.

strXMLPath
The path to the XML document (can be empty).

Return Value
The method returns the new document.

Description
OpenAuthenticFile opens an XML file or database in Authentic View using SPS design strSPSPath.

14.3.2.11.7 OpenFile

Method: OpenFile (*strPath* as String, *bDialog* as Boolean) as Document

Parameters
strPath
Path and file name of file to open.

bDialog
Show dialogs for user input.

Return Value
Returns the opened file on success.

© 2016-2022 Altova GmbH

Description
OpenFile opens the file strPath. If bDialog is TRUE, a file-dialog will be displayed.

Example

```
Dim objDoc As Document
Set objDoc = objSpy.Documents.OpenFile(strFile, False)
```

14.3.2.11.8 OpenURL

Method: OpenURL (*strURL* as String, *nURLType* as SPYURLTypes, *nLoading* as SPYLoading, *strUser* as String, *strPassword* as String) as Document

Parameters
strURL
URL to open as document.

nURLType
Type of document to open. Set to -1 for auto detection.

nLoading
Set nLoading to 0 (zero) if you want to load it from cache or proxy. Otherwise set nLoading to 1.

strUser
Name of the user if required. Can be empty.

strPassword
Password for authentification. Can be empty.

Return Value
The method returns the opened document.

Description
OpenURL opens the URL strURL.

14.3.2.11.9 OpenURLDialog

Method: OpenURLDialog (*strURL* as String, *nURLType* as SPYURLTypes, *nLoading* as SPYLoading, *strUser* as String, *strPassword* as String) as Document

Parameters
strURL
URL to open as document.

nURLType
Type of document to open. Set to -1 for auto detection.

nLoading
Set nLoading to 0 (zero) if you want to load it from cache or proxy. Otherwise set nLoading to 1.

strUser
Name of the user if required. Can be empty.

strPassword
Password for authentification. Can be empty.

Return Value
The method returns the opened document.

Description
OpenURLDialog displays the "open URL" dialog to the user and presets the input fields with the given parameters.

14.3.2.12 DTDSchemaGeneratorDlg

Properties and Methods

Standard automation properties
Application
Parent

DTDSchemaFormat
ValueList
TypeDetection
FrequentElements
MergeAllEqualNamed
ResolveEntities
AttributeTypeDefinition
GlobalAttributes
OnlyStringEnums
MaxEnumLength
OutputPath
OutputPathDialogAction

Description
Use this object to configure the generation of a schema or DTD. The method GenerateDTDOrSchemaEx expects a DTDSchemaGeneratorDlg as parameter to configure the generation as well as the associated user interactions.

14.3.2.12.1 Application

Property: Application as Application (read-only)

Description
Access the Authentic Desktop application object.

Errors
3000 The object is no longer valid.
3001 Invalid address for the return parameter was specified.

 © 2016-2022 Altova GmbH

14.3.2.12.2 AttributeTypeDefinition

Property: AttributeTypeDefinition as SPYAttributeTypeDefinition

Description
Specifies how attribute definitions get merged.

Errors
3000 The object is no longer valid.
3001 Invalid address for the return parameter was specified.

14.3.2.12.3 DTDSchemaFormat

Property: DTDSchemaFormat as SPYDTDSchemaFormat

Description
Sets the schema output format to DTD, or W3C.

Errors
3000 The object is no longer valid.
3001 Invalid address for the return parameter was specified.

14.3.2.12.4 FrequentElements

Property: FrequentElements as SPYFrequentElements

Description
Shall the types for all elements be defined as global? Use that value *spyGlobalComplexType* to define them on global scope. Otherwise, use the value *spyGlobalElements*.

Errors
3000 The object is no longer valid.
3001 Invalid address for the return parameter was specified.

14.3.2.12.5 GlobalAttributes

Property: GlobalAttributes as Boolean

Description
Shall attributes with same name and type be resolved globally?

Errors
3000 The object is no longer valid.
3001 Invalid address for the return parameter was specified.

14.3.2.12.6 MaxEnumLength

Property: MaxEnumLength as Integer

Description
Specifies the maximum number of characters allowed for enumeration names. If one value is longer than this, no enumeration will be generated.

Errors
3000 The object is no longer valid.
3001 Invalid address for the return parameter was specified.

14.3.2.12.7 MergeAllEqualNamed

Property: MergeAllEqualNamed as Boolean

Description
Shall types of all elements with the same name be merged into one type?

Errors
3000 The object is no longer valid.
3001 Invalid address for the return parameter was specified.

14.3.2.12.8 OnlyStringEnums

Property: OnlyStringEnums as Boolean

Description
Specifies if enumerations will be created only for plain strings or all types of values.

Errors
3000 The object is no longer valid.
3001 Invalid address for the return parameter was specified.

14.3.2.12.9 OutputPath

Property: OutputPath as String

Description
Selects the file name for the generated schema/DTD.

Errors

© 2016-2022 Altova GmbH

3000 The object is no longer valid.
3001 Invalid address for the return parameter was specified.

14.3.2.12.10 OutputPathDialogAction

Property: OutputPathDialogAction as SPYDialogAction

Description
Defines how the sub-dialog for selecting the schema/DTD output path gets handled. Set this value to *spyDialogUserInput(2)* to show the dialog with the current value of the OutputPath property as default. Use *spyDialogOK(0)* to hide the dialog from the user.

Errors
3000 The object is no longer valid.
3001 Invalid address for the return parameter was specified.

14.3.2.12.11 Parent

Property: Parent as Dialogs (read-only)

Description
Access the parent of the object.

Errors
3000 The object is no longer valid.
3001 Invalid address for the return parameter was specified.

14.3.2.12.12 ResolveEntities

Property: ResolveEntities as Boolean

Description
Shall all entities be resolved before generation starts? If yes, an info-set will be built.

Errors
3000 The object is no longer valid.
3001 Invalid address for the return parameter was specified.

14.3.2.12.13 TypeDetection

Property: TypeDetection as SPYTypeDetection

Description
Specifies granularity of simple type detection.

Errors
3000 The object is no longer valid.
3001 Invalid address for the return parameter was specified.

14.3.2.12.14 ValueList

Property: ValueList as Integer

Description
Generate not more than this amount of enumeration-facets per type. Set to -1 for unlimited.

Errors
3000 The object is no longer valid.
3001 Invalid address for the return parameter was specified.

14.3.2.13 ElementList

Properties
Count
Item

Methods
RemoveElement

Description
Element lists are used for different purposes during export and import of data. Depending on this purpose, different properties of ElementListItem are used.

It can hold
- a list of table names returned by a call to Application.GetDatabaseTables,
- a list of field names retuned by a call to Application.GetDatabaseImportElementList or Application.GetTextImportElementList,
- a field name filter list used in Application.ImportFromDatabase and Application.ImportFromText,
- a list of table names and counts for their rows and columns as returned by calls to GetExportElementList or
- a field name filter list used in Document.ExportToDatabase and Document.ExportToText.

14.3.2.13.1 Count

Property: Count as long (read-only)

Description
Count of elements in this collection.

© 2016-2022 Altova GmbH

14.3.2.13.2 Item

Method: Item(n as long) as ElementListItem

Description
Gets the element with the index n from this collection. The first item has index 1.

14.3.2.13.3 RemoveElement

Method: RemoveElement(Index as long)

Description
RemoveElement removes the element Index from the collection. The first Item has index 1.

14.3.2.14 ElementListItem

Properties
Name

ElementKind

FieldCount
RecordCount

Description
An element in an ElementList. Usage of its properties depends on the purpose of the element list. For details see ElementList.

14.3.2.14.1 ElementKind

Property: ElementKind as SPYXMLDataKind

Description
Specifies if a field should be imported as XML element (data value of spyXMLDataElement) or attribute (data value of spyXMLDataAttr).

14.3.2.14.2 FieldCount

Property: FieldCount as long (read-only)

Description
Count of fields (i.e. columns) in the table described by this element. This property is only valid after a call to Document.GetExportElementList.

14.3.2.14.3 Name

Property: Name as String (read-only)

Description
Name of the element. This is either the name of a table or a field, depending on the purpose of th element list.

14.3.2.14.4 RecordCount

Property: RecordCount as long (read-only)

Description
Count of records (i.e. rows) in the table described by this element. This property is only valid after a call to Document.GetExportElementList.

14.3.2.15 ExportSettings

Properties

ElementList

EntitiesToText

ExportAllElements
SubLevelLimit

FromAttributes
FromSingleSubElements
FromTextValues

CreateKeys
IndependentPrimaryKey

Namespace

ExportCompleteXML
StartFromElement

Description
ExportSettings contains options used during export of XML data to a database or text file.

14.3.2.15.1 CreateKeys

Property: CreateKeys as Boolean

Description

© 2016-2022 Altova GmbH

This property turns creation of keys (i.e. primary key and foreign key) on or off. Default is True.

14.3.2.15.2 ElementList

Property: ElementList as ElementList

Description
Default is empty list. This list of elements defines which fields will be exported. To get the list of available fields use Document.GetExportElementList. It is possible to prevent exporting columns by removing elements from this list with ElementList.RemoveElement before passing it to Document.ExportToDatabase or Document.ExportToText.

14.3.2.15.3 EntitiesToText

Property: EntitiesToText as Boolean

Description
Defines if XML entities should be converted to text or left as they are during export. Default is True.

14.3.2.15.4 ExportAllElements

Property: ExportAllElements as Boolean

Description
If set to true, all elements in the document will be exported. If set to false, then ExportSettings.SubLevelLimit is used to restrict the number of sub levels to export. Default is true.

14.3.2.15.5 ExportCompleteXML

Property: ExportCompleteXML as Boolean

Description
Defines whether the complete XML is exported or only the element specified by StartFromElement and its children. Default is True.

14.3.2.15.6 FromAttributes

Property: FromAttributes as Boolean

Description
Set FromAttributes to false if no export data should be created from attributes. Default is True.

14.3.2.15.7 FromSingleSubElements

Property: FromSingleSubElements as Boolean

Description
Set FromSingleSubElements to false if no export data should be created from elements. Default is True.

14.3.2.15.8 FromTextValues

Property: FromTextValues as Boolean

Description
Set FromTextValues to false if no export data should be created from text values. Default is True.

14.3.2.15.9 IndependentPrimaryKey

Property: IndependentPrimaryKey as Boolean

Description
Turns creation of independent primary key counter for every element on or off. If ExportSettings.CreateKeys is False, this property will be ignored. Default is True.

14.3.2.15.10 Namespace

Property: Namespace as SPYExportNamespace

Description
The default setting removes all namespace prefixes from the element names. In some database formats the colon is not a legal character. Default is spyNoNamespace.

14.3.2.15.11 StartFromElement

Property: StartFromElement as String

Description
Specifies the start element for the export. This property is only considered when ExportCompleteXML is false.

14.3.2.15.12 SubLevelLimit

Property: SubLevelLimit as Integer

Description
Defines the number of sub levels to include for the export. Default is 0. This property is ignored if ExportSettings.ExportAllElements is true.

© 2016-2022 Altova GmbH

14.3.2.16 FileSelectionDlg

Properties and Methods

Standard automation properties
Application
Parent

Dialog properties
FullName

Acceptance or cancellation of action that caused event
DialogAction

Description
The dialog object allows you to receive information about an event and pass back information to the event handler in the same way as with a user dialog. Use the FileSelectionDlg.FullName to select or modify the file path and set the FileSelectionDlg.DialogAction property to cancel or agree with the action that caused the event.

14.3.2.16.1 Application

Property: Application as Application (read-only)

Description
Access the Authentic Desktop application object.

Errors
2400 The object is no longer valid.
2401 Invalid address for the return parameter was specified.

14.3.2.16.2 DialogAction

Property: DialogAction as SPYDialogAction

Description
If you want your script to perform the file selection operation without any user interaction necessary, simulate user interaction by either setting the property to *spyDialogOK(0)* or *spyDialogCancel(1)*.
To allow your script to fill in the default values but let the user see and react on the dialog, use the value *spyDialogUserInput(2)*. If you receive a FileSelectionDlg object in an event handler, *spyDialogUserInput(2)* is not supported and will be interpreted as *spyDialogOK(0)*.

Errors
2400 The object is no longer valid.
2401 Invalid value for dialog action or invalid address for the return parameter was specified.

14.3.2.16.3 FullName

Property: FullName as String

Description
Access the full path of the file the gets selected by the dialog. Most events that pass a FileSelectionDlg object to you allow you modify this value and thus influence the action that caused the event (e.g. load or save to a different location).

Errors
2400 The object is no longer valid.
2401 Invalid address for the return parameter was specified.

14.3.2.16.4 Parent

Property: Parent as Dialogs (read-only)

Description
Access the parent of the object.

Errors
2400 The object is no longer valid.
2401 Invalid address for the return parameter was specified.

14.3.2.17 FindInFilesDlg

Properties and Methods

Standard automation properties
Application
Parent

Find
RegularExpression
Replace
DoReplace
ReplaceOnDisk
MatchWholeWord
MatchCase
SearchLocation
StartFolder
IncludeSubfolders
SearchInProjectFilesDoExternal
FileExtension
AdvancedXMLSearch
XMLElementNames
XMLElementContents
XMLAttributeNames

© 2016-2022 Altova GmbH

XMLAttributeContents
XMLComments
XMLCData
XMLPI
XMLRest
ShowResult

Description
Use this object to configure the search (or replacement) for strings in files. The method FindInFiles expects a FindInFilesDlg as parameter.

14.3.2.17.1 AdvancedXMLSearch

Property: AdvancedXMLSearch as Boolean

Description
Specifies if the XML search properties (XMLElementNames, XMLElementContents, XMLAttributeNames, XMLAttributeContents, XMLComments, XMLCData, XMLPI and XMLRest) are considered. The default is false.

Errors
3500 The object is no longer valid.
3501 Invalid address for the return parameter was specified.

14.3.2.17.2 Application

Property: Application as Application (read-only)

Description
Access the Authentic Desktop application object.

Errors
3500 The object is no longer valid.
3501 Invalid address for the return parameter was specified.

14.3.2.17.3 DoReplace

Property: DoReplace as Boolean

Description
Specifies if the matched string is replaced by the string defined in Replace. The default is false.

Errors
3500 The object is no longer valid.
3501 Invalid address for the return parameter was specified.

14.3.2.17.4 FileExtension

Property: FileExtension as String

Description
Specifies the file filter of the files that should be considered during the search. Multiple file filters must be delimited with a semicolon (eg: *.xml;*.dtd;a*.xsd). Use the wildcards * and ? to define the file filter.

Errors
3500 The object is no longer valid.
3501 Invalid address for the return parameter was specified.

14.3.2.17.5 Find

Property: Find as String

Description
Specifies the string to search for.

Errors
3500 The object is no longer valid.
3501 Invalid address for the return parameter was specified.

14.3.2.17.6 IncludeSubfolders

Property: IncludeSubfolders as Boolean

Description
Specifies if subfolders are searched too. The default is true.

Errors
3500 The object is no longer valid.
3501 Invalid address for the return parameter was specified.

14.3.2.17.7 MatchCase

Property: MatchCase as Boolean

Description
Specifies if the search is case sensitive. The default is true.

Errors
3500 The object is no longer valid.
3501 Invalid address for the return parameter was specified.

© 2016-2022 Altova GmbH

14.3.2.17.8 MatchWholeWord

Property: MatchWholeWord as Boolean

Description
Specifies whether the whole word or just a part of it must match. The default is false.

Errors
3500 The object is no longer valid.
3501 Invalid address for the return parameter was specified.

14.3.2.17.9 Parent

Property: Parent as Dialogs (read-only)

Description
Access the parent of the object.

Errors
3500 The object is no longer valid.
3501 Invalid address for the return parameter was specified.

14.3.2.17.10 RegularExpression

Property: RegularExpression as Boolean

Description
Specifies if Find contains a regular expression. The default is false.

Errors
3500 The object is no longer valid.
3501 Invalid address for the return parameter was specified.

14.3.2.17.11 Replace

Property: Replace as String

Description
Specifies the replacement string. The matched string is only replaced if DoReplace is set true.

Errors
3500 The object is no longer valid.
3501 Invalid address for the return parameter was specified.

14.3.2.17.12 ReplaceOnDisk

Property: ReplaceOnDisk as Boolean

Description
Specifies if the replacement is done directly on disk. The modified file is not opened. The default is false.

Errors
3500 The object is no longer valid.
3501 Invalid address for the return parameter was specified.

14.3.2.17.13 SearchInProjectFilesDoExternal

Property: SearchInProjectFilesDoExternal as Boolean

Description
Specifies if the external folders in the open project are searched, when a project search is performed. The default is false.

Errors
3500 The object is no longer valid.
3501 Invalid address for the return parameter was specified.

14.3.2.17.14 SearchLocation

Property: SearchLocation as SPYFindInFilesSearchLocation

Description
Specifies the location of the search. The default is spyFindInFiles_Documents.

Errors
3500 The object is no longer valid.
3501 Invalid address for the return parameter was specified.

14.3.2.17.15 ShowResult

Property: ShowResult as Boolean

Description
Specifies if the result is displayed in the Find in Files output window. The default is false.

Errors
3500 The object is no longer valid.
3501 Invalid address for the return parameter was specified.

© 2016-2022 Altova GmbH

14.3.2.17.16 StartFolder

Property: StartFolder as String

Description
Specifies the folder where the disk search starts.

Errors
3500 The object is no longer valid.
3501 Invalid address for the return parameter was specified.

14.3.2.17.17 XMLAttributeContents

Property: XMLAttributeContents as Boolean

Description
Specifies if attribute contents are searched when AdvancedXMLSearch is true. The default is true.

Errors
3500 The object is no longer valid.
3501 Invalid address for the return parameter was specified.

14.3.2.17.18 XMLAttributeNames

Property: XMLAttributeNames as Boolean

Description
Specifies if attribute names are searched when AdvancedXMLSearch is true. The default is true.

Errors
3500 The object is no longer valid.
3501 Invalid address for the return parameter was specified.

14.3.2.17.19 XMLCData

Property: XMLCData as Boolean

Description
Specifies if CData tags are searched when AdvancedXMLSearch is true. The default is true.

Errors
3500 The object is no longer valid.
3501 Invalid address for the return parameter was specified.

14.3.2.17.20 XMLComments

Property: XMLComments as Boolean

Description
Specifies if comments are searched when AdvancedXMLSearch is true. The default is true.

Errors
3500 The object is no longer valid.
3501 Invalid address for the return parameter was specified.

14.3.2.17.21 XMLElementContents

Property: XMLElementContents as Boolean

Description
Specifies if element contents are searched when AdvancedXMLSearch is true. The default is true.

Errors
3500 The object is no longer valid.
3501 Invalid address for the return parameter was specified.

14.3.2.17.22 XMLElementNames

Property: XMLElementNames as Boolean

Description
Specifies if element names are searched when AdvancedXMLSearch is true. The default is true.

Errors
3500 The object is no longer valid.
3501 Invalid address for the return parameter was specified.

14.3.2.17.23 XMLPI

Property: XMLPI as Boolean

Description
Specifies if XML processing instructions are searched when AdvancedXMLSearch is true. The default is true.

Errors
3500 The object is no longer valid.
3501 Invalid address for the return parameter was specified.

© 2016-2022 Altova GmbH

14.3.2.17.24 XMLRest

Property: XMLRest as Boolean

Description
Specifies if the rest of the XML (which is not covered by the other XML search properties) is searched when AdvancedXMLSearch is true. The default is true.

Errors
3500 The object is no longer valid.
3501 Invalid address for the return parameter was specified.

14.3.2.18 FindInFilesResult

Properties and Methods

Standard automation properties
Application
Parent

Count
Item

Path
Document

Description
This object represents a file that matched the search criteria. It contains a list of FindInFilesResultMatch objects that describe the matching position.

14.3.2.18.1 Application

Property: Application as Application (read-only)

Description
Access the Authentic Desktop application object.

Errors
3700 The object is no longer valid.
3701 Invalid address for the return parameter was specified.

14.3.2.18.2 Count

Property: Count as long (read-only)

Description

Count of elements in this collection.

14.3.2.18.3 Document

Property: Path as Document (read-only)

Description
This property returns the Document object if the matched file is already open in XMLSpy.

Errors
3700 The object is no longer valid.
3701 Invalid address for the return parameter was specified.

14.3.2.18.4 Item

Method: Item(n as long) as FindInFilesResultMatch

Description
Gets the element with the index n from this collection. The first item has index 1.

14.3.2.18.5 Parent

Property: Parent as FindInFilesResults (read-only)

Description
Access the parent of the object.

Errors
3700 The object is no longer valid.
3701 Invalid address for the return parameter was specified.

14.3.2.18.6 Path

Property: Path as String (read-only)

Description
Returns the path of the file that matched the search criteria.

Errors
3700 The object is no longer valid.
3701 Invalid address for the return parameter was specified.

© 2016-2022 Altova GmbH

14.3.2.19 FindInFilesResultMatch

Properties and Methods

Standard automation properties
Application
Parent

Line
Position
Length
LineText
Replaced

Description
Contains the exact position in the file of the matched string.

14.3.2.19.1 Application

Property: Application as Application (read-only)

Description
Access the Authentic Desktop application object.

Errors
3800 The object is no longer valid.
3801 Invalid address for the return parameter was specified.

14.3.2.19.2 Length

Property: Length as Long (read-only)

Description
Returns the length of the matched string.

Errors
3800 The object is no longer valid.
3801 Invalid address for the return parameter was specified.

14.3.2.19.3 Line

Property: Line as Long (read-only)

Description
Returns the line number of the match. The line numbering starts with 0.

Errors
3800 The object is no longer valid.
3801 Invalid address for the return parameter was specified.

14.3.2.19.4 LineText

Property: LineText as String (read-only)

Description
Returns the text of the line.

Errors
3800 The object is no longer valid.
3801 Invalid address for the return parameter was specified.

14.3.2.19.5 Parent

Property: Parent as FindInFilesResult (read-only)

Description
Access the parent of the object.

Errors
3800 The object is no longer valid.
3801 Invalid address for the return parameter was specified.

14.3.2.19.6 Position

Property: Position as Long (read-only)

Description
Returns the start position of the match in the line. The position numbering starts with 0.

Errors
3800 The object is no longer valid.
3801 Invalid address for the return parameter was specified.

14.3.2.19.7 Replaced

Property: Replaced as Boolean (read-only)

Description

© 2016-2022 Altova GmbH

True if the matched string was replaced.

Errors
3800 The object is no longer valid.
3801 Invalid address for the return parameter was specified.

14.3.2.20 FindInFilesResults

Properties and Methods

Standard automation properties
Application
Parent

Count
Item

Description
This is the result of the FindInFiles method. It is a list of FindInFilesResult objects.

14.3.2.20.1 Application

Property: Application as Application (read-only)

Description
Access the Authentic Desktop application object.

Errors
3600 The object is no longer valid.
3601 Invalid address for the return parameter was specified.

14.3.2.20.2 Count

Property: Count as long (read-only)

Description
Count of elements in this collection.

14.3.2.20.3 Item

Method: Item(n as long) as FindInFilesResult

Description
Gets the element with the index n from this collection. The first item has index 1.

14.3.2.20.4 Parent

Property: Parent as Application (read-only)

Description
Access the parent of the object.

Errors
3600 The object is no longer valid.
3601 Invalid address for the return parameter was specified.

14.3.2.21 GenerateSampleXMLDlg

Properties and Methods

Standard automation properties
Application
Parent

NonMandatoryAttributes
NonMandatoryElements
RepeatCount
FillAttributesWithSampleData
FillElementsWithSampleData
ContentOfNillableElementsIsNonMandatory
TryToUseNonAbstractTypes
SchemaOrDTDAssignment
LocalNameOfRootElement
NamespaceURIOfRootElement
OptionsDialogAction

Properties that are no longer supported
TakeFirstChoice - obsolete
FillWithSampleData - obsolete
Optimization - obsolete

Description
Used to set the parameters for the generation of sample XML instances based on a W3C schema or DTD.

14.3.2.21.1 Application

Property: Application as Application (read-only)

Description
Access the Authentic Desktop application object.

Errors

© 2016-2022 Altova GmbH

2200 The object is no longer valid.
2201 Invalid address for the return parameter was specified.

14.3.2.21.2 ChoiceMode

Property: `ChoiceMode` as `SPYSampleXMLGenerationChoiceMode`

Description
Specifies which elements will be generated.

Errors

2200 The object is no longer valid.
2201 Invalid address for the return parameter was specified.

14.3.2.21.3 ConsiderSampleValueHints

Property: `ConsiderSampleValueHints` as Boolean

Description
Selects whether to use `SampleValueHints` or not.

Errors
2200 The object is no longer valid.
2201 Invalid address for the return parameter was specified.

14.3.2.21.4 ContentOfNillableElementsIsNonMandatory

Property: ContentOfNillableElementsIsNonMandatory as Boolean

Description
If true, the contents of elements that are nillable will not be treated as mandatory.

Errors
2200 The object is no longer valid.
2201 Invalid address for the return parameter was specified.

14.3.2.21.5 FillAttributesWithSampleData

Property: FillAttributesWithSampleData as Boolean

Description
If true, attributes will have sample content.

Errors
2200 The object is no longer valid.
2201 Invalid address for the return parameter was specified.

14.3.2.21.6 FillElementsWithSampleData

Property: FillElementsWithSampleData as Boolean

Description
If true, elements will have sample content.

Errors
2200 The object is no longer valid.
2201 Invalid address for the return parameter was specified.

14.3.2.21.7 FillWithSampleData - obsolete

Property: FillWithSampleData as Boolean

Description
Do no longer access this property. Use FillAttributesWithSampleData and FillElementsWithSampleData, instead.

Errors
0001 The property is no longer accessible.

14.3.2.21.8 LocalNameOfRootElement

Property: LocalNameOfRootElement as String

Description
Specifies the local name of the root element for the generated sample XML.

Errors
2200 The object is no longer valid.
2201 Invalid address for the return parameter was specified.

© 2016-2022 Altova GmbH

14.3.2.21.9 NamespaceURIOfRootElement

Property: NamespaceURIOfRootElement as String

Description
Specifies the namespace URI of the root element for the generated sample XML.

Errors
2200 The object is no longer valid.
2201 Invalid address for the return parameter was specified.

14.3.2.21.10 NonMandatoryAttributes

Property: NonMandatoryAttributes as Boolean

Description
If true attributes which are not mandatory are created in the sample XML instance file.

Errors
2200 The object is no longer valid.
2201 Invalid address for the return parameter was specified.

14.3.2.21.11 NonMandatoryElements

Property: NonMandatoryElements as Boolean

Description
If `true`, elements which are not mandatory are created in the sample XML instance file.

Errors
2200 The object is no longer valid.
2201 Invalid address was specified for the return parameter.

14.3.2.21.12 Optimization - obsolete

Property: Optimization as SPYSampleXMLGenerationOptimization

Description
Do not use this property any longer. Use ChoiceMode and NonMandatoryElements.

Errors
0001 The property is no longer accessible.

14.3.2.21.13 OptionsDialogAction

Property: OptionsDialogAction as SPYDialogAction

Description
To allow your script to fill in the default values and let the user see and react on the dialog, set this property to the value *spyDialogUserInput(2)*. If you want your script to define all the options in the schema documentation dialog without any user interaction necessary, use *spyDialogOK(0)*. Default is *spyDialogOK*.

Errors
2200 The object is no longer valid.
2201 Invalid value has been used to set the property.
Invalid address for the return parameter was specified.

14.3.2.21.14 Parent

Property: Parent as Dialogs (read-only)

Description
Access the parent of the object.

Errors
2200 The object is no longer valid.
2201 Invalid address for the return parameter was specified.

14.3.2.21.15 RepeatCount

Property: RepeatCount as long

Description
Number of elements to create for repeated types.

Errors
2200 The object is no longer valid.
2201 Invalid address for the return parameter was specified.

14.3.2.21.16 SampleValueHints

Property: `SampleValueHints` as `SPYSampleXMLGenerationSampleValueHints`

Description
Specifies how to select data for the generated sample file.

Errors
2200 The object is no longer valid.

© 2016-2022 Altova GmbH

2201 Invalid address for the return parameter was specified.

14.3.2.21.17 SchemaOrDTDAssignment

Property: SchemaOrDTDAssignment as SPYSampleXMLGenerationSchemaOrDTDAssignment

Description
Specifies in which way a reference to the related schema or DTD - which is this document - will be generated into the sample XML.

Errors
2200 The object is no longer valid.
2201 Invalid address for the return parameter was specified.

14.3.2.21.18 TakeFirstChoice - obsolete

Property: TakeFirstChoice as Boolean

Description
Do no longer use this property.

Errors
0001 The property is no longer accessible.

14.3.2.21.19 TryToUseNonAbstractTypes

Property: TryToUseNonAbstractTypes as Boolean

Description
If true, tries to use a non-abstract type for xsi:type, if element has an abstract type.

Errors
2200 The object is no longer valid.
2201 Invalid address for the return parameter was specified.

14.3.2.22 GridView

Methods
Deselect
Select

SetFocus

Properties
CurrentFocus

IsVisible

Description
GridView Class

14.3.2.22.1 Events

14.3.2.22.1.1 OnBeforeDrag

Event: OnBeforeDrag() as Boolean

XMLSpy scripting environment - VBScript:
```
Function On_BeforeDrag()
	' On_BeforeStartEditing = False  ' to prohibit dragging
End Function
```

XMLSpy scripting environment - JScript:
```
function On_BeforeDrag()
{
	// return false;  /* to prohibit dragging */
}
```

XMLSpy IDE Plugin:
```
IXMLSpyPlugIn.OnEvent (4, ...)   // nEventId = 4
```

Description
This event gets fired on an attempt to drag an XMLData element on the grid view. Return *false* to prevent dragging the data element to a different position.

14.3.2.22.1.2 OnBeforeDrop

Event: OnBeforeDrop(*objXMLData* as XMLData) as Boolean

XMLSpy scripting environment - VBScript:
```
Function On_BeforeDrop(objXMLData)
	' On_BeforeStartEditing = False  ' to prohibit dropping
End Function
```

XMLSpy scripting environment - JScript:
```
function On_BeforeDrop(objXMLData)
{
	// return false;  /* to prohibit dropping */
}
```

XMLSpy IDE Plugin:
```
IXMLSpyPlugIn.OnEvent (5, ...)   // nEventId = 5
```


© 2016-2022 Altova GmbH

Description
This event gets fired on an attempt to drop a previously dragged XMLData element on the grid view. Return *false* to prevent the data element to be moved from its original position to the drop destination position.

14.3.2.22.1.3 OnBeforeStartEditing

Event: OnBeforeStartEditing(*objXMLData* as XMLData, *bEditingName* as Boolean)as Boolean

XMLSpy scripting environment - VBScript:
```
Function On_BeforeStartEditing(objXMLData, bEditingName)
        ' On_BeforeStartEditing = False  ' to prohibit editing the field
End Function
```

XMLSpy scripting environment - JScript:
```
function On_BeforeStartEditing(objXMLData, bEditingName)
{
        // return false;  /* to prohibit editing the field */
}
```

XMLSpy IDE Plugin:
```
IXMLSpyPlugIn.OnEvent (1, ...)   // nEventId = 1
```

Description
This event gets fired before the editing mode for a grid cell gets entered. If the parameter *bEditingName* is true, the name part of the element will be edited, it its value is false, the value part will be edited.

14.3.2.22.1.4 OnEditingFinished

Event: OnEditingFinished(*objXMLData* as XMLData, *bEditingName* as Boolean)

XMLSpy scripting environment - VBScript:
```
Function On_EditingFinished(objXMLData, bEditingName)
End Function
```

XMLSpy scripting environment - JScript:
```
function On_EditingFinished(objXMLData, bEditingName)
{
}
```

XMLSpy IDE Plugin:
```
IXMLSpyPlugIn.OnEvent (2, ...)   // nEventId = 2
```

Description
This event gets fired when the editing mode of a grid cell gets left. The parameter *bEditingName* specifies if the name part of the element has been edited.

14.3.2.22.1.5 OnFocusChanged

Event: OnFocusChanged(*objXMLData* as XMLData, *bSetFocus* as Boolean, *bEditingName* as Boolean)

XMLSpy scripting environment - VBScript:
Function On_FocusChanged(*objXMLData*, *bSetFocus*, *bEditingName*)
End Function

XMLSpy scripting environment - JScript:
function On_FocusChanged(*objXMLData*, *bSetFocus*, *bEditingName*)
{
}

XMLSpy IDE Plugin:
IXMLSpyPlugIn.OnEvent (3, ...) // nEventId = 3

Description
This event gets fired whenever a grid cell receives or looses the cursor focus. If the parameter *bEditingName* is *true*, focus of the name part of the grid element has changed. Otherwise, focus of the value part has changed.

14.3.2.22.2 CurrentFocus

Property: CurrentFocus as XMLData

Description
Holds the XML element with the current focus. This property is read-only.

14.3.2.22.3 Deselect

Method: Deselect(*pData* as XMLData)

Description
Deselects the element pData in the grid view.

14.3.2.22.4 IsVisible

Property: IsVisible as Boolean

Description
True if the grid view is the active view of the document. This property is read-only.

© 2016-2022 Altova GmbH

14.3.2.22.5 Select

Method: Select (*pData* as XMLData)

Description
Selects the XML element pData in the grid view.

14.3.2.22.6 SetFocus

Method: SetFocus (*pFocusData* as XMLData)

Description
Sets the focus to the element pFocusData in the grid view.

14.3.2.23 SchemaDocumentationDlg

Properties and Methods

Standard automation properties
Application
Parent

Interaction and visibility properties
OutputFile
OutputFileDialogAction
OptionsDialogAction
ShowProgressBar
ShowResult

Document generation options and methods
OutputFormat
UseFixedDesign
SPSFile
EmbedDiagrams
DiagramFormat
MultipleOutputFiles
EmbedCSSInHTML
CreateDiagramsFolder
GenerateRelativeLinks

IncludeAll
IncludeIndex
IncludeGlobalAttributes
IncludeGlobalElements
IncludeLocalAttributes
IncludeLocalElements
IncludeGroups
IncludeComplexTypes

IncludeSimpleTypes
IncludeAttributeGroups
IncludeRedefines
IncludeReferencedSchemas

AllDetails
ShowDiagram
ShowNamespace
ShowType
ShowChildren
ShowUsedBy
ShowProperties
ShowSingleFacets
ShowPatterns
ShowEnumerations
ShowAttributes
ShowIdentityConstraints
ShowAnnotations
ShowSourceCode

Description
This object combines all options for schema document generation as they are available through user interface dialog boxes in Authentic Desktop. The document generation options are initialized with the values used during the last generation of schema documentation. However, before using the object you have to set the SetOutputFile property to a valid file path. Use OptionsDialogAction, OutputFileDialogAction and ShowProgressBar to specify the level of user interaction desired. You can use IncludeAll and AllDetails to set whole option groups at once or the individual properties to operate on a finer granularity.

14.3.2.23.1 AllDetails

Method: AllDetails (i_bDetailsOn as Boolean)

Description
Use this method to turn all details options on or off.

Errors
2900 The object is no longer valid.

14.3.2.23.2 Application

Property: Application as Application (read-only)

Description
Access the Authentic Desktop application object.

© 2016-2022 Altova GmbH

Errors
2900 The object is no longer valid.
2901 Invalid address for the return parameter was specified.

14.3.2.23.3 CreateDiagramsFolder

Property: CreateDiagramsFolder as Boolean

Description
Set this property to true, to create a directory for the created images. Otherwise the diagrams will be created next to the documentation. This property is only available when the diagrams are not embedded. The default for the first run is false.

Errors
2900 The object is no longer valid.
2901 Invalid address for the return parameter was specified.

14.3.2.23.4 DiagramFormat

Property: DiagramFormat as SPYImageKind

Description
This property specifies the generated diagram image type. This property is not available for HTML documentation. The property is initialized with the value used during the last call to Document.GenerateSchemaDocumentation. The default for the first run is PNG.

Errors
2900 The object is no longer valid.
2901 Invalid address for the return parameter was specified.

14.3.2.23.5 EmbedCSSInHTML

Property: EmbedCSSInHTML as Boolean

Description
Set this property to true, to embed the CSS data in the generated HTML document. Otherwise a separate file will be created and linked. This property is only available for HTML documentation. The default for the first run is true.

Errors
2900 The object is no longer valid.
2901 Invalid address for the return parameter was specified.

14.3.2.23.6 EmbedDiagrams

Property: EmbedDiagrams as Boolean

Description
Set this property to true, to embed the diagrams in the generated document. This property is not available for HTML documentation. The property is initialized with the value used during the last call to Document.GenerateSchemaDocumentation. The default for the first run is true.

Errors
2900 The object is no longer valid.
2901 Invalid address for the return parameter was specified.

14.3.2.23.7 GenerateRelativeLinks

Property: GenerateRelativeLinks as Boolean

Description
Set this property to true, to create relative paths to local files. This property is not available for HTML documentation. The property is initialized with the value used during the last call to Document.GenerateSchemaDocumentation. The default for the first run is false.

Errors
2900 The object is no longer valid.
2901 Invalid address for the return parameter was specified.

14.3.2.23.8 IncludeAll

Method: IncludeAll (i_bInclude as Boolean)

Description
Use this method to mark or unmark all include options.

Errors
2900 The object is no longer valid.

14.3.2.23.9 IncludeAttributeGroups

Property: IncludeAttributeGroups as Boolean

Description

© 2016-2022 Altova GmbH

Set this property to true, to include attribute groups in the schema documentation. The property is initialized with the value used during the last call to Document.GenerateSchemaDocumentation. The default for the first run is true.

Errors

2900 The object is no longer valid.
2901 Invalid address for the return parameter was specified.

14.3.2.23.10 IncludeComplexTypes

Property: IncludeComplexTypes as Boolean

Description

Set this property to true, to include complex types in the schema documentation. The property is initialized with the value used during the last call to Document.GenerateSchemaDocumentation. The default for the first run is true.

Errors

2900 The object is no longer valid.
2901 Invalid address for the return parameter was specified.

14.3.2.23.11 IncludeGlobalAttributes

Property: IncludeGlobalAttributes as Boolean

Description

Set this property to true, to include global attributes in the schema documentation. The property is initialized with the value used during the last call to Document.GenerateSchemaDocumentation. The default for the first run is true.

Errors

2900 The object is no longer valid.
2901 Invalid address for the return parameter was specified.

14.3.2.23.12 IncludeGlobalElements

Property: IncludeGlobalElements as Boolean

Description

Set this property to true, to include global elements in the schema documentation. The property is initialized with the value used during the last call to Document.GenerateSchemaDocumentation. The default for the first run is true.

Errors

2900 The object is no longer valid.
2901 Invalid address for the return parameter was specified.

14.3.2.23.13 IncludeGroups

Property: IncludeGroups as Boolean

Description
Set this property to true, to include groups in the schema documentation. The property is initialized with the value used during the last call to Document.GenerateSchemaDocumentation. The default for the first run is true.

Errors
2900 The object is no longer valid.
2901 Invalid address for the return parameter was specified.

14.3.2.23.14 IncludeIndex

Property: IncludeIndex as Boolean

Description
Set this property to true, to include an index in the schema documentation. The property is initialized with the value used during the last call to Document.GenerateSchemaDocumentation. The default for the first run is true.

Errors
2900 The object is no longer valid.
2901 Invalid address for the return parameter was specified.

14.3.2.23.15 IncludeLocalAttributes

Property: IncludeLocalAttributes as Boolean

Description
Set this property to true, to include local attributes in the schema documentation. The property is initialized with the value used during the last call to Document.GenerateSchemaDocumentation. The default for the first run is true.

Errors
2900 The object is no longer valid.
2901 Invalid address for the return parameter was specified.

14.3.2.23.16 IncludeLocalElements

Property: IncludeLocalElements as Boolean

Description

© 2016-2022 Altova GmbH

Set this property to true, to include local elements in the schema documentation. The property is initialized with the value used during the last call to Document.GenerateSchemaDocumentation. The default for the first run is true.

Errors

2900 The object is no longer valid.
2901 Invalid address for the return parameter was specified.

14.3.2.23.17 IncludeRedefines

Property: IncludeRedefines as Boolean

Description

Set this property to true, to include redefines in the schema documentation. The property is initialized with the value used during the last call to Document.GenerateSchemaDocumentation. The default for the first run is true.

Errors

2900 The object is no longer valid.
2901 Invalid address for the return parameter was specified.

14.3.2.23.18 IncludeReferencedSchemas

Property: IncludeReferencedSchemas as Boolean

Description

Set this property to true, to include referenced schemas in the schema documentation. The property is initialized with the value used during the last call to Document.GenerateSchemaDocumentation. The default for the first run is true.

Errors

2900 The object is no longer valid.
2901 Invalid address for the return parameter was specified.

14.3.2.23.19 IncludeSimpleTypes

Property: IncludeSimpleTypes as Boolean

Description

Set this property to true, to include simple types in the schema documentation. The property is initialized with the value used during the last call to Document.GenerateSchemaDocumentation. The default for the first run is true.

Errors

2900 The object is no longer valid.
2901 Invalid address for the return parameter was specified.

14.3.2.23.20 MultipleOutputFiles

Property: MultipleOutputFiles as Boolean

Description
Set this property to true, to split the documentation files. The property is initialized with the value used during the last call to Document.GenerateSchemaDocumentation. The default for the first run is false.

Errors
2900 The object is no longer valid.
2901 Invalid value has been used to set the property.
Invalid address for the return parameter was specified.

14.3.2.23.21 OptionsDialogAction

Property: OptionsDialogAction as SPYDialogAction

Description
To allow your script to fill in the default values and let the user see and react on the dialog, set this property to the value *spyDialogUserInput(2)*. If you want your script to define all the options in the schema documentation dialog without any user interaction necessary, use *spyDialogOK(0)*. Default is *spyDialogOK*.

Errors
2900 The object is no longer valid.
2901 Invalid value has been used to set the property.
Invalid address for the return parameter was specified.

14.3.2.23.22 OutputFile

Property: OutputFile as String

Description
Full path and name of the file that will contain the generated documentation. In case of HTML output, additional '.png' files will be generated based on this filename. The default value for this property is an empty string and needs to be replaced before using this object in a call to Document.GenerateSchemaDocumentation.

Errors
2900 The object is no longer valid.
2901 Invalid address for the return parameter was specified.

14.3.2.23.23 OutputFileDialogAction

Property: OutputFileDialogAction as SPYDialogAction

Description

© 2016-2022 Altova GmbH

To allow the user to select the output file with a file selection dialog, set this property to *spyDialogUserInput(2)*. If the value stored in OutputFile should be taken and no user interaction should occur, use *spyDialogOK(0)*. Default is *spyDialogOK*.

Errors

2900 The object is no longer valid.
2901 Invalid value has been used to set the property.
Invalid address for the return parameter was specified.

14.3.2.23.24 OutputFormat

Property: OutputFormat as SPYSchemaDocumentationFormat

Description

Defines the kind of documentation that will be generated: HTML (value=0), MS-Word (value=1), or RTF (value=2). The property gets initialized with the value used during the last call to Document.GenerateSchemaDocumentation. The default for the first run is HTML.

Errors

2900 The object is no longer valid.
2901 Invalid value has been used to set the property.
Invalid address for the return parameter was specified.

14.3.2.23.25 Parent

Property: Parent as Dialogs (read-only)

Description

Access the parent of the object.

Errors

2900 The object is no longer valid.
2901 Invalid address for the return parameter was specified.

14.3.2.23.26 ShowAnnotations

Property: ShowAnnotations as Boolean

Description

Set this property to true, to show the annotations to a type definition in the schema documentation. The property is initialized with the value used during the last call to Document.GenerateSchemaDocumentation. The default for the first run is true.

Errors

2900 The object is no longer valid.
2901 Invalid address for the return parameter was specified.

14.3.2.23.27 ShowAttributes

Property: ShowAttributes as Boolean

Description
Set this property to true, to show the type definitions attributes in the schema documentation. The property is initialized with the value used during the last call to Document.GenerateSchemaDocumentation. The default for the first run is true.

Errors
2900 The object is no longer valid.
2901 Invalid address for the return parameter was specified.

14.3.2.23.28 ShowChildren

Property: ShowChildren as Boolean

Description
Set this property to true, to show the children of a type definition as links in the schema documentation. The property is initialized with the value used during the last call to Document.GenerateSchemaDocumentation. The default for the first run is true.

Errors
2900 The object is no longer valid.
2901 Invalid address for the return parameter was specified.

14.3.2.23.29 ShowDiagram

Property: ShowDiagram as Boolean

Description
Set this property to true, to show type definitions as diagrams in the schema documentation. The property is initialized with the value used during the last call to Document.GenerateSchemaDocumentation. The default for the first run is true.

Errors
2900 The object is no longer valid.
2901 Invalid address for the return parameter was specified.

14.3.2.23.30 ShowEnumerations

Property: ShowEnumerations as Boolean

Description

© 2016-2022 Altova GmbH

Set this property to true, to show the enumerations contained in a type definition in the schema documentation. The property is initialized with the value used during the last call to Document.GenerateSchemaDocumentation. The default for the first run is true.

Errors

2900 The object is no longer valid.
2901 Invalid address for the return parameter was specified.

14.3.2.23.31 ShowIdentityConstraints

Property: ShowIdentityConstraints as Boolean

Description

Set this property to true, to show a type definitions identity constraints in the schema documentation. The property is initialized with the value used during the last call to Document.GenerateSchemaDocumentation. The default for the first run is true.

Errors

2900 The object is no longer valid.
2901 Invalid address for the return parameter was specified.

14.3.2.23.32 ShowNamespace

Property: ShowNamespace as Boolean

Description

Set this property to true, to show the namespace of type definitions in the schema documentation. The property is initialized with the value used during the last call to Document.GenerateSchemaDocumentation. The default for the first run is true.

Errors

2900 The object is no longer valid.
2901 Invalid address for the return parameter was specified.

14.3.2.23.33 ShowPatterns

Property: ShowPatterns as Boolean

Description

Set this property to true, to show the patterns of a type definition in the schema documentation. The property is initialized with the value used during the last call to Document.GenerateSchemaDocumentation. The default for the first run is true.

Errors

2900 The object is no longer valid.

2901 Invalid address for the return parameter was specified.

14.3.2.23.34 ShowProgressBar

Property: ShowProgressBar as Boolean

Description
Set this property to true, to make the window showing the document generation progress visible. Use false, to hide it. Default is false.

Errors
2900 The object is no longer valid.
2901 Invalid address for the return parameter was specified.

14.3.2.23.35 ShowProperties

Property: ShowProperties as Boolean

Description
Set this property to true, to show the type definition properties in the schema documentation. The property is initialized with the value used during the last call to Document.GenerateSchemaDocumentation. The default for the first run is true.

Errors
2900 The object is no longer valid.
2901 Invalid address for the return parameter was specified.

14.3.2.23.36 ShowResult

Property: ShowResult as Boolean

Description
Set this property to true, to automatically open the resulting document when generation was successful. HTML documentation will be opened in Authentic Desktop. To show Word documentation, MS-Word will be started. The property gets initialized with the value used during the last call to Document.GenerateSchemaDocumentation. The default for the first run is true.

Errors
2900 The object is no longer valid.
2901 Invalid address for the return parameter was specified.

14.3.2.23.37 ShowSingleFacets

Property: ShowSingleFacets as Boolean

© 2016-2022 Altova GmbH

Description
Set this property to true, to show the facets of a type definition in the schema documentation. The property is initialized with the value used during the last call to Document.GenerateSchemaDocumentation. The default for the first run is true.

Errors
2900 The object is no longer valid.
2901 Invalid address for the return parameter was specified.

14.3.2.23.38 ShowSourceCode

Property: ShowSourceCode as Boolean

Description
Set this property to true, to show the XML source code for type definitions in the schema documentation. The property is initialized with the value used during the last call to Document.GenerateSchemaDocumentation. The default for the first run is true.

Errors
2900 The object is no longer valid.
2901 Invalid address for the return parameter was specified.

14.3.2.23.39 ShowType

Property: ShowType as Boolean

Description
Set this property to true, to show the type of type definitions in the schema documentation. The property is initialized with the value used during the last call to Document.GenerateSchemaDocumentation. The default for the first run is true.

Errors
2900 The object is no longer valid.
2901 Invalid address for the return parameter was specified.

14.3.2.23.40 ShowUsedBy

Property: ShowUsedBy as Boolean

Description
Set this property to true, to show the used-by relation for type definitions in the schema documentation. The property is initialized with the value used during the last call to Document.GenerateSchemaDocumentation. The default for the first run is true.

Errors

2900 The object is no longer valid.
2901 Invalid address for the return parameter was specified.

14.3.2.23.41 SPSFile

Property: SPSFile as String

Description
Full path and name of the SPS file that will be used to generate the documentation.

Errors
2900 The object is no longer valid.
2901 Invalid address for the return parameter was specified.

14.3.2.23.42 UseFixedDesign

Property: UseFixedDesign as Boolean

Description
Specifies whether the documentation should be created with a fixed design or with a design specified by a SPS file (which requires StyleVision).

Errors
2900 The object is no longer valid.
2901 Invalid address for the return parameter was specified.

14.3.2.24 SpyProject

Methods
CloseProject
SaveProject
SaveProjectAs

Properties
RootItems
ProjectFile

Description
SpyProject Class

14.3.2.24.1 CloseProject

Declaration: CloseProject(*bDiscardChanges* as Boolean, *bCloseFiles* as Boolean, *bDialog* as Boolean)

© 2016-2022 Altova GmbH

Parameters
bDiscardChanges
Set bDiscardChanges to FALSE if you want to save the changes of the open project files and the project.

bCloseFiles
Set bCloseFiles to TRUE to close all open project files.

bDialog
Show dialogs for user input.

Description
CloseProject closes the current project.

14.3.2.24.2 ProjectFile

Declaration: ProjectFile as String

Description
Path and filename of the project.

14.3.2.24.3 RootItems

Declaration: RootItems as SpyProjectItems

Description
Root level of collection of project items.

14.3.2.24.4 SaveProject

Declaration: SaveProject

Description
SaveProject saves the current project.

14.3.2.24.5 SaveProjectAs

Declaration: SaveProjectAs (strPath as String, bDialog as Boolean)

Parameters
strPath
Full path with file name of new project file.

bDialog
If bDialog is TRUE, a file-dialog will be displayed.

Description
SaveProjectAs stores the project data into a new location.

14.3.2.25 SpyProjectItem

Methods
Open

Properties
ChildItems
ParentItem
FileExtensions
ItemType
Name
Path
ValidateWith
XMLForXSLTransformation
XSLForXMLTransformation
XSLTransformationFileExtension
XSLTransformationFolder

Description
SpyProjectItem Class

14.3.2.25.1 ChildItems

Declaration: ChildItems as SpyProjectItems

Description
If the item is a folder, ChildItems is the collection of the folder content.

14.3.2.25.2 FileExtensions

Declaration: FileExtensions as String

Description
Used to set the file extensions if the project item is a folder.

14.3.2.25.3 ItemType

Declaration: ItemType as SPYProjectItemTypes

Description
This property is read-only.

© 2016-2022 Altova GmbH

14.3.2.25.4 Name

Declaration: Name as String

Description
Name of the project item. This property is read-only.

14.3.2.25.5 Open

Declaration: Open as Document

Return Value
The project item opened as document.

Description
Opens the project item.

14.3.2.25.6 ParentItem

Declaration: ParentItem as SpyProjectItem

Description
Parent item of the current project item. Can be NULL (Nothing) if the project item is a top-level item.

14.3.2.25.7 Path

Declaration: Path as String

Description
Path of project item. This property is read-only.

14.3.2.25.8 ValidateWith

Declaration: ValidateWith as String

Description
Used to set the schema/DTD for validation.

14.3.2.25.9 XMLForXSLTransformation

Declaration: XMLForXSLTransformation as String

Description
Used to set the XML for XSL transformation.

14.3.2.25.10 XSLForXMLTransformation

Declaration: XSLForXMLTransformation as String

Description
Used to set the XSL for XML transformation.

14.3.2.25.11 XSLTransformationFileExtension

Declaration: XSLTransformationFileExtension as String

Description
Used to set the file extension for XSL transformation output files.

14.3.2.25.12 XSLTransformationFolder

Declaration: XSLTransformationFolder as String

Description
Used to set the destination folder for XSL transformation output files.

14.3.2.26 SpyProjectItems

Methods
AddFile
AddFolder
AddURL
RemoveItem

Properties
Count
Item

Description
SpyProjectItems Class

14.3.2.26.1 AddFile

Declaration: AddFile (*strPath* as String)

Parameters

strPath
Full path with file name of new project item

Description

© 2016-2022 Altova GmbH

The method adds a new file to the collection of project items.

14.3.2.26.2 AddFolder

Declaration: AddFolder (*strName* as String)

Parameters

strName
Name of the new folder.

Description
The method AddFolder adds a folder with the name strName to the collection of project items.

14.3.2.26.3 AddURL

Declaration: AddURL (*strURL* as String, *nURLType* as SPYURLTypes, *strUser* as String, *strPassword* as String, *bSave* as Boolean)

Description

strURL
URL to open as document.

nURLType
Type of document to open. Set to -1 for auto detection.

strUser
Name of the user if required. Can be empty.

strPassword
Password for authentification. Can be empty.

bSave
Save user and password information.

Description
The method adds an URL item to the project collection.

14.3.2.26.4 Count

Declaration: Count as long

Description
This property gets the count of project items in the collection. The property is read-only.

14.3.2.26.5 Item

Declaration: Item (*n* as long) as SpyProjectItem

Description
Retrieves the n-th element of the collection of project items. The first item has index 1.

14.3.2.26.6 RemoveItem

Declaration: RemoveItem (*pItem* as SpyProjectItem)

Description
RemoveItem deletes the item pItem from the collection of project items.

14.3.2.27 TextImportExportSettings

Properties for import only
ImportFile

Properties for export only
DestinationFolder
FileExtension
CommentIncluded
RemoveDelimiter
RemoveNewline

Properties for import and export
HeaderRow
FieldDelimiter
EnclosingCharacter
Encoding
EncodingByteOrder

Description
TextImportExportSettings contains options common to text import and export functions.

14.3.2.27.1 CommentIncluded

Property: CommentIncluded as Boolean

Description
This property tells whether additional comments are added to the generated text file. Default is true. This property is used only when exporting to text files.

© 2016-2022 Altova GmbH

14.3.2.27.2 DestinationFolder

Property: DestinationFolder as String

Description
The property DestinationFolder sets the folder where the created files are saved during text export.

14.3.2.27.3 EnclosingCharacter

Property: EnclosingCharacter as SPYTextEnclosing

Description
This property defines the character that encloses all field values for import and export. Default is spyNoEnclosing.

14.3.2.27.4 Encoding

Property: Encoding as String

Description
The property Encoding sets the character encoding for the text files for importing and exporting.

14.3.2.27.5 EncodingByteOrder

Property: EncodingByteOrder as SPYEncodingByteOrder

Description
The property EncodingByteOrder sets the byte order for Unicode characters. Default is spyNONE.

14.3.2.27.6 FieldDelimiter

Property: FieldDelimiter as SPYTextDelimiters

Description
The property FieldDelimiter defines the delimiter between the fields during import and export. Default is spyTabulator.

14.3.2.27.7 FileExtension

Property: FileExtension as String

Description
This property sets the file extension for files created on text export.

14.3.2.27.8 HeaderRow

Property: HeaderRow as Boolean

Description
The property HeaderRow is used during import and export. Set HeaderRow true on import, if the first line of the text file contains the names of the columns. Set HeaderRow true on export, if the first line in the created text files should contain the name of the columns. Default value is true.

14.3.2.27.9 ImportFile

Property: ImportFile as String

Description
This property is used to set the text file for import. The string has to be a full qualified path.

14.3.2.27.10 RemoveDelimiter

Property: RemoveDelimiter as Boolean

Description
The property RemoveDelimiter defines whether characters in the text that are equal to the delimiter character are removed. Default is false. This property is used only when exporting to text files.

14.3.2.27.11 RemoveNewline

Property: RemoveNewline as Boolean

Description
The property RemoveNewline defines whether newline characters in the text are removed. Default is false. This property is used only when exporting to text files.

14.3.2.28 TextView

Properties and Methods

Application
Parent

LineFromPosition
PositionFromLine
LineLength
SelText
GetRangeText
ReplaceText

© 2016-2022 Altova GmbH

MoveCaret
GoToLineChar
SelectText
SelectionStart
SelectionEnd
Text
LineCount
Length

Description

14.3.2.28.1 Events

14.3.2.28.1.1 OnBeforeShowSuggestions

Event: OnBeforeShowSuggestions() as Boolean

Description
This event gets fired before a suggestion window is shown. The Document property Suggestions contains a string array that is recommended to the user. It is possible to modify the displayed recommendations during this event. Before doing so you have to assign an empty array to the Suggestions property. The best location for this is the OnDocumentOpened event. To prevent the suggestion window to show up return false and true to continue its display.

Examples
Given below are examples of how this event can be scripted.

XMLSpy scripting environment - VBScript:
```
Function On_BeforeShowSuggestions()
End Function
```

XMLSpy scripting environment - JScript:
```
function On_BeforeShowSuggestions()
{
}
```

XMLSpy IDE Plugin:
```
IXMLSpyPlugIn.OnEvent (33, ...) // nEventId = 33
```

14.3.2.28.1.2 OnChar

Event: OnChar(nChar as Long, bExistSuggestion as Boolean) as Boolean

Description
This event gets fired on each key stroke. The parameter nChar is the key that was pressed and bExistSuggestions tells whether a Authentic Desktop generated suggestions window is displayed after this

key. The Document property Suggestions contains a string array that is recommended to the user. It is possible to modify the displayed recommendations during this event. Before doing so you have to assign an empty array to the Suggestions property. The best location for this is the OnDocumentOpened event. To prevent the suggestion window to show up return false and true to continue its display.
It is also possible to create a new suggestions window when none is provided by Authentic Desktop. Set the Document property Suggestions to a string array with your recommendations and return true.
This event is fired before the OnBeforeShowSuggestions event. If you prevent to show the suggestion window by returning false then OnBeforeShowSuggestions is not fired.

Examples
Given below are examples of how this event can be scripted.

XMLSpy scripting environment - VBScript:
```
Function On_Char(nChar, bExistSuggestions)
End Function
```

XMLSpy scripting environment - JScript:
```
function On_Char(nChar, bExistSuggestions)
{
}
```

XMLSpy IDE Plugin:
```
IXMLSpyPlugIn.OnEvent (35, ...) // nEventId = 35
```

14.3.2.28.2 Application

Property: Application as Application (read-only)

Description
Access the Authentic Desktop application object.

Errors
3900 The object is no longer valid.
3901 Invalid address for the return parameter was specified.

14.3.2.28.3 GetRangeText

Method: GetRangeText(nStart as Long, nEnd as Long) as String

Description
Returns the text in the specified range.

Errors
3900 The object is no longer valid.
3901 Invalid address for the return parameter was specified.

© 2016-2022 Altova GmbH

14.3.2.28.4 GoToLineChar

Method: GoToLineChar(nLine as Long, nChar as Long)

Description
Moves the caret to the specified line and character position.

Errors
3900 The object is no longer valid.
3901 Invalid address for the return parameter was specified.

14.3.2.28.5 Length

Property: Length as Long

Description
Returns the character count of the document.

Errors
3900 The object is no longer valid.
3901 Invalid address for the return parameter was specified.

14.3.2.28.6 LineCount

Property: LineCount as Long

Description
Returns the number of lines in the document.

Errors
3900 The object is no longer valid.
3901 Invalid address for the return parameter was specified.

14.3.2.28.7 LineFromPosition

Method: LineFromPosition(nCharPos as Long) as Long

Description
Returns the line number of the character position.

Errors
3900 The object is no longer valid.
3901 Invalid address for the return parameter was specified.

14.3.2.28.8 LineLength

Method: LineLength(nLine as Long) as Long

Description
Returns the length of the line.

Errors
3900 The object is no longer valid.
3901 Invalid address for the return parameter was specified.

14.3.2.28.9 MoveCaret

Method: MoveCaret(nDiff as Long)

Description
Moves the caret nDiff characters.

Errors
3900 The object is no longer valid.
3901 Invalid address for the return parameter was specified.

14.3.2.28.10 Parent

Property: Parent as Document (read-only)

Description
Access the parent of the object.

Errors
3900 The object is no longer valid.
3901 Invalid address for the return parameter was specified.

14.3.2.28.11 PositionFromLine

Method: PositionFromLine(nLine as Long) as Long

Description
Returns the start position of the line.

Errors
3900 The object is no longer valid.
3901 Invalid address for the return parameter was specified.

© 2016-2022 Altova GmbH

14.3.2.28.12 ReplaceText

Method: ReplaceText(nPosFrom as Long, nPosTill as Long, sText as String)

Description
Replaces the text in the specified range.

Errors
3900 The object is no longer valid.
3901 Invalid address for the return parameter was specified.

14.3.2.28.13 SelectionEnd

Property: SelectionEnd as Long

Description
Returns/sets the text selection end position.

Errors
3900 The object is no longer valid.
3901 Invalid address for the return parameter was specified.

14.3.2.28.14 SelectionStart

Property: SelectionStart as Long

Description
Returns/sets the text selection start position.

Errors
3900 The object is no longer valid.
3901 Invalid address for the return parameter was specified.

14.3.2.28.15 SelectText

Method: SelectText(nPosFrom as Long, nPosTill as Long)

Description
Selects the text in the specified range.

Errors
3900 The object is no longer valid.
3901 Invalid address for the return parameter was specified.

14.3.2.28.16 SelText

Property: SelText as String

Description
Returns/sets the selected text.

Errors
3900 The object is no longer valid.
3901 Invalid address for the return parameter was specified.

14.3.2.28.17 Text

Property: Text as String

Description
Returns/sets the document text.

Errors
3900 The object is no longer valid.
3901 Invalid address for the return parameter was specified.

14.3.2.29 WSDLDocumentationDlg

Properties and Methods

Standard automation properties
Application
Parent

Interaction and visibility properties
GlobalElementsAndTypesOnly
OptionsDialogAction
OutputFile
OutputFileDialogAction
SeparateSchemaDocument
ShowProgressBar
ShowResult

Document generation options and methods
OutputFormat
UseFixedDesign
SPSFile
EmbedDiagrams
DiagramFormat
MultipleOutputFiles
EmbedCSSInHTML
CreateDiagramsFolder

IncludeAll

© 2016-2022 Altova GmbH

IncludeBinding
IncludeImportedWSDLFiles
IncludeMessages
IncludeOverview
IncludePortType
IncludeService
IncludeTypes

AllDetails
ShowBindingDiagram
ShowExtensibility
ShowMessageParts
ShowPort
ShowPortTypeDiagram
ShowPortTypeOperations
ShowServiceDiagram
ShowSourceCode
ShowTypesDiagram
ShowUsedBy

Description
This object combines all options for WSDL document generation as they are available through user interface dialog boxes in Authentic Desktop. The document generation options are initialized with the values used during the last generation of WSDL documentation. However, before using the object you have to set the OutputFile property to a valid file path. Use OptionsDialogAction, OutputFileDialogAction and ShowProgressBar to specify the level of user interaction desired. You can use IncludeAll and AllDetails to set whole option groups at once or the individual properties to operate on a finer granularity.

14.3.2.29.1 AllDetails

Method: AllDetails (i_bDetailsOn as Boolean)

Description
Use this method to turn all details options on or off.

Errors
4300 The object is no longer valid.

14.3.2.29.2 Application

Property: Application as Application (read-only)

Description
Access the Authentic Desktop application object.

Errors

3900 The object is no longer valid.
3901 Invalid address for the return parameter was specified.

14.3.2.29.3 CreateDiagramsFolder

Property: CreateDiagramsFolder as Boolean

Description
Set this property to true, to create a directory for the created images. Otherwise the diagrams will be created next to the documentation. This property is only available when the diagrams are not embedded. The property is initialized with the value used during the last call to Document.GenerateWSDLDocumentation. The default for the first run is false.

Errors
3900 The object is no longer valid.
3901 Invalid address for the return parameter was specified.

14.3.2.29.4 DiagramFormat

Property: DiagramFormat as SPYImageKind

Description
This property specifies the generated diagram image type. This property is not available for HTML documentation. The property is initialized with the value used during the last call to Document.GenerateWSDLDocumentation. The default for the first run is PNG.

Errors
3900 The object is no longer valid.
3901 Invalid address for the return parameter was specified.

14.3.2.29.5 EmbedCSSInHTML

Property: EmbedCSSInHTML as Boolean

Description
Set this property to true, to embed the CSS data in the generated HTML document. Otherwise a separate file will be created and linked. This property is only available for HTML documentation. The property is initialized with the value used during the last call to Document.GenerateWSDLDocumentation. The default for the first run is true.

Errors
3900 The object is no longer valid.
3901 Invalid address for the return parameter was specified.

© 2016-2022 Altova GmbH

14.3.2.29.6 EmbedDiagrams

Property: EmbedDiagrams as Boolean

Description
Set this property to true, to embed the diagrams in the generated document. This property is not available for HTML documentation. The property is initialized with the value used during the last call to Document.GenerateWSDLDocumentation. The default for the first run is true.

Errors
3900 The object is no longer valid.
3901 Invalid address for the return parameter was specified.

14.3.2.29.7 GlobalElementsAndTypesOnly

Property: GlobalElementsAndTypesOnly as Boolean

Description
Returns/sets a value indicating whether a full Schema documentation is done or only Global Elements and Types are documented.

Errors
3900 The object is no longer valid.
3901 Invalid address for the return parameter was specified.

14.3.2.29.8 IncludeAll

Method: IncludeAll (i_bInclude as Boolean)

Description
Use this method to mark or unmark all include options.

Errors
4300 The object is no longer valid.

14.3.2.29.9 IncludeBinding

Property: IncludeBinding as Boolean

Description
Set this property to true, to include bindings in the WSDL documentation. The property is initialized with the value used during the last call to Document.GenerateWSDLDocumentation. The default for the first run is true.

Errors
3900 The object is no longer valid.
3901 Invalid address for the return parameter was specified.

14.3.2.29.10 IncludeImportedWSDLFiles

Property: IncludeImportedWSDLFiles as Boolean

Description
Set this property to true, to include imported WSDL finles in the WSDL documentation. The property is initialized with the value used during the last call to Document.GenerateWSDLDocumentation. The default for the first run is true.

Errors
3900 The object is no longer valid.
3901 Invalid address for the return parameter was specified.

14.3.2.29.11 IncludeMessages

Property: IncludeMessages as Boolean

Description
Set this property to true, to include messages in the WSDL documentation. The property is initialized with the value used during the last call to Document.GenerateWSDLDocumentation. The default for the first run is true.

Errors
3900 The object is no longer valid.
3901 Invalid address for the return parameter was specified.

14.3.2.29.12 IncludeOverview

Property: IncludeOverview as Boolean

Description
Set this property to true, to include an overview in the WSDL documentation. The property is initialized with the value used during the last call to Document.GenerateWSDLDocumentation. The default for the first run is true.

Errors
3900 The object is no longer valid.
3901 Invalid address for the return parameter was specified.

14.3.2.29.13 IncludePortType

Property: IncludePortType as Boolean

Description

© 2016-2022 Altova GmbH

Set this property to true, to include port types in the WSDL documentation. The property is initialized with the value used during the last call to Document.GenerateWSDLDocumentation. The default for the first run is true.

Errors

3900 The object is no longer valid.
3901 Invalid address for the return parameter was specified.

14.3.2.29.14 IncludeService

Property: IncludeService as Boolean

Description

Set this property to true, to include services in the WSDL documentation. The property is initialized with the value used during the last call to Document.GenerateWSDLDocumentation. The default for the first run is true.

Errors

3900 The object is no longer valid.
3901 Invalid address for the return parameter was specified.

14.3.2.29.15 IncludeTypes

Property: IncludeTypes as Boolean

Description

Set this property to true, to include types in the WSDL documentation. The property is initialized with the value used during the last call to Document.GenerateWSDLDocumentation. The default for the first run is true.

Errors

3900 The object is no longer valid.
3901 Invalid address for the return parameter was specified.

14.3.2.29.16 MultipleOutputFiles

Property: MultipleOutputFiles as Boolean

Description

Set this property to true, to split the documentation files. The property is initialized with the value used during the last call to Document.GenerateWSDLDocumentation. The default for the first run is false.

Errors

3900 The object is no longer valid.
3901 Invalid value has been used to set the property.
Invalid address for the return parameter was specified.

14.3.2.29.17 OptionsDialogAction

Property: OptionsDialogAction as SPYDialogAction

Description
To allow your script to fill in the default values and let the user see and react on the dialog, set this property to the value *spyDialogUserInput(2)*. If you want your script to define all the options in the schema documentation dialog without any user interaction necessary, use *spyDialogOK(0)*. Default is *spyDialogOK*.

Errors
3900 The object is no longer valid.
3901 Invalid value has been used to set the property.
Invalid address for the return parameter was specified.

14.3.2.29.18 OutputFile

Property: OutputFile as String

Description
Full path and name of the file that will contain the generated documentation. In case of HTML output, additional '.png' files will be generated based on this filename. The default value for this property is an empty string and needs to be replaced before using this object in a call to Document.GenerateWSDLDocumentation.

Errors
3900 The object is no longer valid.
3901 Invalid address for the return parameter was specified.

14.3.2.29.19 OutputFileDialogAction

Property: OutputFileDialogAction as SPYDialogAction

Description
To allow the user to select the output file with a file selection dialog, set this property to *spyDialogUserInput(2)*. If the value stored in OutputFile should be taken and no user interaction should occur, use *spyDialogOK(0)*. Default is *spyDialogOK*.

Errors
3900 The object is no longer valid.
3901 Invalid value has been used to set the property.
Invalid address for the return parameter was specified.

14.3.2.29.20 OutputFormat

Property: OutputFormat as SPYSchemaDocumentationFormat

Description

© 2016-2022 Altova GmbH

Defines the kind of documentation that will be generated: HTML (value=0), MS-Word (value=1), or RTF (value=2). The property gets initialized with the value used during the last call to Document.GenerateWSDLDocumentation. The default for the first run is HTML.

Errors

3900 The object is no longer valid.
3901 Invalid value has been used to set the property.
Invalid address for the return parameter was specified.

14.3.2.29.21 Parent

Property: Parent as Dialogs (read-only)

Description
Access the parent of the object.

Errors

3900 The object is no longer valid.
3901 Invalid address for the return parameter was specified.

14.3.2.29.22 SeparateSchemaDocument

Property: SeparateSchemaDocument as Boolean

Description
Returns/sets a value indicating whether the Schema documentation should be placed in a separate document.

Errors

3900 The object is no longer valid.
3901 Invalid address for the return parameter was specified.

14.3.2.29.23 ShowBindingDiagram

Property: ShowBindingDiagram as Boolean

Description
Set this property to true, to show binding diagrams in the WSDL documentation. The property is initialized with the value used during the last call to Document.GenerateWSDLDocumentation. The default for the first run is true.

Errors

3900 The object is no longer valid.
3901 Invalid address for the return parameter was specified.

14.3.2.29.24 ShowExtensibility

Property: ShowExtensibility as Boolean

Description
Set this property to true, to show service and binding extensibilities in the WSDL documentation. The property is initialized with the value used during the last call to Document.GenerateWSDLDocumentation. The default for the first run is true.

Errors
3900 The object is no longer valid.
3901 Invalid address for the return parameter was specified.

14.3.2.29.25 ShowMessageParts

Property: ShowMessageParts as Boolean

Description
Set this property to true, to show message parts of messges in the WSDL documentation. The property is initialized with the value used during the last call to Document.GenerateWSDLDocumentation. The default for the first run is true.

Errors
3900 The object is no longer valid.
3901 Invalid address for the return parameter was specified.

14.3.2.29.26 ShowPort

Property: ShowPort as Boolean

Description
Set this property to true, to show service ports in the WSDL documentation. The property is initialized with the value used during the last call to Document.GenerateWSDLDocumentation. The default for the first run is true.

Errors
3900 The object is no longer valid.
3901 Invalid address for the return parameter was specified.

14.3.2.29.27 ShowPortTypeDiagram

Property: ShowPortTypeDiagram as Boolean

Description

© 2016-2022 Altova GmbH

Set this property to true, to show port type diagrams in the WSDL documentation. The property is initialized with the value used during the last call to Document.GenerateWSDLDocumentation. The default for the first run is true.

Errors
3900 The object is no longer valid.
3901 Invalid address for the return parameter was specified.

14.3.2.29.28 ShowPortTypeOperations

Property: ShowPortTypeOperations as Boolean

Description
Set this property to true, to show port type operations in the WSDL documentation. The property is initialized with the value used during the last call to Document.GenerateWSDLDocumentation. The default for the first run is true.

Errors
3900 The object is no longer valid.
3901 Invalid address for the return parameter was specified.

14.3.2.29.29 ShowProgressBar

Property: ShowProgressBar as Boolean

Description
Set this property to true, to make the window showing the document generation progress visible. Use false, to hide it. Default is false.

Errors
3900 The object is no longer valid.
3901 Invalid address for the return parameter was specified.

14.3.2.29.30 ShowResult

Property: ShowResult as Boolean

Description
Set this property to true, to automatically open the resulting document when generation was successful. HTML documentation will be opened in Authentic Desktop. To show Word documentation, MS-Word will be started. The property gets initialized with the value used during the last call to Document.GenerateWSDLDocumentation. The default for the first run is true.

Errors
3900 The object is no longer valid.
3901 Invalid address for the return parameter was specified.

14.3.2.29.31 ShowServiceDiagram

Property: ShowServiceDiagram as Boolean

Description
Set this property to true, to show service diagrams in the WSDL documentation. The property is initialized with the value used during the last call to Document.GenerateWSDLDocumentation. The default for the first run is true.

Errors
3900 The object is no longer valid.
3901 Invalid address for the return parameter was specified.

14.3.2.29.32 ShowSourceCode

Property: ShowSourceCode as Boolean

Description
Set this property to true, to show source code for the includes in the WSDL documentation. The property is initialized with the value used during the last call to Document.GenerateWSDLDocumentation. The default for the first run is true.

Errors
3900 The object is no longer valid.
3901 Invalid address for the return parameter was specified.

14.3.2.29.33 ShowTypesDiagram

Property: ShowTypesDiagram as Boolean

Description
Set this property to true, to show type diagrams in the WSDL documentation. The property is initialized with the value used during the last call to Document.GenerateWSDLDocumentation. The default for the first run is true.

Errors
3900 The object is no longer valid.
3901 Invalid address for the return parameter was specified.

14.3.2.29.34 ShowUsedBy

Property: ShowUsedBy as Boolean

Description

© 2016-2022 Altova GmbH

Set this property to true, to show the used-by relation for types, bindings and messages definitions in the WSDL documentation. The property is initialized with the value used during the last call to Document.GenerateWSDLDocumentation. The default for the first run is true.

Errors
3900 The object is no longer valid.
3901 Invalid address for the return parameter was specified.

14.3.2.29.35 UseFixedDesign

Property: UseFixedDesign as Boolean

Description
Specifies whether the documentation should be created with a fixed design or with a design specified by a SPS file (which requires StyleVision).

Errors
3900 The object is no longer valid.
3901 Invalid address for the return parameter was specified.

14.3.2.29.36 SPSFile

Property: SPSFile as String

Description
Full path and name of the SPS file that will be used to generate the documentation.

Errors
3900 The object is no longer valid.
3901 Invalid address for the return parameter was specified.

14.3.2.30 WSDL20DocumentationDlg

Properties and Methods

Standard automation properties
Application
Parent

Interaction and visibility properties
GlobalElementsAndTypesOnly
OptionsDialogAction
OutputFile
OutputFileDialogAction
SeparateSchemaDocument
ShowProgressBar

ShowResult

Document generation options and methods
OutputFormat
UseFixedDesign
SPSFile
EmbedDiagrams
DiagramFormat
MultipleOutputFiles
EmbedCSSInHTML
CreateDiagramsFolder

IncludeAll
IncludeBinding
IncludeImportedWSDLFiles
IncludeInterface
IncludeOverview
IncludeService
IncludeTypes

AllDetails
ShowBindingDiagram
ShowExtensibility
ShowEndpoint
ShowFault
ShowInterfaceDiagram
ShowOperation
ShowServiceDiagram
ShowSourceCode
ShowTypesDiagram
ShowUsedBy

Description
This object combines all options for WSDL document generation as they are available through user interface dialog boxes in Authentic Desktop. The document generation options are initialized with the values used during the last generation of WSDL documentation. However, before using the object you have to set the OutputFile property to a valid file path. Use OptionsDialogAction, OutputFileDialogAction and ShowProgressBar to specify the level of user interaction desired. You can use IncludeAll and AllDetails to set whole option groups at once or the individual properties to operate on a finer granularity.

14.3.2.30.1 AllDetails

Method: AllDetails (i_bDetailsOn as Boolean)

Description
Use this method to turn all details options on or off.

Errors
4300 The object is no longer valid.

 © 2016-2022 Altova GmbH

14.3.2.30.2 Application

Property: Application as Application (read-only)

Description
Access the Authentic Desktop application object.

Errors
4300 The object is no longer valid.
4301 Invalid address for the return parameter was specified.

14.3.2.30.3 CreateDiagramsFolder

Property: CreateDiagramsFolder as Boolean

Description
Set this property to true, to create a directory for the created images. Otherwise the diagrams will be created next to the documentation. This property is only available when the diagrams are not embedded. The property is initialized with the value used during the last call to Document.GenerateWSD20LDocumentation. The default for the first run is false.

Errors
4300 The object is no longer valid.
4301 Invalid address for the return parameter was specified.

14.3.2.30.4 DiagramFormat

Property: DiagramFormat as SPYImageKind

Description
This property specifies the generated diagram image type. This property is not available for HTML documentation. The property is initialized with the value used during the last call to Document.GenerateWSDL20Documentation. The default for the first run is PNG.

Errors
4300 The object is no longer valid.
4301 Invalid address for the return parameter was specified.

14.3.2.30.5 EmbedCSSInHTML

Property: EmbedCSSInHTML as Boolean

Description

Set this property to true, to embed the CSS data in the generated HTML document. Otherwise a separate file will be created and linked. This property is only available for HTML documentation. The property is initialized with the value used during the last call to Document.GenerateWSDL20Documentation. The default for the first run is true.

Errors

4300 The object is no longer valid.
4301 Invalid address for the return parameter was specified.

14.3.2.30.6 EmbedDiagrams

Property: EmbedDiagrams as Boolean

Description

Set this property to true, to embed the diagrams in the generated document. This property is not available for HTML documentation. The property is initialized with the value used during the last call to Document.GenerateWSDL20Documentation. The default for the first run is true.

Errors

4300 The object is no longer valid.
4301 Invalid address for the return parameter was specified.

14.3.2.30.7 GlobalElementsAndTypesOnly

Property: GlobalElementsAndTypesOnly as Boolean

Description

Returns/sets a value indicating whether a full Schema documentation is done or only Global Elements and Types are documented.

Errors

3900 The object is no longer valid.
3901 Invalid address for the return parameter was specified.

14.3.2.30.8 IncludeAll

Method: IncludeAll (i_bInclude as Boolean)

Description

Use this method to mark or unmark all include options.

Errors

4300 The object is no longer valid.

© 2016-2022 Altova GmbH

14.3.2.30.9 IncludeBinding

Property: IncludeBinding as Boolean

Description
Set this property to true, to include bindings in the WSDL documentation. The property is initialized with the value used during the last call to Document.GenerateWSDL20Documentation. The default for the first run is true.

Errors
4300 The object is no longer valid.
4301 Invalid address for the return parameter was specified.

14.3.2.30.10 IncludeImportedWSDLFiles

Property: IncludeImportedWSDLFiles as Boolean

Description
Set this property to true, to include imported WSDL finles in the WSDL documentation. The property is initialized with the value used during the last call to Document.GenerateWSDL20Documentation. The default for the first run is true.

Errors
4300 The object is no longer valid.
4301 Invalid address for the return parameter was specified.

14.3.2.30.11 IncludeInterface

Property: IncludeInterface as Boolean

Description
Set this property to true, to include interfaces in the WSDL documentation. The property is initialized with the value used during the last call to Document.GenerateWSDL20Documentation. The default for the first run is true.

Errors
4300 The object is no longer valid.
4301 Invalid address for the return parameter was specified.

14.3.2.30.12 IncludeOverview

Property: IncludeOverview as Boolean

Description
Set this property to true, to include an overview in the WSDL documentation. The property is initialized with the value used during the last call to Document.GenerateWSDL20Documentation. The default for the first run is true.

Errors
4300 The object is no longer valid.
4301 Invalid address for the return parameter was specified.

14.3.2.30.13 IncludeService

Property: IncludeService as Boolean

Description
Set this property to true, to include services in the WSDL documentation. The property is initialized with the value used during the last call to Document.GenerateWSDL20Documentation. The default for the first run is true.

Errors
4300 The object is no longer valid.
4301 Invalid address for the return parameter was specified.

14.3.2.30.14 IncludeTypes

Property: IncludeTypes as Boolean

Description
Set this property to true, to include types in the WSDL documentation. The property is initialized with the value used during the last call to Document.GenerateWSDL20Documentation. The default for the first run is true.

Errors
4300 The object is no longer valid.
4301 Invalid address for the return parameter was specified.

14.3.2.30.15 MultipleOutputFiles

Property: MultipleOutputFiles as Boolean

Description
Set this property to true, to split the documentation files. The property is initialized with the value used during the last call to Document.GenerateWSDL20Documentation. The default for the first run is false.

Errors
4300 The object is no longer valid.
4301 Invalid value has been used to set the property.
Invalid address for the return parameter was specified.

14.3.2.30.16 OptionsDialogAction

Property: OptionsDialogAction as SPYDialogAction

© 2016-2022 Altova GmbH

Description
To allow your script to fill in the default values and let the user see and react on the dialog, set this property to the value *spyDialogUserInput(2)*. If you want your script to define all the options in the schema documentation dialog without any user interaction necessary, use *spyDialogOK(0)*. Default is *spyDialogOK*.

Errors
4300 The object is no longer valid.
4301 Invalid value has been used to set the property.
Invalid address for the return parameter was specified.

14.3.2.30.17 OutputFile

Property: OutputFile as String

Description
Full path and name of the file that will contain the generated documentation. In case of HTML output, additional '.png' files will be generated based on this filename. The default value for this property is an empty string and needs to be replaced before using this object in a call to Document.GenerateWSDL20Documentation.

Errors
4300 The object is no longer valid.
4301 Invalid address for the return parameter was specified.

14.3.2.30.18 OutputFileDialogAction

Property: OutputFileDialogAction as SPYDialogAction

Description
To allow the user to select the output file with a file selection dialog, set this property to *spyDialogUserInput(2)*. If the value stored in OutputFile should be taken and no user interaction should occur, use *spyDialogOK(0)*. Default is *spyDialogOK*.

Errors
4300 The object is no longer valid.
4301 Invalid value has been used to set the property.
Invalid address for the return parameter was specified.

14.3.2.30.19 OutputFormat

Property: OutputFormat as SPYSchemaDocumentationFormat

Description
Defines the kind of documentation that will be generated: HTML (value=0), MS-Word (value=1), or RTF (value=2). The property gets initialized with the value used during the last call to Document.GenerateWSDL20Documentation. The default for the first run is HTML.

Errors

4300 The object is no longer valid.
4301 Invalid value has been used to set the property.
Invalid address for the return parameter was specified.

14.3.2.30.20 Parent

Property: Parent as Dialogs (read-only)

Description
Access the parent of the object.

Errors
4300 The object is no longer valid.
4301 Invalid address for the return parameter was specified.

14.3.2.30.21 SeparateSchemaDocument

Property: SeparateSchemaDocument as Boolean

Description
Returns/sets a value indicating whether the Schema documentation should be placed in a separate document.

Errors
3900 The object is no longer valid.
3901 Invalid address for the return parameter was specified.

14.3.2.30.22 ShowBindingDiagram

Property: ShowBindingDiagram as Boolean

Description
Set this property to true, to show binding diagrams in the WSDL documentation. The property is initialized with the value used during the last call to Document.GenerateWSDL20Documentation. The default for the first run is true.

Errors
4300 The object is no longer valid.
4301 Invalid address for the return parameter was specified.

14.3.2.30.23 ShowEndpoint

Property: ShowEndpoint as Boolean

Description

© 2016-2022 Altova GmbH

Set this property to true, to show service endpoints in the WSDL documentation. The property is initialized with the value used during the last call to Document.GenerateWSDL20Documentation. The default for the first run is true.

Errors
4300 The object is no longer valid.
4301 Invalid address for the return parameter was specified.

14.3.2.30.24 ShowExtensibility

Property: ShowExtensibility as Boolean

Description
Set this property to true, to show service and binding extensibilities in the WSDL documentation. The property is initialized with the value used during the last call to Document.GenerateWSDL20Documentation. The default for the first run is true.

Errors
4300 The object is no longer valid.
4301 Invalid address for the return parameter was specified.

14.3.2.30.25 ShowFault

Property: ShowFault as Boolean

Description
Set this property to true, to show faults in the WSDL documentation. The property is initialized with the value used during the last call to Document.GenerateWSDL20Documentation. The default for the first run is true.

Errors
4300 The object is no longer valid.
4301 Invalid address for the return parameter was specified.

14.3.2.30.26 ShowInterfaceDiagram

Property: ShowInterfaceDiagram as Boolean

Description
Set this property to true, to show interface diagrams in the WSDL documentation. The property is initialized with the value used during the last call to Document.GenerateWSDL20Documentation. The default for the first run is true.

Errors
4300 The object is no longer valid.
4301 Invalid address for the return parameter was specified.

14.3.2.30.27 ShowOperation

Property: ShowOperation as Boolean

Description
Set this property to true, to show interface and binding operations in the WSDL documentation. The property is initialized with the value used during the last call to Document.GenerateWSDL20Documentation. The default for the first run is true.

Errors
4300 The object is no longer valid.
4301 Invalid address for the return parameter was specified.

14.3.2.30.28 ShowProgressBar

Property: ShowProgressBar as Boolean

Description
Set this property to true, to make the window showing the document generation progress visible. Use false, to hide it. Default is false.

Errors
4300 The object is no longer valid.
4301 Invalid address for the return parameter was specified.

14.3.2.30.29 ShowResult

Property: ShowResult as Boolean

Description
Set this property to true, to automatically open the resulting document when generation was successful. HTML documentation will be opened in Authentic Desktop. To show Word documentation, MS-Word will be started. The property gets initialized with the value used during the last call to Document.GenerateWSDL20Documentation. The default for the first run is true.

Errors
4300 The object is no longer valid.
4301 Invalid address for the return parameter was specified.

14.3.2.30.30 ShowServiceDiagram

Property: ShowServiceDiagram as Boolean

Description

© 2016-2022 Altova GmbH

Set this property to true, to show service diagrams in the WSDL documentation. The property is initialized with the value used during the last call to Document.GenerateWSDL20Documentation. The default for the first run is true.

Errors
4300 The object is no longer valid.
4301 Invalid address for the return parameter was specified.

14.3.2.30.31 ShowSourceCode

Property: ShowSourceCode as Boolean

Description
Set this property to true, to show source code for the includes in the WSDL documentation. The property is initialized with the value used during the last call to Document.GenerateWSDL20Documentation. The default for the first run is true.

Errors
4300 The object is no longer valid.
4301 Invalid address for the return parameter was specified.

14.3.2.30.32 ShowTypesDiagram

Property: ShowTypesDiagram as Boolean

Description
Set this property to true, to show type diagrams in the WSDL documentation. The property is initialized with the value used during the last call to Document.GenerateWSDL20Documentation. The default for the first run is true.

Errors
4300 The object is no longer valid.
4301 Invalid address for the return parameter was specified.

14.3.2.30.33 ShowUsedBy

Property: ShowUsedBy as Boolean

Description
Set this property to true, to show the used-by relation for types, bindings and messages definitions in the WSDL documentation. The property is initialized with the value used during the last call to Document.GenerateWSDL20Documentation. The default for the first run is true.

Errors
4300 The object is no longer valid.
4301 Invalid address for the return parameter was specified.

14.3.2.30.34 SPSFile

Property: SPSFile as String

Description
Full path and name of the SPS file that will be used to generate the documentation.

Errors
4300 The object is no longer valid.
4301 Invalid address for the return parameter was specified.

14.3.2.30.35 UseFixedDesign

Property: UseFixedDesign as Boolean

Description
Specifies whether the documentation should be created with a fixed design or with a design specified by a SPS file (which requires StyleVision).

Errors
4300 The object is no longer valid.
4301 Invalid address for the return parameter was specified.

14.3.2.31 XBRLDocumentationDlg

Properties and Methods

Standard automation properties
Application
Parent

Interaction and visibility properties
OptionsDialogAction
OutputFile
OutputFileDialogAction
ShowProgressBar
ShowResult

Document generation options and methods
OutputFormat
UseFixedDesign
SPSFile
EmbedDiagrams
DiagramFormat

© 2016-2022 Altova GmbH

EmbedCSSInHTML
CreateDiagramsFolder

IncludeAll
IncludeOverview
IncludeNamespacePrefixes
IncludeGlobalElements
IncludeDefinitionLinkroles
IncludePresentationLinkroles
IncludeCalculationLinkroles

AllDetails
ShowDiagram
ShowSubstitutiongroup
ShowItemtype
ShowBalance
ShowPeriod
ShowAbstract
ShowNillable
ShowLabels
ShowReferences
ShowLinkbaseReferences

ShortQualifiedName
ShowImportedElements

Description
This object combines all options for XBRL document generation as they are available through user interface dialog boxes in Authentic Desktop. The document generation options are initialized with the values used during the last generation of XBRL documentation. However, before using the object you have to set the OutputFile property to a valid file path. Use OptionsDialogAction, OutputFileDialogAction and ShowProgressBar to specify the level of user interaction desired. You can use IncludeAll and AllDetails to set whole option groups at once or the individual properties to operate on a finer granularity.

14.3.2.31.1 AllDetails

Method: AllDetails (i_bDetailsOn as Boolean)

Description
Use this method to turn all details options on or off.

Errors
4400 The object is no longer valid.

14.3.2.31.2 Application

Property: Application as Application (read-only)

Description
Access the Authentic Desktop application object.

Errors
4400 The object is no longer valid.
4401 Invalid address for the return parameter was specified.

14.3.2.31.3 CreateDiagramsFolder

Property: CreateDiagramsFolder as Boolean

Description
Set this property to true, to create a directory for the created images. Otherwise the diagrams will be created next to the documentation. This property is only available when the diagrams are not embedded. The property is initialized with the value used during the last call to Document.GenerateXBRLDocumentation. The default for the first run is false.

Errors
4400 The object is no longer valid.
4401 Invalid address for the return parameter was specified.

14.3.2.31.4 DiagramFormat

Property: DiagramFormat as SPYImageKind

Description
This property specifies the generated diagram image type. This property is not available for HTML documentation. The property is initialized with the value used during the last call to Document.GenerateXBRLDocumentation. The default for the first run is PNG.

Errors
4400 The object is no longer valid.
4401 Invalid address for the return parameter was specified.

14.3.2.31.5 EmbedCSSInHTML

Property: EmbedCSSInHTML as Boolean

Description
Set this property to true, to embed the CSS data in the generated HTML document. Otherwise a separate file will be created and linked. This property is only available for HTML documentation. The property is initialized

© 2016-2022 Altova GmbH

with the value used during the last call to Document.GenerateXBRLDocumentation. The default for the first run is true.

Errors
4400 The object is no longer valid.
4401 Invalid address for the return parameter was specified.

14.3.2.31.6 EmbedDiagrams

Property: EmbedDiagrams as Boolean

Description
Set this property to true, to embed the diagrams in the generated document. This property is not available for HTML documentation. The property is initialized with the value used during the last call to Document.GenerateXBRLDocumentation. The default for the first run is true.

Errors
4400 The object is no longer valid.
4401 Invalid address for the return parameter was specified.

14.3.2.31.7 IncludeAll

Method: IncludeAll (i_blnclude as Boolean)

Description
Use this method to mark or unmark all include options.

Errors
4400 The object is no longer valid.

14.3.2.31.8 IncludeCalculationLinkroles

Property: IncludeCalculationLinkroles as Boolean

Description
Set this property to true, to include calculation linkroles in the XBRL documentation. The property is initialized with the value used during the last call to Document.GenerateXBRLDocumentation. The default for the first run is true.

Errors
4400 The object is no longer valid.
4401 Invalid address for the return parameter was specified.

14.3.2.31.9 IncludeDefinitionLinkroles

Property: IncludeDefinitionLinkroles as Boolean

Description
Set this property to true, to include definition linkroles in the XBRL documentation. The property is initialized with the value used during the last call to Document.GenerateXBRLDocumentation. The default for the first run is true.

Errors
4400 The object is no longer valid.
4401 Invalid address for the return parameter was specified.

14.3.2.31.10 IncludeGlobalElements

Property: IncludeGlobalElements as Boolean

Description
Set this property to true, to include global elements in the XBRL documentation. The property is initialized with the value used during the last call to Document.GenerateXBRLDocumentation. The default for the first run is true.

Errors
4400 The object is no longer valid.
4401 Invalid address for the return parameter was specified.

14.3.2.31.11 IncludeNamespacePrefixes

Property: IncludeNamespacePrefixes as Boolean

Description
Set this property to true, to include namespace prefixes in the XBRL documentation. The property is initialized with the value used during the last call to Document.GenerateXBRLDocumentation. The default for the first run is true.

Errors
4400 The object is no longer valid.
4401 Invalid address for the return parameter was specified.

14.3.2.31.12 IncludeOverview

Property: IncludeOverview as Boolean

Description
Set this property to true, to include an overview in the XBRL documentation. The property is initialized with the value used during the last call to Document.GenerateXBRLDocumentation. The default for the first run is true.

© 2016-2022 Altova GmbH

Errors
4400 The object is no longer valid.
4401 Invalid address for the return parameter was specified.

14.3.2.31.13 IncludePresentationLinkroles

Property: IncludePresentationLinkroles as Boolean

Description
Set this property to true, to include presentation linkroles in the XBRL documentation. The property is initialized with the value used during the last call to Document.GenerateXBRLDocumentation. The default for the first run is true.

Errors
4400 The object is no longer valid.
4401 Invalid address for the return parameter was specified.

14.3.2.31.14 OptionsDialogAction

Property: OptionsDialogAction as SPYDialogAction

Description
To allow your script to fill in the default values and let the user see and react on the dialog, set this property to the value *spyDialogUserInput(2)*. If you want your script to define all the options in the schema documentation dialog without any user interaction necessary, use *spyDialogOK(0)*. Default is *spyDialogOK*.

Errors
4400 The object is no longer valid.
4401 Invalid value has been used to set the property.
Invalid address for the return parameter was specified.

14.3.2.31.15 OutputFile

Property: OutputFile as String

Description
Full path and name of the file that will contain the generated documentation. In case of HTML output, additional '.png' files will be generated based on this filename. The default value for this property is an empty string and needs to be replaced before using this object in a call to Document.GenerateXBRLDocumentation.

Errors
4400 The object is no longer valid.
4401 Invalid address for the return parameter was specified.

14.3.2.31.16 OutputFileDialogAction

Property: OutputFileDialogAction as SPYDialogAction

Description
To allow the user to select the output file with a file selection dialog, set this property to *spyDialogUserInput(2)*. If the value stored in OutputFile should be taken and no user interaction should occur, use *spyDialogOK(0)*. Default is *spyDialogOK*.

Errors
4400 The object is no longer valid.
4401 Invalid value has been used to set the property.
Invalid address for the return parameter was specified.

14.3.2.31.17 OutputFormat

Property: OutputFormat as SPYSchemaDocumentationFormat

Description
Defines the kind of documentation that will be generated: HTML (value=`0`), MS-Word (value=`1`), or RTF (value=`2`). The property gets initialized with the value used during the last call to Document.GenerateXBRLDocumentation. The default for the first run is HTML.

Errors
4400 The object is no longer valid.
4401 Invalid value has been used to set the property.
Invalid address for the return parameter was specified.

14.3.2.31.18 Parent

Property: Parent as Dialogs (read-only)

Description
Access the parent of the object.

Errors
4400 The object is no longer valid.
4401 Invalid address for the return parameter was specified.

14.3.2.31.19 ShortQualifiedName

Property: ShortQualifiedName as Boolean

Description

© 2016-2022 Altova GmbH

Set this property to true, to use short qualified names in the XBRL documentation. The property is initialized with the value used during the last call to Document.GenerateXBRLDocumentation. The default for the first run is true.

Errors
4400 The object is no longer valid.
4401 Invalid address for the return parameter was specified.

14.3.2.31.20 ShowAbstract

Property: ShowAbstract as Boolean

Description
Set this property to true, to show abstracts in the XBRL documentation. The property is initialized with the value used during the last call to Document.GenerateXBRLDocumentation. The default for the first run is true.

Errors
4400 The object is no longer valid.
4401 Invalid address for the return parameter was specified.

14.3.2.31.21 ShowBalance

Property: ShowBalance as Boolean

Description
Set this property to true, to show balances in the XBRL documentation. The property is initialized with the value used during the last call to Document.GenerateXBRLDocumentation. The default for the first run is true.

Errors
4400 The object is no longer valid.
4401 Invalid address for the return parameter was specified.

14.3.2.31.22 ShowDiagram

Property: ShowDiagram as Boolean

Description
Set this property to true, to show diagrams in the XBRL documentation. The property is initialized with the value used during the last call to Document.GenerateXBRLDocumentation. The default for the first run is true.

Errors
4400 The object is no longer valid.
4401 Invalid address for the return parameter was specified.

14.3.2.31.23 ShowImportedElements

Property: ShowImportedElements as Boolean

Description
Set this property to true, to show imported elements in the XBRL documentation. The property is initialized with the value used during the last call to Document.GenerateXBRLDocumentation. The default for the first run is true.

Errors
4400 The object is no longer valid.
4401 Invalid address for the return parameter was specified.

14.3.2.31.24 ShowItemtype

Property: ShowItemtype as Boolean

Description
Set this property to true, to show item types in the XBRL documentation. The property is initialized with the value used during the last call to Document.GenerateXBRLDocumentation. The default for the first run is true.

Errors
4400 The object is no longer valid.
4401 Invalid address for the return parameter was specified.

14.3.2.31.25 ShowLabels

Property: ShowLabels as Boolean

Description
Set this property to true, to show labels in the XBRL documentation. The property is initialized with the value used during the last call to Document.GenerateXBRLDocumentation. The default for the first run is true.

Errors
4400 The object is no longer valid.
4401 Invalid address for the return parameter was specified.

14.3.2.31.26 ShowLinkbaseReferences

Property: ShowLinkbaseReferences as Boolean

Description

© 2016-2022 Altova GmbH

Set this property to true, to show linkbase references in the XBRL documentation. The property is initialized with the value used during the last call to Document.GenerateXBRLDocumentation. The default for the first run is true.

Errors
4400 The object is no longer valid.
4401 Invalid address for the return parameter was specified.

14.3.2.31.27 ShowNillable

Property: ShowNillable as Boolean

Description
Set this property to true, to show nillable properties in the XBRL documentation. The property is initialized with the value used during the last call to Document.GenerateXBRLDocumentation. The default for the first run is true.

Errors
4400 The object is no longer valid.
4401 Invalid address for the return parameter was specified.

14.3.2.31.28 ShowPeriod

Property: ShowPeriod as Boolean

Description
Set this property to true, to show periods in the XBRL documentation. The property is initialized with the value used during the last call to Document.GenerateXBRLDocumentation. The default for the first run is true.

Errors
4400 The object is no longer valid.
4401 Invalid address for the return parameter was specified.

14.3.2.31.29 ShowProgressBar

Property: ShowProgressBar as Boolean

Description
Set this property to true, to make the window showing the document generation progress visible. Use false, to hide it. Default is false.

Errors
4400 The object is no longer valid.
4401 Invalid address for the return parameter was specified.

14.3.2.31.30 ShowReferences

Property: ShowReferences as Boolean

Description
Set this property to true, to show references in the XBRL documentation. The property is initialized with the value used during the last call to Document.GenerateXBRLDocumentation. The default for the first run is true.

Errors
4400 The object is no longer valid.
4401 Invalid address for the return parameter was specified.

14.3.2.31.31 ShowResult

Property: ShowResult as Boolean

Description
Set this property to true, to automatically open the resulting document when generation was successful. HTML documentation will be opened in Authentic Desktop. To show Word documentation, MS-Word will be started. The property gets initialized with the value used during the last call to Document.GenerateXBRLDocumentation. The default for the first run is true.

Errors
4400 The object is no longer valid.
4401 Invalid address for the return parameter was specified.

14.3.2.31.32 ShowSubstitutiongroup

Property: ShowSubstitutiongroup as Boolean

Description
Set this property to true, to show substitution groups in the XBRL documentation. The property is initialized with the value used during the last call to Document.GenerateXBRLDocumentation. The default for the first run is true.

Errors
4400 The object is no longer valid.
4401 Invalid address for the return parameter was specified.

14.3.2.31.33 SPSFile

Property: SPSFile as String

Description

© 2016-2022 Altova GmbH

Full path and name of the SPS file that will be used to generate the documentation.

Errors
4400 The object is no longer valid.
4401 Invalid address for the return parameter was specified.

14.3.2.31.34 UseFixedDesign

Property: UseFixedDesign as Boolean

Description
Specifies whether the documentation should be created with a fixed design or with a design specified by a SPS file (which requires StyleVision).

Errors
4400 The object is no longer valid.
4401 Invalid address for the return parameter was specified.

14.3.2.32 XMLData

Properties
Kind
Name
TextValue

HasChildren
MayHaveChildren
Parent

Methods
GetFirstChild
GetNextChild
GetCurrentChild

InsertChild
InsertChildAfter
InsertChildBefore
AppendChild

EraseAllChildren
EraseChild
EraseCurrentChild

IsSameNode

CountChildren
CountChildrenKind

GetChild
GetChildAttribute
GetChildElement
GetChildKind
GetNamespacePrefixForURI

HasChildrenKind
SetTextValueXMLEncoded

Description
The XMLData interface provides direct XML-level access to a document. You can read and directly modify the XML representation of the document. However, please, note the following restrictions:

- The XMLData representation is only valid when the document is shown in grid view or authentic view.
- When in authentic view, additional XMLData elements are automatically inserted as parents of each visible document element. Typically this is an XMLData of kind spyXMLDataElement with the Name property set to 'Text'.
- When you use the XMLData interface while in a different view mode you will not receive errors, but changes are not reflected to the view and might get lost during the next view switch.

Note also:

- Setting a new text value for an XML element is possible if the element does not have non-text children. A text value can be set even if the element has attributes.
- When setting a new text value for an XML element which has more than one text child, the latter will be deleted and replaced by one new text child.
- When reading the text value of an XML element which has more than one text child, only the value of the first text child will be returned.

14.3.2.32.1 AppendChild

Declaration: AppendChild (*pNewData* as XMLData)

Description
AppendChild appends pNewData as last child to the XMLData object.

Errors
1500 The XMLData object is no longer valid.
1505 Invalid XMLData kind was specified.
1506 Invalid address for the return parameter was specified.
1507 Element cannot have Children
1512 Cyclic insertion - new data element is already part of document
1514 Invalid XMLData kind was specified for this position.
1900 Document must not be modified

Example

```
Dim objCurrentParent As XMLData
Dim objNewChild As XMLData
```


© 2016-2022 Altova GmbH

```
Set objNewChild = objSpy.ActiveDocument.CreateChild(spyXMLDataElement)
Set objCurrentParent = objSpy.ActiveDocument.RootElement

objCurrentParent.AppendChild objNewChild

Set objNewChild = Nothing
```

14.3.2.32.2 CountChildren

Declaration: CountChildren as long

Description
CountChildren gets the number of children.

Available with TypeLibrary version 1.5

Errors
1500 The XMLData object is no longer valid.

14.3.2.32.3 CountChildrenKind

Declaration: CountChildrenKind (*nKind* as SPYXMLDataKind) as long

Description
CountChildrenKind gets the number of children of the specific kind.

Available with TypeLibrary version 1.5

Errors
1500 The XMLData object is no longer valid.

14.3.2.32.4 EraseAllChildren

Declaration: EraseAllChildren

Description
EraseAllChildren deletes all associated children of the XMLData object.

Errors
1500 The XMLData object is no longer valid.
1900 Document must not be modified

Example
The sample erases all elements of the active document.

```
Dim objCurrentParent As XMLData

Set objCurrentParent = objSpy.ActiveDocument.RootElement
objCurrentParent.EraseAllChildren
```

14.3.2.32.5 EraseChild

Method: EraseChild (Child as XMLData)

Description
Deletes the given child node.

Errors
1500 Invalid object.
1506 Invalid input xml
1510 Invalid parameter.

14.3.2.32.6 EraseCurrentChild

Declaration: EraseCurrentChild

Description
EraseCurrentChild deletes the current XMLData child object. Before you call EraseCurrentChild you must initialize an internal iterator with XMLData.GetFirstChild. After deleting the current child, EraseCurrentChild increments the internal iterator of the XMLData element. No error is returned when the last child gets erased and the iterator is moved past the end of the child list. The next call to EraseCurrentChild however, will return error 1503.

Errors
1500 The XMLData object is no longer valid.
1503 No iterator is initialized for this XMLData object, or the iterator points past the last child.
1900 Document must not be modified

Examples

```
// ---------------------------------------
// XMLSpy scripting environment - JScript
// erase all children of XMLData
// ---------------------------------------
// let's get an XMLData element, we assume that the
// cursor selects the parent of a list in grid view
var objList = Application.ActiveDocument.GridView.CurrentFocus;

// the following line would be shorter, of course
//      objList.EraseAllChildren ();

// but we want to demonstrate the usage of EraseCurrentChild
if ((objList != null) && (objList.HasChildren))
```


© 2016-2022 Altova GmbH

```
{
        try
        {
                objEle = objList.GetFirstChild(-1);
                while (objEle != null)
                        objList.EraseCurrentChild();
                        // no need to call GetNextChild
        }
        catch (err)
                // 1503 - we reached end of child list
                { if ((err.number & 0xffff) != 1503) throw (err); }
}
```

14.3.2.32.7 GetChild

Declaration: GetChild (*position* as long) as XMLData

Return Value
Returns an XML element as XMLData object.

Description
GetChild() returns a reference to the child at the given index (zero-based).

Available with TypeLibrary version 1.5

Errors
1500 The XMLData object is no longer valid.
1510 Invalid address for the return parameter was specified.

14.3.2.32.8 GetChildAttribute

Method: GetChildAttribute (strName as string) child as XMLData object (`NULL` on error)

Description
Retrieves the attribute having the given name.

Errors
1500 Invalid object.
1510 Invalid parameter.

14.3.2.32.9 GetChildElement

Method: GetChildElement (strName as string, nIndex as long) child as XMLData object (`NULL` on error)

Description
Retrieves the Nth child element with the given name.

Errors
1500 Invalid object.
1510 Invalid parameter.

14.3.2.32.10 GetChildKind

Declaration: GetChildKind (*position* as long, *nKind* as SPYXMLDataKind) as XMLData

Return Value
Returns an XML element as XMLData object.

Description
GetChildKind() returns a reference to a child of this kind at the given index (zero-based). The position parameter is relative to the number of children of the specified kind and not to all children of the object.

Available with TypeLibrary version 1.5

Errors
1500 The XMLData object is no longer valid.
1510 Invalid address for the return parameter was specified.

14.3.2.32.11 GetCurrentChild

Declaration: GetCurrentChild as XMLData

Return Value
Returns an XML element as XMLData object.

Description
GetCurrentChild gets the current child. Before you call GetCurrentChild you must initialize an internal iterator with XMLData.GetFirstChild.

Errors
1500 The XMLData object is no longer valid.
1503 No iterator is initialized for this XMLData object.
1510 Invalid address for the return parameter was specified.

© 2016-2022 Altova GmbH

14.3.2.32.12 GetFirstChild

Declaration: GetFirstChild (*nKind* as SPYXMLDataKind) as XMLData

Return Value
Returns an XML element as XMLData object.

Description
GetFirstChild initializes a new iterator and returns the first child. Set nKind = -1 to get an iterator for all kinds of children.
REMARK: The iterator is stored inside the XMLData object and gets destroyed when the XMLData object gets destroyed. Be sure to keep a reference to this object as long as you want to use GetCurrentChild, GetNextChild or EraseCurrentChild.

Errors
1500 The XMLData object is no longer valid.
1501 Invalid XMLData kind was specified.
1504 Element has no children of specified kind.
1510 Invalid address for the return parameter was specified.

Example
See the example at XMLData.GetNextChild.

14.3.2.32.13 GetNamespacePrefixForURI

Method: GetNamespacePrefixForURI (strURI as string) strNS as string

Description
Returns the namespace prefix of the supplied URI.

Errors
1500 Invalid object.
1510 Invalid parameter.

14.3.2.32.14 GetNextChild

Declaration: GetNextChild as XMLData

Return Value
Returns an XML element as XMLData object.

Description
GetNextChild steps to the next child of this element. Before you call GetNextChild you must initialize an internal iterator with XMLData.GetFirstChild.

Check for the last child of the element as shown in the sample below.

Errors

1500 The XMLData object is no longer valid.
1503 No iterator is initialized for this XMLData object.
1510 Invalid address for the return parameter was specified.

Examples

```
' ---------------------------------------------
' VBA code snippet - iterate XMLData children
' ---------------------------------------------
On Error Resume Next
Set objParent = objSpy.ActiveDocument.RootElement

'get elements of all kinds
Set objCurrentChild = objParent.GetFirstChild(-1)

Do
        'do something useful with the child

        'step to next child
        Set objCurrentChild = objParent.GetNextChild
Loop Until (Err.Number - vbObjectError = 1503)
```

```
// ---------------------------------------
// XMLSpy scripting environment - JScript
// iterate through children of XMLData
// ---------------------------------------
try
{
        var objXMLData = ... // initialize somehow
        var objChild = objXMLData.GetFirstChild(-1);

        while (true)
        {
                // do something usefull with objChild

                objChild = objXMLData.GetNextChild();
        }
}
catch (err)
{
        if ((err.number & 0xffff) == 1504)
                ; // element has no children
        else if ((err.number & 0xffff) == 1503)
                ;       // last child reached
        else
                throw (err);
}
```


© 2016-2022 Altova GmbH

14.3.2.32.15 GetTextValueXMLDecoded

Method: GetTextValueXMLDecoded ()as string

Description
Gets the decoded text value of the XML.

Errors
1500 Invalid object.
1510 Invalid parameter.

14.3.2.32.16 HasChildren

Declaration: HasChildren as Boolean

Description
The property is true if the object is the parent of other XMLData objects. This property is read-only.

Errors
1500 The XMLData object is no longer valid.
1510 Invalid address for the return parameter was specified.

14.3.2.32.17 HasChildrenKind

Declaration: HasChildrenKind (*nKind* as SPYXMLDataKind) as Boolean

Description
The method returns true if the object is the parent of other XMLData objects of the specific kind.

Available with TypeLibrary version 1.5

Errors
1500 The XMLData object is no longer valid.
1510 Invalid address for the return parameter was specified.

14.3.2.32.18 InsertChild

Declaration: InsertChild (*pNewData* as XMLData)

Description
InsertChild inserts the new child before the current child (see also XMLData.GetFirstChild, XMLData.GetNextChild to set the current child).

Errors

1500 The XMLData object is no longer valid.
1503 No iterator is initialized for this XMLData object.
1505 Invalid XMLData kind was specified.
1506 Invalid address for the return parameter was specified.
1507 Element cannot have Children
1512 Cyclic insertion - new data element is already part of document
1514 Invalid XMLData kind was specified for this position.
1900 Document must not be modified

14.3.2.32.19 InsertChildAfter

Method: InsertChildBefore (Node as XMLData, NewData as XMLData)

Description
Inserts a new XML node (supplied with the second parameter) after the specified node (first parameter).

Errors

1500 Invalid object.
1506 Invalid input xml
1507 No children allowed
1510 Invalid parameter.
1512 Child is already added
1514 Invalid kind at position

14.3.2.32.20 InsertChildBefore

Method: InsertChildBefore (Node as XMLData, NewData as XMLData)

Description
Inserts a new XML node (supplied with the second parameter) before the specified node (first parameter).

Errors

1500 Invalid object.
1506 Invalid input xml
1507 No children allowed
1510 Invalid parameter.
1512 Child is already added
1514 Invalid kind at position

14.3.2.32.21 IsSameNode

Declaration: IsSameNode (*pNodeToCompare* as XMLData) as Boolean

© 2016-2022 Altova GmbH

Description

Returns true if pNodeToCompare references the same node as the object itself.

Errors
1500 The XMLData object is no longer valid.
1506 Invalid address for the return parameter was specified.

14.3.2.32.22 Kind

Declaration: Kind as SPYXMLDataKind

Description
Kind of this XMLData object. This property is read-only.

Errors
1500 The XMLData object is no longer valid.
1510 Invalid address for the return parameter was specified.

14.3.2.32.23 MayHaveChildren

Declaration: MayHaveChildren as Boolean

Description
Indicates whether it is allowed to add children to this XMLData object.
This property is read-only.

Errors
1500 The XMLData object is no longer valid.
1510 Invalid address for the return parameter was specified.

14.3.2.32.24 Name

Declaration: Name as String

Description
Used to modify and to get the name of the XMLData object.

Errors
1500 The XMLData object is no longer valid.
1510 Invalid address for the return parameter was specified.

14.3.2.32.25 Parent

Declaration: Parent as XMLData

Return value

Parent as XMLData object. Nothing (or NULL) if there is no parent element.

Description
Parent of this element. This property is read-only.

Errors
1500 The XMLData object is no longer valid.
1510 Invalid address for the return parameter was specified.

14.3.2.32.26 SetTextValueXMLEncoded

Method: SetTextValueXMLEncoded (*strVal* as String)

Description
Sets the encoded text value of the XML.

Errors
1500 Invalid object.
1513 Modification not allowed.

14.3.2.32.27 TextValue

Declaration: TextValue as String

Description
Used to modify and to get the text value of this XMLData object.

Errors
1500 The XMLData object is no longer valid.
1510 Invalid address for the return parameter was specified.

14.3.3 Enumerations

This is a list of all enumerations used by the Authentic Desktop API. If your scripting environment does not support enumerations use the number-values instead.

14.3.3.1 ENUMApplicationStatus

Description
Enumeration to specify the current Application status.

Possible values:

© 2016-2022 Altova GmbH

eApplicationRunning	= 0
eApplicationAfterLicenseCheck	= 1
eApplicationBeforeLicenseCheck	= 2
eApplicationConcurrentLicenseCheckFailed	= 3
eApplicationProcessingCommandLine	= 4

14.3.3.2 SPYAttributeTypeDefinition

Description
Attribute type definition that can be selected for generation of Sample XML.
This type is used with the method GenerateDTDOrSchema and GenerateDTDOrSchemaEx.

Possible values:

spyMergedGlobal	= 0
spyDistinctGlobal	= 1
spyLocal	= 2

14.3.3.3 SPYAuthenticActions

Description
Actions that can be performed on AuthenticRange objects.

Possible values:

spyAuthenticInsertAt	= 0
spyAuthenticApply	= 1
spyAuthenticClearSurr	= 2
spyAuthenticAppend	= 3
spyAuthenticInsertBefore	= 4
spyAuthenticRemove	= 5

14.3.3.4 SPYAuthenticDocumentPosition

Description
Relative and absolute positions used for navigating with AuthenticRange objects.

Possible values:

spyAuthenticDocumentBegin	= 0
spyAuthenticDocumentEnd	= 1
spyAuthenticRangeBegin	= 2
spyAuthenticRangeEnd	= 3

14.3.3.5 SPYAuthenticElementActions

Description
Actions that can be used with the obsolete object GetAllowedElements (superseded by AuthenticRange.CanPerformActionWith).

Possible values:

k_ActionInsertAt	= 0
k_ActionApply	= 1
k_ActionClearSurr	= 2
k_ActionAppend	= 3
k_ActionInsertBefore	= 4
k_ActionRemove	= 5

14.3.3.6 SPYAuthenticElementKind

Description
Enumeration of the different kinds of elements used for navigation and selection within the AuthenticRange and AuthenticView objects.

Possible values:

spyAuthenticChar	= 0
spyAuthenticWord	= 1
spyAuthenticLine	= 3
spyAuthenticParagraph	= 4
spyAuthenticTag	= 6
spyAuthenticDocument	= 8
spyAuthenticTable	= 9
spyAuthenticTableRow	= 10
spyAuthenticTableColumn	= 11

14.3.3.7 SPYAuthenticMarkupVisibility

Description
Enumeration values to customize the visibility of markup with MarkupVisibility.

Possible values:

spyAuthenticMarkupHidden	= 0
spyAuthenticMarkupSmall	= 1
spyAuthenticMarkupLarge	= 2
spyAuthenticMarkupMixed	= 3

© 2016-2022 Altova GmbH

14.3.3.8 SPYAuthenticToolbarButtonState

Description
Authentic toolbar button states are given by the following enumeration:

Possible values:

authenticToolbarButtonDefault	= 0
authenticToolbarButtonEnabled	= 1
authenticToolbarButtonDisabled	= 2

14.3.3.9 SPYDatabaseKind

Description
Values to select different kinds of databases for import. See DatabaseConnection.DatabaseKind for its use.

Possible values:

spyDB_Access	= 0
spyDB_SQLServer	= 1
spyDB_Oracle	= 2
spyDB_Sybase	= 3
spyDB_MySQL	= 4
spyDB_DB2	= 5
spyDB_Other	= 6
spyDB_Unspecified	= 7
spyDB_PostgreSQL	= 8
spyDB_iSeries	= 9

14.3.3.10 SPYDialogAction

Description
Values to simulate different interactions on dialogs. See Dialogs for all dialogs available.

Possible values:

spyDialogOK	= 0	// simulate click on OK button
spyDialogCancel	= 1	// simulate click on Cancel button
spyDialogUserInput	= 2	// show dialog and allow user interaction

14.3.3.11 SPYDOMType

Description
Enumeration values to parameterize generation of C++ code from schema definitions.

Possible values:

spyDOMType_msxml4	= 0	Obsolete
spyDOMType_xerces	= 1	
spyDOMType_xerces3	= 2	
spyDOMType_msxml6	= 3	

`spyDOMType_xerces` indicates Xerces 2.x usage; `spyDOMType_xerces3` indicates Xerces 3.x usage.

14.3.3.12 SPYDTDSchemaFormat

Description
Enumeration to identify the different schema formats.

Possible values:

spyDTD	= 0
spyW3C	= 1

14.3.3.13 SPYEncodingByteOrder

Description
Enumeration values to specify encoding byte ordering for text import and export.

Possible values:

spyNONE	= 0
spyLITTLE_ENDIAN	= 1
spyBIG_ENDIAN	= 2

14.3.3.14 SPYExportNamespace

Description
Enumeration type to configure handling of namespace identifiers during export.

Possible values:

spyNoNamespace	= 0
spyReplaceColonWithUnderscore	= 1

14.3.3.15 SPYFindInFilesSearchLocation

Description
The different locations where a search can be performed. This type is used with the FindInFilesDlg dialog.

Possible values:

spyFindInFiles_Documents	= 0
spyFindInFiles_Project	= 1
spyFindInFiles_Folder	= 2

© 2016-2022 Altova GmbH

14.3.3.16 SPYFrequentElements

Description
Enumeration value to parameterize schema generation.

Possible values:

spyGlobalElements	= 0
spyGlobalComplexType	= 1

14.3.3.17 SPYImageKind

Description
Enumeration values to parameterize image type of the generated documentation. These values are used in SchemaDocumentationDialog.DiagramFormat.

Possible values:

spyImageType_PNG	= 0
spyImageType_EMF	= 1

14.3.3.18 SPYImportColumnsType

Description
Enumeration to specify different Import columns types.

Possible values:

spyImportColumns_Element	= 0
spyImportColumns_Attribute	= 1

14.3.3.19 SPYKeyEvent

Description
Enumeration type to identify the different key events. These events correspond with the equally named windows messages.

Possible values:

spyKeyDown	= 0
spyKeyUp	= 1
spyKeyPressed	= 2

14.3.3.20 SPYKeyStatus

Description
Enumeration type to identify the key status.

Possible values:

spyLeftShiftKeyMask	= 1
spyRightShiftKeyMask	= 2
spyLeftCtrlKeyMask	= 4
spyRightCtrlKeyMask	= 8
spyLeftAltKeyMask	= 16
spyRightAltKeyMask	= 32

14.3.3.21 SPYLibType

Description
Enumeration values to parameterize generation of C++ code from schema definitions.

Possible values:

spyLibType_static	= 0
spyLibType_dll	= 1

14.3.3.22 SPYLoading

Description
Enumeration values to define loading behaviour of URL files.

Possible values:

spyUseCacheProxy	= 0
spyReload	= 1

14.3.3.23 SPYMouseEvent

Description
Enumeration type that defines the mouse status during a mouse event. Use the enumeration values as bitmasks rather then directly comparing with them.

Examples

```
' to check for ctrl-leftbutton-down in VB
If (i_eMouseEvent = (XMLSpyLib.spyLeftButtonDownMask Or XMLSpyLib.spyCtrlKeyDownMask)) Then
        ' react on ctrl-leftbutton-down
End If

' to check for double-click with any button in VBScript
If (((i_eMouseEvent And spyDoubleClickMask) <> 0) Then
        ' react on double-click
End If
```

Possible values:

spyNoButtonMask	= 0

© 2016-2022 Altova GmbH

```
spyMouseMoveMask          = 1
spyLeftButtonMask         = 2
spyMiddleButtonMask       = 4
spyRightButtonMask        = 8
spyButtonUpMask           = 16
spyButtonDownMask         = 32
spyDoubleClickMask        = 64
spyShiftKeyDownMask       = 128
spyCtrlKeyDownMask        = 256
spyLeftButtonDownMask     = 34   // spyLeftButtonMask | spyButtonDownMask
spyMiddleButtonDownMask   = 36   // spyMiddleButtonMask | spyButtonDownMask
spyRightButtonDownMask    = 40   // spyRightButtonMask | spyButtonDownMask
spyLeftButtonUpMask       = 18   // spyLeftButtonMask | spyButtonUpMask
spyMiddleButtonUpMask     = 20   // spyMiddleButtonMask | spyButtonUpMask
spyRightButtonUpMask      = 24   // spyRightButtonMask | spyButtonUpMask
spyLeftDoubleClickMask    = 66   // spyRightButtonMask | spyButtonUpMask
spyMiddleDoubleClickMask  = 68   // spyMiddleButtonMask | spyDoubleClickMask
spyRightDoubleClickMask   = 72   // spyRightButtonMask | spyDoubleClickMask
```

14.3.3.24 SPYNumberDateTimeFormat

Description
Enumeration value to configure database connections.

Possible values:

```
spySystemLocale      = 0
spySchemaCompatible  = 1
```

14.3.3.25 SPYProgrammingLanguage

Description
Enumeration values to select the programming language for code generation from schema definitions.

Only available/enabled in the Enterprise edition. An error is returned, if accessed by any other version.

Possible values:

```
spyUndefinedLanguage  = -1
spyJava               = 0
spyCpp                = 1
spyCSharp             = 2
```

14.3.3.26 SPYProjectItemTypes

Description
Enumeration values to identify the different elements in project item lists. See SpyProjectItem.ItemType.

Possible values:

spyUnknownItem = 0
spyFileItem = 1
spyFolderItem = 2
spyURLItem = 3

14.3.3.27 SPYProjectType

Description
Enumeration values to parameterize generation of C# from schema definitions.

Possible values:

spyVisualStudioProject	= 0	Obsolete
spyVisualStudio2003Project	= 1	Obsolete
spyBorlandProject	= 2	Obsolete
spyMonoMakefile	= 3	Obsolete
spyVisualStudio2005Project	= 4	Obsolete
spyVisualStudio2008Project	= 5	Obsolete
spyVisualStudio2010Project	= 6	For C++ code also
spyVisualStudio2013Project	= 7	For C++ code also
spyVisualStudio2015Project	= 8	For C++ code also
spyVisualStudio2017Project	= 9	For C++ code also
spyVisualStudio2019Project	=10	For C++ code also

14.3.3.28 SpySampleXMLGenerationChoiceMode

Description
This enumeration is used in `GenerateSampleXMLDlg.ChoiceMode`:

spySampleXMLGen_FirstBranch = 0
spySampleXMLGen_AllBranches = 1
spySampleXMLGen_ShortestBranch = 2

14.3.3.29 SPYSampleXMLGenerationOptimization (Obsolete)

This enumeration is OBSOLETE since v2014.

Description
Specify the elements that will be generated in the Sample XML.
This enumeration is used in GenerateSampleXMLDlg.

Possible values:

spySampleXMLGen_Optimized	= 0

© 2016-2022 Altova GmbH

spySampleXMLGen_NonMandatoryElements = 1
spySampleXMLGen_Everything = 2

14.3.3.30 SpySampleXMLGenerationSampleValueHints

Description
This enumeration is used in `GenerateSampleXMLDlg.SampleValueHints`

spySampleXMLGen_FirstFit = 0
spySampleXMLGen_RandomFit = 1
spySampleXMLGen_CycleThrough = 2

14.3.3.31 SPYSampleXMLGenerationSchemaOrDTDAssignment

Description
Specifies what kind of reference to the schema/DTD should be added to the generated Sample XML.
This enumeration is used in GenerateSampleXMLDlg.

Possible values:
spySampleXMLGen_AssignRelatively = 0
spySampleXMLGen_AssignAbsolutely = 1
spySampleXMLGen_DoNotAssign = 2

14.3.3.32 SPYSchemaDefKind

Description
Enumeration type to select schema diagram types.

Possible values:
spyKindElement = 0
spyKindComplexType = 1
spyKindSimpleType = 2
spyKindGroup = 3
spyKindModel = 4
spyKindAny = 5
spyKindAttr = 6
spyKindAttrGroup = 7
spyKindAttrAny = 8

spyKindIdentityUnique = 9
spyKindIdentityKey = 10
spyKindIdentityKeyRef = 11
spyKindIdentitySelector = 12
spyKindIdentityField = 13
spyKindNotation = 14
spyKindInclude = 15
spyKindImport = 16
spyKindRedefine = 17
spyKindFacet = 18
spyKindSchema = 19
spyKindCount = 20

14.3.3.33 SPYSchemaDocumentationFormat

Description
Enumeration values to parameterize generation of schema documentation. These values are used in SchemaDocumentationDialog.OutputFormat.

Possible values:
spySchemaDoc_HTML = 0
spySchemaDoc_MSWord = 1
spySchemaDoc_RTF = 2
spySchemaDoc_PDF = 3

14.3.3.34 SPYSchemaExtensionType

Description
Enumeration to specify different Schema Extension types.

Possible values:

spySchemaExtension_None = 0
spySchemaExtension_SQL_XML = 1
spySchemaExtension_MS_SQL_Server = 2
spySchemaExtension_Oracle = 3

14.3.3.35 SPYSchemaFormat

Description
Enumeration to specify different Schema Format types.

Possible values:

spySchemaFormat_Hierarchical = 0
spySchemaFormat_Flat = 1

© 2016-2022 Altova GmbH

14.3.3.36 SPYTextDelimiters

Description
Enumeration values to specify text delimiters for text export.

Possible values:

spyTabulator	= 0
spySemicolon	= 1
spyComma	= 2
spySpace	= 3

14.3.3.37 SPYTextEnclosing

Description
Enumeration value to specify text enclosing characters for text import and export.

Possible values:

spyNoEnclosing	= 0
spySingleQuote	= 1
spyDoubleQuote	= 2

14.3.3.38 SPYTypeDetection

Description
Enumeration to select how type detection works during GenerateDTDOrSchema and GenerateDTDOrSchemaEx.

Possible values:

spyBestPossible	= 0
spyNumbersOnly	= 1
spyNoDetection	= 2

14.3.3.39 SPYURLTypes

Description
Enumeration to specify different URL types.

Possible values:

spyURLTypeAuto	= -1
spyURLTypeXML	= 0
spyURLTypeDTD	= 1

14.3.3.40 SPYValidateXSDVersion

Description
Enumeration values that select what XSD version to use. The XSD version that is selected depends on both (i) the presence/absence—and, if present, the value—of the `/xs:schema/@vc:minVersion` attribute of the XSD document, and (ii) the value of this enumeration.

`spyValidateXSDVersion_1_0` selects `XSD 1.0` if `vc:minVersion` is absent, or is present with any value.
`spyValidateXSDVersion_1_1` selects `XSD 1.1` if `vc:minVersion` is absent, or is present with any value.
`spyValidateXSDVersion_AutoDetect` selects `XSD 1.1` if `vc:minVersion=1.1`. If the `vc:minVersion` attribute is absent, or is present with a value other than `1.1`, then `XSD 1.0` is selected.

Possible values

`spyValidateXSDVersion_AutoDetect`	`= 0`
`spyValidateXSDVersion_1_1`	`= 1`
`spyValidateXSDVersion_1_0`	`= 2`

14.3.3.41 SPYValidateErrorFormat

Description
Enumeration values that select the format of the error message.

Possible values

`spyValidateErrorFormat_Text`	`= 0`
`spyValidateErrorFormat_ShortXML`	`= 1`
`spyValidateErrorFormat_LongXML`	`= 2`

14.3.3.42 SPYViewModes

Description
Enumeration values that define the different view modes for XML documents. The mode *spyViewAuthentic(4)* identifies the mode that was intermediately called DocEdit mode and is now called Authentic mode. The mode *spyViewJsonSchema* identifies a mode which is mapped to the Schema Design View on the GUI but is distinguished internally.

Possible values:

spyViewGrid	= 0
spyViewText	= 1
spyViewBrowser	= 2
spyViewSchema	= 3

© 2016-2022 Altova GmbH

```
spyViewContent          = 4     // obsolete
spyViewAuthentic        = 4
spyViewWSDL             = 5
spyViewZIP              = 6
spyViewEditionInfo      = 7
spyViewXBRL             = 8
spyViewJsonSchema       = 9
```

14.3.3.43 SPYVirtualKeyMask

Description
Enumeration type for the most frequently used key masks that identify the status of the virtual keys. Use these values as bitmasks rather then directly comparing with them. When necessary, you can create further masks by using the 'logical or' operator.

Examples

```
'  VBScript sample: check if ctrl-key is pressed
If ((i_nVirtualKeyStatus And spyCtrlKeyMask) <> 0)) Then
        ' ctrl-key is pressed
End If

'  VBScript sample: check if ONLY ctrl-key is pressed
If (i_nVirtualKeyStatus == spyCtrlKeyMask) Then
        ' exactly ctrl-key is pressed
End If

// JScript sample: check if any of the right virtual keys is pressed
if ((i_nVirtualKeyStatus & (spyRightShiftKeyMask | spyRightCtrlKeyMask | spyRightAltKeyMask)) != 0)
{
        ; ' right virtual key is pressed
}
```

Possible values:

```
spyNoVirtualKeyMask     = 0
spyLeftShiftKeyMask     = 1
spyRightShiftKeyMask    = 2
spyLeftCtrlKeyMask      = 4
spyRightCtrlKeyMask     = 8
spyLeftAltKeyMask       = 16
spyRightAltKeyMask      = 32
spyShiftKeyMask         = 3     // spyLeftShiftKeyMask | spyRightShiftKeyMask
spyCtrlKeyMask          = 12    // spyLeftCtrlKeyMask | spyRightCtrlKeyMask
spyAltKeyMask           = 48    // spyLeftAltKeyMask | spyRightAltKeyMask
```

14.3.3.44 SPYXMLDataKind

Description
The different types of XMLData elements available for XML documents.

Possible values:

spyXMLDataXMLDocStruct	= 0
spyXMLDataXMLEntityDocStruct	= 1
spyXMLDataDTDDocStruct	= 2
spyXMLDataXML	= 3
spyXMLDataElement	= 4
spyXMLDataAttr	= 5
spyXMLDataText	= 6
spyXMLDataCData	= 7
spyXMLDataComment	= 8
spyXMLDataPI	= 9
spyXMLDataDefDoctype	= 10
spyXMLDataDefExternalID	= 11
spyXMLDataDefElement	= 12
spyXMLDataDefAttlist	= 13
spyXMLDataDefEntity	= 14
spyXMLDataDefNotation	= 15
spyXMLDataKindsCount	= 16

© 2016-2022 Altova GmbH

14.4 ActiveX Integration

The Authentic Desktop user interface and the functionality described in this section can be integrated into custom applications that can consume ActiveX controls. ActiveX technology enables a wide variety of languages to be used for integration, such as C++, C#, VB.NET, HTML. (Note that ActiveX components integrated in HTML must be run with Microsoft Internet Explorer versions and platforms that support ActiveX). All components are full OLE Controls. Integration into Java is provided through wrapper classes.

To integrate the ActiveX controls into your custom code, the Authentic Desktop Integration Package must be installed (see https://www.altova.com/components/download). Ensure that you install Authentic Desktop first, and then the Authentic Desktop Integration Package. Other prerequisites apply, depending on language and platform (see Prerequisites).

You can flexibly choose between two different levels of integration: application level and document level.

Integration at application level means embedding the complete interface of Authentic Desktop (including its menus, toolbars, panes, etc) as an ActiveX control into your custom application. For example, in the most simple scenario, your custom application could consist of only one form that embeds the Authentic Desktop graphical user interface. This approach is easier to implement than integration at document level but may not be suitable if you need flexibility to configure the Authentic Desktop graphical user interface according to your custom requirements.

Integration at document level means embedding Authentic Desktop into your own application piece-by-piece. This includes implementing not only the main Authentic Desktop control but also the main document editor window, and, optionally, any additional windows. This approach provides greater flexibility to configure the GUI, but requires advanced interaction with ActiveX controls in your language of choice.

The sections Integration at the Application Level and Integration at Document Level describe the key steps at these respective levels. The ActiveX Integration Examples section provides examples in C#, HTML, and Java. Looking through these examples will help you to make the right decisions quickly. The Object Reference section describes all COM objects that can be used for integration, together with their properties and methods.

For information about using Authentic Desktop as a Visual Studio plug-in, see Authentic Desktop in Visual Studio.

14.4.1 Prerequisites

To integrate the Authentic Desktop ActiveX control into a custom application, the following must be installed on your computer:

- Authentic Desktop
- The Authentic Desktop Integration Package, available for download at https://www.altova.com/components/download

To integrate the 64-bit ActiveX control, install the 64-bit versions of Authentic Desktop and Authentic Desktop Integration Package. For applications developed under Microsoft .NET platform with Visual Studio, both the 32-bit and 64-bit versions of Authentic Desktop and Authentic Desktop Integration Package must be installed, as explained below.

Microsoft .NET (C#, VB.NET) with Visual Studio

To integrate the Authentic Desktop ActiveX control into a 32-bit application developed under Microsoft .NET, the following must be installed on your computer:

- Microsoft .NET Framework 4.0 or later
- Visual Studio 2012/2013/2015/2017/2019/2022
- Authentic Desktop 32-bit and Authentic Desktop Integration Package 32-bit
- The ActiveX controls must be added to the Visual Studio toolbox (see Adding the ActiveX Controls to the Toolbox).

If you want to integrate the 64-bit ActiveX control, the following prerequisites apply in addition to the ones above:

- Authentic Desktop 32-bit and Authentic Desktop Integration Package 32-bit must still be installed (this is required to provide the 32-bit ActiveX control to the Visual Studio designer, since Visual Studio runs on 32-bit)
- Authentic Desktop 64-bit and Authentic Desktop Integration Package 64-bit must be installed (provides the actual 64-bit ActiveX control to your custom application at runtime)
- In Visual Studio, create a 64-bit build configuration and build your application using this configuration. For an example, see Running the Sample C# Solution.

Java

To integrate the Authentic Desktop ActiveX control into Java application using the Eclipse development environment, the following must be installed on your computer:

- Java Runtime Environment (JRE) or Java Development Kit (JDK) 7 or later
- Eclipse
- Authentic Desktop and Authentic Desktop Integration Package

Note: To run the 64-bit version of the Authentic Desktop ActiveX control, use a 64-bit version of Eclipse, as well as the 64-bit version of Authentic Desktop and the Authentic Desktop Integration Package.

Authentic Desktop integration and deployment on client computers

If you create a .NET application and intend to distribute it to other clients, you will need to install the following on the client computer(s):

- Authentic Desktop
- The Authentic Desktop Integration Package
- The custom integration code or application.

14.4.2 Adding the ActiveX Controls to the Toolbox

To use the Authentic Desktop ActiveX controls in an application developed with Visual Studio, the controls must first be added to the Visual Studio Toolbox, as follows:

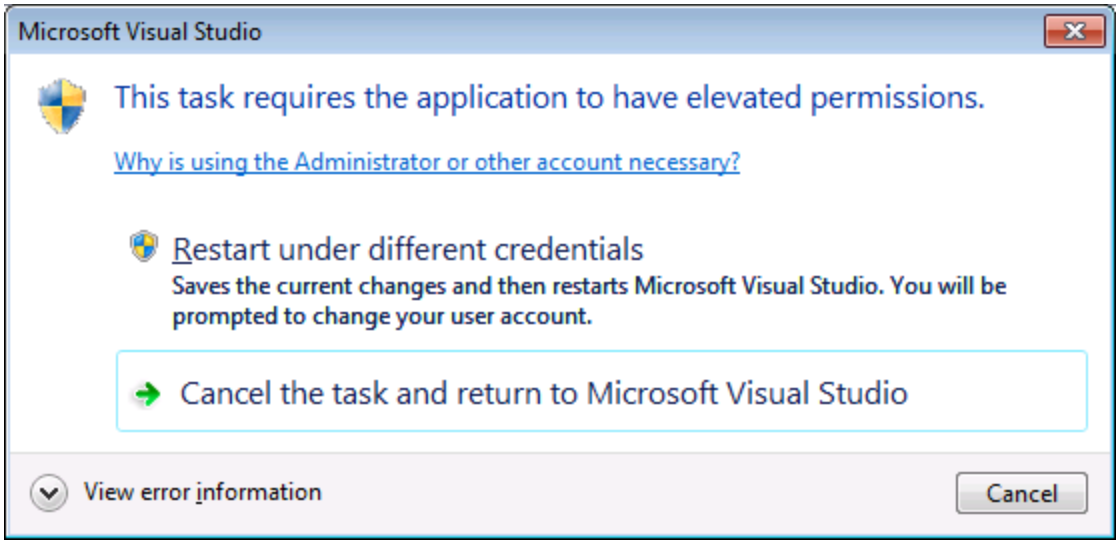
1. On the **Tools** menu of Visual Studio, click **Choose Toolbox Items**.

© 2016-2022 Altova GmbH

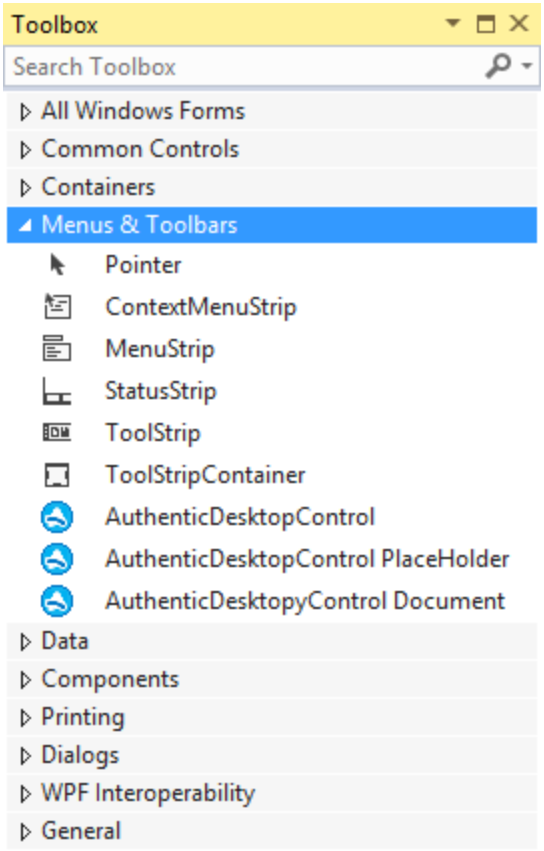
2. On the **COM Components** tab, select the check boxes next to the Authentic DesktopControl, Authentic DesktopControl Document, and Authentic DesktopControl Placeholder.

In case the controls above are not available, follow the steps below:

1. On the **COM Components** tab, click **Browse**, and select the **AuthenticControl.ocx** file from the Authentic Desktop installation folder. Remember that the Authentic Desktop Integration Package must be installed; otherwise, this file is not available, see Prerequisites.
2. If prompted to restart Visual Studio with elevated permissions, click **Restart under different credentials**.



If the steps above were successful, the Authentic Desktop ActiveX controls become available in the Visual Studio Toolbox.



Note: For an application-level integration, only the **AuthenticDesktopControl** ActiveX control is used (see Integration at Application Level). The **AuthenticDesktopControl Document** and **AuthenticDesktopControl Placeholder** controls are used for document-level integration (see Integration at Document Level).

14.4.3 Integration at Application Level

Integration at application level allows you to embed the complete interface of Authentic Desktop into a window of your application. With this type of integration, you get the whole user interface of Authentic Desktop, including all menus, toolbars, the status bar, document windows, and helper windows. Customization of the application's user interface is restricted to what Authentic Desktop provides. This includes rearrangement and resizing of helper windows and customization of menus and toolbars.

The only ActiveX control you need to integrate is `AuthenticDesktopControl`. Do not instantiate or access `AuthenticDesktopControlDocument` or `AuthenticDesktopControlPlaceHolder` ActiveX controls when integrating at application-level.

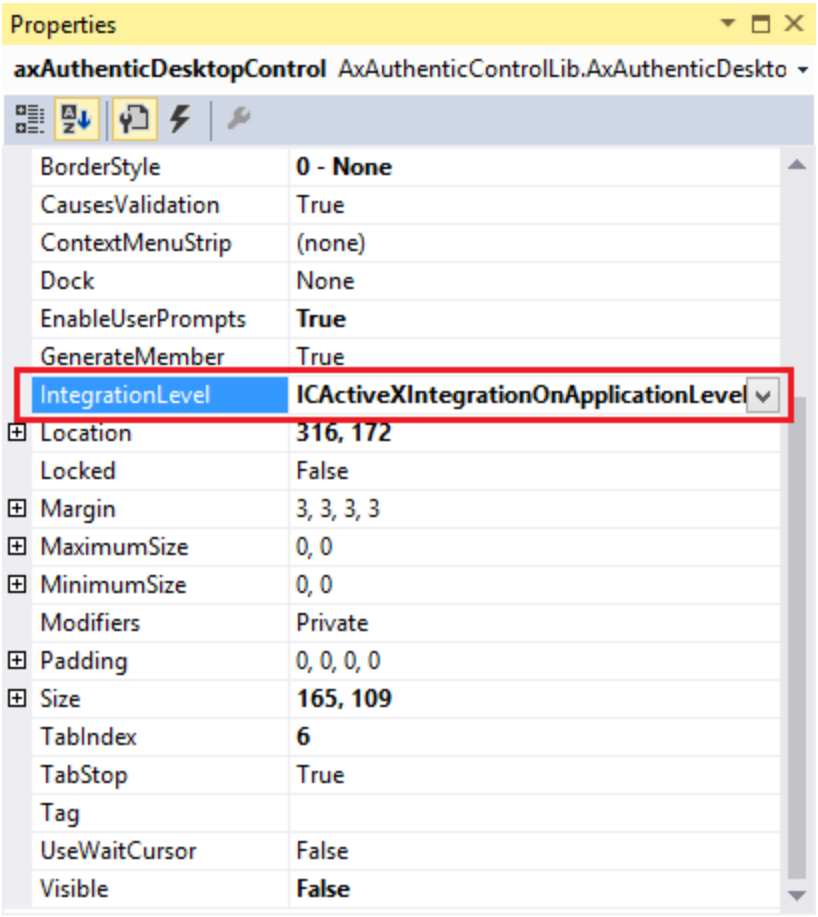
If you have any initialization to do or if you want to automate some behaviour of Authentic Desktop, use the properties, methods, and events described for `AuthenticDesktopControl`. Consider using `AuthenticDesktopControl.Application` for more complex access to Authentic Desktop functionality.

© 2016-2022 Altova GmbH

For an example that shows how the Authentic Desktop application can be embedded in an HTML page, see HTML Integration at Application Level.

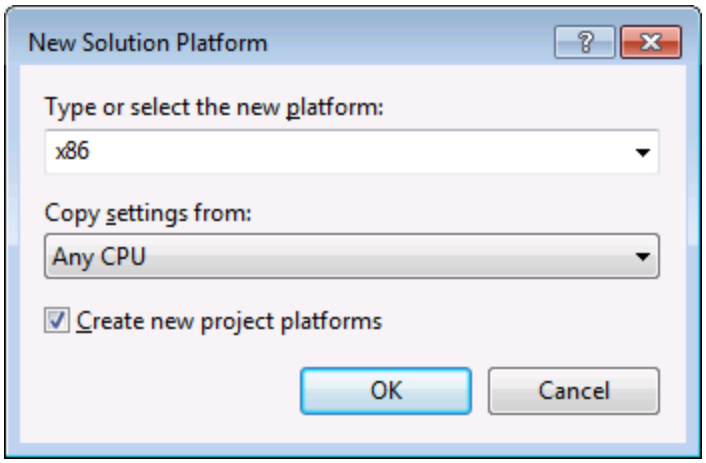
In C# or VB.NET with Visual Studio, the steps to create a basic, one-form application which integrates the Authentic Desktop ActiveX controls at application level are as follows:

1. Check that all prerequisites are met (see Prerequisites).
2. Create a new Visual Studio Windows Forms project with a new empty form.
3. If you have not done that already, add the ActiveX controls to the toolbox (see Adding the ActiveX Controls to the Toolbox).
4. Drag the **AuthenticDesktopControl** from the toolbox onto your new form.
5. Select the **AuthenticDesktopControl** on the form, and, in the Properties window, set the **IntegrationLevel** property to **ICActiveXIntegrationOnApplicationLevel**.



6. Create a build platform configuration that matches the platform under which you want to build (x86, x64). Here is how you can create the build configuration:

   a. Right-click the solution in Visual Studio, and select **Configuration Manager**.
   b. Under **Active solution platform**, select **New...** and then select the x86 or x64 configuration (in this example, **x86**).



You are now ready to build and run the solution in Visual Studio. Remember to build using the configuration that matches your target platform (x86, x64).

14.4.4 Integration at Document Level

Compared to integration at application level, integration at document level is a more complex, yet more flexible way to embed Authentic Desktop functionality into your application by means of ActiveX controls. With this approach, your code can access selectively the following parts of the Authentic Desktop user interface:

- Document editing window
- Project window
- Info window
- Messages window
- Entry helper windows (Elements, Attributes, Entities)
- Output window

As mentioned in Integration at Application Level, for an ActiveX integration at application level, only one control is required, namely the **AuthenticDesktopControl**. However, for an ActiveX integration at document level, Authentic Desktop functionality is provided by the following ActiveX controls:

- AuthenticDesktopControl
- AuthenticDesktopControl Document
- AuthenticDesktopControl Placeholder

These controls are supplied by the **AuthenticControl.ocx** file available in the application installation folder of Authentic Desktop. When you develop the ActiveX integration with Visual Studio, you will need to add these controls to the Visual Studio toolbox (see Adding the ActiveX Controls to the Toolbox).

The basic steps to integrate the ActiveX controls at document level into your application are as follows:

1. First, instantiate **AuthenticDesktopControl** in your application. Instantiating this control is mandatory; it enables support for the **AuthenticDesktopControl Document** and **AuthenticDesktopControl Placeholder** controls mentioned above. It is important to set the `IntegrationLevel` property to **ICActiveXIntegrationOnDocumentLevel** (or "1"). To hide the control from the user, set its **Visible** property to **False**.

© 2016-2022 Altova GmbH

Note: When integrating at document level, do not use the **Open** method of the **AuthenticDesktopControl**; this might lead to unexpected results. Use the corresponding open methods of **AuthenticDesktopControl Document** and **AuthenticDesktopControl PlaceHolder** instead.

2. Create at least one instance of **AuthenticDesktopControl Document** in your application. This control supplies the document editing window of Authentic Desktop to your application and can be instantiated multiple times if necessary.

   Use the method **Open** to load any existing file. To access document-related functionality, use the **Path** and **Save** or methods and properties accessible via the property **Document**.

Note: The control does not support a read-only mode. The value of the property **ReadOnly** is ignored.

3. Optionally, add to your application the **AuthenticDesktopControl Placeholder** control for each additional window (other than the document window) that must be available to your application.

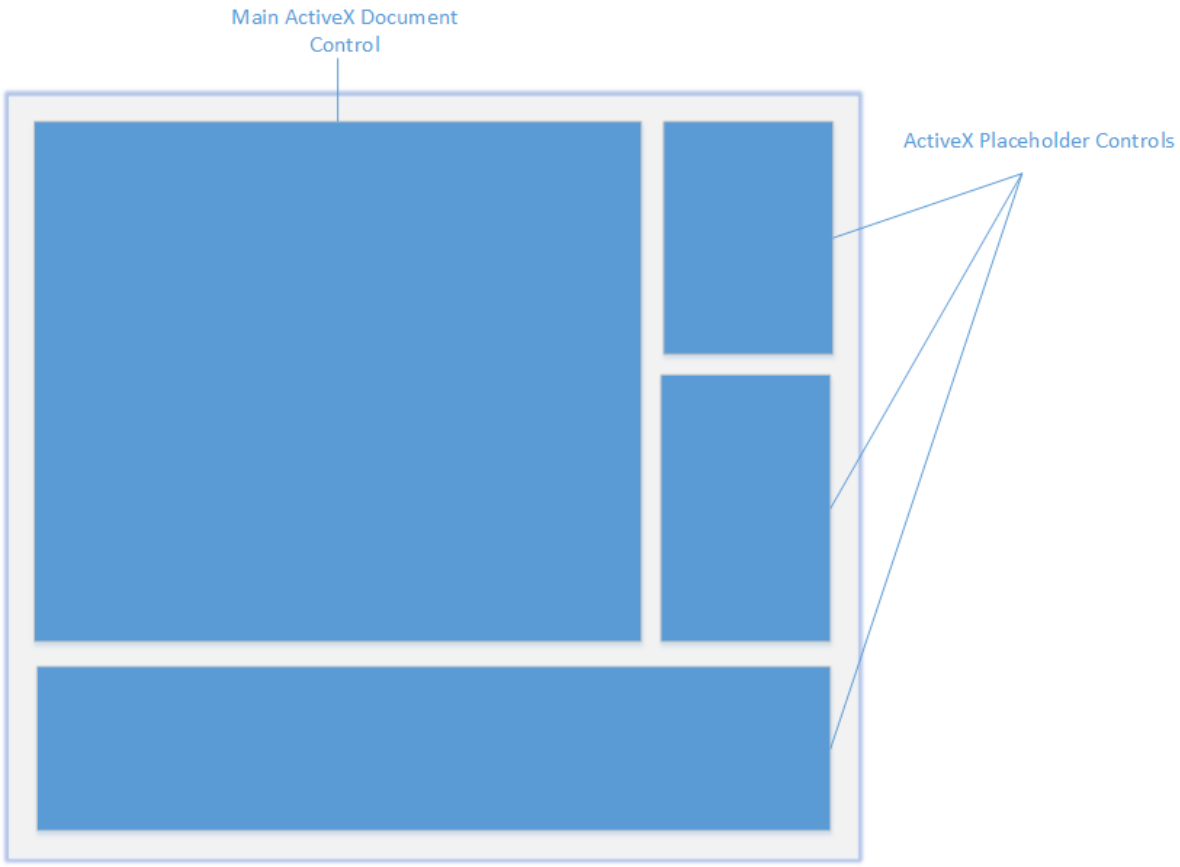
   Instances of **AuthenticDesktopControl PlaceHolder** allow you to selectively embed additional windows of Authentic Desktop into your application. The window kind (for example, Project window) is defined by the property **PlaceholderWindowID**. Therefore, to set the window kind, set the property **PlaceholderWindowID**. For valid window identifiers, see AuthenticDesktopControlPlaceholderWindow.

Note: Use only one **AuthenticDesktopControl PlaceHolder** for each window identifier.

   For placeholder controls that select the Authentic Desktop project window, additional methods are available. Use **OpenProject** to load a Authentic Desktop project. Use the property Project and the methods and properties from the Authentic Desktop automation interface to perform any other project related operations.

For example, in C# or VB.NET with Visual Studio, the steps to create a basic, one-form application which integrates the Authentic Desktop ActiveX controls at document level could be similar to those listed below. Note that your application may be more complex if necessary; however, the instructions below are important to understand the minimum requirements for an ActiveX integration at document level.

1. Create a new Visual Studio Windows Forms project with a new empty form.
2. If you have not done that already, add the ActiveX controls to the toolbox (see Adding the ActiveX Controls to the Toolbox).
3. Drag the **AuthenticDesktopControl** from the toolbox onto your new form.
4. Set the **IntegrationLevel** property of the **AuthenticDesktopControl** to **ICActiveXIntegrationOnDocumentLevel**, and the **Visible** property to **False**. You can do this either from code or from the **Properties** window.
5. Drag the **AuthenticDesktopControl Document** from the toolbox onto the form. This control provides the main document window of Authentic Desktop to your application, so you may need to resize it to a reasonable size for a document.
6. Optionally, add one or more **AuthenticDesktopControl Placeholder** controls to the form (one for each additional window type that your application needs, for example, the **Project** window). You will typically want to place such additional placeholder controls either below or to the right or left of the main document control, for example:



7. Set the **PlaceholderWindowID** property of each **AuthenticDesktopControl Placeholder** control to a valid window identifier. For the list of valid values, see AuthenticDesktopControlPlaceholderWindow.
8. Add commands to your application (at minimum, you will need to open, save and close documents), as shown below.

Querying Authentic Desktop Commands

When you integrate at document level, no Authentic Desktop menu or toolbar is available to your application. Instead, you can retrieve the required commands, view their status, and execute them programmatically, as follows:

- To retrieve all available commands, use the `CommandsList` property of the **AuthenticDesktopControl**.
- To retrieve commands organized according to their menu structure, use the `MainMenu` property.
- To retrieve commands organized by the toolbar in which they appear, use the `Toolbars` property.
- To send commands to Authentic Desktop, use the `Exec` method.
- To query if a command is currently enabled or disabled, use the `QueryStatus` method.

This enables you to flexibly integrate Authentic Desktop commands into your application's menus and toolbars.

Your installation of Authentic Desktop also provides you with command label images used within Authentic Desktop. See the folder **<ApplicationFolder>\Examples\ActiveX\Images** of your Authentic Desktop installation for icons in GIF format. The file names correspond to the command names as they are listed in the Command Reference section.

© 2016-2022 Altova GmbH

General considerations

To automate the behaviour of Authentic Desktop, use the properties, methods, and events described for the **AuthenticDesktopControl**, **AuthenticDesktopControl Document**, and **AuthenticDesktopControl Placeholder**.

For more complex access to Authentic Desktop functionality, consider using the following properties:

- `AuthenticDesktopControl.Application`
- `AuthenticDesktopControlDocument.Document`
- `AuthenticDesktopControlPlaceHolder.Project`

These properties give you access to the Authentic Desktop automation interface (AuthenticDesktopAPI)

Note: To open a document, always use `AuthenticDesktopControlDocument.Open` or `AuthenticDesktopControlDocument.New` on the appropriate document control. To open a project, always use `AuthenticDesktopControlPlaceHolder.OpenProject` on a placeholder control embedding a Authentic Desktop project window.

For examples that show how to instantiate and access the necessary controls in different programming environments, see ActiveX Integration Examples.

14.4.5 ActiveX Integration Examples

This section contains examples of Authentic Desktop document-level integration using different container environments and programming languages. Source code for all examples is available in the folder `<ApplicationFolder>\Examples\ActiveX` of your Authentic Desktop installation.

14.4.5.1 C#

A basic ActiveX integration example solution for C# and Visual Studio is available in the folder `<ApplicationFolder>\Examples\ActiveX\C#`. Before you compile the source code and run the sample, make sure that all prerequisites are met (see Running the Sample C# Solution).

14.4.5.1.1 Running the Sample C# Solution

The sample Visual Studio solution available in the folder **<ApplicationFolder>\Examples\ActiveX\C#** illustrates how to consume the Authentic Desktop ActiveX controls. Before attempting to build and run this solution, note the following steps:

Step 1: Check the prerequisites

Visual Studio 2010 or later is required to open the sample solution. For the complete list of prerequisites, see Prerequisites.

Step 2: Copy the sample to a directory where you have write permissions

To avoid running Visual Studio as an Administrator, copy the source code to a directory where you have write permissions, instead of running it from the default location.

Step 3: Check and set all required control properties

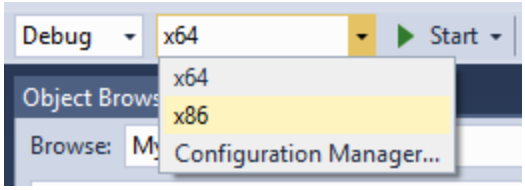
The sample application contains one instance of AuthenticDesktopControlDocument and several instances of AuthenticDesktopControlPlaceHolder controls. Double-check that the following properties of these controls are set as shown in the table below:

Control name	Property	Property value
axAuthenticDesktopControl	IntegrationLevel	ICActiveXIntegrationOnDocumentLevel
axAuthenticDesktopControlHelperWndEntities	PlaceholderWindowID	2
axAuthenticDesktopControlHelperWndAttributes	PlaceholderWindowID	1
axAuthenticDesktopControlHelperWndElements	PlaceholderWindowID	0
axAuthenticDesktopControlHelperWndInfo	PlaceholderWindowID	18
axAuthenticDesktopControlHelperWndProject	PlaceholderWindowID	4

Here is how you can view or set the properties of an ActiveX control:

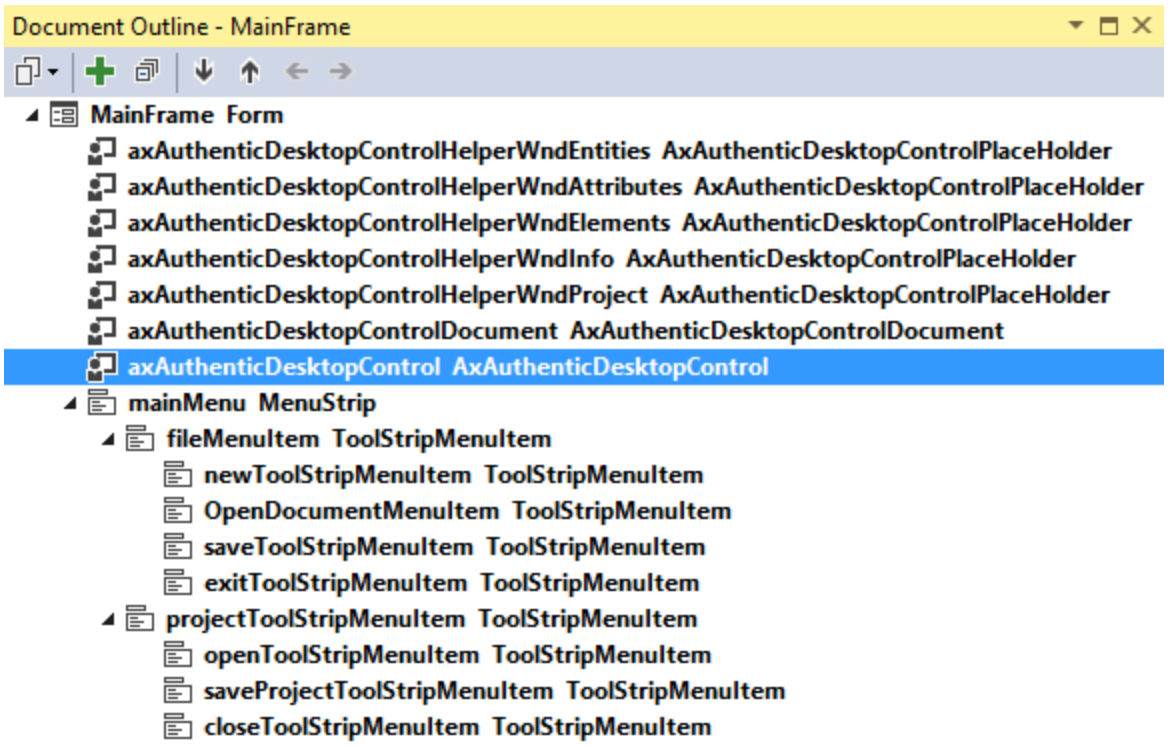
1. Open the **MDIMain.cs** form in the designer window.

Note: On 64-bit Windows, it may be necessary to change the build configuration of the Visual Studio solution to "x86" **before** opening the designer window. If you need to build the sample as a 64-bit application, see Prerequisites.

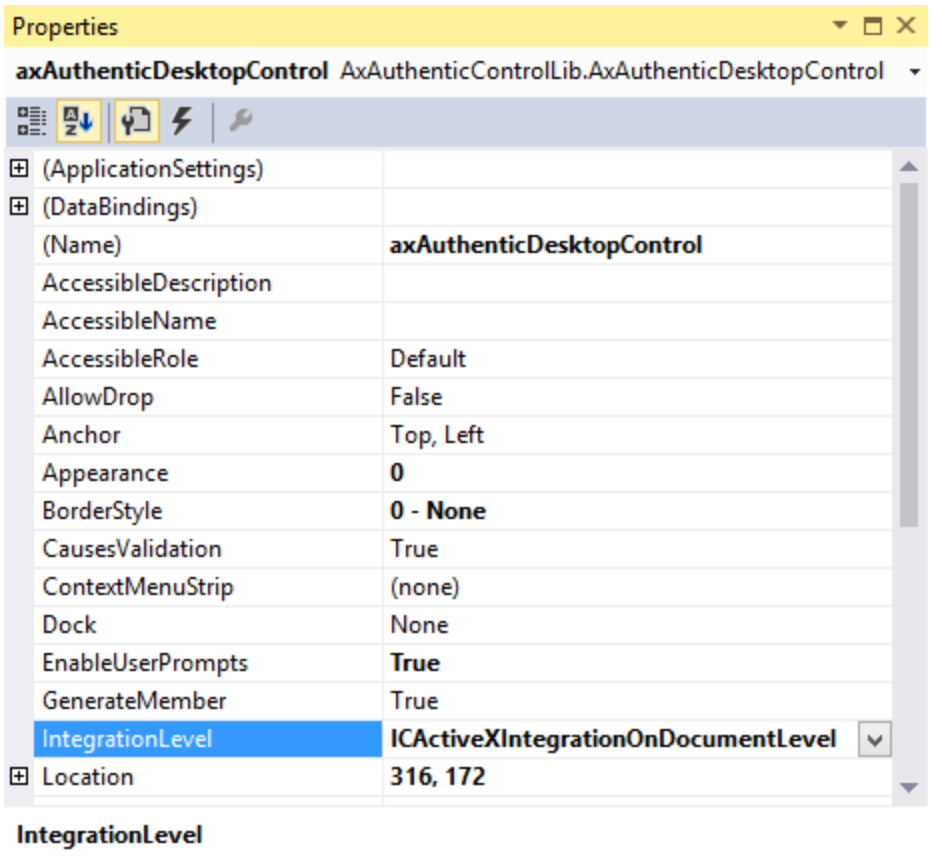


2. Open the **Document Outline** window of Visual Studio (On the **View** menu, click **Other Windows | Document Outline**).

© 2016-2022 Altova GmbH



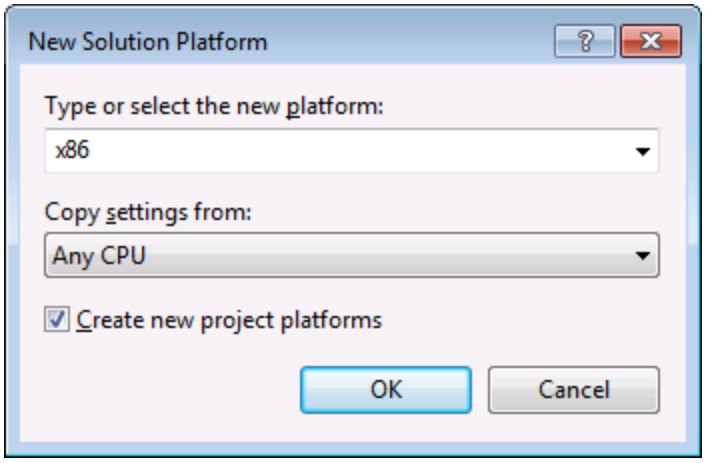
3. Click an ActiveX control in the **Document Outline** window, and edit its required property in the **Properties** window, for example:



Step 4: Set the build platform

- Create a build platform configuration that matches the platform under which you want to build (x86, x64). Here is how you can create the build configuration:

  a. Right-click the solution in Visual Studio, and select **Configuration Manager**.
  b. Under **Active solution platform**, select **New...** and then select the x86 or x64 configuration (in this example, **x86**).



© 2016-2022 Altova GmbH

You are now ready to build and run the solution in Visual Studio. Remember to build using the configuration that matches your target platform (x86, x64); otherwise, runtime errors might occur.

14.4.5.2 HTML

The code listings in this section show how to integrate the AuthenticDesktopControl at application level and document level. Source code for all examples is available in the folder `<ApplicationFolder>\Examples\ActiveX\HTML` of your Authentic Desktop installation.

Note: ActiveX controls in an HTML page are supported only by Internet Explorer when it runs as a 32-bit application. When Internet Explorer 10 or 11 runs in 64-bit mode, it does not load ActiveX controls. The default browser security settings will normally block ActiveX, so you will need to explicitly allow blocked content to run on the page when prompted by Internet Explorer.

14.4.5.2.1 HTML Integration at Application Level

This example shows a simple integration of the Authentic Desktop control at application-level into a HTML page. The integration is described in the following sections:

- Instantiate a AuthenticDesktopControl in HTML code.
- Implement buttons to load documents and automate code-generation tasks.
- Define actions for some application events.

The code for this example is available at the following location in your Authentic Desktop installation: `<ApplicationFolder>\Examples\ActiveX\HTML\AuthenticActiveX_ApplicationLevel.htm`.

14.4.5.2.1.1 Instantiate the Control

The HTML `Object` tag is used to create an instance of the AuthenticDesktopControl. The `Classid` is that of AuthenticDesktopControl. Width and height specify the window size. No additional parameters are necessary, since application-level is the default.

```
<OBJECT id="objAuthenticDesktopControl"
      Classid="clsid:4A3E7996-89B1-4d7b-9D87-615CC7C4CB47"
      width="1000"
      height="700"
      VIEWASTEXT>
</OBJECT>
```

14.4.5.2.1.2 Add Button to Open Default Document

As a simple example of how to automate some tasks, we add a button to the page:

```
<input type="button" value="Open" onclick="BtnOpen()">
```

When clicked, a predefined document will be opened in the AuthenticDesktopControl. The `MakeAbsolutePath` method creates an absolute path using the location of the script as a base path.

```
function BtnOpen()
{
   if(strPath.value.length > 0)
   {
      var absolutePath = MakeAbsolutePath(strPath.value);
      var objDoc = objAuthenticDesktopControl.Open(absolutePath);
      if (objDoc == null)
         alert("Unable to locate " + absolutePath + " at: " +
objAuthenticDesktopControl.BaseHref);
   }
   else
   {
      alert("Please set path for the document first!");
      strPath.focus();
   }
}
```

14.4.5.2.1.3 Connect to Custom Events

The example implements two event callbacks for AuthenticDesktopControl custom events to show the principle:

```
<!-- ------------------------------------------------------- -->
<!--  custom event 'OnDocumentOpened" of AuthenticDesktopControl object  -->
<SCRIPT FOR="objAuthenticDesktopControl" event="OnDocumentOpened( objDocument )"
LANGUAGE="javascript">
   // alert("Document '" + objDocument.Name + "' opened!");
</SCRIPT>

<!-- ------------------------------------------------------- -->
<!--  custom event 'OnDocumentClosed" of AuthenticDesktopControl object  -->
<SCRIPT FOR="objAuthenticDesktopControl" event="OnDocumentClosed( objDocument )"
LANGUAGE="javascript">
   // alert("Document '" + objDocument.Name + "' closed!");
</SCRIPT>
```

14.4.5.2.2 HTML Integration at Document Level

This example shows an integration of the Authentic Desktop control at document-level into a HTML page. The following topics are covered:

- Instantiate a AuthenticDesktopControl ActiveX control object in HTML code
- Instantiate a AuthenticDesktopControlDocument ActiveX control to allow editing a Authentic Desktop file
- Instantiate one AuthenticDesktopControlPlaceHolder for a AuthenticDesktopControl project window
- Instantiate one AuthenticDesktopControlPlaceHolder to alternatively host one of the Authentic Desktop helper windows

© 2016-2022 Altova GmbH

This example is available in its entirety in the file `Authentic DesktopActiveX_DocumentLevel.htm` within the `<ApplicationFolder>\Examples\ActiveX\HTML\` folder of your Authentic Desktop installation.

14.4.5.2.2.1 Instantiate the AuthenticDesktopControl

The HTML `OBJECT` tag is used to create an instance of the AuthenticDesktopControl. The Classid is that of AuthenticDesktopControl. Width and height are set to 0 since we use this control as manager control without use for its user interface. The integration level is specified as a parameter within the `OBJECT` tag.

```
<object id="objAuthenticDesktopControl" classid="clsid:4A3E7996-89B1-4d7b-9D87-
615CC7C4CB47" width="0" height="0" VIEWASTEXT>
   <param name="IntegrationLevel" value="1">
</object>
```

14.4.5.2.2.2 Create Editor Window

The HTML `OBJECT` tag is used to embed an editing window. The additional custom parameter specifies that the control is to be initialized with a new empty document.

```
<object id="objDoc1" classid="clsid:EBD19852-F38E-4274-B48E-47A8EA8EF450" width="800"
height="500" VIEWASTEXT>
   <param name="NewDocument">
</object>
```

14.4.5.2.2.3 Create Project Window

The HTML `OBJECT` tag is used to create a AuthenticDesktopControlPlaceHolder window. The parameter defines the placeholder to show the Authentic Desktop project window.

```
<!-- create Project window placeholder control.          -->
<!-- the editor with focus will automatically direct its -->
<!-- output to the appropriate helper window.           -->
<object id="objProjectWindow" classid="clsid:B80F1F5A-B739-4de8-BC76-DDAE2E848C42"
width="200" height="250" VIEWASTEXT>
   <param name="PlaceholderWindowID" value="-1">
</object>
```

14.4.5.2.2.4 Create Placeholder for Helper Windows

The **AuthenticDesktopControlPlaceHolder** control is required to host an application helper window, see also Integration at Document Level. For example, in the code listing below, the HTML `object` tag is used to instantiate a control that will host the **Message** window.

```
<!-- create Message window placeholder control.          -->
<!-- the editor with focus will automatically direct its -->
<!-- output to the appropriate helper window.            -->
<object id="objMessageWindow" classid="clsid:B80F1F5A-B739-4de8-BC76-DDAE2E848C42"
width="800" height="200" VIEWASTEXT>
   <param name="PlaceholderWindowID" value="-1">
</object>
```

Notice that the `PlaceholderWindowID` parameter is set to `-1`, which means that, initially, no helper window is shown. Whenever a file is open, the method `DisplayHelperWindowsContent()` is invoked. This method assigns the appropriate `PlaceholderWindowID` to each control. For the list of possible values of `PlaceholderWindowID`, see `AuthenticDesktopControlPlaceholderWindow`.

14.4.5.3 Java

Authentic Desktop ActiveX components can be accessed from Java code. Java integration is provided by the libraries listed below. These libraries are available in the folder `<ApplicationFolder>\Examples\JavaAPI` of your Authentic Desktop installation, after you have installed both Authentic Desktop and the Authentic Desktop Integration Package (see also Prerequisites).

- `AltovaAutomation.dll`: a JNI wrapper for Altova automation servers (in case of the 32-bit installation of Authentic Desktop)
- `AltovaAutomation_x64.dll`: a JNI wrapper for Altova automation servers (in case of the 64-bit installation of Authentic Desktop)
- `AltovaAutomation.jar`: Java classes to access Altova automation servers
- `AuthenticActiveX.jar`: Java classes that wrap the Authentic ActiveX interface
- `AuthenticActiveX_JavaDoc.zip`: a Javadoc file containing help documentation for the Java interface

Note: In order to use the Java ActiveX integration, the .dll and .jar files must be included in the Java class search path.

Example Java project

An example Java project is supplied with your product installation. You can test the Java project and modify and use it as you like. For more details, see Example Java Project.

Rules for mapping the ActiveX Control names to Java

For the documentation of ActiveX controls, see Object Reference. Note that the object naming conventions are slightly different in Java compared to other languages. Namely, the rules for mapping between the ActiveX controls and the Java wrapper are as follows:

- ***Classes and class names***
  For every component of the Authentic Desktop ActiveX interface a Java class exists with the name of the component.

- ***Method names***
  Method names on the Java interface are the same as used on the COM interfaces but start with a small letter to conform to Java naming conventions. To access COM properties, Java methods that prefix the property name with get and set can be used. If a property does not support write-access, no setter method is available. Example: For the `IntegrationLevel` property of the

© 2016-2022 Altova GmbH

AuthenticDesktopControl, the Java methods `getIntegrationLevel` and `setIntegrationLevel` are available.

- ***Enumerations***
  For every enumeration defined in the ActiveX interface, a Java enumeration is defined with the same name and values.

- ***Events and event handlers***
  For every interface in the automation interface that supports events, a Java interface with the same name plus 'Event' is available. To simplify the overloading of single events, a Java class with default implementations for all events is provided. The name of this Java class is the name of the event interface plus '`DefaultHandler`'. For example:
  `AuthenticDesktopControl`: Java class to access the application
  `AuthenticDesktopControlEvents`: Events interface for the AuthenticDesktopControl
  `AuthenticDesktopControlEventsDefaultHandler`: Default handler for `AuthenticDesktopControlEvents`

Exceptions to mapping rules

There are some exceptions to the rules listed above. These are listed below:

Interface	Java name
`AuthenticDesktopControlDocument`, method `New`	`newDocument`
`Document`, method `SetEncoding`	`setFileEncoding`
`AuthenticView`, method `Goto`	`gotoElement`
`AuthenticRange`, method `Goto`	`gotoElement`
`AuthenticRange`, method `Clone`	`cloneRange`

This section

This section shows how some basic Authentic Desktop ActiveX functionality can be accessed from Java code. It is organized into the following sub-sections:

- Example Java Project
- Creating the ActiveX Controls
- Loading Data in the Controls
- Basic Event Handling
- Menus
- UI Update Event Handling
- Creating an XML Tree

14.4.5.3.1 Example Java Project

The Authentic Desktop installation package contains an example Java project, located in the ActiveX Examples folder of the application folder: `<ApplicationFolder>\Examples\ActiveX\Java\`.

The Java example shows how to integrate the AuthenticDesktopControl in a common desktop application created with Java. You can test it directly from the command line using the batch file `BuildAndRun.bat`, or you can compile and run the example project from within Eclipse. See below for instructions on how to use these procedures.

File list

The Java examples folder contains all the files required to run the example project. These files are listed below:

`.classpath`	Eclipse project helper file
`.project`	Eclipse project file
`AltovaAutomation.dll`	Java-COM bridge: DLL part (for the 32-bit installation)
`AltovaAutomation_x64.dll`	Java-COM bridge: DLL part (for the 64-bit installation)
`AltovaAutomation.jar`	Java-COM bridge: Java library part
`AuthenticActiveX.jar`	Java classes of the Authentic Desktop ActiveX control
`AuthenticActiveX_JavaDoc.zip`	`Javadoc` file containing help documentation for the Java API
`AuthenticDesktopContainer.java`	Java example source code
`AuthenticDesktopContainerEventHandler.java`	Java example source code
`BuildAndRun.bat`	Batch file to compile and run example code from the command line prompt. Expects folder where Java Virtual Machine resides as parameter.
`XMLTreeDialog.java`	Java example source code

What the example does

The example places one AuthenticDesktop document editor window, the Project window, the Info window and an Authentic entry helper in an AWT frame window. It reads out the File menu defined for Authentic and creates an AWT menu with the same structure. You can use this menu or the project window to open and work with files in the document editor.

You can modify the example in any way you like.

The following specific features are described in code listings:

- Creating the ActiveX Controls: Starts Authentic Desktop, which is registered as an automation server, or activates Authentic Desktop if it is already running.
- Loading Data in the Controls: Locates one of the example documents installed with Authentic Desktop and opens it.
- Basic Event Handling: Changes the view of all open documents to Text View. The code also shows how to iterate through open documents.
- Menus: Validates the active document and shows the result in a message box. The code shows how to use output parameters.

© 2016-2022 Altova GmbH

- UI Update Event Handling: Shows how to handle Authentic Desktop events.
- Creating an XML Tree: Shows how to create an XML tree and prepare it for modal activation.

Updating the path to the Examples folder

Before running the provided sample, you may need to edit the **AuthenticDesktopContainer.java** file. Namely, check that the following path refers to the actual folder where the Authentic Desktop example files are stored on your operating system:

```
// Locate samples installed with the product.
final String strExamplesFolder = System.getenv( "USERPROFILE" ) + "\\Documents\\Altova\
\Authentic2023\\AuthenticExamples\\";
```

Running the example from the command line

To run the example from the command line:

1. Check that all prerequisites are met (see Prerequisites).
2. Open a command prompt window, change the current directory to the sample Java project folder, and type:

   ```
   buildAndRun.bat "<Path-to-the-Java-bin-folder>"
   ```

3. Press **Enter**.

The Java source in `AuthenticDesktopContainer.java` will be compiled and then executed.

Compiling and running the example in Eclipse

To import the sample Java project into Eclipse:

1. Check that all prerequisites are met (see Prerequisites).
2. On the **File** menu, click **Import**.
3. Select **Existing Projects into Workspace**, and browse for the Eclipse project file located at `<ApplicationFolder>\Examples\ActiveX\Java\`. Since you may not have write-access in this folder, it is recommended to select the **Copy projects into workspace** check box on the Import dialog box.

To run the example application, right-click the project in Package Explorer and select the command **Run as | Java Application**.

Help for Java API classes is available through comments in code as well as the Javadoc view of Eclipse. To enable the Javadoc view in Eclipse, select the menu command **Window | Show View | JavaDoc**.

14.4.5.3.2 Creating the ActiveX Controls

The code listing below show how ActiveX controls can be created. The constructors will create the Java wrapper objects. Adding these Canvas-derived objects to a panel or to a frame will trigger the creation of the wrapped ActiveX object.

```
01    /**
02     * Authentic Desktop manager control - always needed
03     */
04    public static AuthenticDesktopControl         authenticDesktopControl = null;
05
06    /**
07     * Authentic Desktop document editing control
08     */
09    public static AuthenticDesktopControlDocument     authenticDesktopDocument = null;
10
11    /**
12     * Tool windows - Authentic Desktop place-holder controls
13     */
14    private static AuthenticDesktopControlPlaceHolder   authenticDesktopInfoToolWindow =
null;
15    private static AuthenticDesktopControlPlaceHolder
authenticDesktopEHElementToolWindow = null;
16    private static AuthenticDesktopControlPlaceHolder   authenticDesktopProjectToolWindow
= null;
17
18    // Create the Authentic Desktop ActiveX control, The parameter determines that we want
      // to place document controls and place-holder controls individually.
19    // It gives us full control over the menu, as well.
20       authenticDesktopControl = new AuthenticDesktopControl(
         ICActiveXIntegrationLevel.ICActiveXIntegrationOnDocumentLevel.getValue() );
21
22       authenticDesktopDocument = new AuthenticDesktopControlDocument();
23       authenticDesktopDocument.setPreferredSize( new Dimension ( 640, 480 ) );
24       frame.add( authenticDesktopDocument, BorderLayout.CENTER );
25
26    // Create a project window and open the sample project in it
27       authenticDesktopProjectToolWindow = new AuthenticDesktopControlPlaceHolder(
         XMLSpyControlPlaceholderWindow.XMLSpyControlProjectWindowToolWnd.getValue() );
28       authenticDesktopProjectToolWindow.setPreferredSize( new Dimension( 200, 200 ) );
```

14.4.5.3.3 Loading Data in the Controls

The code listing below show how data can be loaded in the ActiveX controls.

```
1     // Locate samples installed with the product.
2        final String strExamplesFolder = System.getenv( "USERPROFILE" ) +
         "\\Documents\\Altova\\Authentic2022\\AuthenticExamples\\";
3        authenticDesktopProjectToolWindow = new
AuthenticDesktopControlPlaceHolder( XMLSpyControlPlaceholderWindow.XMLSpyControlProjectWind
owToolWnd.getValue() );
```

14.4.5.3.4 Basic Event Handling

The code listing below shows how basic events can be handled. When calling the AuthenticDesktopControl's `open` method, or when trying to open a file via the menu or Project tree, the `onOpenedOrFocused` event is sent to the attached event handler. The basic handling for this event is opening the file by calling the Authentic DesktopDocumentControl's `open` method.

© 2016-2022 Altova GmbH

```
// Open the PXF file when button is pressed
      btnOpenPxf.addActionListener( new ActionListener() {
        public void actionPerformed(ActionEvent e) {
          try {
            authenticDesktopControl.open( strExamplesFolder + "OrgChart.pxf" );
          } catch (AutomationException e1) {
            e1.printStackTrace();
          }
        }
      } );
      public void onOpenedOrFocused( String i_strFileName, boolean
i_bOpenWithThisControl, boolean i_bFileAlreadyOpened ) throws AutomationException
  {
    // Handle the New/Open events coming from the Project tree or from the menus
    if ( !i_bFileAlreadyOpened )
    {
      // This is basically an SDI interface, so open the file in the already existing
document control
      try {
        AuthenticDesktopContainer.authenticDesktopDocument.open( i_strFileName );
        AuthenticDesktopContainer.authenticDesktopDocument.requestFocusInWindow();
      } catch (Exception e) {
        e.printStackTrace();
      }
    }
  }
```

14.4.5.3.5 Menus

The code listing below shows how menu items can be created. Each `Authentic DesktopCommand` object gets a corresponding `MenuItem` object, with the `ActionCommand` set to the ID of the command. The actions generated by all menu items are handled by the same function, which can perform specific handlings (like reinterpreting the closing mechanism) or can delegate the execution to the `AuthenticDesktopControl` object by calling its `exec` method. The `menuMap` object that is filled during menu creation is used later (see section UI Update Event Handling).

```
 // Load the file menu when the button is pressed
      btnMenu.addActionListener( new ActionListener() {
        public void actionPerformed(ActionEvent e) {
          try {
            // Create the menubar that will be attached to the frame
            MenuBar mb = new MenuBar();
            // Load the main menu's first item - the File menu
            XMLSpyCommand xmlSpyMenu =
xmlSpyControl.getMainMenu().getSubCommands().getItem( 0 );
            // Create Java menu items from the Commands objects
            Menu fileMenu = new Menu();
            handlerObject.fillMenu( fileMenu, xmlSpyMenu.getSubCommands() );
            fileMenu.setLabel( xmlSpyMenu.getLabel().replace( "&", "" ) );
            mb.add( fileMenu );
            frame.setMenuBar( mb );
            frame.validate();
          } catch (AutomationException e1) {
            e1.printStackTrace();
```

```
18            }
19            // Disable the button when the action has been performed
20            ((AbstractButton) e.getSource()).setEnabled( false );
21          }
22        } ) ;
23 /** * Populates a menu with the commands and submenus contained in an XMLSpyCommands
object  */
24        public void fillMenu(Menu newMenu, XMLSpyCommands xmlSpyMenu) throws
AutomationException
25    {
26      // For each command/submenu in the xmlSpyMenu
27      for ( int i = 0 ; i < xmlSpyMenu.getCount() ; ++i  )
28      {
29        XMLSpyCommand xmlSpyCommand = xmlSpyMenu.getItem( i );
30        if ( xmlSpyCommand.getIsSeparator() )
31          newMenu.addSeparator();
32        else
33        {
34          XMLSpyCommands subCommands = xmlSpyCommand.getSubCommands();
35          // Is it a command (leaf), or a submenu?
36          if ( subCommands.isNull() || subCommands.getCount() == 0 )
37          {
38            // Command -> add it to the menu, set its ActionCommand to its ID and store it
in the menuMap
39            MenuItem mi = new MenuItem( xmlSpyCommand.getLabel().replace( "&", "" ) );
40            mi.setActionCommand( "" + xmlSpyCommand.getID() );
41            mi.addActionListener( this );
42            newMenu.add( mi );
43            menuMap.put( xmlSpyCommand.getID(), mi );
44          }
45          else
46          {
47            // Submenu -> create submenu and repeat recursively
48            Menu newSubMenu = new Menu();
49            fillMenu( newSubMenu, subCommands );
50            newSubMenu.setLabel( xmlSpyCommand.getLabel().replace( "&", "" ) );
51            newMenu.add( newSubMenu );
52          }
53        }
54      }
55    }
56
57     /**
58     * Action handler for the menu items
59     * Called when the user selects a menu item; the item's action command corresponds to
the command table for XMLSpy
60     */
61    public void actionPerformed( ActionEvent e )
62    {
63      try
64      {
65        int iCmd = Integer.parseInt( e.getActionCommand() );
66        // Handle explicitly the Close commands
67        switch ( iCmd )
68        {
69          case 57602:       // Close
70          case 34050:       // Close All
71            AuthenticDesktopContainer.initXmlSpyDocument();
72            break;
```


© 2016-2022 Altova GmbH

```
73          default:
74            AuthenticDesktopContainer.xmlSpyControl.exec( iCmd );
75            break;
76        }
77      }
78      catch ( Exception ex )
79      {
80        ex.printStackTrace();
81      }
82
83    }
```

14.4.5.3.6 UI Update Event Handling

The code listing below shows how a UI-Update event handler can be created.

```
01    /**
02     * Call-back from the XMLSpyControl.
03     * Called to enable/disable commands
04     */
05    @Override
06    public void onUpdateCmdUI() throws AutomationException
07    {
08      // A command should be enabled if the result of queryStatus contains the Supported
(1) and Enabled (2) flags
09      for ( java.util.Map.Entry<Integer, MenuItem> pair : menuMap.entrySet() )
10
       pair.getValue().setEnabled( AuthenticDesktopContainer.authenticDesktopControl.querySt
atus( pair.getKey() ) > 2 );
11    }
12    /**
13     * Call-back from the XMLSpyControl.
14     * Usually called while enabling/disabling commands due to UI updates
15     */
16    @Override
17    public boolean onIsActiveEditor( String i_strFilePath ) throws AutomationException
18    {
19      try {
20        return
AuthenticDesktopContainer.authenticDesktopDocument.getDocument().getFullName().equalsIgnore
Case( i_strFilePath );
21      } catch ( Exception e ) {
22        return false;
23      }
24    }
```

14.4.5.3.7 Creating an XML Tree

The listing below loads an XML data object as nodes in a tree.

```
01 // access required XMLSpy Java-COM classes
02 import com.altova.automation.XMLSpy.XMLData;
03
```

```
04 // access AWT and Swing components
05 import java.awt.*;
06 import javax.swing.*;
07 import javax.swing.tree.*;
08
09 /**
10  * A simple example of a tree control loading the structure from an XMLData object.
11  * The class receives an XMLData object, loads its nodes in a JTree, and prepares
12  * for modal activation.
13  *
14  * Feel free to modify and extend this sample.
15  *
16  * @author Altova GmbH
17  */
18 class XMLTreeDialog extends JDialog
19 {
20   /**
21    * The tree control
22    */
23   private JTree myTree;
24
25   /**
26    * Root node of the tree control
27    */
28   private DefaultMutableTreeNode top ;
29
30   /**
31    * Constructor that prepares the modal dialog containing the filled tree control
32    * @param xml   The data to be displayed in the tree
33    * @param parent  Parent frame
34    */
35   public XMLTreeDialog( XMLData xml, Frame parent )
36   {
37     // Construct the modal dialog
38     super( parent, "XML tree", true );
39     // Arrange controls in the dialog
40     top = new DefaultMutableTreeNode("root");
41     myTree = new JTree(top);
42     setContentPane( new JScrollPane( myTree ) );
43     // Build up the tree
44     fillTree( top, xml );
45     myTree.expandRow( 0 );
46   }
47
48   /**
49    * Loads the nodes of an XML element under a given tree node
50    * @param node  Target tree node
51    * @param elem  Source XML element
52    */
53   private void fillTree( DefaultMutableTreeNode node, XMLData elem)
54   {
55     try
56     {
57       // There are several ways to iterate through child elements: either using the
getFirstChild/getNextChild,
58       // or by incrementing an index up to countChildren and calling getChild [as shown
below].
```


© 2016-2022 Altova GmbH

```
59        // If you only want to get children of one kind, you should use
countChildrenKind/getChildKind,
60        // or provide a kind to the getFirstChild before iterating with the getNextChild.
61        int nSize = elem.countChildren() ;
62        for ( int i = 0 ; i < nSize ; ++i)
63        {
64          // Create a new tree node for each child element, and continue recursively
65          XMLData newElem = elem.getChild(i) ;
66          DefaultMutableTreeNode newNode = new DefaultMutableTreeNode( newElem.getName() )
;
67          node.add( newNode ) ;
68          fillTree( newNode, newElem ) ;
69        }
70      }
71      catch (Exception e)
72      {
73        e.printStackTrace();
74      }
75    }
76
77 }
```

14.4.6 Command Reference

This section lists the names and identifiers of all menu commands that are available within Authentic Desktop. Every sub-section lists the commands from the corresponding top-level menu of Authentic Desktop. The command tables are organized as follows:

- The "Menu Item" column shows the command's menu text as it appears in Authentic Desktop, to make it easier for you to identify the functionality behind the command.
- The "Command Name" column specifies the string that can be used to get an icon with the same name from **ActiveX\Images** folder of the Authentic Desktop installation directory.
- The "ID" column shows the numeric identifier of the column that must be supplied as argument to methods which execute or query this command.

To execute a command, use the `AuthenticDesktopControl.Exec` or the `AuthenticDesktopControlDocument.Exec` methods. To query the status of a command, use the `AuthenticDesktopControl.QueryStatus` or `AuthenticDesktopControlDocument.QueryStatus` methods.

Depending on the edition of Authentic Desktop you have installed, some of these commands might not be supported.

14.4.6.1 "File" Menu

The "File" menu has the following commands:

Menu item	Command name	ID
New...	ID_FILE_NEW	57600
Open...	ID_FILE_OPEN	57601

Menu item	Command name	ID
Reload	IDC_FILE_RELOAD	34065
Encoding...	IDC_ENCODING	34061
Close	ID_FILE_CLOSE	57602
Close All	IDC_CLOSE_ALL	34050
Close All But Active	IDC_CLOSE_OTHERS	34271
Save	ID_FILE_SAVE	57603
Save As...	ID_FILE_SAVE_AS	57604
Save All	ID_FILE_SAVE_ALL	34208
Send by Mail...	ID_FILE_SEND_MAIL	57612
Print...	ID_FILE_PRINT	57607
Print Preview	IDC_PRINT_PREVIEW	34104
Print Setup...	ID_FILE_PRINT_SETUP	57606
Recent File	ID_FILE_MRU_FILE1	57616
Exit	ID_APP_EXIT	57665

14.4.6.2 "Edit" Menu

The "Edit" menu has the following commands:

Menu item	Command name	ID
Undo	ID_EDIT_UNDO	57643
Redo	ID_EDIT_REDO	57644
Cut	ID_EDIT_CUT	57635
Copy	ID_EDIT_COPY	57634
Paste	ID_EDIT_PASTE	57637
Delete	ID_EDIT_CLEAR	57632
Select All	ID_EDIT_SELECT_ALL	57642
Find...	ID_EDIT_FIND	57636
Find Next	ID_EDIT_REPEAT	57640

© 2016-2022 Altova GmbH

Menu item	Command name	ID
Replace...	ID_EDIT_REPLACE	57641

14.4.6.3 "Project" Menu

The "Project" menu has the following commands:

Menu item	Command name	ID
New Project	IDC_ICPROJECTGUI_NEW	37200
Open Project...	IDC_ICPROJECTGUI_OPEN	37201
Reload Project	IDC_ICPROJECTGUI_RELOAD	37202
Close Project	IDC_ICPROJECTGUI_CLOSE	37203
Save Project	IDC_ICPROJECTGUI_SAVE	37204
Save Project As...	IDC_ICPROJECTGUI_SAVE_AS	37207
Enable Source Control	ID_SCC_ENABLE	38602
Add Files to Project...	IDC_ICPROJECTGUI_ADD_FILES_TO_PROJECT	37205
Add Global Resource to Project...	IDC_ICPROJECTGUI_ADD_GLOBAL_RESOURCE_TO_PROJECT	37239
Add URL to Project...	IDC_ICPROJECTGUI_ADD_URL_TO_PROJECT	37206
Add Active File to Project	IDC_ICPROJECTGUI_ADD_ACTIVE_FILE_TO_PROJECT	37208
Add Active and Related Files to Project	IDC_ICPROJECTGUI_ADD_ACTIVE_AND_RELATED_FILES_TO_PROJECT	37209
Add Project Folder to Project...	IDC_ICPROJECTGUI_ADD_FOLDER_TO_PROJECT	37210
Add External Folder to Project...	IDC_ICPROJECTGUI_ADD_EXT_FOLDER_TO_PROJECT	37211
Add External Web Folder to Project...	IDC_ICPROJECTGUI_ADD_EXT_URL_FOLDER_TO_PROJECT	37212
Script settings...	IDC_PROJECT_SCRIPT_SETTINGS	34136
Properties...	IDC_ICPROJECTGUI_PROJECT_PROPERTIES	37223

Menu item	Command name	ID
Recent Project	IDC_ICPROJECTGUI_RECENT	37224

14.4.6.4 "XML" Menu

The "XML" menu has the following commands:

Menu item	Command name	ID
Check Well-Formedness	IDC_CHECK_WELL_FORM	34049
Validate XML	IDC_VALIDATE	32954

14.4.6.5 "XSL/XQuery" Menu

The "XSL/XQuery" menu has the following commands:

Menu item	Command name	ID
XSL Transformation	IDC_TRANSFORM_XSL	33006
XSL-FO Transformation	IDC_TRANSFORM_XSLFO	33007
XSL Parameters / XQuery Variables...	IDC_TRANSFORM_XSL_PARAMS	33008

14.4.6.6 "Authentic" Menu

The "Authentic" menu has the following commands:

Menu item	Command name	ID
New Document...	IDC_AUTHENTIC_NEW_FILE	34036
Edit Database Data...	IDC_AUTHENTIC_EDIT_DB	34035
Edit StyleVision Stylesheet	IDC_EDIT_SPS	34060
Select New Row with XML Data for Editing...	IDC_CHANGE_WORKING_DB_XML_CELL	32861
XML Signature...	IDC_AUTHENTICGUI_XMLSIGNATURE	32862
Define XML Entities...	IDC_DEFINE_ENTITIES	32805
Hide Markup	IDC_MARKUP_HIDE	32855
Show Small Markup	IDC_MARKUP_SMALL	32858

© 2016-2022 Altova GmbH

Menu item	Command name	ID
Show Large Markup	IDC_MARKUP_LARGE	32856
Show Mixed Markup	IDC_MARKUP_MIXED	32857
Toggle Bold	IDC_AUTHENTICGUI_RICHEDIT_TOGGLEBOLD	32813
Toggle Italic	IDC_AUTHENTICGUI_RICHEDIT_TOGGLEITALIC	32814
Toggle Underline	IDC_AUTHENTICGUI_RICHEDIT_TOGGLEUNDERLINE	32815
Toggle Strikethrough	IDC_AUTHENTICGUI_RICHEDIT_TOGGLESTRIKETHROUGH	32816
Foreground Color	IDC_AUTHENTICGUI_RICHEDIT_COLOR_FOREGROUND	32824
Background Color	IDC_AUTHENTICGUI_RICHEDIT_COLOR_BACKGROUND	32830
Align Left	IDC_AUTHENTICGUI_RICHEDIT_ALIGN_LEFT	32818
Center	IDC_AUTHENTICGUI_RICHEDIT_ALIGN_CENTER	32819
Align Right	IDC_AUTHENTICGUI_RICHEDIT_ALIGN_RIGHT	32820
Append Row	IDC_ROW_APPEND	32806
Insert Row	IDC_ROW_INSERT	32809
Duplicate Row	IDC_ROW_DUPLICATE	32808
Move Row Up	IDC_ROW_MOVE_UP	32811
Move Row Down	IDC_ROW_MOVE_DOWN	32810
Delete Row	IDC_ROW_DELETE	32807
Generate an HTML document	IDC_PXF_GENERATE_HTML	34283
Generate an RTF document	IDC_PXF_GENERATE_RTF	34284
Generate a PDF document	IDC_PXF_GENERATE_PDF	34285
Generate a Word 2007+ document	IDC_PXF_GENERATE_DOCX	34286
Trusted Locations...	IDC_TRUSTED_LOCATIONS	34288

14.4.6.7 "View" Menu

The "View" menu has the following commands:

Menu item	Command name	ID
Authentic View	IDC_VIEW_CONTENT	34177
Browser View	IDC_VIEW_BROWSER	34176
Text View Settings	IDC_TEXTVIEW_SETTINGS	34119

14.4.6.8 "Browser" Menu

The "Browser" menu has the following commands:

Menu item	Command name	ID
Back	IDC_STEP_BACK	32958
Forward	IDC_STEP_FORWARD	32957
Stop	IDC_BROWSER_STOP	34047
Refresh	IDC_BROWSER_REFRESH	34046
Largest	IDC_BROWSER_FONT_LARGEST	34041
Larger	IDC_BROWSER_FONT_LARGE	34040
Medium	IDC_BROWSER_FONT_MEDIUM	34042
Smaller	IDC_BROWSER_FONT_SMALL	34043
Smallest	IDC_BROWSER_FONT_SMALLEST	34044

14.4.6.9 "Tools" Menu

The "Tools" menu has the following commands:

Menu item	Command name	ID
Spelling...	IDC_SPELL_CHECK	34154
Spelling Options...	IDC_SPELL_OPTIONS	34155
Scripting Editor...	ID_SCRIPTFORMEDITOR_EDIT_PROJECT	39666

© 2016-2022 Altova GmbH

Menu item	Command name	ID
none	ID_SCRIPTFORMEDITOR_EXECUTE_MACRO_MENU_UPPDATE	39600
	IDC_TOOLS_ENTRY	34292
Global Resources	IDC_GLOBALRESOURCES	37401
	IDC_GLOBALRESOURCES_SUBMENUENTRY1	37408
Customize...	IDC_APP_TOOLS_CUSTOMIZE	32959
Options...	IDC_SETTINGS	34133
	ID_SCRIPTING_MACROITEMS	34249

14.4.6.10 "Window" Menu

The "Window" menu has the following commands:

Menu item	Command name	ID
Cascade	ID_WINDOW_CASCADE	57650
Tile horizontally	ID_WINDOW_TILE_HORZ	57651
Tile vertically	ID_WINDOW_TILE_VERT	57652
Project window	IDC_PROJECT_WINDOW	34128
Info window	IDC_INFO_WINDOW	34085
Entry Helpers	IDC_ENTRY_HELPERS	34062
Output windows	IDC_OUTPUT_DIALOGBARS	34004
Project and Entry Helpers	IDC_PROJECT_ENTRYHELPERS	34006
All on/off	IDC_ALL_BARS	34031

14.4.6.11 "Help" Menu

The "Help" menu has the following commands:

Menu item	Command name	ID
Table of Contents...	IDC_HELP_CONTENTS	32966
Index...	IDC_HELP_INDEX	32967

Menu item	Command name	ID
Search...	IDC_HELP_SEARCH	32969
Keyboard Map...	IDC_HELP_KEYMAPDLG	32968
Software Activation...	IDC_ACTIVATION	32970
Order Form...	IDC_OPEN_ORDER_PAGE	32971
Registration...	IDC_REGISTRATION	32972
Check for Updates...	IDC_CHECK_FOR_UPDATES	32973
XMLSpy Product Comparison...	IDC_PRODUCT_COMPARISON	32955
Support Center...	IDC_OPEN_SUPPORT_PAGE	32961
FAQ on the Web...	IDC_OPEN_FAQ_PAGE	32962
Download Components and Free Tools...	IDC_OPEN_COMPONENTS_PAGE	32963
Authentic on the Internet..	IDC_OPEN_HOME_PAGE	32964
Authentic Training...	IDC_OPEN_TRAINING_PAGE	32965
About Authentic...	ID_APP_ABOUT	57664

14.4.7 Object Reference

Objects:
Authentic DesktopCommand
Authentic DesktopCommands
AuthenticDesktopControl
AuthenticDesktopControlDocument
AuthenticDesktopControlPlaceHolder

To give access to standard Authentic Desktop functionality, objects of the **Authentic Desktop automation interface** can be accessed as well. See AuthenticDesktopControl.Application, AuthenticDesktopControlDocument.Document and AuthenticDesktopControlPlaceHolder.Project for more information.

14.4.7.1 Authentic DesktopCommand

Properties:
ID
Label
Name
IsSeparator
ToolTip
StatusText

© 2016-2022 Altova GmbH

Accelerator
SubCommands

Description:
A command object can be one of the following: an executable command, a command container (for example, a menu, submenu, or toolbar), or a menu separator. To determine what kind of information is stored in the current Command object, query its ID, IsSeparator, and SubCommands properties, as follows.

The Command object is...	When...
An executable command	• ID is greater than zero • IsSeparator is false • SubCommands is empty
A command container	• ID is zero • IsSeparator is false • SubCommands contains a collection of Command objects.
Separator	• ID is zero • IsSeparator is true

14.4.7.1.1 Accelerator

Property: Accelerator as string

Description:
Returns the accelerator key defined for the command. If the command has no accelerator key assigned, this property returns the empty string. The string representation of the accelerator key has the following format:

```
[ALT+][CTRL+][SHIFT+]key
```

Where key is converted using the Windows Platform SDK function GetKeyNameText.

14.4.7.1.2 ID

Property: ID as long

Description:
This property gets the unique identifier of the command. A command's ID is required to execute the command (using Exec) or query its status (using QueryStatus). If the command is a container for other commands (for example, a top-level menu), or a separator, the ID is 0.

14.4.7.1.3 IsSeparator

Property: IsSeparator as boolean

Description:

The property returns `true` if the command object is a menu separator; `false` otherwise. See also Command.

14.4.7.1.4 Label

Property: `Label` as string

Description:
This property gets the text of the command as it is displayed in the graphical user interface of Authentic Desktop. If the command is a separator, "Label" is an empty string. This property may also return an empty string for some toolbar commands that do not have any GUI text associated with them.

14.4.7.1.5 Name

Property: `Name` as string

Description:
This property gets the unique name of the command. This value can be used to get the icon file of the command, where it is available. The available icon files can be found in the folder **<ApplicationFolder>\Examples\ActiveX\Images** of your Authentic Desktop installation.

14.4.7.1.6 StatusText

Property: `Label` as string

Description:
The status text is the text shown in the status bar of Authentic Desktop when the command is selected. It applies only to command objects that are not separators or containers of other commands; otherwise, the property is an empty string.

14.4.7.1.7 SubCommands

Property: `SubCommands` as `Commands`

Description:
The `SubCommands` property gets the collection of `Command` objects that are sub-commands of the current command. The property is applicable only to commands that are containers for other commands (menus, submenus, or toolbars). Such container commands have the `ID` set to 0, and the `IsSeparator` property set to `false`.

14.4.7.1.8 ToolTip

Property: `ToolTip` as string

Description:

© 2016-2022 Altova GmbH

This property gets the text that is shown as a tool-tip for each command. If the command does not have a tooltip text, the property returns an empty string.

14.4.7.2 Authentic DesktopCommands

Properties:
Count
Item

Description:
Collection of Command objects to get access to command labels and IDs of the AuthenticDesktopControl. Those commands can be executed with the Exec method and their status can be queried with QueryStatus.

14.4.7.2.1 Count

Property: `Count` as long

Description:
Number of Command objects on this level of the collection.

14.4.7.2.2 Item

Property: `Item` (`n` as long) as Command

Description:
Gets the command with the index `n` in this collection. Index is 1-based.

14.4.7.3 AuthenticDesktopControl

Properties:
IntegrationLevel
Appearance
Application
BorderStyle
CommandsList
EnableUserPrompts
MainMenu
Toolbars

Methods:
Open
Exec
QueryStatus

Events:
OnUpdateCmdUI
OnOpenedOrFocused

OnCloseEditingWindow
OnFileChangedAlert
OnContextChanged
OnDocumentOpened
OnValidationWindowUpdated

This object is a complete ActiveX control and should only be visible if the Authentic Desktop library is used in the Application Level mode.

14.4.7.3.1 Properties

The following properties are defined:

IntegrationLevel
EnableUserPrompts
Appearance
BorderStyle

Command related properties:
CommandsList
MainMenu
Toolbars

Access to AuthenticDesktopAPI:
Application

14.4.7.3.1.1 Appearance

Property: `Appearance` as short

Dispatch Id: *-520*

Description:
A value not equal to `0` displays a client edge around the control. Default value is `0`.

14.4.7.3.1.2 Application

Property: `Application` as Application

Dispatch Id: *1*

Description:
The `Application` property gives access to the `Application` object of the complete Authentic Desktop automation server API. The property is read-only.

© 2016-2022 Altova GmbH

14.4.7.3.1.3 BorderStyle

Property: `BorderStyle` as short

Dispatch Id: *-504*

Description:
A value of `1` displays the control with a thin border. Default value is `0`.

14.4.7.3.1.4 CommandsList

Property: `CommandList` as `Commands` (read-only)

Dispatch Id: *1004*

Description:
This property returns a flat list of all commands defined available with AuthenticDesktopControl. To get commands organized according to their menu structure, use `MainMenu`. To get toolbar commands, use `Toolbars`.

```
public void GetAllAuthenticCommands()
{
    // Get all commands from the Authentic ActiveX control assigned to the current form
    AuthenticControlLib.XMLSpyCommands commands =
this.axAuthenticDesktopControl1.CommandList;
    // Loop through all commands
    for (int i = 0; i < commands.Count; i++)
    {
      // Get each command by index and output it to the console
      AuthenticControlLib.XMLSpyCommand cmd = axAuthenticDesktopControl1.CommandList[i];
      Console.WriteLine("{0} {1} {2}", cmd.ID, cmd.Name, cmd.Label.Replace("&", ""));
    }
}
```

C# example

14.4.7.3.1.5 EnableUserPrompts

Property: `EnableUserPrompts` as boolean

Dispatch Id: *1006*

Description:
Setting this property to *false*, disables user prompts in the control. The default value is *true*.

14.4.7.3.1.6 IntegrationLevel

Property: `IntegrationLevel` as `ICActiveXIntegrationLevel`

Dispatch Id: *1000*

Description:
The `IntegrationLevel` property determines the operation mode of the control. See also Integration at Application Level and Integration at Document Level for more information.

Note: It is important to set this property immediately after the creation of the `AuthenticDesktopControl` object.

14.4.7.3.1.7 MainMenu

Property: `MainMenu` as `Command` (read-only)

Dispatch Id: *1003*

Description:
This property provides information about the structure and commands available in the AuthenticDesktopControl main menu, as a `Command` object. The `Command` object contains all available submenus of Authentic Desktop (for example "File", "Edit", "View" etc.). To access the submenu objects, use the `SubCommands` property of the `MainMenu` property. Each submenu is also a `Command` object. For each submenu, you can then further iterate through their `SubCommands` property in order to get their corresponding child commands and separators (this technique may be used, for example, to create the application menu programmatically). Note that some menu commands act as containers ("parents") for other menu commands, in which case they also have a `SubCommands` property. To get the structure of all menu commands programmatically, you will need a recursive function.

```
public void GetAuthenticMenus()
{
    // Get the main menu from the Authentic ActiveX control assigned to the current form
    AuthenticControlLib.XMLSpyCommand mainMenu =
this.axAuthenticDesktopControl1.MainMenu;

    // Loop through entries of the main menu (e.g. File, Edit, etc.)
    for (int i = 0; i < mainMenu.SubCommands.Count; i++)
    {
      AuthenticControlLib.XMLSpyCommand menu = mainMenu.SubCommands[i];
      Console.WriteLine("{0} menu has {1} children items (including separators)",
menu.Label.Replace("&", ""), menu.SubCommands.Count);
    }
}
```

C# example

© 2016-2022 Altova GmbH

14.4.7.3.1.8 Toolbars

Property: `Toolbars` as Commands (read-only)

Dispatch Id: *1005*

Description:
This property provides information about the structure of AuthenticDesktopControl toolbars, as a `Command` object. The `Command` object contains all available toolbars of Authentic Desktop. To access the toolbars, use the `SubCommands` property of the `Toolbars` property. Each toolbar is also a `Command` object. For each toolbar, you can then further iterate through their `SubCommands` property in order to get their commands (this technique may be used, for example, to create the application's toolbars programmatically).

```
public void GetAuthenticToolbars()
{
    // Get the application toolbars from the Authentic ActiveX control assigned to the
current form
    AuthenticControlLib.XMLSpyCommands toolbars =
this.axAuthenticDesktopControl1.Toolbars;

    // Iterate through all toolbars
    for (int i = 0; i < toolbars.Count; i++)
    {
      AuthenticControlLib.XMLSpyCommand toolbar = toolbars[i];
      Console.WriteLine();
      Console.WriteLine("The toolbar \"{0}\" has the following commands:",
toolbar.Label);

      // Iterate through all commands of this toolbar
      for (int j = 0; j < toolbar.SubCommands.Count; j++)
      {
         AuthenticControlLib.XMLSpyCommand cmd = toolbar.SubCommands[j];
         // Output only command objects that are not separators
         if ( ! cmd.IsSeparator)
         {
            Console.WriteLine("{0}, {1}, {2}", cmd.ID, cmd.Name, cmd.Label.Replace("&",
""));
         }
      }
    }
}
```

C# example

14.4.7.3.2 Methods

The following methods are defined:

Open
Exec
QueryStatus

14.4.7.3.2.1 Exec

Method: `Exec` (`nCmdID` as long) as boolean

Dispatch Id: *6*

Description:
This method calls the Authentic Desktop command with the ID `nCmdID`. If the command can be executed, the method returns `true`. To get a list of all available commands, use `CommandsList`. To retrieve the status of any command, use `QueryStatus`.

14.4.7.3.2.2 Open

Method: `Open` (`strFilePath` as string) as boolean

Dispatch Id: *5*

Description:
The result of the method depends on the extension passed in the argument `strFilePath`. If the file extension is `.sps`, a new document is opened. If the file extension is `.svp`, the corresponding project is opened. If a different file extension is passed into the method, the control tries to load the file as a new component into the active document.

Do not use this method to load documents or projects when using the control in document-level integration mode. Instead, use `AuthenticDesktopControlDocument.Open` and `AuthenticDesktopControlPlaceHolder.OpenProject`.

14.4.7.3.2.3 QueryStatus

Method: `QueryStatus` (`nCmdID` as long) as long

Dispatch Id: *7*

Description:
`QueryStatus` returns the enabled/disabled and checked/unchecked status of the command specified by `nCmdID`. The status is returned as a bit mask.

Bit	Value	Name	Meaning
0	1	Supported	Set if the command is supported.
1	2	Enabled	Set if the command is enabled (can be executed).
2	4	Checked	Set if the command is checked.

This means that if `QueryStatus` returns `0` the command ID is not recognized as a valid Authentic Desktop command. If `QueryStatus` returns a value of `1` or `5`, the command is disabled.

© 2016-2022 Altova GmbH

14.4.7.3.3 Events

The AuthenticDesktopControl ActiveX control provides the following connection point events:

OnUpdateCmdUI
OnOpenedOrFocused
OnCloseEditingWindow
OnFileChangedAlert
OnContextChanged

OnDocumentOpened
OnValidationWindowUpdated

14.4.7.3.3.1 OnCloseEditingWindow

Event: `OnCloseEditingWindow` (i_strFilePath as String) as boolean

Dispatch Id: *1002*

Description:
This event is triggered when Authentic Desktop needs to close an already open document. As an answer to this event, clients should close the editor window associated with *i_strFilePath*. Returning *true* from this event indicates that the client has closed the document. Clients can return *false* if no specific handling is required and AuthenticDesktopControl should try to close the editor and destroy the associated document control.

14.4.7.3.3.2 OnContextChanged

Event: `OnContextChanged` (`i_strContextName` as String, `i_bActive` as bool) as bool

Dispatch Id: *1004*

Description:
This event is not used in Authentic Desktop

14.4.7.3.3.3 OnDocumentOpened

Event: `OnDocumentOpened` (`objDocument` as Document)

Dispatch Id: *1*

Description:
This event is triggered whenever a document is opened. The argument `objDocument` is a `Document` object from the Authentic Desktop automation interface and can be used to query for more details about the document, or perform additional operations. When integrating on document-level, it is often better to use the event `AuthenticDesktopControlDocument.OnDocumentOpened` instead.

14.4.7.3.3.4 OnFileChangedAlert

Event: `OnFileChangedAlert` (`i_strFilePath` as String) as bool

Dispatch Id: *1001*

Description:
This event is triggered when a file loaded with AuthenticDesktopControl is changed on the hard disk by another application. Clients should return true, if they handled the event, or false, if Authentic Desktop should handle it in its customary way, i.e. prompting the user for reload.

14.4.7.3.3.5 OnLicenseProblem

Event: `OnLicenseProblem` (`i_strLicenseProblemText` as String)

Dispatch Id: *1005*

Description:
This event is triggered when AuthenticDesktopControl detects that no valid license is available for this control. In case of restricted user licenses this can happen some time after the control has been initialized. Integrators should use this event to disable access to this control's functionality. After returning from this event, the control will block access to its functionality (e.g. show empty windows in its controls and return errors on requests).

14.4.7.3.3.6 OnOpenedOrFocused

Event: `OnOpenedOrFocused` (`i_strFilePath` as String, `i_bOpenWithThisControl` as bool)

Dispatch Id: *1000*

Description:
When integrating at application level, this event informs clients that a document has been opened, or made active by Authentic Desktop.

When integrating at document level, this event instructs the client to open the file `i_strFilePath` in a document window. If the file is already open, the corresponding document window should be made the active window.

if `i_bOpenWithThisControl` is `true`, the document must be opened with AuthenticDesktopControl, since internal access is required. Otherwise, the file can be opened with different editors.

14.4.7.3.3.7 OnToolWindowUpdated

Event: `OnToolWindowUpdated(pToolWnd` as long `)`

Dispatch Id: *1006*

Description:

© 2016-2022 Altova GmbH

This event is triggered when the tool window is updated.

14.4.7.3.3.8 OnUpdateCmdUI

Event: `OnUpdateCmdUI ()`

Dispatch Id: *1003*

Description:
Called frequently to give integrators a good opportunity to check status of Authentic Desktop commands using `AuthenticDesktopControl.QueryStatus`. Do not perform long operations in this callback.

14.4.7.3.3.9 OnValidationWindowUpdated

Event: `OnValidationWindowUpdated` ()

Dispatch Id: *3*

Description:
This event is triggered whenever the validation output window is updated with new information.

14.4.7.4 AuthenticDesktopControlDocument

Properties:
`Appearance`
`BorderStyle`
`Document`
`IsModified`
`Path`
`ReadOnly`

Methods:
`Exec`
`New`
`Open`
`QueryStatus`
`Reload`
`Save`
`SaveAs`

Events:
`OnDocumentOpened`
`OnDocumentClosed`
`OnModifiedFlagChanged`
`OnContextChanged`
`OnFileChangedAlert`
`OnActivate`

If the AuthenticDesktopControl is integrated in the Document Level mode each document is displayed in an own object of type `AuthenticDesktopControlDocument`. The `AuthenticDesktopControlDocument` contains only one document at the time but can be reused to display different files one after another.

This object is a complete ActiveX control.

14.4.7.4.1 Properties

The following properties are defined:

`ReadOnly`
`IsModified`
`Path`
`Appearance`
`BorderStyle`

Access to AuthenticDesktopAPI:
`Document`

14.4.7.4.1.1 Appearance

Property: `Appearance` as short

Dispatch Id: *-520*

Description:
A value not equal to `0` displays a client edge around the document control. Default value is `0`.

14.4.7.4.1.2 BorderStyle

Property: `BorderStyle` as short

Dispatch Id: *-504*

Description:
A value of `1` displays the control with a thin border. Default value is `0`.

14.4.7.4.1.3 Document

Property: `Document` as Document

Dispatch Id: *1*

Description:
The `Document` property gives access to the `Document` object of the Authentic Desktop automation server API. This interface provides additional functionality which can be used with the document loaded in the control. The property is read-only.

© 2016-2022 Altova GmbH

14.4.7.4.1.4 IsModified

Property: `IsModified` as boolean (read-only)

Dispatch Id: *1006*

Description:
`IsModified` is *true* if the document content has changed since the last open, reload or save operation. It is *false*, otherwise.

14.4.7.4.1.5 Path

Property: `Path` as string

Dispatch Id: *1005*

Description:
Sets or gets the full path name of the document loaded into the control.

14.4.7.4.1.6 ReadOnly

Property: `ReadOnly` as boolean

Dispatch Id: *1007*

Description:
Using this property you can turn on and off the read-only mode of the document. If `ReadOnly` is `true` it is not possible to do any modifications.

14.4.7.4.2 Methods

The following methods are defined:

Document handling:
`New`
`Open`
`Reload`
`Save`
`SaveAs`

Command Handling:
`Exec`
`QueryStatus`

14.4.7.4.2.1 Exec

Method: `Exec` (`nCmdID` as long) as boolean

Dispatch Id: *8*

Description:
`Exec` calls the Authentic Desktop command with the ID `nCmdID`. If the command can be executed, the method returns `true`. This method should be called only if there is currently an active document available in the application.

To get commands organized according to their menu structure, use the `MainMenu` property of AuthenticDesktopControl. To get toolbar commands, use the `Toolbars` property of the AuthenticDesktopControl.

14.4.7.4.2.2 New

Method: `New` () as boolean

Dispatch Id: *1000*

Description:
This method initializes a new document inside the control.

14.4.7.4.2.3 Open

Method: `Open` (`strFileName` as string) as boolean

Dispatch Id: *1001*

Description:
`Open` loads the file `strFileName` as the new document into the control.

14.4.7.4.2.4 QueryStatus

Method: `QueryStatus` (`nCmdID` as long) as long

Dispatch Id: *9*

Description:
`QueryStatus` returns the enabled/disabled and checked/unchecked status of the command specified by `nCmdID`. The status is returned as a bit mask.

Bit	Value	Name	Meaning
0	1	Supported	Set if the command is supported.

© 2016-2022 Altova GmbH

1	2	Enabled	Set if the command is enabled (can be executed).
2	4	Checked	Set if the command is checked.

This means that if `QueryStatus` returns `0` the command ID is not recognized as a valid Authentic Desktop command. If `QueryStatus` returns a value of `1` or `5` the command is disabled. The client should call the `QueryStatus` method of the document control if there is currently an active document available in the application.

14.4.7.4.2.5 Reload

Method: `Reload` () as boolean

Dispatch Id: *1002*

Description:
`Reload` updates the document content from the file system.

14.4.7.4.2.6 Save

Method: `Save` () as boolean

Dispatch Id: *1003*

Description:
`Save` saves the current document at the location `Path`.

14.4.7.4.2.7 SaveAs

Method: `SaveAs` (`strFileName` as string) as boolean

Dispatch Id: *1004*

Description:
`SaveAs` sets `Path` to *strFileName* and then saves the document to this location.

14.4.7.4.3 Events

The AuthenticDesktopControlDocument ActiveX control provides following connection point events:

`OnDocumentOpened`
`OnDocumentClosed`
`OnModifiedFlagChanged`
`OnContextChanged`
`OnFileChangedAlert`
`OnActivate`
`OnSetEditorTitle`

14.4.7.4.3.1 OnActivate

Event: `OnActivate` ()

Dispatch Id: *1005*

Description:
This event is triggered when the document control is activated, has the focus, and is ready for user input.

14.4.7.4.3.2 OnContextChanged

Event: `OnContextChanged` (`i_strContextName` as String, `i_bActive` as bool) as bool

Dispatch Id: *1004*

Description: None

14.4.7.4.3.3 OnDocumentClosed

Event: `OnDocumentClosed` (`objDocument` as Document)

Dispatch Id: *1001*

Description:
This event is triggered whenever the document loaded into this control is closed. The argument `objDocument` is a `Document` object from the Authentic Desktop automation interface and should be used with care.

14.4.7.4.3.4 OnDocumentOpened

Event: `OnDocumentOpened` (`objDocument` as Document)

Dispatch Id: *1000*

Description:
This event is triggered whenever a document is opened in this control. The argument `objDocument` is a `Document` object from the Authentic Desktop automation interface, and can be used to query for more details about the document, or perform additional operations.

14.4.7.4.3.5 OnDocumentSaveAs

Event: `OnContextDocumentSaveAs` (`i_strFileName` as String)

Dispatch Id: *1007*

Description:

© 2016-2022 Altova GmbH

This event is triggered when this document gets internally saved under a new name.

14.4.7.4.3.6 OnFileChangedAlert

Event: `OnFileChangedAlert` () as bool

Dispatch Id: *1003*

Description:
This event is triggered when the file loaded into this document control is changed on the hard disk by another application. Clients should return true, if they handled the event, or false, if Authentic Desktop should handle it in its customary way, i.e. prompting the user for reload.

14.4.7.4.3.7 OnModifiedFlagChanged

Event: `OnModifiedFlagChanged` (`i_bIsModified` as boolean)

Dispatch Id: *1002*

Description:
This event gets triggered whenever the document changes between modified and unmodified state. The parameter *i_bIsModifed* is *true* if the document contents differs from the original content, and *false*, otherwise.

14.4.7.4.3.8 OnSetEditorTitle

Event: OnSetEditorTitle ()

Dispatch Id: *1006*

Description:
This event is being raised when the contained document is being internally renamed.

14.4.7.5 AuthenticDesktopControlPlaceHolder

Properties available for all kinds of placeholder windows:
`PlaceholderWindowID`

Properties for project placeholder window:
`Project`

Methods for project placeholder window:
`OpenProject`
`CloseProject`

The `AuthenticDesktopControlPlaceHolder` control is used to show the additional Authentic Desktop windows like Overview, Library or Project window. It is used like any other ActiveX control and can be placed anywhere in the client application.

14.4.7.5.1 Properties

The following properties are defined:

PlaceholderWindowID

Access to AuthenticDesktopAPI:
Project

14.4.7.5.1.1 Label

Property: Label as String (read-only)

Dispatch Id: *1001*

Description:
This property gives access to the title of the placeholder. The property is read-only.

14.4.7.5.1.2 PlaceholderWindowID

Property: PlaceholderWindowID as AuthenticDesktopControlPlaceholderWindow

Dispatch Id: *1*

Description:
This property specifies which Authentic Desktop window should be displayed in the client area of the control. The PlaceholderWindowID can be set at any time to any valid value of the AuthenticDesktopControlPlaceholderWindow enumeration. The control changes its state immediately and shows the new Authentic Desktop window.

14.4.7.5.1.3 Project

Property: Project as Project (read-only)

Dispatch Id: *2*

Description:
The Project property gives access to the Project object of the Authentic Desktop automation server API. This interface provides additional functionality which can be used with the project loaded into the control. The property will return a valid project interface only if the placeholder window has PlaceholderWindowID with a value of Authentic DesktopXProjectWindow (=3). The property is read-only.

© 2016-2022 Altova GmbH

14.4.7.5.2 Methods

The following method is defined:

OpenProject
CloseProject

14.4.7.5.2.1 OpenProject

Method: `OpenProject` (`strFileName` as `string`) as `boolean`
Dispatch Id: *3*
Description:
`OpenProject` loads the file `strFileName` as the new project into the control. The method will fail if the placeholder window has a **`PlaceholderWindowID`** different to `Authentic DesktopXProjectWindow (=3)`.

14.4.7.5.2.2 CloseProject

Method: `CloseProject ()`

Dispatch Id: *4*

Description:
`CloseProject` closes the project loaded by the control. The method will fail if the placeholder window has a `PlaceholderWindowID` different to `Authentic DesktopXProjectWindow (=3)`.

14.4.7.5.3 Events

The AuthenticDesktopControlPlaceholder ActiveX control provides following connection point events:

OnModifiedFlagChanged

14.4.7.5.3.1 OnModifiedFlagChanged

Event: `OnModifiedFlagChanged` (`i_bIsModified` as boolean)

Dispatch Id: *1*

Description:
This event gets triggered only for placeholder controls with a `PlaceholderWindowID` of `Authentic DesktopXProjectWindow (=3)`. The event is fired whenever the project content changes between modified and unmodified state. The parameter *i_bIsModifed* is *true* if the project contents differs from the original content, and *false*, otherwise.

14.4.7.5.3.2 OnSetLabel

Event: `OnSetLabel(i_strNewLabel as string)`

Dispatch Id: *1000*

Description:
Raised when the title of the placeholder window is changed.

14.4.7.6 Enumerations

The following enumerations are defined:

`ICActiveXIntegrationLevel`
`AuthenticDesktopControlPlaceholderWindow`

14.4.7.6.1 ICActiveXIntegrationLevel

Possible values for the IntegrationLevel property of the AuthenticDesktopControl.

```
ICActiveXIntegrationOnApplicationLevel  = 0
ICActiveXIntegrationOnDocumentLevel     = 1
```

14.4.7.6.2 AuthenticDesktopControlPlaceholderWindow

This enumeration contains the list of the supported additional Authentic Desktop windows.

```
AuthenticDesktopControlNoToolWnd                    =  -1
AuthenticDesktopControlEntryHelperTopToolWnd        =   0
AuthenticDesktopControlEntryHelperMiddleToolWnd     =   1
AuthenticDesktopControlEntryHelperBottomToolWnd     =   2
AuthenticDesktopControlValidatorOutputToolWnd       =   3
AuthenticDesktopControlProjectWindowToolWnd         =   4
AuthenticDesktopControlInfoToolWnd                  =  18
```


© 2016-2022 Altova GmbH

15 Annexes

Ces annexes contiennent des informations techniques concernant Authentic Desktop et des informations importantes concernant la licence. Chaque annexe contient les sous-sections suivantes :

Données techniques

- SE et exigences de mémoire
- Parseur XML Altova
- Moteurs XSLT et XQuery Altova
- Prise en charge Unicode
- Utilisation Internet

Informations de licence

- Distribution électronique de logiciel
- Droits de la propriété intellectuelle et copyright
- Contrat de licence de l'utilisateur final

15.1 Données techniques

Cette section contient des informations utiles concernant certains aspects techniques de votre logiciel. Cette information est organisée dans les sections suivantes :

- SE et exigences de mémoire
- Moteurs Altova
- Prise en charge Unicode
- Utilisation Internet

15.1.1 SE et exigences de mémoire

Système d'exploitation

Les applications logicielles d'Altova sont disponibles pour les plateformes suivantes :

- Windows 7 SP1 avec mise à jour de la plateforme, Windows 8, Windows 10, Windows 11
- Windows Server 2008 R2 SP1 avec mise à jour de la plateforme ou plus récent

Mémoire

Étant donné que le logiciel est rédigé en C++, il ne nécessite pas la performance d'un Java Runtime Environment et nécessite généralement moins de mémoire que les applications basées sur Java comparables. Néanmoins, chaque document est chargé entièrement dans la mémoire de manière à ce qu'il puisse être parsé complètement et pour améliorer la vitesse de la consultation et de l'édition. Les exigences de mémoire augmentent avec la taille du document.

Les exigences de mémoire sont aussi affectées par un historique de la fonction Annuler non limité. Si vous coupez/collez sans arrêt de larges sélections dans des documents volumineux, la mémoire disponible peut baisser rapidement.

15.1.2 Moteurs Altova

Validateur XML

Lors de l'ouverture d'un document XML, l'application utilise son validateur XML intégré pour vérifier la bonne formation, pour valider le document par rapport à un schéma (si spécifié), et de générer des arborescences et infosets. Le validateur XML est aussi utilisé pour fournir une édition intelligente pendant que vous éditez des documents et pour afficher dynamiquement toute erreur de validation qui peut se produire.

Le validateur XML intégré met en place les spécifications de la Final Recommendation of the W3C's XML Schema 1.0 et 1.1. Les nouveaux développements recommandés par le Groupe de travail de Schéma XML de W3C sont incorporés en continu dans le validateur XML, afin que les produits Altova vous donnent un environnement de développement de pointe.

© 2016-2022 Altova GmbH

Moteurs XSLT et XQuery

Les produits Altova utilisent les moteurs Altova XSLT 1.0, 2.0 et 3.0, et les moteurs Altova XQuery 1.0 et 3.1. Si un de ces moteurs est inclus dans le produit, la documentation concernant le comportement spécifique à la mise en place pour chaque moteur est indiquée dans les annexes de la documentation.

Note: Altova MapForce génère du code utilisant les moteurs XSLT 1.0, 2.0 et XQuery 1.0.

15.1.3 Prise en charge Unicode

Les produits XML d'Altova propose une prise en charge complète d'Unicode. Pour éditer un document XML, vous devrez utiliser une police d'écriture qui sera également prise en charge par ce document.

Veuillez noter que la plupart des polices ne contient qu'un sous-ensemble très spécifique de la plage Unicode et qu'elles sont donc généralement taillées à la mesure du système d'écriture correspondant. Si vous voyez apparaître un texte déformé, une des raisons peut être que la police que vous avez sélectionnée ne contient pas les symboles exigés. Il est donc utile de disposer d'une police qui couvre toute la plage Unicode, surtout lors de l'édition de documents XML dans des langues ou des systèmes d'écriture variés. Une police Unicode typique utilisée sur les PC Windows est Arial Unicode MS.

Dans le dossier `/Examples` de votre dossier d'application, vous trouverez un fichier XHTML appelé `UnicodeUTF-8.html` qui contient la phrase suivante dans plusieurs langues et systèmes d'écriture :

- *When the world wants to talk, it speaks Unicode*
- *Quand le monde veut communiquer, il parle en Unicode*
- *Wenn die Welt miteinander spricht, spricht sie Unicode*
- 世界的に話すなら、Unicode です。

Ouvrez ce fichier XHTML pour voir un aperçu des possibilités d'Unicode et pour indiquer que les systèmes d'écriture sont pris en charge par les polices disponibles sur votre PC.

15.1.4 Utilisation Internet

Les applications Altova initieront des connexions Internet pour vous dans les situations suivantes :

- Si vous cliquez sur "Demander code-clé d'évaluation" dans le dialogue d'Enregistrement (**Aide | Activation du logiciel**), les trois champs dans le dialogue d'enregistrement seront transférés vers notre serveur web au moyen d'une connexion http (port 80) normale et le code-clé d'évaluation sera renvoyé au client par le biais d'un e-mail SMTP normal.
- Dans certains produits Altova, vous pouvez ouvrir un fichier dans Internet (**Fichier | Ouvrir | Passer à URL**). Dans ce cas, le document est extrait à l'aide d'une des méthodes de protocole et connexions suivantes : HTTP (normalement port 80), FTP (normalement port 20/21), HTTPS (normalement port 443). Vous pouvez aussi exécuter un serveur HTTP sur le port 8080. (Dans le dialogue URL, spécifier le port après le nom de serveur et un double point.)
- Si vous ouvrez un document XML qui réfère à un Schéma XML ou à un DTD et que le document est spécifié par une URL, le document de schéma référencé est aussi extrait par le biais d'une connexion HTTP (port 80) ou un autre protocole spécifié dans l'URL (voir Point 2 ci-dessus). Un document de

schéma sera aussi extrait lorsqu'un fichier XML est validé. Veuillez noter que la validation peut avoir lieu automatiquement lors de l'ouverture d'un document si vous avez instruit l'application de procéder de cette manière (dans l'onglet Fichier du dialogue Options (**Outils | Options**)).

- Dans les applications Altova utilisant WSDL et SOAP, les connexions de service web sont définies par les documents WSDL.
- Si vous utilisez la commande **Envoyer par courrier électronique** (**Fichier | Envoyer par courrier électronique**) dans XMLSpy, la sélection ou le fichier actuel est envoyé avec tout programme d'e-mail conforme à MAPI et installé sur le PC de l'utilisateur.
- Fait partie de l'Activation du logiciel et du LiveUpdate tel que décrit ultérieurement dans l'Accord de licence de logiciel Altova.

© 2016-2022 Altova GmbH

15.2 Informations de licence

Cette section contient des informations concernant :

- la distribution de ce logiciel
- le contrat de licence régissant l'usage de ce logiciel

Veuillez lire ces informations attentivement. Elles ont force obligatoire puisque vous avez accepté ces termes lors de l'installation de ce logiciel.

Pour consulter les termes de toute licence Altova, rendez-vous sur la page des informations juridiques Altova sur le site web Altova.

15.2.1 Distribution électronique de logiciel

Ce produit est disponible par le biais de la distribution électronique de logiciel, une méthode de distribution qui fournit les avantages uniques suivants :

- Vous pouvez évaluer gratuitement le logiciel pendant 30 jours avant de vous décider à l'achat. (*Note : Altova MobileTogether Designer dispose d'une une licence gratuite.*)
- Une fois que vous avez décidé d'acheter le logiciel, vous pouvez passer vos commandes en ligne sur le site web Altova et vous obtiendrez en quelques minutes un produit bénéficiant d'une pleine licence.
- Lorsque vous passez une commande en ligne, vous disposerez toujours de la dernière version de nos logiciels.
- Le pack de produits comprend une aide sur écran qui peut être accédé depuis l'intérieur de l'interface de l'application. La dernière version du manuel d'utilisateur est disponible sous www.altova.com (i) sous format HTML pour une navigation en ligne, et (ii) sous format PDF pour le téléchargement (et pour imprimer si vous préférez avoir recours à une documentation en papier).

Période d'évaluation de 30 jours

Après avoir téléchargé le produit, vous pourrez évaluer celui-ci gratuitement pour une période de jusqu'à 30 jours. Au bout d'environ 20 jours de cette période d'évaluation, le logiciel commencera à vous rappeler qu'il n'est pas encore sous licence. Le message de rappel s'affichera une fois à chaque fois que vous démarrerez l'application. Si vous souhaitez continuer à utiliser le programme à l'issue de la période d'évaluation de 30 jours, vous devrez acheter une licence de produit, qui est fournie sous la forme d'un fichier de licence contenant un code-clé. Déverrouiller le produit en chargeant le fichier de licence dans le dialogue d'activation du logiciel de votre produit.

Vous pouvez acheter des licences de produit dans la boutique en ligne du https://shop.altova.com/.

Transmettre le logiciel à d'autres collaborateurs dans votre entreprise à des fins d'évaluation

Si vous souhaitez distribuer la version d'évaluation dans le cadre de votre réseau d'entreprise, ou si vous prévoyez de l'utiliser sur un PC qui n'est pas connecté à Internet, vous pourrez uniquement distribuer le fichier d'installation, à condition qu'il ne soit pas modifié de quelque manière que ce soit. Toute personne accédant au programme d'installation du logiciel que vous avez fourni doit demander son propre code-clé d'évaluation de 30 jours et devra aussi acheter une licence à l'issue de la période d'évaluation afin de pouvoir continuer à utiliser le produit.

15.2.2 Altova Contrat de licence de l'utilisateur final pour Authentic

- Le contrat Altova de licence de l'utilisateur final pour Authentic est disponible ici : https://www.altova.com/legal/authentic-eula
- La politique de confidentialité d'Altova est disponible ici : https://www.altova.com/privacy

© 2016-2022 Altova GmbH

Index

© 2016-2022 Altova GmbH

© 2016-2022 Altova GmbH

B

C

D

© 2016-2022 Altova GmbH

© 2016-2022 Altova GmbH

E

© 2016-2022 Altova GmbH

F

G

H

I

J

L

© 2016-2022 Altova GmbH

M

N

O

© 2016-2022 Altova GmbH

P

R

© 2016-2022 Altova GmbH

S

© 2016-2022 Altova GmbH

T

U

© 2016-2022 Altova GmbH

V

W

X

© 2016-2022 Altova GmbH

© 2016-2022 Altova GmbH