Altova Authentic 2024 Browser Edition

Benutzer- und Referenzhandbuch

Altova Authentic 2024 Browser Edition
Benutzer- und Referenzhandbuch

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: 2024

© 2018-2024 Altova GmbH

Inhaltsverzeichnis

1 Einführung

Mit der **Altova® Authentic® 2024 Browser Edition** können Benutzer XML- und Datenbank-Content über eine Benutzeroberfläche bearbeiten, die einen Textverarbeitungsprogramm sehr ähnlich sieht. Die Browser Edition kann in jede beliebige Webseite eingebettet werden und ermöglicht die Bearbeitung direkt über den Browser. Die Authentic Browser Edition steht als Plug-in für **Microsoft Internet Explorer** zur Verfügung.

Authentic Browser steht in Form einer Trusted und einer Untrusted Version[10] zur Verfügung zur Verfügung. Für beide Varianten ist eine Lizenz erforderlich.

© 2018-2024 Altova GmbH

1.1 Vorteile von Authentic Browser

Die Authentic Browser Lösung zur Bearbeitung von XML-Dateien weist eine Reihe von Vorteilen auf. Einige davon sind unten aufgelistet:

- Ermöglicht mehreren Benutzern den Zugriff und die Bearbeitung eines XML-Dokuments über ihre Browser. Derzeit wird **Microsoft Internet Explorer 5.5 oder höher** unterstützt. Die 64-Bit-Version von Internet Explorer 10 und 11 wird nicht unterstützt.
- Vereinfacht das firmenweite Deployment und die Wartung von Applikationen enorm und senkt gleichzeitig die Gesamtinvestitionskosten.
- Basiert auf offenen Standards wie XML Schema und XSLT.
- Ist vollständig Unicode-kompatibel.
- Verwendet die Altova Authentic-Ansicht, die auf dem gebräuchlichen Internet Explorer Browser oder Firefox basiert. In der Authentic-Ansicht können XML-Dateien auf WYSIWYG-Art bearbeitet werden, d.h. der Benutzer muss den zugrunde liegenden XML-Code nicht anzeigen lassen.
- Benötigt keine weitere Software, da das Authentic Browser Plug-in ein Browser Add-on ist.
- Authentic Browser ist ein ActiveX Control, wobei die COM-Schnittstelle über das Authentic 69 Objekt definiert wird. Das vollständige Objektmodell ist im Abschnitt Benutzerreferenz: Objekte 69 dieser Dokumentation beschrieben.

1.2 Funktionsweise

Um ein Authentic Browser-Projekt zu implementieren, benötigen Sie einen Server und einen oder mehrere Client Computer, die mit dem Server verbunden sind.

Authentic Browser Server

Der Authentic Browser Server hat die folgenden Funktionen:

- Auf dem Server sind die Dateien gespeichert, die im Zusammenhang mit dem zu bearbeitenden Altova Authentic XML-Dokument stehen. Es sind dies die folgenden Dateien:

  1. Das XML-Schema (XSD), auf dem das editierbare XML-Dokument basiert;
  2. Das zu bearbeitende XML-Dokument;
  3. Die SPS- oder PXF-Datei, mit der das Layout und die Eingabemechanismen für das XML-Dokuments in der Authentic-Ansicht festgelegt sind.

- Auf dem Server ist die HTML-Seite für das Authentic Plug-in[22] gespeichert. Über diese HTML-Seite greifen Sie auf die Authentic-Ansicht zu und bearbeiten sie. Sie enthält die Anweisungen für den Zugriff auf das XML-Dokument und bildet den Behälter für das Authentic-Ansichtsfenster, in das das XML-Dokument zur Bearbeitung geladen wird. Um diese Seite aufzurufen, muss ihre URL in den Client Browser eingegeben werden.
- Wenn Sie mit der Enterprise Edition von Authentic Browser arbeiten, muss auf dem Server die Lizenz für die Enterprise Edition gespeichert sein. Enterprise-Lizenzen werden für einen oder mehrere bestimmte Server ausgestellt.
- Falls Sie eine On-Demand-Installation des Authentic Browser Plug-in planen, wird/werden auf dem Server das/die Authentic Browser Erweiterungspaket(e) (CAB-Dateien) zum Downloaden und Installieren des Plug-in auf Client-Rechnern gespeichert. (Andernfalls wird das Plug-in direkt auf Client-Rechnern installiert)

Eine Beschreibung der einzelnen Schritte zum Einrichten des Servers finden Sie im Abschnitt Einrichten des Servers[13].

Authentic Browser Clients

Ein XML-Dokument kann in der Altova Authentic-Ansicht angezeigt und bearbeitet werden, wenn dem XML-Dokument eine SPS- oder PXF-Datei zugewiesen wurde. Die Bearbeitung in der Authentic-Ansicht erfolgt auf den Client-Rechnern, die folgendermaßen eingerichtet werden müssen:

- Auf jedem Client-Rechner muss **Internet Explorer 5.5 oder höher (32-Bit oder 64-Bit)** installiert sein.
- Falls keine On-Demand-Installation geplant ist, muss das Authentic Browser Plug-in direkt auf dem Client-Rechner installiert sein.

Eine Beschreibung der einzelnen Schritte zum Einrichten des Client finden Sie im Abschnitt Einrichten des Client[39].

Funktionsweise von Authentic Browser

Nachdem der Server und die Clients wie oben beschrieben eingerichtet wurden, gibt der Benutzer die URL der HTML-Seite für das Authentic Plug-in in den Client Browser ein. Wenn das Plug-in noch nicht auf dem Client

© 2018-2024 Altova GmbH

Browser installiert wurde, kann die HTML-Seite eine Anleitung zur Durchführung einer On-Demand-Installation des Plug-in im Client Browser enthalten.

Nachdem das Plug-in auf dem Client installiert wurde, öffnet der Code auf der HTML-Seite im Browser-Fenster ein Authentic-Ansichts-Bearbeitungsfenster. Das zu bearbeitende XML-Dokument wird vom Server aus in das Authenticsfenster geladen und der Benutzer kann mit der Bearbeitung beginnen und die Änderungen direkt im XML-Dokument speichern.

© 2018-2024 Altova GmbH

1.3 Authentic Browser Versionen

Die Authentic Browser-Versionen stehen gemäß den folgenden Kriterien zur Verfügung:

- *Sprache:* Englisch (EN), Deutsch (DE), Spanisch (ES), Französisch (FR), Japanisch (JA)
- *Trusted/Untrusted:* Trusted, Untrusted
- *Client Browser:* Microsoft Internet Explorer (32-Bit und 64-Bit). Siehe auch Browser-Anforderungen 40.

Für jede unterstützte Sprache (Englisch, Deutsch, Spanisch, Französisch, Japanisch) gibt es jeweils eine Trusted und Untrusted Version für Microsoft Internet Explorer 32-Bit und 64-Bit. Sie können eine, einige oder alle dieser Versionen des Authentic Browser Plug-in auf einem Client-Rechner installieren. Jede Version wird als separates Plug-in im Add-on Manager des Browsers angezeigt.

Die verschiedenen Versionen für Englisch (EN), Deutsch (DE), Spanisch (ES), Französisch (FR) und Japanisch (JA) sind weiter unten mit ihren Class IDs (für CAB-Dateien) aufgelistet. Sie müssen die entsprechende Class ID auf der HTML-Seite für das Authentic Plug-in 22 definieren.

- ***CAB-Dateien und ihre Class IDs (identische Class IDs für 32-Bit und 64-Bit Internet Explorer)***

EN	Trusted	B4628728-E3F0-44a2-BEC8-F838555AE780
EN	Untrusted	A5985EA9-3332-4ddf-AD7F-F6E98BFEAF94
DE	Trusted	91DDF44A- DFD1- 4F47- 8EE3- 4CBE874584F7
DE	Untrusted	28A640E8- EAEE- 4B5D- BEBE- BFA956081E66
ES	Trusted	23B503E7-269B-45CE-BAB2-22AA97BED8E2
ES	Untrusted	8AD3EF86-AC1E-4574-8C13-DE5B6CBECEBE
FR	Trusted	1B768F46-A9E8-4a88-91B0-2917FE47612A
FR	Untrusted	070F4B50-26F7-48cc-9AC1-52D562C1E749
JA	Trusted	5B15DB5A-1720-4264-BB65-70C3F7A860DA
JA	Untrusted	4B9512D2-A3D3-46e3-82C1-34248BBDCE58

Je nach Anforderung können Sie eine oder mehrere dieser Versionen von der Altova Website herunterladen.

Beachten Sie zu den verschiedenen Authentic Browser Versionen die folgenden Punkte:

- Alle Versionen sind Unicode-Versionen und jede davon bietet vollständige Unterstützung für mehrere Zeichensätze im XML-Dokument. Unicode-Versionen werden auf den folgenden Client-Arbeitsrechnern unterstützt: Windows 10, Windows 11.
- Es gibt zwar für die 32-Bit- und die 64-Bit-Version von Internet Explorer je eine separate CAB-Datei, doch die Class IDs der beiden .CAB-Dateien (für die 32-Bit- und für die 64-Bit IE Browser Version) sind identisch und werden in der obigen Tabelle für die veschiedenen EN/DE/ED/JA-Versionen der Trusted und der Untrusted Version angegeben.
- Die Trusted Version, erlaubt keinen Zugriff auf lokale Dateien und ist daher als "safe for scripting" gekennzeichnet. Sie kann in einem Browser-basierten Szenario verwendet und von jeder Webseite aus ohne Sicherheitswarnungen durch den Client aufgerufen werden.

© 2018-2024 Altova GmbH

- Die Untrusted Version ist für die Verwendung im Intranet gedacht oder für die Verwendung der Authentic Browser Edition als ActiveX Control in Ihrer Applikation. Sie bietet Zugriff auf lokale Dateien und ist daher **nicht** als "safe for scripting" gekennzeichnet. Wenn Sie versuchen, diese Version in einem Browser-Fenster zu verwenden, wird der Benutzer um Erlaubnis gefragt. Sie können selbst entscheiden, ob ActiveX Controls aktiviert oder deaktiviert werden sollen oder ob Sie vom Browser gefragt werden sollen, ob ActiveX Controls aktiviert werden sollen (nähere Informationen dazu finden Sie im browserspezifischen Abschnitt: Internet Explorer 9 44).

Hinweise: Informationen zur Installation und Konfiguration Ihres Browser-Diensts (z.B. Internet Information Services von Microsoft) finden Sie in der Dokumentation des jeweiligen Anbieters.

1.4 Info zu dieser Dokumentation

Diese Dokumentation ist das Handbuch zum Authentic Browser Plug-in und in die folgenden Abschnitte gegliedert:

- eine Einführung[6] zu (i) den Vorteilen der Verwendung von Authentic Browser[7]; (ii) der Funktionsweise von Authentic Browser[8]; sowie (iii) den verschiedenen Authentic Browser-Versionen[10].
- einen Abschnitt Einrichten des Servers[13], in dem beschrieben wird, wie Sie einen Server für ein Authentic Browser-Projekt einrichten. Dieser Abschnitt enthält eine ausführliche Beschreibung der HTML-Seite für das Authentic Plug-in[22] und einen Abschnitt, in dem erklärt wird, wie Sie die Authentic Browser-Erweiterungspakete für die On-Demand-Installation[38] verwenden.
- Einen Client[39]-Abschnitt, der einen Abschnitt über die verschiedenen Arten der Installation des Authentic Browser Plug-in in Client Browsern[41] enthält.
- einen aus drei Teilen bestehenden Referenzabschnitt (Benutzerreferenz: Mechanismen[48], Benutzerreferenz: Objekte[69] und Benutzerreferenz: Enumerationen[191]). Hier werden die Mechanismen, Objekte und Enumerationen beschrieben, mit Hilfe derer Sie die Authentic-Ansicht in Authentic Browser erstellen können.

Weitere Dokumentation zu diesem Thema

Zu diesem Thema gibt es zwei weitere Handbücher von Altova:

- Informationen zum Erstellen von StyleVision Power Stylesheets (SPSs) finden Sie auf der Altova Website in der Dokumentation zum Altova Produkt StyleVision. Ein SPS ist die Datei, in der definiert ist, wie die Authentic-Ansicht eines XML-Dokuments aussieht. Diese Dokumentation ist relevant, wenn Sie die Authentic-Ansichtsseite für die XML-Bearbeitung erstellen.
- Dokumentation zur Verwendung der Authentic-Ansicht. Wenn Sie die Authentic-Ansicht verwenden, lesen Sie im Authentic Desktop-Benutzerhandbuch das Tutorial zur Authentic-Ansicht und den Abschnitt zur Verwendung. Dieses Handbuch steht auf derAltova Website zur Verfügung.

© 2018-2024 Altova GmbH

2 Einrichten des Servers

Um Authentic Browser auf Ihrem Server zu installieren, gehen Sie folgendermaßen vor:

Konfigurieren des Browser-Diensts

Installieren und konfigurieren Sie den Browser-Dienst 15 Ihres Servers. Wenn Sie Internet Information Services (IIS) von Microsoft verwenden, beachten Sie, dass bei der Installation ein Standardverzeichnis mit dem Namen `Inetpub` sowie Unterverzeichnisse angelegt werden. Das Root-Verzeichnis des Servers wäre dann: `//Inetpub/wwwroot`. Dies ist das Verzeichnis, das Sie mit der IP-Adresse des Servers erreichen, wenn Sie (in Ihrer Browser Service-Konfiguration) kein anderes als das Root-Verzeichnis angeben. Nähere Informationen zum Installieren und Konfigurieren Ihres Browser-Diensts finden Sie in der Dokumentation Ihres Anbieters.

Einrichten der XSD-, XML- und SPS/PXF-Datei

Mit Hilfe der SPS-Datei kann das XML-Dokument in editierbarem Format in der Authentic-Ansicht angezeigt werden. Es basiert auf einer XML-Schema (XSD-Datei) und wird in StyleVision von Altova erstellt. Die SPS-Datei muss zusammen mit der XSDL-Datei und dem zu bearbeitenden XML-Dokument unter einem Netzwerkpfad gespeichert sein, auf den alle Client-Rechner Zugriff haben (normalerweise ist dies der Authentic Browser Server). Nähere Informationen zur Einrichtung der Dateien auf dem Server finden Sie im Abschnitt XSD-, XML- und SPS/PXF-Datei 19.

Erstellen der HTML-Seite für das Authentic Plug-in

Über die HTML-Seite für das Authentic Plug-in 22 können Sie zur Bearbeitung auf die Authentic-Ansicht zugreifen. Sie enthält Anweisungen, wie das XML-Dokument aufgerufen wird und dient als Behälter für das Authentic-Ansichtsfenster, in das das XML-Dokument zur Bearbeitung geladen wird. Um diese Seite aufzurufen, wird ihre URL in den Client Browser eingegeben. Diese HTML-Seite muss korrekt erstellt und auf dem Server gespeichert werden. Eine Anleitung zur Erstellung der HTML-Seite finden Sie im Abschnitt HTML-Seite für das Authentic Plug-in 22.

Speichern von Erweiterungspaketen für die On-Demand-Installation

Laden Sie Authentic Browser (eine gezippte CAB-Datei) von der Altova Website in einen beliebigen Ordner auf dem Server herunter. Wenn Sie die Enterprise Edition von Authentic Browser verwenden, muss das Paket auf dem Server gespeichert werden, für den die Enterprise Lizenz registriert wurde. **Entpacken Sie diese Datei nicht.** Auf der Website steht für jede Sprachversion (Englisch, Deutsch, Spanisch, Französisch,Japanisch) jeweils eine Trusted und eine Untrusted Version von Authentic Browser in vier Formaten (CAB 32-Bit, CAB 64-Bit) zur Verfügung. Informationen zur Auswahl der Version 10 finden Sie in den Unterabschnitten dieses Abschnitts.

Hinweis:

Bevor Sie Authentic Browser installieren, vergewissern Sie sich bitte, dass nicht gerade eine frühere Version von Authentic Browser ausgeführt wird, da die neue Version sonst eventuell nicht korrekt registriert wird, was zu einer fehlerhaften Installation führen kann. Falls dies passiert, registrieren Sie das Plugin durch Ausführen von: `regsvr32 C:\Windows\Downloaded Program`

`Files\AuthenticPlugin.dll`. (Beachten Sie bitte, dass Sie zum Ausführen dieses Programms (`regsvr32.exe`) Administratorrechte benötigen.)

© 2018-2024 Altova GmbH

2.1 IIS: Konfigurieren des Browser-Diensts

Internet Information Services (IIS) 6 von Microsoft unterstützt nur Dateitypen, die in den MIME Types für die jeweilige Site (Website oder Ordner) definiert sind. Benötigte Dateitypen müssen daher zur Liste der MIME Types für diese Site hinzugefügt werden.

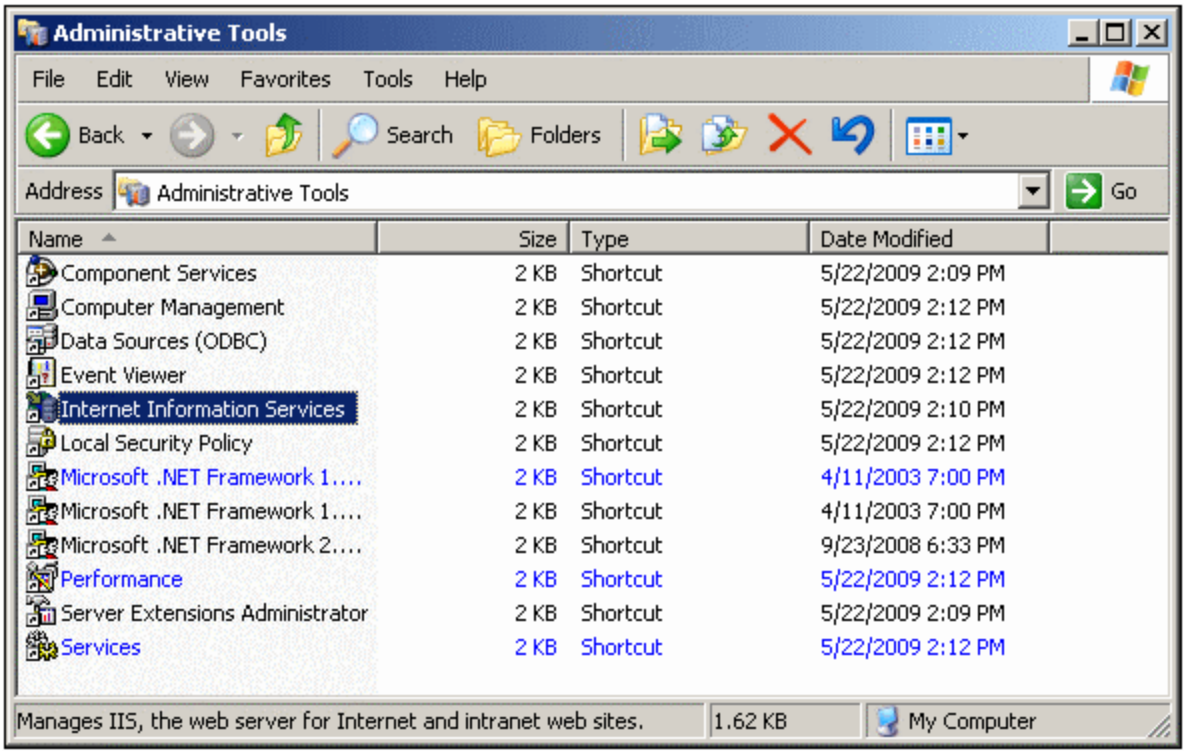
Für die Arbeit mit Authentic Browser werden die folgenden Dateitypen benötigt. Diese müssen hinzugefügt werden:

Dateierweiterung	MIME Type	Kommentar
xsd	text/plain	
sps	text/plain	
pxf	application/x-zip-compressed	

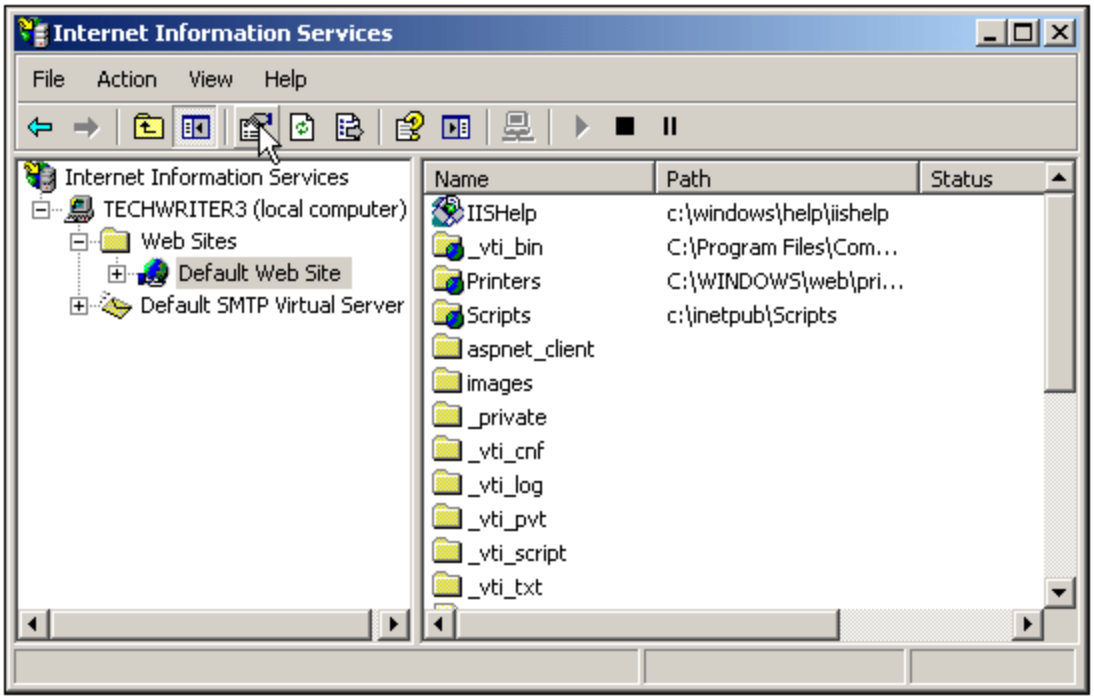
Hinzufügen von MIME Types für eine Site in Internet Information Services

Um auf einem Windows XP-Betriebssystem zur Liste der MIME Types für eine bestimmte Website einen MIME Type hinzuzufügen, gehen Sie folgendermaßen vor. Die Vorgehensweise ist auch auf anderen unterstützten Systemen (Windows 10, Windows 11) ähnlich.

1. Öffnen Sie die Systemsteuerung und doppelklicken Sie auf "Verwaltung".
2. Doppelklicken Sie anschließend im angezeigten Ordner auf Internet Information Services (*Abbildung unten*).

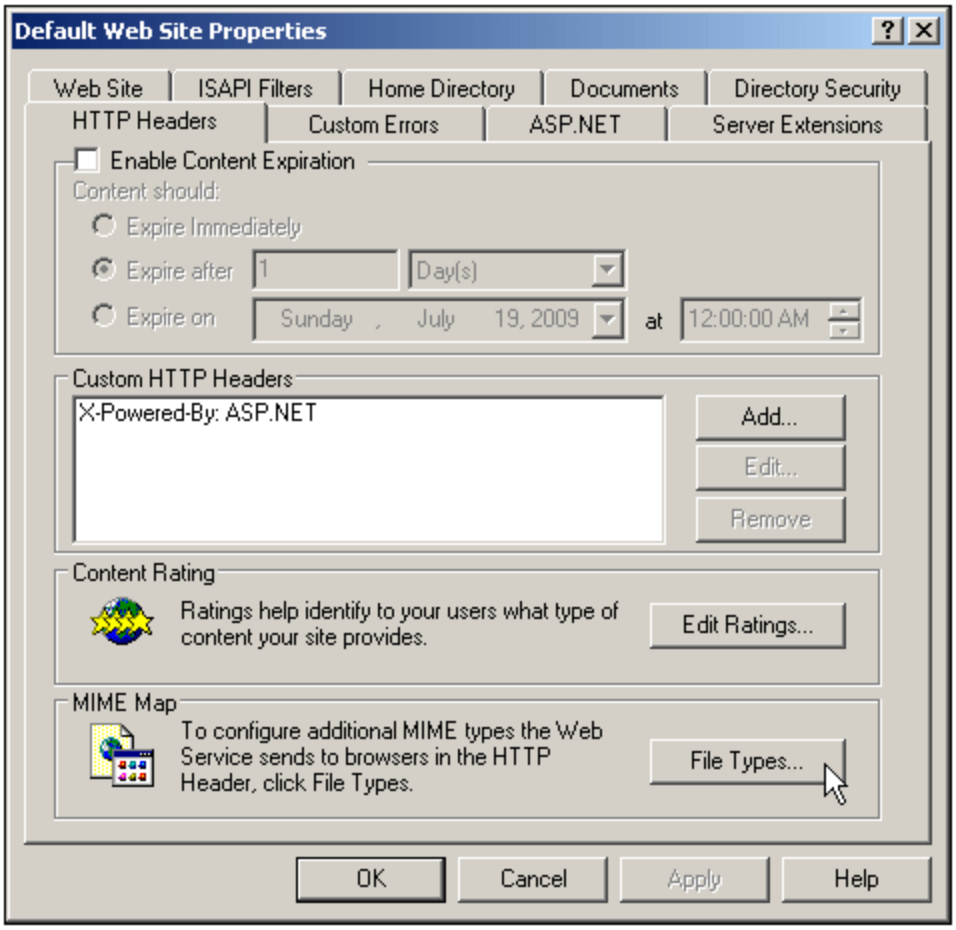


3. Wählen Sie im Ordner "Internet Information Services (IIS)" (*Abbildung unten*) zuerst im Ordnerbereich die gewünschte Site aus (Website oder Ordner) und klicken Sie anschließend auf die Schaltfläche **Eigenschaften** (*in der Abbildung unter dem Cursor*) oder im Kontextmenü, das durch Rechtsklick aufgerufen wird, auf den Befehl **Eigenschaften**.

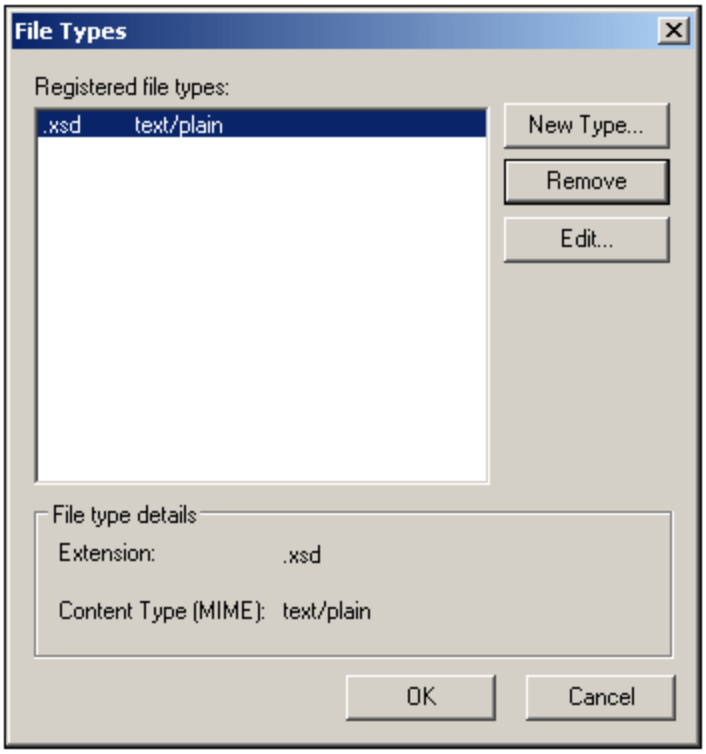


 © 2018-2024 Altova GmbH

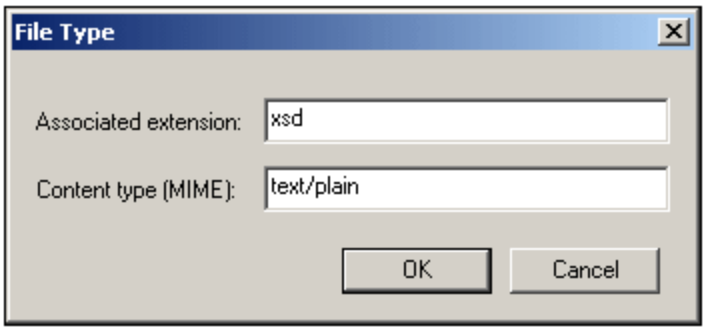
4. Klicken Sie im Dialogfeld "Eigenschaften" auf dem Register "HTTP Headers" i Bereich "MIME-Zuordnungen" auf die Schaltfläche **Dateitypen** (*siehe Abbildung unten*).



5. Klicken Sie im Dialogfeld "Dateitypen" (*Abbildung unten*) auf die Schaltfläche **Neuer Typ...**.



6. Geben Sie im daraufhin angezeigten Dialogfeld die gewünschte Erweiterung und ihren MIME Type ein. Informationen zu den erforderlichen Dateitypen und ihren entsprechenden MIME Types finden Sie in der Tabelle oben.



7. Bestätigen Sie Ihre Auswahl mit **OK**.

Eine Beschreibung, wie man über Web-DAV Fernzugriff auf einen Ordner auf einem internen Server einrichten kann, finden Sie in diesem Windows IT Pro-Artikel.

© 2018-2024 Altova GmbH

2.2 XSD-, XML- und SPS/PXF-Datei

Hauptsächlich wird ein Authentic Browser Plug-in Projekt implementiert, um ein XML-Dokument in der Authentic-Ansicht anzeigen und bearbeiten zu können. In diesem Format wird der XML-Markup-Code eines XML-Dokuments ausgeblendet und Sie können das Dokument auf einer benutzerdefinierten WYSIWYG-Benutzeroberfläche bearbeiten. Das Layout und die Dateneingabemechanismen für ein XML-Dokument in der Authentic-Ansicht werden in einer SPS-Datei definiert. Daher müssen die SPS-Datei, die XML-Datei und die XML-Schema-Datei (XSD-Datei), auf der das SPS und das XML-Dokument basieren auf dem Netzwerk gespeichert sein, damit Sie vom Authentic Browser Client Zugriff darauf haben.

SPS-Dateien

Eine SPS-Datei wird in StyleVision von Altova erstellt. Sie basiert auf demselben XML-Schema wie die XML-Datei. In StyleVision kann das SPS mittels Drag-and-Drop und Seitenlayoutfunktionen erstellt werden. In der SPS-Datei werden das Layout und die Dateneingabemechanismen für das in der Authentic-Ansicht angezeigte XML-Dokument festgelegt.

Der Pfad der XML-, SPS- und XSD-Datei wird auf der HTML-Seite für das Authentic Plug-in[22] definiert. Die XML-Datei wird in das Authentic-Ansichtsfenster (in die HTML-Seite) geladen und gemäß dem in der SPS-Datei definierten Design angezeigt. Die XML-Datei und die SPS-Datei basieren auf demselben XML-Schema (wobei die XML-Datei auf einem Teil des größeren Schemas, das die SPS-Datei verwendet, basieren kann).

Anmerkung: Damit Authentic Browser korrekt funktioniert, muss in der SPS-Datei **unbedingt ein Schema als Hauptschema zugewiesen worden sein**. Nähere Informationen zum Erstellen von StyleVision Power Stylesheets (SPSs) finden Sie im Benutzerhandbuch des Altova-Produkts "StyleVision". Eine Beschreibung dazu finden Sie auch auf der Altova Website

XML-Dateien

Der Benutzer der Authentic-Ansicht bearbeitet das XML-Dokument auf einem Client-Rechner durch Aufrufen der HTML-Seite für das Authentic Plug-in[22]. Diese Seite enthält ein Authentic-Ansichtsfenster, in das die HTML-Datei geladen wird. Der Benutzer der Authentic-Ansicht bearbeitet das Dokument im Client Browser und speichert seine Änderungen wieder im XML-Dokument.

XSD-Dateien

Die XSD-Datei dient zu zwei Zwecken: zum Validieren des XML-Dokuments und zur Bereitstellung der allgemeinen Dokumentstruktur, auf der das SPS basiert. Sowohl die XML- als auch die SPS-Datei referenzieren die XSD-Datei. Daher muss auch die XSD-Datei unter dem von der XML- und der SPS-Datei referenzierten Pfad gespeichert sein.

Pfad zur XSD-, XML- und SPS/PXF-Datei

Die XSD-, die XML- und die SPS- (oder PXF)-Datei müssen unter einem Netzwerkpfad gespeichert sein, der von allen Client-Rechnern aus aufgerufen werden kann, d.h. am besten auf dem Authentic Browser Server. Am besten, alle diese Dateie werden in einem einzigen Ordner auf dem Server gespeichert und die Referenzen innerhalb dieser Dateien aufeinander über relative Pfade definiert. So sollte z.B. die Referenz auf die XSD-Datei in der XML-Datei als relativer Pfad gespeichert werden.

Anmerkung zu datenbankbasierten SPS

Falls Sie Authentic Browser verwenden wollen, um Datenbanken (DB) über ein Datenbank-basiertes StyleVision Power Stylesheet (SPS) aufzurufen oder zu editieren, müssen Sie die folgenden Einstellungen vornehmen, um sicherzustellen, dass der Client eine Verbindung zur Datenbank aufbauen kann.

Connection Informationen im SPS: Alle für die Herstellung der Verbindung zur Datenbank erforderlichen Informationen sind in einem Connection String im SPS gespeichert. Der Connection String im SPS wird bei der Erstellung des StyleVision Power Stylesheet in StyleVision erstellt. Das Verfahren zur Herstellung einer Verbindung zu MS Access Datenbank unterscheidet sich von den für andere Datenbanken verwendeten Verfahren. Für andere Datenbanken wird eine ADO-Verbindung verwendet. Im Folgenden sind die Einstellungen beschrieben, die Sie für diese beiden Verbindungsarten vornehmen müssen.

- **Für MS Access-Datenbanken:** Im Connection String für MS Access Datenbanken muss ein UNC-Pfad verwendet werden, damit der Client eine Verbindung zur Datenbank aufbauen kann. Dieser UNC-Pfad wird beim Erstellen des StyleVision Power Stylesheet in StyleVision definiert. Sie müssen allerdings sicherstellen, dass der Ordner, in dem die Datenbank gespeichert ist (oder ein übergeordneter Ordner) freigegeben wurde. (In Windows XP können Sie diese Einstellungen durch Rechtsklick auf den Ordner und Auswahl des Befehls "**Freigabe und Sicherheit...**" vornehmen). Außerdem müssen Sie auf diesem Rechner die erweiterte Einstellung zur Dateifreigabe aktivieren (Klicken Sie auf **Arbeitsplatz | Extras | Ordneroptionen** und deaktivieren Sie das Kontrollkästchen "einfache Datenfreigabe verwenden"). Auf den Clients sind keine speziellen Einstellungen erforderlich.

  Hinweis: Das Format für den im Connection String angegebenen UNC-Pfad ist: `\\servername\sharename\path\file.mdb`, wobei `servername` der Name des Servers ist, `sharename` der Name des freigegebenen Ordners (der in den Freigabeeinstellungen für den jeweiligen Ordner auf dem Server definiert wurde), `path` der Pfad zur Datenbank und `file.mdb` der Name der MS Access Datenbank im freigegebenen Ordner, bzw. in einem Unterordner dieses Ordners.

- **Für ADO-Verbindungen:** Der ADO Connection String im SPS wird beim Erstellen des SPS in StyleVision definiert. Er enthält die erforderlichen Connection-Informationen sowie die Sicherheitsinformationen. Sie müssen sicherstellen, dass der beim Testen des Connection String in StyleVision verwendete Treiber auch auf allen Client-Rechnern vorhanden ist, auf denen Authentic Browser installiert ist, damit das SPS eine Verbindung zwischen Datenbank und Client aufbauen kann. Beachten Sie außerdem, dass ADO-Datenbankverbindungen nur mit lokalen Dateipfaden und nicht mit `http://` URLs funktionieren.

Anmerkung zu PXF-Dateien

Ein SPS-Design, in dem XSLT 2.0 verwendet wird, kann als PXF (Portable XML Form)-Datei gespeichert werden. Das PXF-Dateiformat wurde von Altova speziell zu dem Zweck entwickelt, das SPS-Design mit damit in Zusammenhang stehenden Dateien (wie z.B. der Schema-Datei, der XML-Quelldatei, im Design verwendeten Bilddateien und XSLT-Dateien für die Transformation der XML-Quelldatei in ein Ausgabeformat) zu verpacken. Der Vorteil des PXF-Dateiformats ist, dass alle für die Bearbeitung in der Authentic-Ansicht und die Generierung von Ausgabedateien anhand der Authentic-Ansicht benötigten Dateien ganz einfach in einer einzigen Datei weitergeleitet werden können.

Anmerkung: Wenn eine PXF-Datei auf einem Webserver gespeichert ist und mit dem Authentic Browser Plug-in verwendet werden soll, müssen Sie sicherstellen, dass der Server die Datei nicht blockiert. Fügen Sie dazu (z.B. über das Fenster IIS-Verwaltung) den folgenden MIME Type für PXF (`.pxf`)-Dateierweiterungen hinzu: `application/x-zip-compressed`.

© 2018-2024 Altova GmbH

2.3 HTML-Seite für das Authentic Plug-in

Die HTML-Seite für das Authentic Plug-in führt die folgenden wichtigen Funktionen aus:

1. Wenn die HTML-Seite für das Authentic Plug-in das erste Mal auf dem Client-Rechner geöffnet wird, wird das Authentic Browser Plug-in aufgrund des Codes in der HTML-Seite vom Server auf den Client heruntergeladen und dort installiert. Dieser Code muss korrekt sein, damit die richtige CAB-Datei auf dem Server gefunden wird.
2. Der Code in der HTML-Seite richtet die Authentic-Ansicht im Browser-Fenster ein, dabei werden auch die Größe der Authentic-Ansicht und die Pfade zu der XML-, XSD- und SPS-Datei konfiguriert.
3. Die zu bearbeitende XML-Datei sowie die Schema-Datei und das SPS, auf dem die XML-Datei basiert, werden definiert.
4. Der HTML-Code enthält `SCRIPT`-Elemente, Definitionen für Subroutinen sowie Ereignisbehandlungen. So kann z.B. festgelegt werden, welche Aktion ausgeführt werden soll, wenn man auf der HTML-Seite auf eine Schaltfläche klickt. Beispiele dazu finden Sie unter Internet Explorer Beispiel 1: Einfach 29.

Anmerkung: Wenn Sie die **Enterprise Edition von Authentic Browser** verwenden, muss die HTML-Seite auf dem Server gespeichert werden, für den die Enterprise Lizenz registriert wurde.

Einbettung von Authentic Browser

Um das Authentic Browser Plug-in mit Internet Explorer verwenden zu können, muss in die HTML-Seite, die das Plug-in herunterlädt, ein Objekt, das das Plug-in identifiziert, eingebettet werden. Dies erfolgt über das HTML-`OBJECT`-Element:

```
<OBJECT clsid="clsid:<CLSID>"/>
```

Nähere Informationen zu CLSID-Typwerten finden Sie unter Authentic Browser Versionen 10.

In diesem Abschnitt

In den Unterabschnitten dieses Abschnitts wird beschrieben, wie die oben aufgelisteten Funktionen auf der HTML-Seite implementiert werden sollen. Diese Abschnitten sind aufgrund der unterschiedlichen Methoden, wie die Authentic Browser DLL für bestimmte Browser heruntergeladen wird, auf der ersten Ebene nach Browser-Art gegliedert:

- Lizenzierung für die Enterprise Edition 23 enthält eine Beschreibung zur Lizenzierung für die Enterprise Edition von Authentic Browser
- Internet Explorer 24 enthält eine Beschreibung, wie man eine CAB-Datei (mit der Erweiterung `.cab`) herunter über ein HTML `OBJECT`-Element herunterlädt.
- Unabhängige Browser 33: Enthält eine Beschreibung, wie man feststellt, welche Browser-Art den Request sendet und wie man die richtige Plug-In-Version für diesen Browser herunterlädt.

In diesen Unterabschnitten finden Sie Beschreibungen und Beispiele zur Verwendung der HTML-Elemente `OBJECT` 24 und `SCRIPT` 28. Außerdem finden Sie Beispiele für vollständige HTML-Seiten, die das Authentic Browser Plug-in aufrufen. Nähere Informationen zu einzelnen Objekten finden Sie im Abschnitt Benutzerreferenz 47 unter der jeweiligen Objektbeschreibung 69.

 © 2018-2024 Altova GmbH

2.3.1 Lizenzierung für die Enterprise Edition

Die Lizenz für die Authentic Browser Enterprise Edition können Sie auf der Altova Website erwerben.

Übersicht über das Einrichten der Lizenz für die Enterprise Edition

Unten finden Sie eine kurze Übersicht, welche Schritte erforderlich sind, um die Lizenzinformationen für die Authentic Browser Enterprise Edition korrekt einzurichten.

- Für die Authentic Browser Enterprise Edition werden die folgenden gültigen Lizenzinformationen benötigt: (i) der ***Server-Name,*** für den die Lizenz gilt; (ii) der ***Name des Unternehmens,*** für das die Lizenz registriert wurde und (iii) der ***Lizenzschlüssel***. Diese Informationen werden Ihnen in der Lizenz-E-Mail zugesandt, die Sie nach Erwerb Ihrer Authentic Browser Enterprise Edition-Lizenz erhalten.
- Die Lizenzinformationen müssen auf der HTML-Seite für das Authentic Plug-in[22] vorhanden sein. Wie dies auf der HTML-Seite erfolgt, wird unten beschrieben. (Beachten Sie: Es gibt keine spezifische Lizenzschlüsseldatei, die von Authentic Browser referenziert wird.)
- Wenn die auf der HTML-Seite für das Authentic Plug-in[22] eingetragenen Informationen gültig sind, so werden die Funktionalitäten der Enterprise Edition in der Authentic Browser-Datei freigegeben.
- Die HTML-Seite für das Authentic Plug-in[22] muss auf dem Server gespeichert sein, für den die Lizenz gilt.

So geben Sie die Lizenzinformationen ein

Die drei Lizenzschlüsselparameter (Servername, Firmenname und Lizenzschlüssel) können auf mehrere Arten auf der HTML-Seite für das Authentic Plug-in[22] eingetragen werden:

- Als Parameterwerte des Elements `OBJECT`. Eine Anleitung dazu finden Sie im Abschnitt zur HTML-Seite für Internet Explorer[24].
- Wenn die HTML-Seite Browser-unabhängig sein soll, können die Lizenzparameter wie in der Codeliste im Browser-unabhängigen Beispiel[33] gezeigt, registriert werden.
- Der Lizenzparameter kann auch direkt im Objekt gesetzt werden, wie in der Codeliste, in der die Codeliste im Browser-unabhängigen Beispiel[33] adaptiert wurde, unten gezeigt wird.

```
<SCRIPT LANGUAGE="javascript" FOR=objPlugIn EVENT="ControlInitialized">
// event subscription if running on Internet Explorer
if ( isIEOnWindows() )
{
 InitAuthenticPluginPage();
}
</SCRIPT>

<SCRIPT type="text/javascript" LANGUAGE="javascript" >
function InitAuthenticPluginPage( )
{
var serverstr='DevAuthBrowTest';
var basedir='Authentic/';
objPlugIn.LicServer = 'DevAuthBrowTest';
objPlugIn.LicCompany = 'Altova';
objPlugIn.LicKey = 'XXXXXXXXXX';
objPlugIn.SchemaLoadObject.URL = 'http://' + serverstr + basedir + 'OrgChart.xsd';
objPlugIn.XMLDataLoadObject.URL = 'http://' + serverstr + basedir + 'OrgChart.xml' ;
objPlugIn.DesignDataLoadObject.URL = 'http://' + serverstr + basedir +
'OrgChart.sps';
```

```
objPlugIn.StartEditing();
}
</SCRIPT>
```

2.3.2 Internet Explorer

Wenn die HTML-Seite für das Authentic Plug-in in Internet Explorer (32-Bit oder 64-Bit) geöffnet werden soll, muss sie die folgenden Elemente enthalten:

- ein HTML `OBJECT`[24]-Element, das (i) die richtige DLL für das Authentic Plug-In (32-Bit oder 64-Bit) vom Server auf den Client lädt und (ii) die Größe des Authentic-Fensters im Browser des Client definiert. Das `OBJECT-`[24]Element enthält den Pfad zum Authentic Browser Plug-in. **Beachten Sie, dass das Authentic Plug-in in einer Version für die 32-Bit- und in einer Version für die 64-Bit-Version von Internet Explorer zur Verfügung steht (außer für die 64-Bit-Version von IE 10 und 11), daher muss die richtige Authentic Browser Plug-in-Version (**.cab **Datei) auf den Client heruntergeladen werden.** Beispielcode, der die X-Bit Verson von Internet Explorer automatisch überprüft und das richtige Authentic Plug-in herunterlädt, finden Sie im Abschnitt Browser-unabhängiges Beispiel[33].
- eines oder mehrere HTML `SCRIPT`[28]-Elemente. Ein `SCRIPT`-Element definiert die verschiedenen Subroutinen, die in der HTML-Seite und bei der Ereignisverarbeitung verwendet werden. Mit Hilfe eines `SCRIPT`-Elements können Sie definieren, welches XML-Dokument bearbeitet werden soll und welches XML-Schema und welche SPS-Datei dem XML-Dokument zugrunde gelegt werden soll.

Dieser Abschnitt

Dieser Abschnitt ist in die folgenden Unterabschnitte gegliedert:

- Das OBJECT-Element[24]: Hier wird beschrieben, wie das HTML `OBJECT`[24] Element in einer HTML-Seite für das Authentic Plug-in verwendet wird.
- Das SCRIPT-Element[28]: Hier wird beschrieben, wie HTML `SCRIPT`[28] Elemente in einer HTML-Seite für das Authentic Plug-in verwendet werden.
- Beispiele für komplette HTML-Seiten: IE Beispiel 1, einfach[29] und IE Beispiel 2: Tabelle sortieren[30].

Informationen zu einzelnen Objekten finden Sie im Abschnitt "Benutzerreferenz"[47] unter der entsprechenden Object-Beschreibung[69].

Hinweis: Das Authentic Browser Plug-in wird für **Internet Explorer 5.5 oder höher** unterstützt, allerdings wird die 64-Bit-Version von Internet Explorer 10 und 11 nicht unterstützt.

2.3.2.1 Das OBJECT-Element

Das `OBJECT`-Element hat die folgenden Funktionen:

- Es gibt dem Authentic Browser Plug-In einen Namen (über das `id`-Attribut).
- Es wählt aus, welche Version des Authentic Browser Plug-In[10] verwendet werden soll (über den Wert des `classid`-Attributs).

© 2018-2024 Altova GmbH

- Es definiert eine `.CAB`-Datei und eine Versionsnummer (codebase-Attribut). Die `.CAB`-Datei enthält eine DLL, die ein COM-Objekt (das Authentic Browser Plug-In) mit einer ID registriert, die der Wert des `classid`-Attributs ist. Normalerweise wird sowohl die 32-Bit- als auch die 64-Bit-Version des Authentic Browser Plug-in auf dem Server gespeichert. Jede Version wird durch einen unterschiedlichen Dateinamen gekennzeichnet. Das `codebase` Attribut definiert den Dateinamen der erforderlichen `.CAB`-Datei (*siehe Codefragment weiter unten*). Die Class IDs für die 32-Bit und die 64-Bit `.CAB` Dateien sind identisch und werden in der Tabelle für die verschiedenen EN/DE-Versionen der Trusted/Untrusted Version[10] aufgelistet.

  Dateiname für die 32-Bit Version: `AuthenticBrowserEdition.CAB`
  Dateiname für die 64-Bit Version: `AuthenticBrowserEdition_x64.CAB`

- Definiert (über das **`style`**-Attribut) die Größe des Authentic-Fensters im Browser des Client.
- Es kann eine beliebige Anzahl von Parametern definieren.

OBJECT-Beispielelement

Hier sehen Sie ein Beispiel für ein HTML `OBJECT`-Element. Es wählt die Trusted Unicode-Version aus (über den im `classid`-Attribut definierten Wert) und legt als Größe des Authentic-Fensters im Browser des Client 600 x 500 Pixel fest. Im Folgenden sind alle Attribute und Parameter beschrieben, die dem `OBJECT`-Element zur Verfügung stehen.

Anmerkung: Bei der unten angegebenen Versionsnummer handelt es sich nicht notwendigerweise um die aktuelle Versionsnummer. Nähere Informationen dazu finden Sie unter codebase[26].

Für das 32-Bit Authentic Browser Plug-in:

```
<OBJECT id="objPlugIn" style="WIDTH:600px; HEIGHT:500px"
   codeBase="http://yourserver/cabfiles/AuthenticBrowserEdition.CAB#Version=12,3,0,0"
   classid=clsid:B4628728-E3F0-44a2-BEC8-F838555AE780>
   <PARAM NAME="XMLDataURL" VALUE="http://yourserver/OrgChart.xml">
   <PARAM NAME="SPSDataURL" VALUE="http://yourserver/OrgChart.sps">
   <PARAM NAME="SchemaDataURL" VALUE="http://yourserver/OrgChart.xsd">
</OBJECT>
```

Für das 64-Bit Authentic Browser Plug-in:

```
<OBJECT id="objPlugIn" style="WIDTH:600px; HEIGHT:500px"
   codeBase="http://yourserver/cabfiles/AuthenticBrowserEdition_x64.CAB#Version=12,3,0,0"
   classid="clsid:B4628728-E3F0-44a2-BEC8-F838555AE780">
   <PARAM NAME="XMLDataURL" VALUE="http://yourserver/OrgChart.xml">
   <PARAM NAME="SPSDataURL" VALUE="http://yourserver/OrgChart.sps">
   <PARAM NAME="SchemaDataURL" VALUE="http://yourserver/OrgChart.xsd">
</OBJECT>
```

`id`

Der Wert des `id`-Attributs wird als Name der Authentic Browser Plug-In Objekte verwendet, wenn diese in Skripts verwendet werden. So wird z.B. mit `objPlugIn.SchemaLoadObject.URL` das Objekt aufgerufen, das die Schema-Datei lädt. Nähere Informationen dazu siehe Das SCRIPT-Element[28].

`style`

Dies ist das normale HTML **style**-Attribut. Es dient zur Angabe der Größe des Authentic-Fensters im Browser des Client.

codebase
Das codebase-Attribut gibt den Pfad zur **.CAB**-Datei an. Beachten Sie, dass die **.CAB**-Datei für das 32-Bit Authentic Browser Plug-in und die .CAB-Datei für das 64-Bit Authentic Browser Plug-in nicht dieselben sind, nämlich: AuthenticBrowserEdition.CAB und AuthenticBrowserEdition_x64.CAB.

Der Wert der optionalen #Version-Erweiterung gibt die Versionsnummer der Komponente an, die derzeit auf dem Server zur Verfügung steht. Falls der Client eine frühere Version verwendet und im codebase Attribut eine neuere Version definiert ist, wird die neuere Version vom Server installiert. Wenn die #Version-Erweiterung nicht angegeben wurde, erfolgt die Aktualisierung erst, wenn die Komponente manuell vom Client entfernt wurde. Die aktuelle Versionsnummer der Komponente ist in den Eigenschaften der **.dll**-Datei der **CAB**-Datei der Komponente aufgelistet. (Klicken Sie mit der rechten Maustaste auf die Datei und wählen Sie den Befehl "Eigenschaften").

classid
Die Class IDs für die 32-Bit und die 64-Bit .CAB-Dateien sind identisch und werden im Kapitel Authentic Browser Versionen[10] aufgelistet.

Ab Version 5.0 des Browser Plug-In ist der **classid**-Wert für Unicode-Versionen ab Version 5.0 ein anderer als der früherer Unicode-Versionen. Wenn Sie daher die Unicode .CAB-Datei auf Ihrem Server von einer Version vor Version 5.0 aktualisieren, stellen Sie sicher, dass Sie die **classid**-Werte in Ihren HTML-Dateien ändern. Beachten Sie: Wenn eine neue **.CAB**-Datei auf dem Server dieselbe CLSID hat wie eine auf dem Client bereits installierte .CAB-Datei, wird die alte .CAB-Datei auf dem Client nicht automatisch durch die neue ersetzt. Sie müssen die zuvor installierte .CAB-Datei entfernen, bevor Sie die neue .CAB-Datei herunterladen. Jede Sprachversion hat einen anderen CLSID-Wert.

Parameter

Sie können beliebig viele, der folgenden Parameter verwenden.

LicServer
Der Name des Servers, für den der Lizenzschlüssel für die Authentic Browser Enterprise Edition gültig ist.

LicKey
Der Lizenzschlüssel zum Validieren der Verwendung der Authentic Browser Enterprise Edition.

LicCompany
Der Firmenname zum Validieren der Verwendung der Authentic Browser Enterprise Edition.

XMLDataURL
Eine absolute URL, die den Speicherort der zu editierenden XML-Datei angibt. Für Untrusted Versionen können Sie auch einen vollständigen lokalen Pfad angeben.

XMLDataSaveURL
Eine absolute URL, die den Ordner angibt, in dem die XML-Datei gespeichert werden soll. Für Untrusted Versionen können Sie auch einen vollständigen lokalen Pfad angeben.

SPSDataURL

© 2018-2024 Altova GmbH

Eine absolute URL, die den Pfad zum StyleVision Power Stylesheet (.sps-Datei) angibt. Für Untrusted Versionen können Sie auch einen vollständigen lokalen Pfad angeben.

SchemaDataURL
Eine absolute URL, die den Pfad zur verknüpften Schema-Datei angibt. Für Untrusted Versionen können Sie auch einen vollständigen lokalen Pfad angeben.

TextStateBmpURL
Der Ordner, in dem Bitmap-Grafiken für Textstatus-Symbole gespeichert werden sollen.

TextStateToolbarLine
Die Zeile der Symbolleiste, in der Textstatus-Symbole platziert werden sollen. Der Standardwert ist 1.

AutoHideUnusedCommandGroups
Gibt an, ob Gruppen nicht verwendeter Symbolleistenbefehle ausgeblendet werden sollen. Der Standardwert ist True.

ToolbarsEnabled
Gibt die allgemeine Unterstützung für Symbolleisten an. Der Standardwert ist True.

ToolbarTooltipsEnabled
Gibt an, ob Tooltips aktiviert oder deaktiviert sind.

HideSaveButton
Bei Einstellung "True" wird die standardmäßig eingeblendete Schaltfläche "Speichern" aus der Authentic-Symbolleiste entfernt.

BaseURL
Gibt die Basis-URL für relative Pfade an.

SaveButtonUsePOST
Bei Einstellung "True" wird beim Speichern des Dokuments der Befehl HTTP POST verwendet, anstelle von PUT.

EntryHelpersEnabled
Bei Einstellung "True" werden die Authentic-Eingabehilfen eingeblendet.

EntryHelperSize
Breite der Eingabehilfefenster in Pixeln.

EntryHelperAlignment
Definiert die Position der Eingabehilfen im Dokumentenfenster.

0 = Symbolleiste am oberen Rand des Dokuments anzeigen
1 = Symbolleiste am linken Rand des Dokuments anzeigen
2 = Symbolleiste am unteren Rand des Dokuments anzeigen
3 = Symbolleiste am rechten Rand des Dokuments anzeigen

EntryHelperWindows
Gibt an, welche der Eingabehilfenfenster eingeblendet werden sollen.

1 = Elemente
2 = Attribute

4 = Entities

Zulässig ist jede Kombination (bit-check)

SaveButtonAutoEnable
Siehe Authentic.SaveButtonAutoEnable 91

LoaderSettingsFileURL
Gibt die URL der LoaderSettingsFile für die Paketverwaltung an.

2.3.2.2 Das SCRIPT-Element

Die SCRIPT-Elemente definieren die Ereignishandler und Subroutinen, die von innerhalb der HTML-Datei aus aufgerufen werden können.

Im Folgenden sehen Sie ein Beispiel für ein Skript zur Ereignisbehandlung:

```
<SCRIPT LANGUAGE="javascript" FOR=objPlugIn EVENT="ControlInitialized">
      objPlugIn.SchemaLoadObject.URL = "http://yourserver/OrgChart.xsd"
      objPlugIn.XMLDataLoadObject.URL = "http://yourserver/OrgChart.xml"
      objPlugIn.DesignDataLoadObject.URL = "http://yourserver/OrgChart.sps"
      objPlugIn.StartEditing
</SCRIPT>
```

Hier sehen Sie ein Beispiel für ein Skript mit Subroutinen:

```
<SCRIPT ID=clientEventHandlers LANGUAGE=vbscript>
   Sub BtnOnClick
      objPlugIn.SchemaLoadObject.URL = "http://yourserver/OrgChart.xsd"
      objPlugIn.XMLDataLoadObject.URL = "http://yourserver/OrgChart.xml"
      objPlugIn.DesignDataLoadObject.URL = "http://yourserver/OrgChart.sps"
      objPlugIn.StartEditing
   End Sub
   Sub OnClickFind
      objPlugIn.FindDialog
   End Sub
   Sub BtnOnTestProp
      If objPlugIn.IsRowInsertEnabled Then
         msgbox "true"
      Else
         msgbox "false"
      End If
   End Sub
</SCRIPT>
```

Skript-Sprachen
Das Authentic Browser Plug-In wurde mit JavaScript and VBScript getestet.

Ereignisbehandlung
Der Wert des ID-Attributs des OBJECT-Elements im HTML-Textkörper wird als der Wert des FOR-Attributs definiert. Authentic Browser Plug-In-Objekte, die aufgerufen werden, müssen einen Namen haben, der diesem Wert entspricht. Eine Liste von Ereignissen finden Sie auch unter Ereignisse: Referenz 50.

 © 2018-2024 Altova GmbH

Subroutinen
Für jedes Ereignis, das Sie in der HTML-Datei definieren möchten, können Subroutinen erstellt werden. Der Objektname für das Authentic Browser Plug-In muss mit dem Wert des `ID`-Attributs des `OBJECT`-Elements im HTML-Textkörper identisch sein. Im oben gezeigten Beispiel ist das Präfix `objPlugIn`. Dieses Präfix muss der Wert des `ID`-Attributs des `OBJECT`-Elements sein. Die im Authentic Browser Plug-In verfügbaren Methoden, Eigenschaften und Sub-Objekte werden im Referenzteil dieser Dokumentation beschrieben.

2.3.2.3 IE Beispiel 1: Einfach

Der unten gezeigte HTML-Code generiert eine Seite, die die folgenden Funktionen hat:

- Sie installiert die Trusted Unicode-Version von Authentic Browser am Client, falls dies nicht bereits geschehen ist.
- Innerhalb des body-Tags ist ein Fenster von der Größe 600px mal 500px definiert, in das Authentic Browser geladen wird.
- Unterhalb des Authentic Browser-Fensters befindet sich eine Reihe mit vier Schaltflächen
- Die Authentic-Ansicht von `OrgChart.xml` ist geladen.
- Die Schaltflächen **Suchen** und **Ersetzen** dienen zum Aufrufen des Such- bzw. Ersetzungsdialogfelds.
- Die Schaltfläche **Speichern** dient zum Speichern der Änderungen in einer Datei mit dem Namen `SaveFile.xml`, die im Root-Verzeichnis des Servers abgelegt ist.
- Die Schaltfläche **Eigenschaft testen** dient zum Testen einer einfachen Eigenschaft.

Wenn diese HTML-Seite auf dem Client geöffnet wird, kann der Benutzer die XML-Datei `OrgChart.xml` bearbeiten und die bearbeitete Datei als `saveFile.xml` speichern.

Anhand dieser einfachen HTML-Seite können Sie testen, ob Authentic Browser korrekt funktioniert. Stellen Sie dabei sicher, dass Sie zur Angabe der CAB-Datei, der `xsd`-, `xml`- und der `sps`-Datei und aller anderen auf dem Server gespeicherten Ressourcen die richtigen URLs verwenden. Beachten Sie, dass die Groß- und Kleinschreibung auf einigen Servern eine Rolle spielt. Wenn eine Datei daher nicht gefunden wird, überprüfen Sie die Groß- und Kleinschreibung von Dateinamen und Befehlen im Code. Sie können dieses Beispiel erweitern und ändern, um komplexere Lösungen mit Hilfe von Authentic Browser zu erstellen. Nähere Informationen dazu finden Sie auch unter dem OBJECT-Element[24].

```
<html>
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=windows-1252">
    <title>Minimal XMLSpyDocEditPlugIn page</title>

    <!-- Script for handling the ControlInitialized event -->
    <SCRIPT LANGUAGE="javascript" FOR="objPlugIn" EVENT="ControlInitialized">
         objPlugIn.SchemaLoadObject.URL = "http://yourserver/OrgChart.xsd"
         objPlugIn.XMLDataLoadObject.URL = "http://yourserver/OrgChart.xml"
         objPlugIn.DesignDataLoadObject.URL = "http://yourserver/OrgChart.sps"
         objPlugIn.StartEditing()
    </SCRIPT>

    <!-- Script with subroutines -->
    <SCRIPT ID=clientEventHandlers LANGUAGE=vbscript>
      Sub OnClickFind
         objPlugIn.FindDialog
      End Sub

      Sub OnClickReplace
```

```
        objPlugIn.ReplaceDialog
      End Sub

      Sub BtnOnSave
        objPlugIn.XMLDataSaveUrl = "http://yourserver/SaveFile.xml"
        objPlugIn.Save
      End Sub

      Sub BtnOnTestProp
        If objPlugIn.IsRowInsertEnabled Then
          msgbox "true"
        Else
          msgbox "false"
        End If
      End Sub
    </SCRIPT>
  </head>

  <body>
    <!-- Object element has id with value that must be used -->
    <!-- as name of Authentic Browser Plug-in objects -->
    <!-- Classid selects the Trusted Unicode version -->
    <OBJECT id="objPlugIn"
    <!-- CodeBase selects 32-bit CAB file (AuthenticBrowserEdition.CAB) -->
    <!-- or 64-bit CAB file (AuthenticBrowserEdition_x64.CAB) -->
      CodeBase="http://yourserver/AuthenticBrowserEdition.CAB#Version=12,3,0,0"
    <!-- Class Id for 32-bit and 64-bit CAB files is the same -->
      Classid="clsid:B4628728-E3F0-44a2-BEC8-F838555AE780" width="600" height="500">
    </OBJECT>
    <p>
      <input type="button" value="Find" name="B4" onclick="OnClickFind()">
      <input type="button" value="Replace" name="B5" onclick="OnClickReplace()">
      <input type="button" value="Save" name="B6" onclick="BtnOnSave()">
      <input type="button" value="Test property" name="B7" onclick="BtnOnTestProp">
    </p>
  </body>
</html>
```

2.3.2.4 IE Beispiel 2: Tabelle sortieren

Dies ist eine HTML-Beispieldatei mit einem eingebetteten JavaScript. Für das Beispiel muss das Authentic Browser Plug-In (CAB-Datei und Lizenzdatei) auf Ihrem Computer installiert werden. Beachten Sie, dass die Groß- und Kleinschreibung auf einigen Servern eine Rolle spielt. Wenn eine Datei daher nicht gefunden wird, überprüfen Sie die Groß- und Kleinschreibung von Dateinamen und Befehlen im Code.

Der Code zeigt an:

- wie das Browser Plug-In aufgerufen werden soll. Passen Sie den Code bitte an und geben Sie den Pfad zu Ihrer CAB-Datei und dem Class Identifier Ihrer Browser Plug-In-Version an (trusted oder untrusted).
- wie eine Datei in das Browser Plug-In geladen werden soll. Passen Sie bitte den Code an und geben Sie den Pfad zu Ihrem Beispieldokument an.
- wie Schaltflächen für einfache Cursor-Positionierungen implementiert werden.
- wie komplexere Befehle wie z.B. die Sortierung von Tabellen implementiert werden.
- wie das `SelectionChanged`-Ereignis verwendet wird.

Nähere Informationen dazu finden Sie auch unter dem OBJECT-Element [24].

© 2018-2024 Altova GmbH

```
<html>
   <head>
      <meta http-equiv="Content-Type" content="text/html; charset=windows-1252">
      <title>test page For Authentic Browser Plug-in</title>

      <SCRIPT LANGUAGE="javascript" For="objPlugIn" EVENT="ControlInitialized">
         var strSampleRoot = "http://myRoot/myPath/myDocBaseName";
         objPlugIn.SchemaLoadObject.URL = strSampleRoot + ".xsd";
         objPlugIn.XMLDataLoadObject.URL = strSampleRoot + ".xml";
         objPlugIn.DesignDataLoadObject.URL = strSampleRoot + ".sps";
         objPlugIn.StartEditing();
      </SCRIPT>

      <SCRIPT ID="clientEventHandlers" LANGUAGE="javascript">
         var objCurrentRange = Null;

         Function BtnDocumentBegin() { objPlugIn.AuthenticView.DocumentBegin.Select(); }
         Function BtnDocumentEnd() { objPlugIn.AuthenticView.DocumentEnd.Select(); }
         Function BtnWholeDocument() { objPlugIn.AuthenticView.WholeDocument.Select(); }
         Function BtnSelectNextWord()
{ objPlugIn.AuthenticView.Selection.SelectNext(1).Select(); }
         Function BtnSortDepartmentOnClick()
         {
            var objCursor = Null;
            var objTableStart = Null;
            var objBubble = Null;
            var strField1 = "";
            var strField1 = "";
            var nColIndex = 0;
            var nRows = 0;

            objCursor = objPlugIn.AuthenticView.Selection;
            If (objCursor.IsInDynamicTable())
            {
               // calculate current column index
               nColIndex = 0;
               While (True)
               {
                  try { objCursor.GotoPrevious(11); }
               catch (err) { break; }
               nColIndex++;
            }

            // GoTo begin of table
            objTableStart = objCursor.ExpandTo(9).CollapsToBegin().Clone();

            // count number of table rows
            nRows = 1;
            While (True)
            {
               try { objTableStart.GotoNext(10); }
            catch (err) { break; }
            nRows++;
```

```
            }

            // bubble sort through table
            For (var i = 0; i < nRows - 1; i++) {
               for(var j = 0; j < nRows-i-1; j++) {
                  objBubble = objCursor.ExpandTo(9).CollapsToBegin().Clone();
                  // Select correct column in jth table row
                  objBubble.GotoNext(6).Goto(10,j,2).Goto(11,nColIndex,2).ExpandTo(6);
                  strField1 = objBubble.Text;
                  strField2 = objBubble.GotoNext(10).Goto(11,nColIndex,2).ExpandTo(6).Text;
                  if(strField1 > strField2) {
                      if(!objBubble.MoveRowUp()) {
                             alert('Table row move is not allowed!');
                             return;
                      }
                  }
               }
            }
         }
      </SCRIPT>
   </head>

   <body>
      <Object id="objPlugIn"
      <!-- CodeBase selects 32-bit CAB file (AuthenticBrowserEdition.CAB) -->
      <!-- or 64-bit Cab file (AuthenticBrowserEdition_x64.CAB) -->
         codeBase="http://myCabfileLocation/AuthenticBrowserEdition.CAB#Version=12,3,0,0"
       <!-- Class Id for 32-bit and 64-bit CAB files is the same -->
         classid="clsid:B4628728-E3F0-44a2-BEC8-F838555AE780"
         width="100%"
         height="80%"
         VIEWASTEXT>
         <PARAM NAME="EntryHelpersEnabled" VALUE="TRUE">
         <PARAM NAME="SaveButtonAutoEnable" VALUE="TRUE">
      </Object>
      <TABLE>
         <TR>
            <TD><Input Type="button" value="Goto Begin" id="B1"
onclick="BtnDocumentBegin()"></TD>
            <TD><Input Type="button" value="Goto End" name="B2"
onclick="BtnDocumentEnd()"></TD>
            <TD><Input Type="button" value="Whole Document" name="B3"
onclick="BtnWholeDocument()"></TD>
            <TD><Input Type="button" value="Select Next Word" name="B4"
onclick="BtnSelectNextWord()"></TD>
         </TR>
         <TR>
            <TD><Input Type="button" value="Sort Table by this Column" id="B6"
onclick="BtnSortDepartmentOnClick()"</TD>
         </TR>
      </TABLE>
      <TABLE id=SelTable border=1>
         <TR><TD id=SelTable_FirstTextPosition></TD><TD
id=SelTable_LastTextPosition></TD></TR>
```


© 2018-2024 Altova GmbH

```
        <TR><TD id=SelTable_FirstXMLData></TD><TD id=SelTable_FirstXMLDataOffset></TD></TR>
        <TR><TD id=SelTable_LastXMLData></TD><TD id=SelTable_LastXMLDataOffset></TD></TR>
        <TR><TD id=SelTable_Text></TD></TR>
      </TABLE>
   </body>

   <SCRIPT LANGUAGE=javascript For=objPlugIn EVENT=selectionchanged>
      var CurrentSelection =  Null;
      CurrentSelection = objPlugIn.AuthenticView.Selection;
      SelTable_FirstTextPosition.innerHTML = CurrentSelection.FirstTextPosition;
      SelTable_LastTextPosition.innerHTML = CurrentSelection.LastTextPosition;
      SelTable_FirstXMLData.innerHTML = CurrentSelection.FirstXMLData.Parent.Name;
      SelTable_FirstXMLDataOffset.innerHTML = CurrentSelection.FirstXMLDataOffset;
      SelTable_LastXMLData.innerHTML = CurrentSelection.LastXMLData.Parent.Name;
      SelTable_LastXMLDataOffset.innerHTML = CurrentSelection.LastXMLDataOffset;
   </SCRIPT>

</html>
```

2.3.3 Browser-unabhängig

In manchen Projekten ist es nicht immer bekannt, welcher Browser auf dem Client verwendet wird. In diesen Fällen kann auf dem Server Authentic Browser für unterschiedliche Internet Explorer-Versionen[10] gespeichert werden. Sie können in die HTML-Seite ein Skript einfügen, mit dem ermittelt wird, mit welchem Browser die HTML-Seite geöffnet wurde, sodass das richtige Authentic Browser Plug-in[10] geladen werden kann.

Wenn auf dem Client Internet Explorer verwendet wird, so muss die richtige **.CAB**-Datei (für den 32-Bit oder den 64-Bit Internet Explorer) ausgewählt werden, um vom Server heruntergeladen zu werden. Mit Hilfe eines Skripts kann ermittelt werden, welche X-Bit Version von Internet Explorer verwendet wird, damit die richtige .CAB Datei vom Server heruntergeladen wird.

In der Beispieldatei unten geschieht Folgendes: Der Browser wird ermittelt, die korrekte Authentic Browser Version wird geladen wird und einige Funktionen werden ausgeführt. Informationen zu einzelnen Objekten finden Sie im Abschnitt "Benutzerreferenz"[47] unter der entsprechenden Object-Beschreibung[69].

Beispieldatei

Der unten gezeigte HTML-Code generiert eine Seite, die die folgenden Funktionen hat:

- Sie überprüft, welcher Browser auf dem Client installiert ist.
- Wenn als Browser Internet Explorer installiert ist, wird überprüft, ob es sich um ein 32-Bit oder ein 64-Bit-System handelt. Anschließend wird die richtige .CAB-Datei[24] (für den 32-Bit oder den 64-Bit Internet Explorer) ausgewählt.
- Das Authentic Browser-Fenster in der Seite hat eine Breite, die 100 % von der des Browser-Fensters beträgt und 60 % seiner Höhe.
- Unterhalb des Authentic Browser-Fensters befindet sich eine Reihe mit fünf Schaltflächen
- Die Schaltfläche **Bearbeitung starten** dient zum Laden der Authentic-Ansicht von OrgChart.xml, welche sich im Root-Verzeichnis Ihres Servers befindet.
- Die Schaltflächen **Suchen** und **Ersetzen** dienen zum Aufrufen des Such- bzw. Ersetzungsdialogfelds.
- Die Schaltfläche **Speichern** dient zum Speichern der Änderungen in einer Datei mit dem Namen SaveFile_OrgChart.xml, die im Root-Verzeichnis des Servers abgelegt ist.

- Die Schaltfläche **Eigenschaft testen** dient zum Testen einer einfachen Eigenschaft.

Wenn diese HTML-Seite auf dem Client geöffnet wird, kann der Benutzer die XML-Datei `OrgChart.xml` bearbeiten und die bearbeitete Datei als `SaveFile_OrgChart.xml` speichern.

Anhand dieser einfachen HTML-Seite können Sie testen, ob Authentic Browser korrekt funktioniert. Stellen Sie dabei sicher, dass Sie die richtige IP-Adresse und den richtigen Pfad zu den entsprechenden Dateien in den URLs der `XPI`-Datei, der `xsd`-, `xml`- und der `sps`-Datei und aller anderen auf dem Server gespeicherten Ressourcen verwenden. Beachten Sie, dass die Groß- und Kleinschreibung auf einigen Servern eine Rolle spielt. Wenn eine Datei daher nicht gefunden wird, überprüfen Sie die Groß- und Kleinschreibung von Dateinamen und Befehlen im Code. Sie können dieses Beispiel erweitern und ändern, um komplexere Lösungen mit Hilfe von Authentic Browser zu erstellen.

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
   <head>
      <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
      <title>Orgchart.sps Scriptable Plug-in Test - browser independent</title>
      <script type="text/javascript">
         <!--
         function BtnOnSave() { objPlugIn.Save();}

         function InitAuthenticPluginPage( )
         {

          var schema= document.getElementById('xsd');
          var instance=document.getElementById('xml');
          var design=document.getElementById('sps');
          objPlugIn.XMLDataLoadObject.URL =instance.innerHTML;
          objPlugIn.DesignDataLoadObject.URL = design.innerHTML;
          objPlugIn.SchemaLoadObject.URL= schema.innerHTML;
         // alert(schema.innerHTML+"  "+instance.innerHTML+" " +design.innerHTML);

         /*
          var serverstr='your-server/';
          var basedir='Authentic/';
          objPlugIn.SchemaLoadObject.URL = 'http://' + serverstr + basedir + 'OrgChart.xsd';
          objPlugIn.XMLDataLoadObject.URL = 'http://' + serverstr + basedir + 'OrgChart.xml'
;
          objPlugIn.DesignDataLoadObject.URL = 'http://' + serverstr + basedir +
'OrgChart.sps';
         */

          objPlugIn.StartEditing();


         }

         function Unload()
         {
         }
         //-->
```


© 2018-2024 Altova GmbH

```
      </script>
   <style type="text/css">@page { margin-left:0.60in; margin-right:0.60in; margin-
top:0.79in; margin-bottom:0.79in } @media       screen { br.altova-page-break { display:
none; } } @media print { br.altova-page-break { page-break-before: always; } }
   </style>
   </head>

   <body id="bodyId" onunload="Unload()">
      <table border="1">
         <tbody>
            <tr><th><span>DesignLoadURL</span></th><td id="sps">http://your-
server/Authentic/Orgchart.sps</td></tr>
            <tr><th><span>SchemaLoadURL</span></th><td id="xsd">http://your-
server/Authentic/Orgchart.xsd</td></tr>
            <tr><th><span>XMLDataLoadURL</span></th><td id="xml">http://your-
server/Authentic/Orgchart.xml</td></tr>
            <tr><th><span>XMLDataSaveURL</span></th><td id="xmlsave">http://your-
server/Authentic/SaveFile_OrgChart.xml</td></tr>
         </tbody>
      </table>
      <center><h3><span>Authentic Platformindependent Plug-in Enterprise
Edition</span></h3></center>
      <span> </span>
      <center>
      <script language="JavaScript" type="text/javascript">
         // return true if the page loads in Internet Explorer
         function isIEOnWindows()
         {
          return ((navigator.userAgent.indexOf('MSIE') != -1) &&
(navigator.userAgent.indexOf('Win') != -1))
         }

         //return true if Browser is 64bit
         function is64bitBrowser()
         {
          return ((navigator.userAgent.indexOf('Win64') != -1)&&
(navigator.userAgent.indexOf('x64') != -1))
         }

         //return Codebase for 32 bit  or 64 bit
         function getCodeBase()
         {
           if ( is64bitBrowser() ){
            return('CodeBase="http://your-
server/AuthenticBrowserEdition_x64.CAB#Version=12,2,0,0" ')
           }
           else {
            return('CodeBase="http://your-
server/AuthenticBrowserEdition.CAB#Version=12,2,0,0" ')
           }
         }

         // Create the plugin object instance, according to the browser loading the page
```

```
        // -IE uses <OBJECT> tag for embeding plugins and supports CODEBASE attribute
        // to indicate a .cab file for the installation if the plugin is not
        // currently installed

        function createObject( codebase, clsid)
        {if ( isIEOnWindows() )
         {
         document.write ( '<OBJECT ' +
           'id="objPlugIn" ' +
           getCodeBase() +
           'Classid="clsid:B4628728-E3F0-44a2-BEC8-F838555AE780" ' +
           'width="100%" ' +
           'height="60%" ' +
           '>' +
           '<PARAM NAME="XMLDataSaveUrl" VALUE="http://your-
server/Authentic/SaveFile_OrgChart.xml"> ' +
           '<PARAM NAME="EntryHelpersEnabled" VALUE="TRUE"> ' +
           '<PARAM NAME="SaveButtonAutoEnable" VALUE="TRUE"> ' +
           '<PARAM NAME="LicServer" VALUE="your-server"> ' +
           '<PARAM NAME="LicCompany" VALUE="Altova"> ' +
           '<PARAM NAME="LicKey" VALUE="XXXXXXXXXX"> ' +
           '<\/OBJECT>');
        }
        }

        createObject();
        // after running createObject the plugin object exists. Initialize the javascript
variable to be used in the scripts
        var objPlugIn = document.getElementById('objPlugIn');
     </script>

        <br><br>
        <button onclick="objPlugIn.StartEditing()"><span>Start Editing</span></button>
        <button onclick="objPlugIn.FindDialog()"><span>Find</span></button>
        <button onclick="objPlugIn.ReplaceDialog();"><span>Replace</span></button>
        <button onclick="BtnOnSave()"><span>Save</span></button>
        <button onclick="alert
( objPlugIn.IsRowInsertEnabled );"><span>Test</span><br></button>
        </center>

     <script language="javascript" type="text/javascript">
          // event subscription if running on Firefox
           if ( isFirefoxOnWindows() )
           {
           objPlugIn.addEventListener("ControlInitialized", InitAuthenticPluginPage,
false);
           }
     </script>

     <script event="ControlInitialized" for="objPlugIn" language="javascript"
type="text/javascript">
          // event subscription if running on Internet Explorer
           if ( isIEOnWindows() )
```


© 2018-2024 Altova GmbH

```
            {
            InitAuthenticPluginPage();
            //if ( isIEx64OnWindows() ) alert("IE x64");
            }
      </script>

   </body>
</html>
```

Anmerkung: Das oben gezeigte Skript enthält Lizenzinformationen zum Aktivieren der Authentic Browser Enterprise Edition.

2.4 Erweiterungspakete für die On-Demand-Installation

Falls Sie eine On-Demand-Installation des Authentic Browser Plug-in planen, muss/müssen das/die gezippte(n) Authentic Browser Erweiterungspaket(e) (CAB-Datei/en) auf dem Server gespeichert werden. Wenn die HTML-Seite für das Authentic Plug-in zum ersten Mal auf dem Client Browser aufgerufen wird, wird das entsprechende Erweiterungspaket aufgrund der Anweisungen auf der HTML-Seite vom Server auf den Client heruntergeladen, entpackt und installiert.

Die erforderlichen Authentic Browser Erweiterungspakete für die On-Demand-Installation können von der Altova Website heruntergeladen werden. Sie müssen auf dem Server in Form einer gezippten CAB-Datei gespeichert werden.

Internet Explorer v5.5 oder höher (32-Bit)	CAB-Datei (für 32-Bit IE)
Internet Explorer (64-Bit)	CAB-Datei (für 64-Bit IE)

CAB-Dateien für Internet Explorer stehen für 32-Bit IE Browser und für 64-Bit IE Browser zur Verfügung. Sie können beide CAB-Dateiarten (32-Bit und 64-Bit) auf dem Server speichern. Die HTML-Seite für das Authentic Plug-in 22 könnte ein Skript enthalten, in dem festgelegt ist, ob es sich beim Browser um eine 32-Bit- oder eine 64-Bit-Version handelt und das die richtige CAB-Datei anschließend herunterlädt. Eine Anleitung zur Erstellung eines solchen Skripts finden Sie im Abschnitt HTML-Seite für das Authentic Plug-in | Browser-unabhängig 33.

Herunterladen und Speichern der CAB/XPI/CRX-Datei

Die CAB-Datei muss von der Altova Website heruntergeladen werden und kann in jedem Ordner Ihres Servers gespeichert werden. Wenn Sie die Enterprise Edition von Authentic Browser verwenden, **muss das Paket auf dem Server gespeichert werden, für den die Enterprise Lizenz registriert wurde**.

Die Installationsdatei (CAB) ist ein gezipptes Dateiformat. **Entpacken Sie diese Datei nicht**. Die Extraktion und Installation der Datei auf dem Client erfolgt automatisch, wenn der Client die HTML-Seite für das Authentic Plug-in 22 zum ersten Mal öffnet. Der Pfad der CAB-Datei auf dem Server ist in der HTML-Seit für das Authentic Plug-in 22 definiert.

© 2018-2024 Altova GmbH

3 Einrichten des Client

Der Client-Rechner ist der Rechner, auf dem die HTML-Seite für das Authentic Plug-in[22] geöffnet und die Altova Authentic XML-Seite bearbeitet wird. Zu diesem Zweck muss das Authentic Browser Plug-in als Add-on in dem Client Browser installiert werden, über den die HTML-Seite für das Authentic Plug-in[22] geöffnet wird.

Das Authentic Browser Plug-in kann automatisch vom Server[41] im Client Browser installiert werden, wenn die HTML-Seite für das Authentic Plug-in[22] im Client Browser geöffnet wird. Alternativ dazu kann das Authentic Browser Plug-in direkt, entweder manuell[41] oder mittels eines zentral verwalteten MSI-basierten Push-Systems[42] im Client Browser installiert werden.

In diesem Abschnitt wird beschrieben, wie Sie die Client-Rechner für diese Funktionen einrichten. Es werden die folgenden Punkte beschrieben:

- die Internet Browser-Anforderungen[40], um die Authentic Browser-Funktionalität aktivieren zu können.
- die verschiedenen Methoden der Installation des Authentic Browser Plug-in[41] als Add-on in unterstützten Browsern auf dem Client-Rechner. Es wird auch beschrieben, wie Sie das Authentic Browser Plug-in upgraden[43] und deinstallieren[43].
- welche IE9-Sicherheitseinstelllungen[44] definiert werden müssen, damit das Plug-in ungehindert ausgeführt werden kann.
- welche IE10-Sicherheitseinstellungen[46] definiert werden müssen, damit das Plug-in ungehindert ausgeführt werden kann. Beachten Sie, dass in Internet Explorer 10 der Modus auf Kompatibilitätsmodus gesetzt werden muss, damit das Authentic Browser Plug-in ordnungsgemäß funktioniert.

3.1 Browser-Anforderungen

Damit eine Altova Authentic XML-Seite auf einem Client-Rechner angezeigt werden kann, wird ein Internet Browser, der Plug-ins unterstützt, benötigt. Da die Unterstützung für die NPAPI-Architektur in einer Reihe von Browsern nicht mehr angeboten wird, wird die Auhentic Browser Edition derzeit nur auf **Microsoft Internet Explorer 32-Bit ab Version 9** unterstützt. Wenn Sie einen anderen Internet Browser verwenden möchten, müssen Sie eine ältere Version, die Plug-ins unterstützt, verwenden.

Die nachstehende Liste enthält Informationen über den Grad der Unterstützung für das Authentic Browser Plug-in in den wichtigsten Internet Browsern.

- *Microsoft Internet Explorer 32-Bit* ab Version 9 unterstützt Authentic Browser.
- *Microsoft Internet Explorer 64-Bit* Version 9 ist die höchste verwendbare 64-Bit-Version. Neuere Versionen sind Microsoft Edge.
- *Microsoft Edge* unterstützt Authentic Browser nicht, da dieser Browser ActiveX Controls als Plug-ins nie unterstützt hat.

Anmerkung: Internet Explorer **muss** installiert sein, da die Benutzeroberfläche der Authentic-Ansicht (die im Browser-Fenster angezeigt wird) mit Hilfe von Internet Explorer generiert wird.

© 2018-2024 Altova GmbH

3.2 Authentic Browser Plug-in

In diesem Abschnitt werden die verschiedenen Arten der Installation von Authentic Browser auf einem Client Computer sowie das Upgrade-Verfahren und die Deinstallation von Authentic Browser beschrieben.

Installation mehrerer Versionen

Es gibt mehrere Versionen von Authentic Browser. Für jede unterstützte Sprache (Englisch, Deutsch, Spanisch, Französisch, Japanisch) gibt es jeweils eine Trusted und eine Untrusted Version für jeden unterstützten Browser (Microsoft Internet Explorer 32-Bit und 64-Bit, Mozilla Firefox).

Sie können eine, einige oder alle dieser Versionen des Authentic Browser Plug-in auf einem Client-Rechner installieren. Jede Version wird als separates Plug-in im Add-on Manager des Browsers angezeigt.

Nähere Informationen zu den verschiedenen Versionen finden Sie unter Authentic Browser Versionen. 10

3.2.1 On Demand-Installation

Die beste Art, das Authentic Browser Plug-in zu installieren, ist die automatische Installation nach Bedarf. Dabei geschieht Folgendes:

1. Die CAB-Erweiterungspakete sind auf dem Server gespeichert.
2. Wenn die HTML-Seite für das Authentic Plug-in 22 im Browser des Client geöffnet wird, wird das entsprechende Erweiterungspaket aufgrund von Anweisungen auf der HTML-Seite heruntergeladen und das Authentic Browser Plug-in wird im Client Browser installiert.

Definieren Sie die URL der CAB-Datei über das Attribut `CODEBASE` des Elements `OBJECT`. Nähere Informationen dazu finden Sie im Beispiel zu Internet Explorer 24.

3.2.2 Manuelle Installation über MSI

Es gibt keine Möglichkeit, ein in einer CAB-Datei enthaltenes Internet Explorer Add-on manuell zu installieren. Die manuelle Installation kann jedoch über MSI erfolgen.

Laden Sie eine MSI-Installationsdatei (Microsoft Windows Installer) für Authentic Browser von der Altova Website herunter. Dabei handelt es sich um eine ausführbare Installationsdatei (mit der Erweiterung `.exe`), die das Authentic Browser Plug-in in den derzeit auf dem Client-Rechner installierten unterstützten Browsern (Microsoft Internet Explorer 32-Bit und 64-Bit) installiert.

Installation

Laden Sie die MSI-Installationsdatei auf den Client-Rechner herunter und doppelklicken Sie darauf, um die Installation zu starten. Das Plug-in wird (standardmäßig im Ordner `C:\Programme (x86)\Altova\AuthenticBrowserPlugin\...`) installiert und im selben Schritt in die derzeit installierten Browser integriert. Über das Dialogfeld "Benutzerdefinierte Installation" können Sie das Plug-in selektiv für unterschiedliche Browser installieren.

Die installierte Browser-Version wird auf dem Client-Rechner in der Liste *Programme hinzufügen/entfernen* angezeigt.

MSI-Versionen

Es gibt für jede unterstützte Sprache (Englisch, Deutsch, Spanisch, Französisch, Japanisch), für jeden Trusted- und Untrusted-Typ und jeweils für die 32-Bit- und die 64-Bit-Version eine eigene MSI-Installationsdatei. So gibt es z.B. eine MSI-Installationsdatei für 32-Bit-Browser Trusted Englisch, eine weitere für 64-Bit Browser Trusted Englisch, eine weitere für 32-Bit-Browser Untrusted Englisch usw.

Nähere Informationen zu den verschiedenen Versionen finden Sie unter Authentic Browser Versionen. 10

3.2.3 Push-Installation über MSI

Wenn die Installation neuer Software zentral von der IT-Abteilung durchgeführt wird und die Installer über ein Administration Framework verteilt werden, ist die Installation über einen nativen MSI Installer (Microsoft Windows Installer) von Vorteil. Dies gilt v.a. für Internet Explorer, wo Sie zum Installieren eines Plug-in Administratorrechte benötigen.

Sie können eine MSI-Installationsdatei (Microsoft Windows Installer) für Authentic Browser von der Altova Website herunterladen. Dabei handelt es sich um eine ausführbare Installationsdatei (mit der Erweiterung `.exe`), die das Authentic Browser Plug-in in den derzeit auf dem Client-Rechner installierten unterstützten Browsern (Microsoft Internet Explorer 32-Bit und 64-Bit) installiert.

Installation

Laden Sie die MSI-Installationsdatei auf den Client-Rechner herunter und doppelklicken Sie darauf, um die Installation zu starten. Das Plug-in wird (standardmäßig im Ordner `C:\Programme (x86)\Altova\AuthenticBrowserPlugin\...`) installiert und im selben Schritt in die derzeit installierten Browser integriert. Über das Dialogfeld "Benutzerdefinierte Installation" können Sie das Plug-in selektiv für unterschiedliche Browser installieren.

Um die Installation stumm im Hintergrund auszuführen, verwenden Sie den Befehlszeilenparameter `/q`.

Die installierte Browser-Version wird auf dem Client-Rechner in der Liste *Programme hinzufügen/entfernen* angezeigt.

MSI-Versionen

Es gibt für jede unterstützte Sprache (Englisch, Deutsch, Spanisch, Französisch, Japanisch), für jeden Trusted- und Untrusted-Typ und jeweils für die 32-Bit- und die 64-Bit-Version eine eigene MSI-Installationsdatei.

© 2018-2024 Altova GmbH

So gibt es z.B. eine MSI-Installationsdatei für 32-Bit-Browser Trusted Englisch, eine weitere für 64-Bit Browser Trusted Englisch, eine weitere für 32-Bit-Browser Untrusted Englisch usw.

Nähere Informationen zu den verschiedenen Versionen finden Sie unter Authentic Browser Versionen. 10

3.2.4 Automatische Updates

Die Verfügbarkeit von Authentic Browser Upgrades und die Installation dieser Updates für Internet Explorer funktioniert folgendermaßen:

- Der `codebase` Attributwert des `object` Elements auf der HTML-Seite für das Authentic Plug-in 22 muss das Suffix `#Version=...` erhalten, um die Versionsnummer des auf dem Server verfügbaren Plug-in anzugeben.
- Wenn die Version des auf dem Client Computer installierten Plug-in niedriger ist als diese Version, wird der Benutzer gefragt, ob ein Upgrade für das Plug-in installiert werden soll.

3.2.5 Deinstallation, Deaktivierung

Wenn die Installation über MSI erfolgt ist, so wird die installierte Browser-Version auf dem Client-Rechner in der Liste *Programme hinzufügen/entfernen* angezeigt. Wenn Sie das Produkt über die Liste *Programme hinzufügen/entfernen* deinstallieren, wird es aus dem Browser und der MSI-Liste entfernt.

Wenn ein Plug-in in einem Browser installiert wurde, wird es im Add-on Manager des Browsers als aktiviert angezeigt. Wenn Sie das Plug-in aus dem Add-on Manager des Browsers entfernen, wird es deaktiviert, bleibt aber auf dem Rechner installiert.

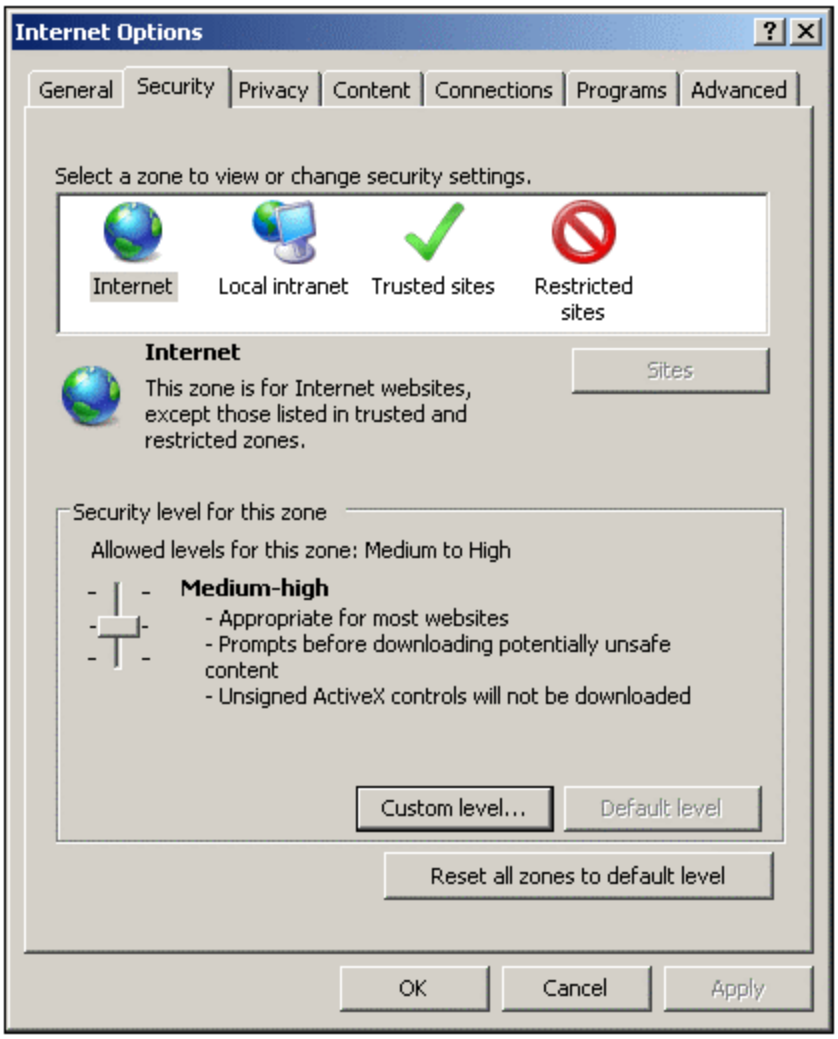
3.3 IE9 Sicherheitseinstellungen

Wenn die Untrusted Version von Authentic Browser[10] in Internet Explorer 9 verwendet wird, wird möglicherweise die folgende Meldung angezeigt:

Ein ActiveX-Steuerelement wurde blockiert. Diese Seite wird eventuell nicht richtig angezeigt.

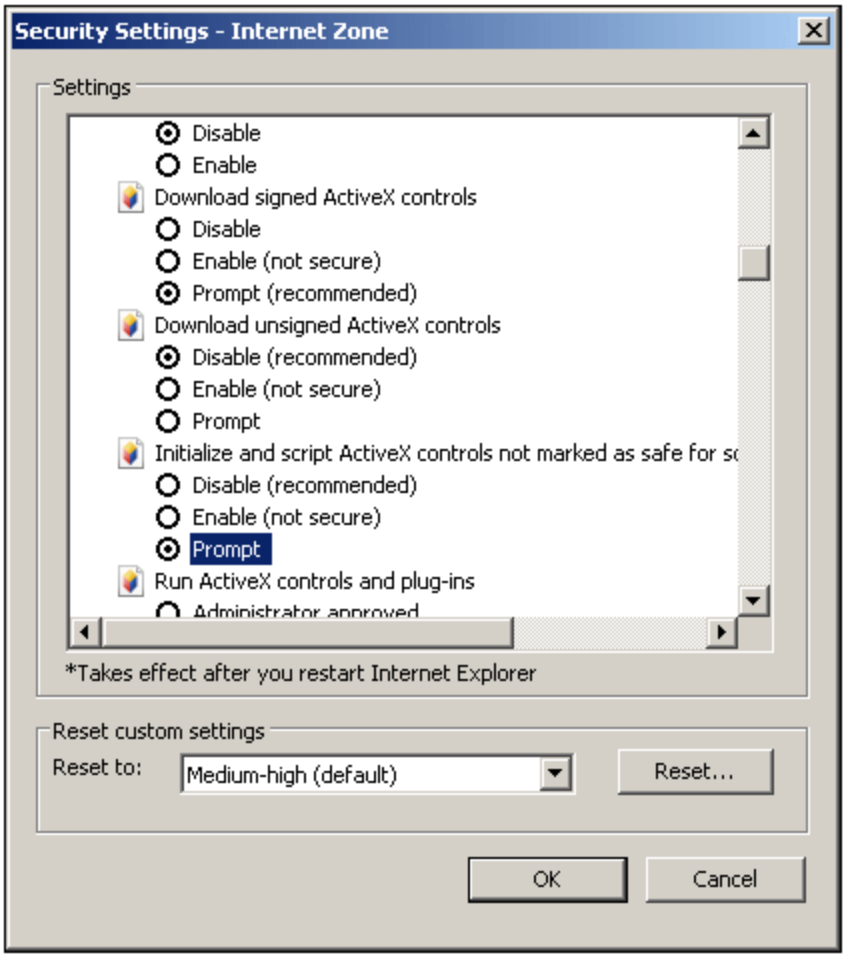
Um ActiveX Controls zu aktivieren, müssen Sie die entsprechende Sicherheitsoption in den Internet-Optionen von Internet Explorer einstellen. Gehen Sie dazu folgendermaßen vor:

1. Öffnen Sie das Dialogfeld "Internetoptionen" (*Abbildung unten*) von Internet Explorer durch Auswahl des Menübefehls **Extras | Internetoptionen**.



2. Wählen Sie das Register "Sicherheit" (*siehe Abbildung oben*) und anschließend die Zone, für die Sie die Einstellung vornehmen möchten (Internet oder lokales Intranet).
3. Klicken Sie anschließend auf die Schaltfläche "**Stufe anpassen**". Daraufhin wird das Dialogfeld "Sicherheitseinstellungen" angezeigt. (*Abbildung unten*).

© 2018-2024 Altova GmbH



4. Scrollen Sie hinunter zum Abschnitt *ActiveX-Steuerelemente und Plugins* und innerhalb dieses Abschnitts zur Einstellung *ActiveX-Steuerelemente initialisieren und ausführen, die nicht als "sicher für Skripting" markiert sind*. Wählen Sie die Option *Bestätigen* (*siehe Abbildung oben*).
5. Starten Sie Internet Explorer neu, damit die neue Einstellung wirksam wird.

Ab diesem Zeitpunkt werden Sie, jedes Mal, wenn eine HTML-Seite für das Authentic Plug-in[22] versucht, eine ActiveX-Datei zu laden, von Internet Explorer gefragt, ob das Control geladen werden soll oder nicht.

3.4 IE 10 Sicherheitseinstellungen

Internet Explorer 10 (IE10) hat zwei Browser-Modi:

- Metro-Stil: hierbei wird das Programm im Enhanced Protection-Sicherheitsmodus ausgeführt und
- Desktop, bei dem die Sicherheitseinstellungen im Kompatibilitätsmodus verwendet werden.

Um das Authentic Browser Plug-in ausführen zu können, müssen Sie den IE10 Kompatibilitätsmodus (**Extras| Optionen | Sicherheit**) einstellen.

© 2018-2024 Altova GmbH

4 Benutzerreferenz

In diesem Abschnitt sind die Mechanismen, Objekte und Enumerationen von Authentic Browser aufgelistet und beschrieben.

4.1 Mechanismen

Dieser Abschnitt enthält eine Beschreibung einiger der am häufigsten verwendeten Authentic Browser-Mechanismen.

4.1.1 Ereignisse: Connection Point für IE

In Authentic Browser steht eine Reihe von Connection Point-Ereignissen zur Verfügung (siehe auch Ereignisse: Referenz 50), für die Sie Ereignishandler in den SCRIPT-Blöcken auf Ihrer HTML-Seite bereitstellen können.

Anmerkung: Die Beschreibung in diesem Abschnitt bezieht sich **nur auf Internet Explorer**.

In den folgenden Beispielen sehen Sie `SCRIPT` Blöcke für die Ereignisse "ControlInitialized" und "SelectionChanged":

`ControlInitialized`
Dieses Connection Point-Ereignis wird sofort nach Erstellung und Initialisierung des Controls ausgelöst. Ein zusätzliches Initialisierungsskript für das Control kann innerhalb der `ControlInitialized`-Ereignisbehandlungsroutine abgearbeitet werden.

```
<SCRIPT LANGUAGE=javascript FOR=objPlugIn EVENT=controlinitialized>
	// add your code here
</SCRIPT>
```

`SelectionChanged`
Das `SelectionChanged`-Ereignis wird jedes Mal ausgelöst, wenn die aktuelle Auswahl in der Ansicht geändert wird. Verwenden Sie einen `SCRIPT`-Block, um den jeweiligen Code für das Ereignis auszuführen.

```
<SCRIPT LANGUAGE=javascript FOR=objPlugIn EVENT=selectionchanged>
	// add your code here
</SCRIPT>
```

Bitte beachten Sie, dass das Authentic.event 78-Objekt nicht befüllt wird, wenn dieses Ereignis vorkommt. Wenn in Ihrem Skript Ereignishandler registriert sind, enthalten die Eigenschaften des Authentic.event 78-Objekts Werte aus dem letzten Ereignis, das aufgetreten ist.

Das Authentic.CurrentSelection 74-Objekt enthält nun gültige Informationen.

Laden von SPS-, XSD- und XML-Dateien ohne Zutun des Benutzers

Wenn die `.sps`-, `.xsd`- und die `.xml`-Datei beim Laden der HTML-Seite ohne Zutun des Benutzers geladen werden sollen, ist die beste Methode, einen Ereignishandler zu erzeugen, der das `ControlInitialized`-Event verarbeitet. Alternativ dazu können Sie auch ein PropertyBag verwenden (siehe zweites Beispiel):

Empfohlene Methode:

```
<SCRIPT LANGUAGE="javascript" FOR=objPlugIn EVENT="ControlInitialized">
objPlugIn.SchemaLoadObject.URL = "http://yourserver/OrgChart.xsd"
```


© 2018-2024 Altova GmbH

```
objPlugIn.XMLDataLoadObject.URL = "http://yourserver/OrgChart.xml"
objPlugIn.DesignDataLoadObject.URL = "http://yourserver/OrgChart.sps"
objPlugIn.StartEditing()
</SCRIPT>
```

oder

```
<OBJECT id=objPlugIn style="WIDTH:600px; HEIGHT:500px"
codeBase="http://yourserver/cabfiles/AuthenticBrowserEdition.CAB#Version=12,3,0,0"
classid=clsid:B4628728-E3F0-44a2-BEC8-F838555AE780>
<PARAM NAME="XMLDataURL" VALUE="http://yourserver/OrgChart.xml">
<PARAM NAME="SPSDataURL" VALUE="http://yourserver/OrgChart.sps">
<PARAM NAME="SchemaDataURL" VALUE="http://yourserver/OrgChart.xsd">
</OBJECT>
```

Es ist nicht ratsam, diese Dateien in einen Ereignishandler zu laden, der das "onload"-Ereignis des "body"-Elements abarbeitet, da das Authentic Browser Plug-In-Control möglicherweise erst initialisiert wird, nachdem das "onload"-Ereignis ausgelöst wurde. In diesem Fall stehen die Methoden und Eigenschaften des Plug-In nicht zur Verfügung und die Dateien werden nicht geladen. Nähere Informationen dazu finden Sie auch im Kapitel Das OBJECT-Element 24.

Nicht empfohlen:

```
<SCRIPT LANGUAGE="javascript">
 function load () {

objPlugIn.SchemaLoadObject.URL = "http://yourserver/OrgChart.xsd"
objPlugIn.XMLDataLoadObject.URL = "http://yourserver/OrgChart.xml"
objPlugIn.DesignDataLoadObject.URL = "http://yourserver/OrgChart.sps"
objPlugIn.StartEditing()
}
</SCRIPT>

<body onload = "load files">
```

4.1.2 Ereignisse: Symbolleisten-Schaltfläche

Jede Symbolleisten-Schaltfläche hat ein Standardverhalten, das unter Umständen geändert werden muss. Mit Hilfe des `AuthenticCommand`-Ereignisses können Sie zusätzliche Aufgaben hinzufügen oder das Standardverhalten eines solchen Schaltflächenbefehls völlig neu definieren. In Skripts kann das `AuthenticCommand`-Ereignis dazu verwendet werden, um jedes Mal, wenn der Benutzer auf eine Symbolleisten-Schaltfläche klickt, eine Benachrichtigung zu erhalten. Beachten Sie bitte, dass jeder Befehl (aus der Authentic.UICommands 95-Sammlung) mit einem bestimmten Ereignis verknüpft ist. Um festzustellen, auf welches Symbol der Benutzer geklickt hat, muss das Skript die AuthenticEvent.srcElement 107-Eigenschaft überprüfen, die eine Referenz auf das entsprechende AuthenticCommand 98-Objekt enthält.

Beispiel

```
// event handler for OnDocEditCommand
function OnCommand()
```

```
{
	// we are interested in the k_CommandSave button
	if(objPlugIn.event.srcElement.CommandID == 1)
	{
		// instead of the standard HTTP PUT we want to use
		// a HTTP POST
		objPlugIn.SavePOST();

		// no standard execution follows
		objPlugIn.event.cancelBubble = true;
	}
}
// somewhere in your script
function MyInit()
{
	objPlugIn.attachCallBack("OnDocEditCommand", OnCommand);
}
```

Referenz
Welche Befehle zur Verfügung stehen, finden Sie unter AuthenticToolbarButton.CommandID[145].

4.1.3 Ereignisse: Referenz

Liste von Connection Point Events

Name	seit TypeLib	Beschreibung
SelectionChanged	1.0	Die Auswahl im Editor hat sich geändert. Das Authentic.CurrentSelection[74] Objekt enthält nun gültige Informationen. Die Eigenschaften des Authentic.event[78] Objekts sind nicht definiert.
ControlInitialized	1.2	Das Control ist nun zur Gänze geladen und initialisiert. Die Eigenschaften des Authentic.event[78] Objekts sind nicht definiert.
dragover	1.8	Eine Drag-over-Operation findet statt.
drop	1.8	Eine Drop-Operation wurde ausgeführt.
keydown	1.8	Eine Taste wurde gedrückt und noch nicht losgelassen.
keyup	1.8	Eine Taste wurde losgelassen. Dies ist normalerweise das Ereignis, das auf einen Tastendruck (keystroke) folgt.
keypressed	1.8	Wird bei jeder Eingabe über die Tastatur ausgelöst.
mousemove	1.8	Der Mauszeiger wurde bewegt.
buttonup	1.8	Eine der Maustasten wurde losgelassen.
buttondown	1.8	Eine der Maustasten wurde gedrückt.
contextmenu	1.8	Wird gesendet nach Empfang von WM_CONTEXTMENU.
editpaste	1.8	Wird aufgerufen, bevor eine Einfügeoperation durchgeführt wird.

© 2018-2024 Altova GmbH

editcut	1.8	Wird aufgerufen, bevor eine Ausschneideoperation durchgeführt wird.
editcopy	1.8	Wird aufgerufen, bevor eine Kopieroperation durchgeführt wird.
editclear	1.8	Wird aufgerufen, bevor eine Löschoperation durchgeführt wird.
doceditcommand	1.8	Wird bei Ausführung eines Authentic-Befehls und zur Implementierung eines benutzerdefinierten Befehls-Handlers ausgelöst. Siehe auch Ereignisse: Symbolleisten-Schaltflächen[49]. Die Eigenschaften des Authentic.event[78] Objekts sind nicht definiert.
buttondoubleclick	1.8	Doppelklick auf eine der Maustasten.

Sofern keine Ausnahme in der Beschreibung erwähnt ist, sind die Eigenschaften des Authentic.event[78] Objekts nicht definiert.

Anmerkung: Diese Event Handler sind synchron, d.h. sie werden unmittelbar nach dem Auftreten des Event aufgerufen.

4.1.4 Aufrufen und Ändern von Dokumentinhalt

Zum Aufrufen und Ändern von Dokumentinhalt können Sie das `AuthenticRange-`, `AuthenticView-` und das `Authentic`-Objekt (und ihre Eigenschaften und Methoden) verwenden.

Wo sich die Funktionen der `AuthenticRange-` und der `AuthenticView`-Schnittstelle einerseits und der vom `Authentic`-Objekt bereitgestellten Schnittstelle auf der anderen Seite überlappen, sollte man der `AuthenticRange-` und der `AuthenticView`-Schnittstelle den Vorzug geben. Diese überlappenden Funktionalitäten des `Authentic`-Objekts werden allmählich eingestellt und in zukünftigen Versionen nicht mehr zur Verfügung stehen.

4.1.5 Bearbeitungsoperationen

Wenn XML-Daten in der Authentic-Ansicht angezeigt werden, können Sie einzelne Elemente mit Hilfe der Standardbearbeitungsoperationen Ausschneiden, Kopieren und Einfügen manipulieren. Es dürfen jedoch nicht alle XML-Datenelemente editiert werden. Sie müssen daher zuerst testen, ob eine Bearbeitung möglich ist. Dabei wird folgendermaßen vorgegangen: Überprüfen Sie zuerst, ob die jeweilige Bearbeitungsoperation aktiviert ist. Ist dies der Fall, rufen Sie die entsprechende Editier-Methode auf. Die einzige Methode, für die kein Test verfügbar ist, ist die `EditSelectAll`-Methode, die automatisch alle im Dokument angezeigten Elemente auswählt.

Nachstehend sehen Sie eine Liste von Eigenschaften und Methoden, die Bearbeitungsoperationen durchführen. Jede Eigenschaft gibt einen Boolschen Wert zurück. Die Methoden haben keine Parameter.

IsEditUndoEnabled	Authentic.EditUndo[77]	Macht eine Bearbeitungsoperation rückgängig
IsEditRedoEnabled	Authentic.EditRedo[76]	Stellt eine Bearbeitungsoperation wieder her
IsEditCopyEnabled	Authentic.EditCopy[76]	Kopiert den markierten Text in die Zwischenablage

IsEditCutEnabled	Authentic.EditCut 76	Schneidet den markierten Text aus und legt ihn in die Zwischenablage
IsEditPasteEnabled	Authentic.EditPaste 76	Fügt den Text aus der Zwischenablage an der Cursorposition ein
IsEditClearEnabled	Authentic.EditClear 75	Löscht den markierten Text aus dem XML-Dokument

4.1.6 Suchen und Ersetzen

Die Authentic.FindDialog 78-Methode öffnet ein Suchdialogfeld, in welches der Benutzer einen Suchbegriff eingeben kann. Mit Hilfe der Authentic.FindNext 79-Methode wird die nächste Instanz desselben Elements gesucht. Die `FindNext`-Methode kann mit Hilfe der Boolschen Eigenschaft `IsFindNextEnabled` getestet werden. Eine Variante der `FindDialog`-Methode ist die Authentic.ReplaceDialog 88-Methode. Mit dieser Operation wird ein bestimmtes Element gesucht und durch einen bestimmten Wert, den der Benutzer in das Ersetzungsdialogfeld eingegeben hat, ersetzt. Bei Aufruf der `FindNext`-Methode wird das nächste Element gesucht und ersetzt.

4.1.7 Zeilenoperationen

Sie können in der Authentic-Ansicht eine Struktur sich wiederholender Elemente als dynamische Tabelle anlegen, in der jede Zeile eine Instanz des sich wiederholenden Elements darstellt. Nachdem Sie die dynamische Tabelle erstellt haben, kann der Benutzer, der mit der Authentic-Ansicht arbeitet, die Zeilen und deren Daten manipulieren. Diese Zeilenoperationen können mit Hilfe eines Skripts ausgeführt werden.

Wenn die Zeilenoperationen von einem externen Skript ausgeführt werden sollen, sind zwei Schritte erforderlich:

- Zuerst wird überprüft, ob die Zeile, in der sich der Cursor befindet, eine Eigenschaft verwendet. Eine Eigenschaft wie z.B. `IsRowInsertEnabled` wird verwendet und gibt den Wert "True" oder "False" zurück.
- Wenn der zurückgegebene Wert "True" ist, kann die benötigte Zeilenmethode aufgerufen werden.

Im Folgenden sehen Sie eine Liste von Eigenschaften und Methoden, die Zeilenoperationen ausführen. Jede Eigenschaft gibt einen Boolschen Wert zurück und die Methoden haben keine Parameter.

IsRowInsertEnabled	Authentic.RowInsert 90	Zeileneinfügeoperation
IsRowAppendEnabled	Authentic.RowAppend 89	Operation zum Anhängen einer Zeile
IsRowDeleteEnabled	Authentic.RowDelete 89	Operation zum Löschen einer Zeile
IsRowMoveUpEnabled	Authentic.RowMoveUp 90	Verschiebt die XML-Daten um eine Zeile nach oben
IsRowMoveDownEnabled	Authentic.RowMoveDown 90	Verschiebt die XML-Daten um eine Zeile nach unten
IsRowDuplicateEnabled	Authentic.RowDuplicate 89	Dupliziert die aktuelle Zeile

© 2018-2024 Altova GmbH

4.1.8 Tastaturkürzel

Falls Authentic Browser sich im Eingabemodus befindet, stehen die folgenden Tastaturkürzel zur Verfügung:

Strg + P	Dokument drucken
Strg + Z	Rückgängig
Strg + Y	Wiederherstellen
Strg + X	Ausschneiden
Strg + C	Kopieren
Strg + V	Einfügen
Strg + A	Alle auswählen
Strg + F	Suchdialogfeld öffnen
Strg + H	Ersetzungsdialogfeld öffnen

4.1.9 Textstatus-Schaltflächen

Die Symbolleiste der Authentic-Ansicht von Authentic Browser unterstützt Textstatussymbole. Dabei handelt es sich um Symbole, die Elemente mit vordefinierten Textformatierungseigenschaften einfügen. Um diese Funktion verwenden zu können, muss das Element, das als Textstatus-Symbol erstellt werden soll

- im Schema als globales Template deklariert sein und
- das SPS muss die erforderlichen Definitionen enthalten (siehe StyleVision-Dokumentation)

Das Plug-In benötigt den Pfad zum `.bmp`, das als Textstatus-Symbol oder benutzerdefinierte Schaltfläche in der Symbolleiste verwendet werden soll. Diese URL zur Bilddatei des Symbols wird als der Wert des Authentic.TextStateBmpURL 94 .-Ereignisses verwendet.

4.1.10 Eingabehilfen

Authentic Browser gestattet Ihnen, Eingabehilfen wie in XMLSpy bereitzustellen. Auf diese Art hat ein Benutzer, der in der Authentic-Ansicht arbeitet, jederzeit Zugriff auf die Elemente, Attribute und Entities, die an einer bestimmten Stelle im Dokument zulässig sind. Die Eingabehilfen sind standardmäßig deaktiviert und werden nicht angezeigt, wenn sie nicht explizit aktiviert werden. Sie können die Eingabehilfen über die folgenden `PROPERTYBAG`-Parameter im `OBJECT`-Tag von Internet Explorer 24 aktivieren:

`EntryHelpersEnabled`	TRUE / FALSE
`EntryHelperSize`	Größe in Pixeln
`EntryHelperAlignment`	Nimmt die SPYAuthenticToolbarAllignment 195 -Werte auf
`EntryHelperWindows`	Nimmt die SPYAuthenticEntryHelperWindows 194 -Werte auf. Jede Kombination ist zulässig (bit-check)

Anstatt die Parameter im `OBJECT`-Tag zu verwenden, können Sie auch Eigenschaften und Methoden in der API verwenden:

Eigenschaften

- Authentic.EntryHelpersEnabled 77
- Authentic.EntryHelperAlignment 77
- Authentic.EntryHelperSize 78
- Authentic.EntryHelperWindows 78

Methode

- Authentic.RedrawEntryHelpers 88

4.1.11 Pakete

Das Authentic Browser Plug-in unterstützt Zusatzpakete, die Funktionalitäten des Programmmoduls erweitern. Die Pakete sind auf dem Server gespeichert und werden bei Bedarf über die Authentic Browser-Benutzeroberfläche lokal (auf dem Client) installiert. Sobald ein Paket lokal installiert ist, wird es bei jedem Start aktiviert, bis es von einem Benutzer entfernt wird.

Derzeit unterstützt Authentic Browser Rechtschreibpakete.

Dieser Abschnitt ist in zwei Teile gegliedert:

- Arbeiten mit Paketen 54: Hier wird beschrieben, wie die Verwendung der Pakete funktioniert.
- Rechtschreibprüfungspakete 56: Hier wird beschrieben, wie die Rechtschreibprüfungspakete auf dem Server gespeichert, installiert und verwendet werden.

4.1.11.1 Arbeiten mit Paketen

Pakete sind auf dem Server gespeichert und werden bei Bedarf über die Authentic Browser-Benutzeroberfläche lokal installiert. Sobald ein Paket lokal (auf dem Client) installiert ist, wird es bei jedem Start aktiviert, bis es vom Benutzer entfernt wird.

Paketverwaltung

Die Paketverwaltung ist eine Authentic Browser-Funktionalität, mit der der Benutzer von Authentic Browser Kontrolle über verfügbare Pakete hat. Diese Funktion steht über das Dialogfeld "Paketverwaltung" zur Verfügung.

Um die Paketverwaltungsfunktion zu aktivieren, müssen Sie den `LoaderSettingsFileURL`-Parameter auf der HTML-Seite für das Authentic Plug-in 22 verwenden. Dieser Parameter definiert die URL der LoaderSettings-Datei für die Paketverwaltung (die sich unter jedem Pfad befinden kann, auf den Zugriff besteht):

Auf einer HTML-Seite für Internet Explorer 24 wird der `LoaderSettingsFileURL` Parameter als Child `PARAM`-Element des `OBJECT`-Elements verwendet:

```
    <OBJECT>
  ...
  <PARAM NAME="LoaderSettingsFileURL"
VALUE="http://www.server.com/AuthenticFiles/XMLSpyPlugInLoaderSettings.xml"/>
```


© 2018-2024 Altova GmbH

```
</OBJECT>
```

Die Datei LoaderSettings enthält die Paketdefinitionen zusammen mit den URLs der jeweiligen Pakete. Unten finden Sie eine Liste der Datei LoaderSettings.

Beispiel für eine LoaderSettings XML-Datei

Das Dokument-Element der LoaderSettings XML-Datei ist `loadersettings` (*siehe Liste unten*). Dieses Element kann mehrere Child `package`-Elemente und/oder mehrere `zippackage`-Elemente enthalten.

- Mit dem `package`-Element werden die älteren `.pck` Rechtschreibprüfungspakete referenziert. Es wird von Authentic Browser Versionen vor Version 2011r3 verwendet. Diese Authentic Browser Versionen ignorieren das `zippackage`-Element.
- Mit dem `zippackage`-Element werden die `.zip` Rechtschreibprüfungspakete referenziert. Es wird von Authentic Browser Versionen ab Version 2011r3 verwendet. Diese Authentic Browser Versionen ignorieren das `package` Element.

Nähere Informationen zu den Rechtschreibprüfungspaketen finden Sie im Abschnitt Rechtschreibprüfungspakete 56.

```
<?xml version="1.0" encoding="UTF-8"?>
<loadersettings>

   <zippackage mode="user_demand"  category="spelling">
      <packageurl>Portuguese (Brazilian).zip</packageurl>
      <description>Portuguese (BR) language pack.</description>
   </zippackage>

   <zippackage mode="user_demand"  category="spelling">
      <packageurl>Portuguese (Brazilian).zip</packageurl>
      <description>Portuguese (BR) language pack.</description>
   </zippackage>

   <package mode="user_demand" id="SentrySpellChecker_EAM_only"
      category="spelling" version="1">
      <packageurl>PlugIn/SentrySpellChecker_EAM_only.pck</packageurl>
      <description>Sentry Spellchecker (EN-US)</description>
   </package>

   <package mode="user_demand" id="SentrySpellChecker_EALL"
      category="spelling" version="1">
      <packageurl>PlugIn/SentrySpellChecker_EALL.pck</packageurl>
      <description>Sentry SpellChecker EN with Legal and Medical.</description>
   </package>

</loadersettings>
```

Beachten Sie bitte die folgenden Punkte:

- Der Pfad eines Pakets muss als Inhalt des packageurl Child-Elements des `package` oder `zippackage`-Elements definiert werden. Der Speicherpfad kann absolut oder relativ zur HTML-Datei

sein, die die LoaderSettings Datei aufruft. Das Paket kann sich in jedem beliebigen Ordner auf dem Server befinden.

- Wenn Sie ein Paket aus der XML-Datei entfernen (durch Entfernen eines `package`-Elements), steht das Paket nicht für die Installation auf dem Client zur Verfügung.

- Das mode Attribut des `package` Elements definiert, wieviel Einfluss der Benutzer darauf hat, ob das Paket installiert werden soll. Es können die folgenden Werte verwendet werden:

  1. user: In diesem Fall wird der Benutzer bei jedem Start von Authentic Browser gefragt, ob das Paket installiert werden soll.
  2. user_demand: Installiert das Paket, wenn dies vom Benutzer ausdrücklich verlangt wird. Der Benutzer kann dies entweder über die Symbolleisten-Schaltfläche zur Rechtschreibprüfung oder oder über das Dialogfeld "Paketverwaltung" anfordern.
  3. force: In diesem Fall wird das Paket ohne Benachrichtigung des Benutzers installiert.

- Der Inhalt des `description` Child-Elements von `package` kann bearbeitet werden und kann als beschreibender Text im Dialogfeld "Paketverwaltung" verwendet werden.

Einmalige Installation von Paketen

Wenn eine neuere Version von Authentic Browser installiert wird, müssen bereits zuvor installierte Pakete auf dem Client nicht neu installiert werden. Das bereits installierte Paket wird von der neueren Version verwendet. Authentic Browser auf dem Client PC liest die LoaderSettings-Datei auf dem Server, ruft die Paketinformationen aus dieser Datei ab und überprüft, ob das Paket lokal installiert ist.

Anmerkung: In Version 2011r3 und höher unterstützte Pakete (`.zip`-Pakete) verwenden eine andere Rechtschreibprüfung als die Rechtschreibprüfungspakete früherer Versionen (`.pck`-Pakete). Authentic Browser Versionen ab Version 2011r3 verwenden die `.zip` Rechtschreibprüfungspakete. Ältere Versionen von Authentic Browser verwenden die `.pck` Rechtschreibprüfungspakete.

4.1.11.2 Rechtschreibprüfungspakete

In Authentic Browser werden ab 2011r3 Hunspell-Wörterbücher verwendet.

Installationspfad

Diese Wörterbücher befinden sich auf einem Client-Rechner im Ordner `Lexicons` unter dem folgenden Pfad:

Auf Windows 7/8: `C:\ProgramData\Altova\SpellChecker\Lexicons`
Auf Windows XP: `C:\Documents and Settings\All Users\Application Data\Altova\SharedBetweenVersions\SpellChecker\Lexicons`

Wenn auf dem Client-Rechner ein Altova-Produkt installiert wurde, so wurden unter den oben angeführten Pfaden im Ordner `Lexicons` einige integrierte Wörterbücher installiert. Alle Benutzer des Rechners und alle verschiedenen Hauptversionen von Altova-Produkten (sowohl 32-Bit- als auch 64-Bit-Versionen) haben gemeinsamen Zugriff auf diese installierten Wörterbücher.

© 2018-2024 Altova GmbH

Zusätzliche Wörterbücher können von der Altova Website heruntergeladen werden. Des Weiteren können Hunspell-Wörterbücher auch von anderen Webseiten heruntergeladen werden, z.B. von http://wiki.services.openoffice.org/wiki/Dictionaries oder http://extensions.services.openoffice.org/en/dictionaries. Beachten Sie außerdem, dass, da Hunspell-Wörterbücher auf Myspell-Wörterbüchern basieren, auch Myspell-Wörterbücher im Ordner `Lexicons` installiert werden können. OpenOffice verpackt Wörterbuchdateien als `.oxt` Dateien. Sie können daher die Dateierweiterung in `.zip` ändern und anschließend die benötigten `.aff` und `.dic` Dateien entpacken. Der Benutzer muss sicherstellen, dass er die Lizenzbedingungen für die Verwendung der installierten Wörterbücher erfüllt.

Anmerkung: Die Auswahl der integrierten Wörterbücher, die mit Altova Software verfügbar sind, ist nicht Ausdruck einer Präferenz für bestimmte Sprachen durch Altova, sondern ist größtenteils auf der Verfügbarkeit der Wörterbücher begründet, die mit kommerzieller Software im Rahmen von Lizenzen wie z.B. MPL-, LGPL- oder BSD-Lizenzen weitergegeben werden dürfen. Es gibt noch viele weitere Open Source-Wörterbücher, doch werden diese mit größeren Lizenzeinschränkungen wie z.B. im Rahmen von GPL Lizenzen weitergegeben. Viele dieser Wörterbücher stehen als Teil eines separaten Installationsprogramms unter http://www.altova.com/de/dictionaries zur Verfügung. Es bleibt Ihnen überlassen, ob Sie mit den für das jeweilige Wörterbuch geltenden Lizenzbedingungen einverstanden sind und ob das Wörterbuch sich für die Verwendung mit der auf Ihrem Rechner installierten Software eignet

Wörterbuchdateien und Wörterbuchordnerstruktur auf dem Client

Jedes installierte Wörterbuch auf dem Client-Rechner besteht aus zwei Hunspell-Wörterbuchdateien, die zusammen verwendet werden: einer `.aff`-Datei und einer `.dic`-Datei.

Alle Sprachwörterbücher werden unter den oben aufgelisteten Pfaden im Ordner `Lexicons` installiert. Im Ordner `Lexicons` werden die unterschiedlichen Sprachwörterbücher in jeweils unterschiedlichen Ordnern gespeichert: `<Sprachname>\<Wörterbuchdateien>`. Die Namen dieser Sprachordner werden in Authentic Browser als Namen der Wörterbücher angezeigt. Im Folgenden finden Sie eine Liste der Struktur des Ordners `Lexicons` und der entsprechenden Wörterbuchnamen, wie sie in Authentic Browser angezeigt würden.

```
Lexicons
       |
       |-- Englisch (Britisch)        Wörterbuchname: Englisch (Britisch)
       |          |
       |          |-- .aff Datei
       |          |   .dic Datei
       |
       |-- Englisch (US)              Wörterbuchname: Englisch (US)
       |          |
       |          |-- .aff Datei
       |          |   .dic Datei
       |
       |-- Deutsch                    Wörterbuchname: Deutsch
       |          |
       |          |-- .aff Datei
       |          |   .dic Datei
```

Wörerbuchdateien auf dem Server

Ein Sprachwörterbuch muss auf dem Server als ZIP-Datei gespeichert werden. Wenn die ZIP-Datei auf den Client heruntergeladen und installiert wird, wird im Ordner `Lexicons` ein Sprachordner angelegt (siehe Installationspfad oben) und die entpackte `.aff-` und `.dic`-Datei wird in diesem Ordner gespeichert. So wird z.B. mit der Datei `English (British).zip` der Ordner: `<path>\Lexicons\English (British)` angelegt und die `.aff-` und die `.dic`-Dateien werden in diesem Ordner gespeichert.

Es können auch mehrere ZIP-Datein in einer einzigen ZIP-Datei verpackt werden und diese einzige Datei kann auf den Server platziert werden. Wenn diese ZIP-Datei auf den Client heruntegeladen und dort installiert wird, werden die Namen der innersten ZIP-Dateien als Namen für die Ordner im Ordner "Lexicons" verwendet. So können etwa die Dateien `English (British).zip` und `English (US).zip` in einer Datei namens `English.zip` verpackt werden, die auf den Server gestellt werden kann. Bei der Installation werden im Ordner `Lexicons` zwei Ordner angelegt: `<path>\Lexicons\English (British)` und `<path>\Lexicons\English (US)`.

Funktionsweise der Wörterbuchinstallation

Wenn eine HTML-Seite für das Authentic Plug-in geöffnet wird, können die Wörterbuchpakete auf dem Client entweder automatisch oder auf Anfrage des Benutzers installiert werden. Die Installation erfolgt auf folgende Weise:

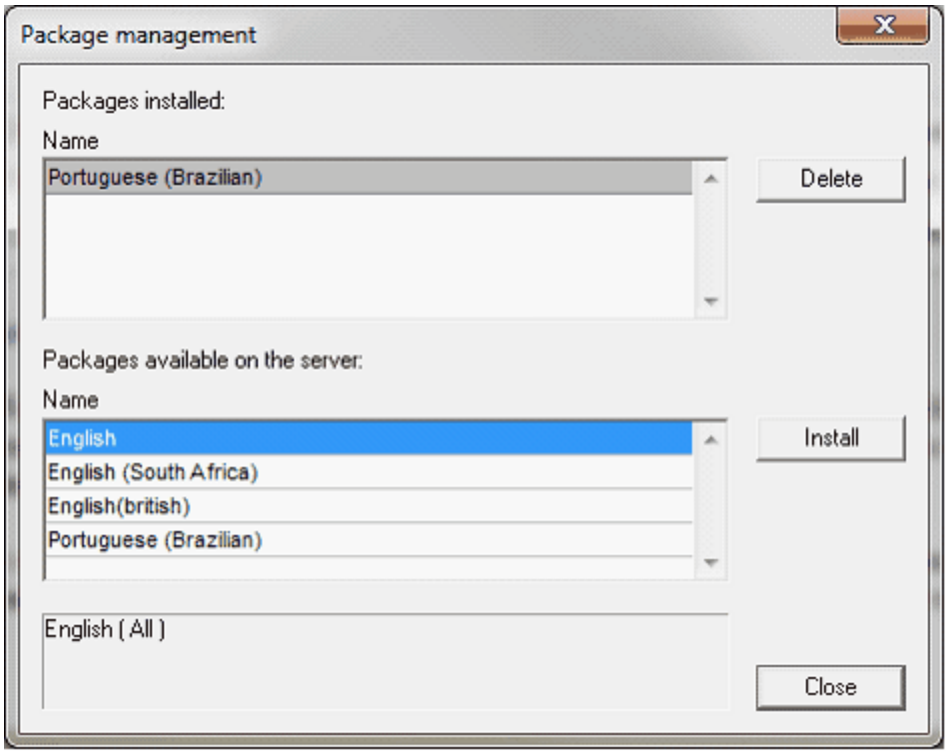
1. Die HTML-Seite, die in Authentic Browser geöffnet wird, referenziert die LoaderSettings-Datei 54.
2. Die LoaderSettings-Datei 54 referenziert die Rechschreibprüfungspakete auf dem Server. Die LoaderSettings-Datei 54 definiert außerdem für jedes Rechtschreibprüfungspaket, ob dieses Paket unmittelbar nach dem Laden der HTML-Seite automatisch installiert werden soll oder ob der Benutzer gefragt werden soll, ob das Paket installiert werden soll.
3. Das Rechtschreibprüfungspaket wird aus der gezippten Datei vom Server aus auf dem Client installiert.

Wörterbuchinstallation

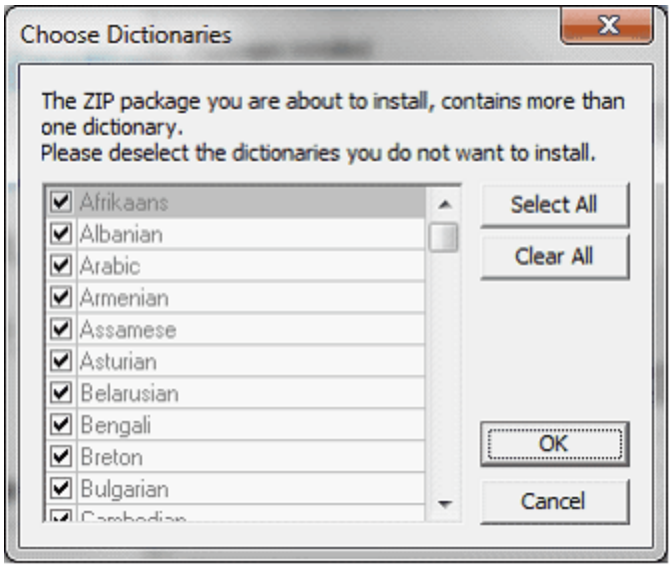
Die Wörterbuchpakete werden auf dem Server als ZIP (`.zip`)-Dateien gespeichert und können lokal über die Authentic Browser Benutzeroberfläche installiert werden. Um ein Rechtschreibprüfungspaket zu installieren, klicken Sie auf der Benutzeroberfläche von Authentic Browser auf die Schaltfläche "Paketverwaltung" Daraufhin wird das Dialogfeld "Paketverwaltung" (*Abbildung unten*) aufgerufen. Das Dialogfeld "Paketverwaltung" wird auch dann angezeigt, wenn kein Wörterbuch installiert ist und Sie auf die Schaltfläche "Rechtschreibprüfung" klicken.

Im Dialogfeld "Paketverwaltung" (*Abbildung unten*) werden alle auf dem Server verfügbaren Pakete im unteren Bereich aufgelistete. Bei den angezeigten Namen handelt es sich um die Namen der ZIP-Dateien auf dem Server. Wählen Sie die gewünschten Dateien aus und klicken Sie auf **Installieren.** Um installierte Pakete zu löschen, wählen Sie sie im oberen Bereich aus und klicken Sie auf **Löschen**.

© 2018-2024 Altova GmbH



Wenn eine für die Installation ausgewählte ZIP-Datei mehrere ZIP-Pakete enthält, wird das Dialogfeld "Pakete auswählen" (*Abbildung unten)* angezeigt. In diesem Dialogfeld können Sie die zu installierenden Pakete mit einem Häckchen versehen bzw. dieses bei Paketen, die nicht installiert werden sollen, entfernen. Klicken Sie auf **OK,** um die ausgewählten Pakete zu installieren. Auf diese Art kann jedes einzelne Sprachwörterbuch separat auf dem Server oder in einer einzigen ZIP-Datei gespeichert werden.



Wenn bei der Installation eines Pakets ein vorhandenes Paket überschrieben wird, so wird eine entsprechende Warnmeldung angezeigt. Der Benutzer von Authentic Browser kann den Vorgang dann fortsetzen oder abbrechen.

4.1.12 Verwendung von XMLData

Mit XMLData haben Sie Zugriff auf die Elemente der aktuell angezeigten XML-Datei. Sie können damit alle erforderlichen Änderungen an den Elementen der XML-Schemastruktur vornehmen. Die Hauptfunktion von XMLData ist:

1. Zugriff auf die Namen und Werte jeder Art von Datenelementen (z.B. Elemente, Attribute)
2. Erstellung von neuen Datenelementen jeder Art
3. Einfügen und Anhängen neuer Elemente
4. Löschen vorhandener Child-Elemente

Die XMLData-Schnittstelle funktioniert in der Authentic API anders als in der XMLSpy API, insbesondere, wenn Sie neue in die XML-Datei einfügen oder bestehende Elemente umbenennen. Lesen Sie dazu bitte die folgenden Abschnitte "*Erstellen und Einfügen neuer XMLData-Objekte*" und "*Name und Wert von Elementen*".

Struktur von XMLData

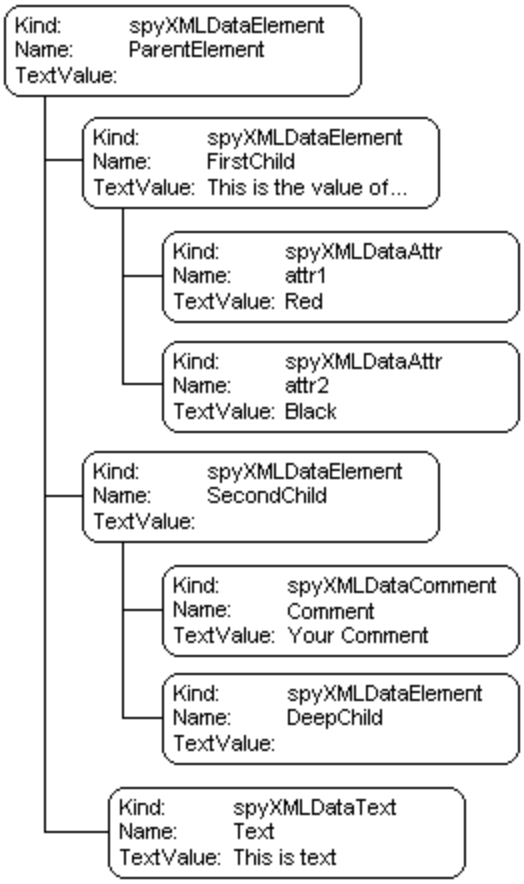
Um diese XMLData-Schnittstelle verwenden können, müssen Sie wissen, wie eine bestehende XML-Datei auf eine XMLData-Struktur gemappt wird. Beachten Sie dabei vor allem, dass XMLData keine separaten Objekte für Attribute hat; die Attribute eines Elements sind gleichzeitig Children des Elements. Ein Element kann andere Arten von Children haben (wie z.B. Child-Elemente und Textinhalt). Die XMLData.Kind 189-Eigenschaft gibt Ihnen die Möglichkeit, zwischen den verschiedenen Child-Typen eines Elements zu unterscheiden. Nachfolgend finden Sie ein Beispiel:

Der folgende XML-Code

```
<ParentElement>
	<FirstChild attr1="Red" attr2="Black">
		This is the value of FirstChild
	</FirstChild>
	<SecondChild>
		<!--Your Comment-->
		</DeepChild>
	</SecondChild>
	This is Text
</ParentElement>
```

wird auf die folgende XMLData-Objektstruktur gemappt:

© 2018-2024 Altova GmbH



Das Parent-Element aller XML-Elemente in einer Datei ist die Eigenschaft Authentic.XMLRoot[97]. Über dieses XMLData-Objekt werden alle anderen XML-Elemente in der Struktur referenziert.

Name und Wert von Elementen

Mit Hilfe der Eigenschaften XMLData.Name[189] und XMLData.TextValue[190] können Sie den Namen und Wert aller XML-Elementtypen abrufen und ändern. Es kann vorkommen, dass mehrere Arten von XMLData-Objekten und leere Elemente keinen damit verknüpften Textwert haben. Es ist nicht ratsam, den Namen eines bestehenden XML-Elements in der Authentic-Ansicht zu ändern, da der Name bestimmt, wie der Inhalt des Elements in der Authentic-Ansicht angezeigt wird. Nähere Informationen zu diesem Thema finden Sie in der Dokumentation zu StyleVision.

Erstellen und Einfügen neuer XMLData-Objekte

Um ein neues XMLData-Objekt (z.B. ein Element, Attribut oder einen Text-Node) einzufügen, gehen Sie folgendermaßen vor:

1. Erstellen Sie mit Hilfe der Authentic.CreateChild[74]-Methode das neue XMLData-Objekt.Definieren Sie den Namen und Wert, bevor Sie es einfügen.
2. Fügen Sie das neue Objekt an der richtigen Stelle mit einer Referenz zu seinem übergeordneten Objekt ein: (i) Wenn das neue Child-Datenelement das letzte Child-Datenelement des Parent-Datenelements werden soll, verwenden Sie die XMLData.AppendChild[181]-Methode. (ii) Falls das neue Child-Datenelement an einer anderen Stelle in der Sequenz der Child-Objekte angelegt werden soll, verwenden Sie entweder XMLData.GetFirstChild[185] oder XMLData.GetNextChild[186], das Child-

Datenelement anzugeben, vor dem das neue Child-Datenelement eingefügt werden soll. Fügen Sie anschließend das neue Child mit Hilfe von XMLData.InsertChild[188] ein. Das neue Child-Datenelement wird unmittelbar vor dem aktuellen Child-Datenelement eingefügt.

Anmerkung: In der Authentic-Ansicht kann ein neues Element zusammen mit seiner Substruktur erstellt werden, da die Substruktur im XML-Schema definiert ist. Ob die Substruktur erstellt wird, kann in den Node-Einstellungen des SPS definiert werden, siehe dazu die Dokumentation zu StyleVision.

Im folgenden Beispiel wird ein drittes Child-Datenelement zwischen die Elemente <FirstChild> und <SecondChild> eingefügt:

```
Dim objParent
Dim objChild
Dim objNewChild

Set objNewChild = objPlugIn.CreateChild(spyXMLDataElement)
objNewChild.Name = "OneAndAHalf"

'objParent is set to <ParentElement>
'GetFirstChild(-1) gets all children of the parent element
'and move to <SecondChild>
Set objChild = objParent.GetFirstChild(-1)
Set objChild = objParent.GetNextChild

objParent.InsertChild objNewChild
Set objNewChild = Nothing
```

Child-Elemente sollten in einer bestimmten Reihenfolge eingefügt werden. Fügen Sie zuerst die Attribute des Elements ein und dann die Child-Elemente des Elements. Der Textinhalt eines Elements muss als Child-Node vom Typ "Text" hinzugefügt werden; verwenden Sie Authentic.CreateChild[74] mit dem Parameterwert 6. Der Textinhalt eines Elements wird als Textwert des Child-Node eingegeben.

Kopieren vorhandener XMLData-Objekte

Die Methoden XMLData.InsertChild und XMLData.AppendChild können nicht zum Einfügen bestehender XMLData-Objekte an einer anderen Stelle in derselben Datei verwendet werden. Diese Methoden funktionieren nur bei neuen XMLData-Objekten. Stattdessen müssen Sie die Objekthierarchie manuell kopieren. Im Folgenden sehen Sie ein Beispiel für eine in JavaScript geschriebene Funktion, mit der XMLData rekursiv kopiert werden:

```
// this function returns a complete copy of the XMLData object
function GetCopy(objXMLData)
{
	var objNew;
	objNew = objPlugIn.CreateChild(objXMLData.Kind);

	objNew.Name = objXMLData.Name;
	objNew.TextValue = objXMLData.TextValue;

	if(objXMLData.HasChildren) {
		var objChild;
		objChild = objXMLData.GetFirstChild(-1);
```

 © 2018-2024 Altova GmbH

```
                while(objChild)     {
                        try {
                                objNew.AppendChild(GetCopy(objChild));
                                objChild = objXMLData.GetNextChild();
                        }
                        catch(e) {
                                objChild = null;
                        }
                }
        }

        return objNew;
}
```

Entfernen von XMLData-Objekten

XMLData bietet zwei Methoden zum Entfernen von Child-Objekten: XMLData.EraseAllChildren[182] und XMLData.EraseCurrentChild[183]. Zum Entfernen von XMLData-Objekten müssen Sie zuerst das Parent-Element der zu löschenden Elemente aufrufen. Verwenden Sie XMLData.GetFirstChild[185] und XMLData.GetNextChild[186], um eine Referenz auf das XMLData-Parent-Objekt zu erhalten. Beispiele zum Löschen von XML-Elementen finden Sie unter der Beschreibung der Methoden XMLData.EraseAllChildren[182] und XMLData.EraseCurrentChild[183].

4.1.13 DOM und XMLData

Die XMLData-Schnittstelle bietet im Vergleich zu DOM weniger Methoden, ist um einiges einfacher und gibt Ihnen vollen Zugriff auf die XML-Struktur des aktuellen Dokuments. Die XMLData-Schnittstelle ist ein minimalistischer Lösungsansatz zum Lesen und Ändern bestehender oder neu erstellter XML-Daten. Sie können aber auch eine DOM-Struktur verwenden, wenn Sie von einer externen Quelle darauf zugreifen können oder die MSXML DOM-Implementierung bevorzugen.

Die nachfolgend beschriebenen Funktionen **ProcessDOMNode()** und **ProcessXMLDataNode()** konvertieren alle Segmente einer XML-Struktur zwischen XMLData und DOM.

So verwenden Sie die **ProcessDOMNode()**-Funktion:

- Übergeben Sie das Root-Element des DOM-Segments, das Sie in **objNode** konvertieren möchten und
- Übergeben Sie das Plug-In-Objekt mit der CreateChild()-Methode in **objCreator.**

So verwenden Sie die **ProcessXMLDataNode()**-Funktion:

- Übergeben Sie das Root-Element des XMLData-Segments in **objXMLData** und
- Übergeben Sie das mit MSXML in **xmlDoc** erzeugte DOMDocument-Objekt.

DOM in XMLData

Im nachstehenden Code wird gezeigt, wie Sie DOM in XMLData konvertieren.

```
////////////////////////////////////////////////////////////
// DOM to XMLData conversion
```

```
function ProcessDOMNode(objNode,objCreator)
{
	var objRoot;
	objRoot = CreateXMLDataFromDOMNode(objNode,objCreator);

	if(objRoot)   {
		if((objNode.nodeValue != null) && (objNode.nodeValue.length > 0))
			objRoot.TextValue = objNode.nodeValue;

		// add attributes
		if(objNode.attributes) {
			var Attribute;
			var oNodeList = objNode.attributes;

			for(var i = 0;i < oNodeList.length; i++) {
				Attribute = oNodeList.item(i);

				var newNode;
				newNode = ProcessDOMNode(Attribute,objCreator);

				objRoot.AppendChild(newNode);
			}
		}

		if(objNode.hasChildNodes)   {
			try {
				// add children
				var Item;
				oNodeList = objNode.childNodes;

				for(var i = 0;i < oNodeList.length; i++) {
					Item = oNodeList.item(i);

					var newNode;
					newNode = ProcessDOMNode(Item,objCreator);

					objRoot.AppendChild(newNode);
				}
			}
			catch(err) {
			}
		}
	}

	return objRoot;
}


function CreateXMLDataFromDOMNode(objNode,objCreator)
{
	var bSetName = true;
```


© 2018-2024 Altova GmbH

```
	var bSetValue = true;

	var nKind = 4;

	switch(objNode.nodeType)   {
		case 2:nKind = 5;break;
		case 3:nKind = 6;bSetName = false;break;
		case 4:nKind = 7;bSetName = false;break;
		case 8:nKind = 8;bSetName = false;break;
		case 7:nKind = 9;break;
	}

	var objNew = null;
	objNew = objCreator.CreateChild(nKind);

	if(bSetName)
		objNew.Name = objNode.nodeName;

	if(bSetValue && (objNode.nodeValue != null))
		objNew.TextValue = objNode.nodeValue;

	return objNew;
}
```

XMLData in DOM

Im nachstehenden Code wird gezeigt, wie Sie XMLData in DOM konvertieren.

```
////////////////////////////////////////////////////////////////
// XMLData to DOM conversion

function ProcessXMLDataNode(objXMLData,xmlDoc)
{
	var objRoot;
	objRoot = CreateDOMNodeFromXMLData(objXMLData,xmlDoc);

	if(objRoot)   {
		if(IsTextNodeEnabled(objRoot) && (objXMLData.TextValue.length > 0))
			objRoot.appendChild(xmlDoc.createTextNode(objXMLData.TextValue));

		if(objXMLData.HasChildren)  {
			try {
				var objChild;
				objChild = objXMLData.GetFirstChild(-1);

				while(true)   {
					if(objChild)  {
						var newNode;
						newNode = ProcessXMLDataNode(objChild,xmlDoc);
```

```
                                        if(newNode.nodeType == 2)  {
                                               // child node is an attribute
                                               objRoot.attributes.setNamedItem(newNode);
                                        }
                                        else
                                               objRoot.appendChild(newNode);
                                 }

                                 objChild = objXMLData.GetNextChild();
                          }
                   }
                   catch(err) {
                   }
            }
      }

      return objRoot;
}

function CreateDOMNodeFromXMLData(objXMLData,xmlDoc)
{
      switch(objXMLData.Kind)    {
            case 4:return xmlDoc.createElement(objXMLData.Name);
            case 5:return xmlDoc.createAttribute(objXMLData.Name);
            case 6:return xmlDoc.createTextNode(objXMLData.TextValue);
            case 7:return xmlDoc.createCDATASection(objXMLData.TextValue);
            case 8:return xmlDoc.createComment(objXMLData.TextValue);
            case 9:return
xmlDoc.createProcessingInstruction(objXMLData.Name,objXMLData.TextValue);
      }

      return xmlDoc.createElement(objXMLData.Name);
}

function IsTextNodeEnabled(objNode)
{
      switch(objNode.nodeType) {
            case 1:
            case 2:
            case 5:
            case 6:
            case 11:return true;
      }

      return false;
}
```


© 2018-2024 Altova GmbH

4.1.14 Authentic Skripterstellung

Durch die Funktion **Authentic-Skripterstellung** können SPS-Designs flexibler und interaktiver gestaltet werden. Diese Designs können in der StyleVision Enterprise und der Professional Edition erstellt werden und in der Authentic-Ansicht der Enterprise und Reporting Edition von Altova-Produkten angezeigt werden.

In der Tabelle unten sehen Sie eine komplette Liste aller Altova-Produkte, die diese Funktion unterstützen. Beachten Sie, dass die interne Skripterstellung in der trusted Version des Authentic Browser Plug-in aus Sicherheitsgründen deaktiviert ist.

Altova-Produkt	Erstellung von Authentic Skripts	Authentic Skripts aktiv
StyleVision Enterprise	Ja	Ja
StyleVision Professional	Ja	Ja
StyleVision Basic *	Nein	Nein
XMLSpy Enterprise	Nein	Ja
XMLSpy Professional	Nein	Ja
AuthenticDesktop Enterprise	Nein	Ja
Authentic Browser Ent Trusted **	Nein	Ja
Authentic Browser Ent Untrusted	Nein	Ja

* *Keine Authentic-Ansicht*
** *Designs mit Skript werden angezeigt. Keine interne Makroausführung oder Event-Behandlung. Externe Events werden ausgelöst.*

Authentic-Skripts verhalten sich in allen Altova-Produkten gleich. Es ist kein produktspezifischer Code und keine produktspezifische Einstellung erforderlich.

Funktionsweise der Authentic Skripterstellung

Die Authentic-Skripterstellung kann beim Design von SPS-Dateien auf zwei Arten eingesetzt werden, um Authentic-Dokumente interaktiv zu machen:

- Durch Zuweisen von Skripts für benutzerdefinierte Aktionen (Makros) zu Design-Elementen, Symbolleisten-Schaltflächen und Kontextmenübefehlen.
- Durch Hinzufügen von Event Handlers zum Design, die auf Events der Authentic-Ansicht reagieren.

Alle Skripts, die erforderlich sind, um Authentic-Dokumente interaktiv zu machen, werden über die StyleVision-Benutzeroberfläche (Enterprise und Professional Edition) erstellt. Die Formulare, Makros und Event Handler werden mit dem Skript-Editor von StyleVision erstellt und diese Skripts werden mit dem SPS gespeichert. Anschließend werden die gespeicherten Skripts in der Design-Ansicht von StyleVision Design-Elementen, Symbolleisten-Schaltflächen und Kontextmenüs zugewiesen. Wenn ein XML-Dokument, das auf dem SPS basiert, in einem Altova-Produkt, das die Authentic-Skripterstellung unterstützt (*siehe Tabelle oben),* geöffnet wird, stehen diese zusätzlichen Funktionen im Dokument zur Verfügung.

Dokumentation zur Authentic Skripterstellung

Die Dokumentation zur Authentic Skripterstellung finden Sie in der Dokumentation zu StyleVision. Online finde Sie diese auf der Altova Website auf der Seite "Produktdokumentation".

© 2018-2024 Altova GmbH

4.2 Objekte

Dieser Abschnitt enthält eine Liste und Beschreibung der verschiedenen Authentic Browser Objekte.

4.2.1 Authentic

Methoden

SetUnmodified 93
ClearUndoRedo 74

Eigenschaften

AuthenticView 73
IsEditClearEnabled 83
IsEditCopyEnabled 83
IsEditCutEnabled 83
IsEditPasteEnabled 84
IsEditRedoEnabled 84
IsEditUndoEnabled 84
IsFindNextEnabled 84
IsRowAppendEnabled 85
IsRowDeleteEnabled 85
IsRowDuplicateEnabled 85
IsRowInsertEnabled 85
IsRowMoveDownEnabled 86
IsRowMoveUpEnabled 86
SchemaLoadObject 92
XMLDataLoadObject 96
DesignDataLoadObject 75
XMLDataSaveUrl 97
XMLRoot 97
CurrentSelection 74
event 78
validationBadData 96
validationMessage 96
ToolbarsEnabled 95
ToolbarTooltipsEnabled 95
AutoHideUnusedCommandGroups 73
TextStateToolbarLine 94
TextStateBmpURL 94
ToolbarRows 95
UICommands 95
XMLTable 97
BaseURL 73
EntryHelpersEnabled 77
EntryHelperSize 78
EntryHelperAlignment 77
EntryHelperWindows 78
EnableModifications 77
Modified 87
SaveButtonAutoEnable 91

Ereignisse

SelectionChanged 93
ControlInitialized 74

Beschreibung

Authentic-Klasse

© 2018-2024 Altova GmbH

4.2.1.1 Authentic.ApplyTextState

Siehe auch

Deklaration: ApplyTextState(*elementName* als String)

Beschreibung
Wendet den vom Parameter elementName definierten Textstatus an oder entfernt ihn. Ein typisches Beispiel für den Parameter elementName wäre "strong" oder "italic".

Ein XML-Dokument enthält Datensegmente, die Subelemente enthalten können. Nehmen wir als Beispiel den folgenden HTML-Code:

```
<b>f r agment </ b>
```

Aufgrund des HTML-Tags <b> wird das Wort "fragment" fett angezeigt. Dies geschieht jedoch nur deshalb, weil der HTML-Parser weiß, dass der Tag <b> "bold", also fett bedeutet. XML bietet viel mehr Flexibilität. Sie können jede beliebige Aktion für jeden beliebigen XML-Tag definieren. Wichtig ist, dass Sie mit XML einen Textstatus auf einen Text anwenden können, jedoch muss dieser Textstatus im Schema enthalten sein.

So bedeutet z.B. im Beispiel mit den Dateien OrgChart.xml OrgChart.sps, OrgChart.xsd der Tag dasselbe wie "bold". Um diesen Textstatus anzuwenden, wird die Methode **ApplyTextState()** aufgerufen. Wie auch bei den die Zeilen- und Bearbeitungsoperationen muss getestet werden, ob der Textstatus angewendet werden kann.
Siehe auch IsTextStateEnabled[86] und IsTextStateApplied[86].

4.2.1.2 Authentic.attachCallBack

Wird nicht mehr verwendet

Verwenden Sie stattdessen die hier[48] beschriebenen Connection Point Events.

***

Siehe auch

Deklaration: attachCallBack(*bstrName* als String,*varCallBack* als Variant)

Beschreibung
In der Authentic-Ansicht stehen Ereignisse zur Verfügung, die mittels benutzerdefinierter Callback-Funktionen abgearbeitet werden können. Alle Ereignishandler akzeptieren keine Parameter und alle Werte, die zurückgegeben werden, werden ignoriert. Um Informationen abzurufen, wenn ein bestimmtes Ereignis abgearbeitet wird, müssen Sie die entsprechenden Eigenschaften des event[78]-Objekts aufrufen.

Liste der derzeit verfügbaren Ereignisse:

ondragover
ondrop
onkeydown
onkeyup
onkeypressed
onmousemove
onbuttonup
onbuttondown
oneditpaste
oneditcut
oneditcopy

Seit Version 3.0.0.0:

ondoceditcommand

Seit Version: 5.3.0.0:

onbuttondoubleclick

JavaScript-Beispiel:

```
// somwhere in your script:
objPlugIn.attachCallBack("ondragover",OnDragOver);
objPlugIn.attachCallBack("ondrop",OnDrop);

// event handlers
function OnDragOver()
{
	if(	!objPlugIn.event.dataTransfer.ownDrag &&
		objPlugIn.event.dataTransfer.type == "TEXT"))
	{
		objPlugIn.event.dataTransfer.dropEffect = 1;
		objPlugIn.event.cancelBubble = true;
	}
}

// OnDrop() replaces the complete text value of the XML
// element with the selection from the drag operation
function OnDrop()
{
	var objTransfer = objPlugIn.event.dataTransfer;

	if(	!objTransfer.ownDrag &&
		(objTransfer.type == "TEXT"))
		objPlugIn.event.srcElement.TextValue = objTransfer.getData();
}
```


© 2018-2024 Altova GmbH

4.2.1.3 AuthenticView

Siehe auch

Deklaration: AuthenticView als AuthenticView[150] (schreibgeschützt)

Beschreibung
Gibt ein Objekt zurück, das Zugriff auf Authentic-Ansicht-spezifische Eigenschaften und Methoden verschafft.

Die AuthenticView[150]-Funktion überlappt sich mit den bestehenden ansichtsspezifischen Funktionen. Zukünftige Versionen der Authentic-Ansicht werden alle ansichtsspezifischen Funktionen enthalten. Das AuthenticView-Objekt ist die empfohlene Schnittstelle für alle zukünftigen Implementierungen.

Beispiele
Nähere Informationen zur Verwendung des AuthenticView-Objekts finden Sie unter "Sortieren dynamischer Tabellen mit bubble-sort[30]".

4.2.1.4 Authentic.AutoHideUnusedCommandGroups

Deklaration: AutoHideUnusedCommandGroups als Boolean

Beschreibung
True, wenn nicht verwendete Symbolleisten-Befehlsgruppen automatisch ausgeblendet werden (z.B. XML-Tabellenbefehle, wenn die aktuelle SPS-Datei XML-Tabellen nicht unterstützt)
Standardwert: true

4.2.1.5 Authentic.BaseURL

Deklaration: BaseURL als String

Beschreibung
Diese Eigenschaft definiert eine alternative URL, die zur Auflösung von relativen Pfadangaben verwendet wird. Wenn keine URL definiert wurde, wird der Ordner der aktuellen XML-Datei verwendet.

4.2.1.6 Authentic.ClearSelection

Deklaration: ClearSelection()

Beschreibung
Mit dieser Methode wird die aktuelle Auswahl aufgehoben. Danach schlägt jeder Versuch, die Auswahl mit Hilfe der CurrentSelection-Eigenschaft zu holen, fehl, bis ein Node ausgewählt wird oder ein erfolgreicher Aufruf von SelectionSet()[93] verarbeitet wurde.

Um einen Node löschen zu können, der die aktuelle Auswahl enthält, muss ein andere Node ausgewählt oder die Auswahl mittels ClearSelection() aufgehoben werden.

4.2.1.7 Authentic.ClearUndoRedo

Deklaration: ClearUndoRedo()

Beschreibung
Mit dieser Methode wird der gesamte Rückgängig/Wiederherstellen-Puffer gelöscht.

4.2.1.8 Authentic.ControlInitialized

Siehe auch

Deklaration: ControlInitialized

Beschreibung
Dieses Ereignis wird bei der Erstellung und Initialisierung des Controls ausgelöst.

Siehe auch Connection Point-Ereignisse[48].

4.2.1.9 Authentic.CreateChild

Siehe auch

Deklaration: CreateChild(*nKind* als SPYXMLDataKind[195]) als XMLData[180]

Rückgabewert
Neuer XML Node

Beschreibung

Die CreateChild-Methode dient zur Erstellung neuer Nodes, die Sie über die XMLData[180]-Schnittstelle in die XML-Struktur des aktuellen Dokuments einfügen können.

Siehe auch XMLData.AppendChild[181] und XMLData.InsertChild[188]

4.2.1.10 Authentic.CurrentSelection

Siehe auch

Deklaration: CurrentSelection als AuthenticSelection[144]

© 2018-2024 Altova GmbH

Beschreibung
Die Eigenschaft verschafft Zugriff auf die aktuelle Auswahl in der Authentic-Ansicht.

Mit dem unten gezeigten Beispielcode wird der komplette Text der aktuellen Auswahl abgerufen:

JavaScript:

```
// somewhere in your script:
GetSelection(objPlugIn.CurrentSelection);

// GetSelection() collects complete text selection
function GetSelection(objSel)
{
	var strText = "";

	var objCurrent = objSel.Start;

	while(!objSel.End.IsSameNode(objCurrent))
	{
		objCurrent = objPlugIn.GetNextVisible(objCurrent);
		strText += objCurrent.TextValue;
	}

	strText += objSel.End.TextValue.substring(0,objSel.EndTextPosition);
	return objSel.Start.TextValue.substr(objSel.StartTextPosition) + strText;
}
```

4.2.1.11 Authentic.DesignDataLoadObject

Siehe auch

Deklaration: DesignDataLoadObject als AuthenticLoadObject[110]

Beschreibung
Das DesignDataLoadObject enthält eine Referenz auf das SPS-Dokument. Mit Hilfe des SPS-Dokuments, das normalerweise in StyleVision erstellt wird, wird die WYSIWYG-Editierumgebung erstellt.

Ein Beispiel dazu finden Sie unter SchemaLoadObject[92].

4.2.1.12 Authentic.EditClear

Siehe auch

Deklaration: EditClear

Beschreibung
Hebt die aktuelle Auswahl auf.

4.2.1.13 Authentic.EditCopy

Siehe auch

Deklaration: EditCopy

Beschreibung
Kopiert die aktuelle Auswahl in die Zwischenablage.

4.2.1.14 Authentic.EditCut

Siehe auch

Deklaration: EditCut

Beschreibung
Schneidet die aktuelle Auswahl aus dem Dokument aus und kopiert sie in die Zwischenablage.

4.2.1.15 Authentic.EditPaste

Siehe auch

Deklaration: EditPaste

Beschreibung
Fügt den Inhalt aus der Zwischenablage in das Dokument ein.

4.2.1.16 Authentic.EditRedo

Siehe auch

Deklaration: EditRedo

Beschreibung
Stellt die zuletzt rückgängig gemachte Aktion wieder her.

© 2018-2024 Altova GmbH

4.2.1.17 Authentic.EditSelectAll

Siehe auch

Deklaration: EditSelectAll

Beschreibung
Mit dieser Methode wird das komplette Dokument ausgewählt.

4.2.1.18 Authentic.EditUndo

Siehe auch

Deklaration: EditUndo

Beschreibung
Die letzte Aktion wird rückgängig gemacht.

4.2.1.19 Authentic.EnableModifications

Deklaration: EnableModifications als Boolean

Beschreibung
"True", wenn die durch Authentic Browser am XML-Inhalt vorgenommenen Änderungen aktiviert sind. Siehe auch Modified[87]-Eigenschaft.

Standardwert: TRUE

4.2.1.20 Authentic.EntryHelperAlignment

Deklaration: EntryHelperAlignment als SPYAuthenticToolbarAllignment[195]

Beschreibung
Diese Eigenschaft dient zum Festlegen der Position der Eingabehilfen. Der Standardwert ist 3. Damit werden die Eingabehilfen an der rechten Seite platziert.

4.2.1.21 Authentic.EntryHelpersEnabled

Deklaration: EntryHelpersEnabled als Boolean

Beschreibung
"True", wenn die Eingabehilfen von Authentic Browser aktiviert sind.
Diese Eigenschaft kann verwendet werden, um die Eingabehilfen zu aktivieren bzw. zu deaktivieren.

Standardwert: FALSE

4.2.1.22 Authentic.EntryHelperSize

Deklaration: EntryHelperSize als Integer

Beschreibung
Mit Hilfe dieser Eigenschaft können Sie die anfangs verwendete Größe der Eingabehilfen in Pixeln angeben. Setzen Sie die Eigenschaft auf -1, wenn dieser Wert ignoriert werden soll.

Standardwert: -1

4.2.1.23 Authentic.EntryHelperWindows

Deklaration: EntryHelperWindows als SPYAuthenticEntryHelperWindows[194]

Beschreibung
Mit Hilfe dieser Eigenschaft können Sie definieren, welche der Eingabehilfen angezeigt werden sollen. Die Werte können kombiniert werden, um mehrere Eingabehilfenfenster anzuzeigen. Der Standardwert ist 7. Bei Auswahl dieses Werts werden alle drei Eingabehilfen eingeblendet.

4.2.1.24 Authentic.event

Siehe auch

Deklaration: event als AuthenticEvent[102]

Beschreibung
Die Ereigniseigenschaft enthält ein Ereignis-Datenobjekt mit Informationen über das aktuelle Ereignis.

4.2.1.25 Authentic.FindDialog

Siehe auch

Deklaration: FindDialog

Beschreibung
Zeigt den FindDialog an.

© 2018-2024 Altova GmbH

Siehe auch Suchen und Ersetzen[52].

4.2.1.26 Authentic.FindNext

Siehe auch

Deklaration: FindNext

Beschreibung
Diese Methode führt eine "Weitersuchen"-Operation aus.

Siehe auch Suchen und Ersetzen[52].

4.2.1.27 Authentic.GetAllAttributes

Siehe auch

Deklaration: GetAllAttributes(*pForElement* als XMLData[180], *pElements* als Variant)

Beschreibung
GetAllAttributes() gibt die für das angegebene Element zulässigen Attribute als String-Array zurück.

JavaScript-Beispiel:

```
function GetAttributes()
{
	var arrElements = new Array(1);

	var objStart = objPlugIn.CurrentSelection.Start;

	var strText;
	strText = "Valid attributes at current selection:\n\n";

	for(var i = 1;i <= 4;i++)
	{
		objPlugIn.GetAllAttributes(objStart, arrElements);
		strText = strText + ListArray(arrElements) + "------------------\n";
	}

	return strText;
}

function ListArray(arrIn)
{
	var strText = "";
```

```
		if(typeof(arrIn) == "object")
		{
			for(var i = 0;i <= (arrIn.length - 1);i++)
				strText = strText + arrIn[i] + "\n";
		}

		return strText;
	}
```

VBScript-Beispiel:

```
	Sub DisplayAllowedAttributes
		dim arrElements()

		dim objStart
		dim objEnd
		set objStart = objPlugIn.CurrentSelection.Start
		set objEnd = objPlugIn.CurrentSelection.End

		dim strText
		strText = "Valid attributes at current selection:" & chr(13) & chr(13)

		dim i

		For i = 1 To 4
			objView.GetAllAttributes objStart, arrElements
			strText = strText & ListArray(arrElements) & "----------------" &
chr(13)
		Next

		msgbox strText
	End Sub

	Function ListArray(arrIn)
		dim strText

		If IsArray(arrIn) Then
			dim i

			For i = 0 To UBound(arrIn)
				strText = strText & arrIn(i) & chr(13)
			Next
		End If

		ListArray = strText
 End Function
```


© 2018-2024 Altova GmbH

4.2.1.28 Authentic.GetAllowedElements

Siehe auch

Deklaration: GetAllowedElements(*nAction* als SPYAuthenticElementActions[193],*pStartElement* als XMLData[180],*pEndElement* als XMLData[180],*pElements* als Variant)

Beschreibung
GetAllowedElements() gibt die für die verschiedenen von nAction definierten Aktionen zulässigen Elemente zurück.

JavaScript-Beispiel:

```
function GetAllowed()
{
	var arrElements = new Array(1);

	var objStart = objPlugIn.CurrentSelection.Start;
	var objEnd = objPlugIn.CurrentSelection.End;

	var strText;
	strText = "valid elements at current selection:\n\n";

	for(var i = 0;i <= 4;i++)  {
		objPlugIn.GetAllowedElements(i,objStart,objEnd,arrElements);
		strText = strText + ListArray(arrElements) + "------------------\n";
	}

	return strText;
}

function ListArray(arrIn)
{
	var strText = "";

	if(typeof(arrIn) == "object")     {
		for(var i = 0;i <= (arrIn.length - 1);i++)
			strText = strText + arrIn[i] + "\n";
	}

	return strText;
}
```

VBScript-Beispiel:

```
Sub DisplayAllowed
	dim arrElements()

	dim objStart
	dim objEnd
```

```
		set objStart = objPlugIn.CurrentSelection.Start
		set objEnd = objPlugIn.CurrentSelection.End

		dim strText
		strText = "Valid elements at current selection:" & chr(13) & chr(13)

		dim i

		For i = 1 To 4
			objView.GetAllowedElements i,objStart,objEnd,arrElements
			strText = strText & ListArray(arrElements) & "---------------" &
chr(13)
		Next

		msgbox strText
	End Sub

	Function ListArray(arrIn)
		dim strText

		If IsArray(arrIn) Then
			dim i

			For i = 0 To UBound(arrIn)
				strText = strText & arrIn(i) & chr(13)
			Next
		End If

		ListArray = strText
	End Function
```

4.2.1.29 Authentic.GetFileVersion

Siehe auch

Deklaration: GetFileVersion(*strVersion* als String)

Beschreibung
Die Methode gibt die Version der Komponente als String im Format 5.0.0.0 zurück.

4.2.1.30 Authentic.GetNextVisible

Siehe auch

Deklaration: GetNextVisible(*pElement* als XMLData[180]) als XMLData[180]

Beschreibung
Die Methode liefert das nächste sichtbare XML-Element im Dokument.

© 2018-2024 Altova GmbH

4.2.1.31 Authentic.GetPreviousVisible

Siehe auch

Deklaration: GetPreviousVisible(*pElement* als XMLData[180]) als XMLData[180]

Beschreibung
Die Methode liefert das vorhergehende XML-Element im Dokument.

4.2.1.32 Authentic.IsEditClearEnabled

Siehe auch

Deklaration: IsEditClearEnabled als Boolean

Beschreibung
True, wenn EditClear[75] möglich ist.

Siehe auch Bearbeitungsoperationen[51].

4.2.1.33 Authentic.IsEditCopyEnabled

Siehe auch

Deklaration: IsEditCopyEnabled als Boolean

Beschreibung
True, wenn Kopieren in die Zwischenablage möglich ist.

Siehe auch EditCopy[76] und Bearbeitungsoperationen[51].

4.2.1.34 Authentic.IsEditCutEnabled

Siehe auch

Deklaration: IsEditCutEnabled als Boolean

Beschreibung
True, wenn EditCut[76] derzeit möglich ist.

Siehe auch Bearbeitungsoperationen[51].

4.2.1.35 Authentic.IsEditPasteEnabled

Siehe auch

Deklaration: IsEditPasteEnabled als Boolean

Beschreibung
True, wenn EditPaste[76] möglich ist.

Siehe auch Bearbeitungsoperationen[51].

4.2.1.36 Authentic.IsEditRedoEnabled

Siehe auch

Deklaration: IsEditRedoEnabled als Boolean

Beschreibung
True, wenn EditRedo[76] derzeit möglich ist.

Siehe auch Bearbeitungsoperationen[51].

4.2.1.37 Authentic.IsEditUndoEnabled

Siehe auch

Deklaration: IsEditUndoEnabled als Boolean

Beschreibung
True, wenn EditUndo[77] möglich ist.

Siehe auch Bearbeitungsoperationen[51].

4.2.1.38 Authentic.IsFindNextEnabled

Siehe auch

Deklaration: IsFindNextEnabled als Boolean

Beschreibung
True, wenn FindNext derzeit möglich ist. False, wenn der Suchbegriff nicht mehr vorkommt.

© 2018-2024 Altova GmbH

Siehe auch Suchen und Ersetzen[52] und FindDialog[78].

4.2.1.39 Authentic.IsRowAppendEnabled

Siehe auch

Deklaration: IsRowAppendEnabled als Boolean

Beschreibung
True, wenn RowAppend[89] möglich ist.

Siehe auch Zeilenoperationen[52].

4.2.1.40 Authentic.IsRowDeleteEnabled

Siehe auch

Deklaration: IsRowDeleteEnabled als Boolean

Beschreibung
True, wenn RowDelete[89] möglich ist.

Siehe auch Zeilenoperationen[52].

4.2.1.41 Authentic.IsRowDuplicateEnabled

Siehe auch

Deklaration: IsRowDuplicateEnabled als Boolean

Beschreibung
True, wenn RowDuplicate[89] möglich ist.

Siehe auch Zeilenoperationen[52].

4.2.1.42 Authentic.IsRowInsertEnabled

Siehe auch

Deklaration: IsRowInsertEnabled als Boolean

Beschreibung
True, wenn RowInsert[90] möglich ist.

Siehe auch Zeilenoperationen[52].

4.2.1.43 Authentic.IsRowMoveDownEnabled

Siehe auch

Deklaration: IsRowMoveDownEnabled als Boolean

Beschreibung
True, wenn RowMoveDown[90] möglich ist.

Siehe auch Zeilenoperationen[52].

4.2.1.44 Authentic.IsRowMoveUpEnabled

Siehe auch

Deklaration: IsRowMoveUpEnabled als Boolean

Beschreibung
True, wenn RowMoveUp[90] möglich ist.

Siehe auch Zeilenoperationen[52].

4.2.1.45 Authentic.IsTextStateApplied

Siehe auch

Deklaration: IsTextStateApplied(*elementName* als String) als Boolean

Beschreibung
Überprüft, ob der Textstatus bei der aktuellen Selektion/Position bereits angewendet wurde. Typische Beispiele für den Parameter elementName sind "strong" und "italic".

4.2.1.46 Authentic.IsTextStateEnabled

Siehe auch

Deklaration: IsTextStateEnabled(*elementName* als String) als Boolean

© 2018-2024 Altova GmbH

Beschreibung
Überprüft, ob ein Textstatus für die aktuelle Selektion/Position angewendet werden kann. Typische Beispiele für den Parameter elementName sind "strong" und "italic".

4.2.1.47 Authentic.LoadXML

Siehe auch

Deklaration: LoadXML(*xmlString* als String)

Beschreibung
Lädt den angegebenen String als das aktuelle XML-Dokument in die Authentic-Ansicht. Der neue Inhalt wird sofort angezeigt.

4.2.1.48 Authentic.MarkUpView

Siehe auch

Deklaration: MarkUpView(*kind* als SPYAuthenticMarkupVisibility[194])

Beschreibung
Standardmäßig wird das Dokument wie ein HTML-Dokument angezeigt. Manchmal ist es jedoch hilfreich, wenn die Editier-Tags angezeigt werden. Mit Hilfe dieser Methode können Sie verschiedene Arten von Markup-Tags anzeigen.

4.2.1.49 Authentic.Modified

Deklaration: Modified als Boolean

Beschreibung
True, wenn XML-Inhalt geändert wurde.

Diese Eigenschaft ist schreibgeschützt.

4.2.1.50 Authentic.Print

Siehe auch

Deklaration: Print

Beschreibung
Druckt das aktuell editierte Dokument.

4.2.1.51 Authentic.PrintPreview

Siehe auch

Deklaration: PrintPreview

Beschreibung
Zeigt eine Druckvorschau des gerade editierten Dokuments an.

4.2.1.52 Authentic.RedrawEntryHelpers

Deklaration: RedrawEntryHelpers()

Beschreibung
RedrawEntryHelpers holt die Werte aus den Eigenschaften EntryHelpersEnabled[77], EntryHelperAlignment[77], EntryHelperSize[78] und EntryHelperWindows[78] und zeichnet die Eingabehilfenfenster neu.

4.2.1.53 Authentic.ReloadToolbars

Deklaration: ReloadToolbars()

Beschreibung
ReloadToolbars liest die ToolbarRows[95] Sammlung und zeichnet die Symbolleisten und Ansichten neu.

4.2.1.54 Authentic.ReplaceDialog

Siehe auch

Deklaration: ReplaceDialog

Beschreibung
Zeigt den ReplaceDialog an.

Siehe auch Suchen und Ersetzen[52].

© 2018-2024 Altova GmbH

4.2.1.55 Authentic.Reset

Veraltet

Verwenden Sie statt dessen `Authentic.StartEditing` 94.

Siehe auch

Deklaration: Reset

Beschreibung
Wurde durch andere Funktionen ersetzt und es wird angeraten diese nicht mehr zu verwenden.

4.2.1.56 Authentic.RowAppend

Siehe auch

Deklaration: RowAppend

Beschreibung
Hängt eine Zeile an der aktuellen Position an.

Siehe auch Zeilenoperationen 52.

4.2.1.57 Authentic.RowDelete

Siehe auch

Deklaration: RowDelete

Beschreibung
Löscht die aktuell ausgewählte(n) Zeile(n).

Siehe auch Zeilenoperationen 52.

4.2.1.58 Authentic.RowDuplicate

Siehe auch

Deklaration: RowDuplicate

Beschreibung
Diese Methode dupliziert die aktuell ausgewählten Zeilen.

Siehe auch Zeilenoperationen[52].

4.2.1.59 Authentic.RowInsert

Siehe auch

Deklaration: RowInsert

Beschreibung
Fügt eine neue Zeile unmittelbar oberhalb der aktuellen Auswahl ein.

Siehe auch Zeilenoperationen[52].

4.2.1.60 Authentic.RowMoveDown

Siehe auch

Deklaration: RowMoveDown

Beschreibung
Verschiebt die aktuelle Zeile um eine Position nach unten.

Siehe auch Zeilenoperationen[52].

4.2.1.61 Authentic.RowMoveUp

Siehe auch

Deklaration: RowMoveUp

Beschreibung
Verschiebt die aktuelle Zeile um eine Position nach oben.

Siehe auch Zeilenoperationen[52].

4.2.1.62 Authentic.Save

Siehe auch

© 2018-2024 Altova GmbH

Deklaration: Save

Beschreibung
Speichert das Dokument unter der in der Eigenschaft XMLDataSaveUrl[97] definierten URL. Bei Untrusted Versionen können Sie auch einen vollständigen lokalen Pfad verwenden.

Das Plug-In sendet einen HTTP PUT Request an den Server, um die aktuell angezeigte XML-Datei zu speichern.

4.2.1.63 Authentic.SaveButtonAutoEnable

Deklaration: SaveButtonAutoEnable als Boolean

Beschreibung
Ist diese Eigenschaft auf TRUE gesetzt, wird der Ein/Aus-Status der Schaltfläche "Speichern" in der Symbolleiste des Control entsprechend dem Modified[87] Flag des Dokuments gesetzt.

Der Standardwert ist FALSE.

4.2.1.64 Authentic.SavePOST

Siehe auch

Deklaration: SavePOST

Beschreibung
Speichert das Dokument unter der von der Eigenschaft XMLDataSaveUrl[97] definierten URL. Bei Untrusted Versionen können Sie auch einen vollständigen lokalen Pfad verwenden. Das Plug-In sendet einen HTTP POST Request an den Server, um die aktuell angezeigte XML-Datei zu speichern.

Überprüfen, ob eine Datei gespeichert wurde oder nicht
Wenn das Authentic Plug-In eine HTTP-Antwort >= 300 erhält, geht es davon aus, dass die Datei **nicht** gespeichert wurde und zeigt daraufhin (standardmäßig) eine Meldung an, die den HTTP-Fehler enthält. Anschließend wird eine zweite (deaktivierbare) Meldung angezeigt, in der der Benutzer darauf hingewiesen wird, dass die Datei nicht gespeichert wurde. Es obliegt dem Applikationsentwickler, sicherzustellen, dass die korrekte HTTP-Antwort zurückgegeben wird, je nachdem ob das Dokument gespeichert werden konnte oder nicht. Der entsprechende PHP-Code könnte etwa folgendermaßen aussehen:

```
<?php
// suppress error messages to prevent any output
// being generated before headers can be sent
error_reporting (0);
$error = false;
$handle = fopen ( "result.xml", "w+" );
if (! $handle)
   $error = true;
else
{
```

```
    if (! fwrite($handle, $HTTP_RAW_POST_DATA))
           $error = true;
    else
           fclose($handle);
}
if ($error)
    header( "HTTP/1.1 500 Server Error" );
?>
```

4.2.1.65 Authentic.SaveXML

Siehe auch

Deklaration: SaveXML als String

Rückgabewert
XML-Struktur als String

Beschreibung
Speichert die aktuellen XML-Daten in einem String, der an den Caller zurückgegeben wird.

4.2.1.66 Authentic.SchemaLoadObject

Siehe auch

Deklaration: SchemaLoadObject als AuthenticLoadObject[110]

Beschreibung
Das SchemaLoadObject enthält eine Referenz zum XML-Schema-Dokument für die aktuelle XML-Datei. Das Schema-Dokument wird normalerweise mit XMLSpy generiert.

Beispiel

```
objPlugIn.SchemaLoadObject.URL = "http://www.YOURSERVER.com/OrgChart.xsd"
objPlugIn.XMLDataLoadObject.URL = "http://www.YOURSERVER.com/OrgChart.xml"
objPlugIn.DesignDataLoadObject.URL = "http://www.YOURSERVER.com/OrgChart.sps"
objPlugIn.StartEditing
```

Der obige Code definiert alle URL-Eigenschaften der zu ladenden Objekte und ruft StartEditing[94] zum Laden und Anzeigen der Dateien auf. Bei Untrusted Versionen können Sie auch einen vollständigen lokalen Pfad verwenden. Der aktuelle Inhalt und Status des Plug-In werden zurückgesetzt.

© 2018-2024 Altova GmbH

4.2.1.67 Authentic.SelectionChanged

Siehe auch

Deklaration: SelectionChanged als VT_0019

Beschreibung
Dieses Ereignis wird ausgelöst, wenn der Benutzer die aktuelle Auswahl ändert.

Siehe auch Connection Point-Ereignisse[48].

4.2.1.68 Authentic.SelectionMoveTabOrder

Siehe auch

Deklaration: SelectionMoveTabOrder(*bForward* als Boolean,*bTag* als Boolean)

Beschreibung
SelectionMoveTabOrder() verschiebt die aktuelle Auswahl nach vorne oder nach hinten.

Wenn bTag "false" ist und die letzte Zelle einer Tabelle ausgewählt ist, wird eine neue Zeile hinzugefügt.

4.2.1.69 Authentic.SelectionSet

Siehe auch

Deklaration: SelectionSet(*pStartElement* als XMLData[180],*nStartPos* als long,*pEndElement* als XMLData[180],*nEndPos* als long) als Boolean

Beschreibung
Verwenden Sie SelectionSet(), um eine neue Auswahl in der Authentic-Ansicht zu definieren. Sie können pEndElement auf Null (nichts) setzen, wenn die Auswahl nicht mehr als ein (pStartElement) XML-Element umfassen soll.

4.2.1.70 Authentic.SetUnmodified

Deklaration: SetUnmodified()

Beschreibung
Nach Aufruf dieser Methode wird der aktuelle Zustand der Rückgängig/Wiederholen-Puffers als ungeänderter Zustand des zugrunde liegenden XML-Dokuments verwendet und der Modified[87] Flag wird auf FALSE gesetzt.

4.2.1.71 Authentic.StartEditing

Siehe auch

Deklaration: StartEditing als Boolean

Rückgabewert
True, wenn alle Dateien erfolgreich geladen wurden und angezeigt werden.

Beschreibung
Startet die Editierung des aktuellen Dokuments. Es ist wichtig, dass zuerst die Eigenschaften der Lade-Objekte SchemaLoadObject 92, DesignDataLoadObject 75 und XMLDataLoadObject 96 festgelegt werden.

4.2.1.72 Authentic.StartSpellChecking

Deklaration: StartSpellChecking()

Beschreibung
Dieser Befehl öffnet das Dialogfeld für die Rechtschreibprüfung, wenn ein Paket mit dem Rechtschreibprüf-Prozessor und den erforderlichen Wörterbüchern verfügbar und aktiviert ist.

4.2.1.73 Authentic.TextStateBmpURL

Deklaration: TextStateBmpURL als String

Beschreibung
Die URL, von der die Bitmaps für die Textstatus-Symbole abgerufen werden sollen.
Wenn keine URL definiert ist, werden keine Textstatus-Symbole angezeigt.

Beispiele
objPlugIn.TextStateBmpURL = "<http://plugin.xmlspy.com/textstates/>"

```
<PARAM NAME="TextStateBmpURL" VALUE="http://plugin.xmlspy.com/textstates/">
```

4.2.1.74 Authentic.TextStateToolbarLine

Deklaration: TextStateToolbarLine als long

Beschreibung
Die Symbolleiste (Zeilennummer) in der die Textstatus-Symbole angezeigt werden sollen. Textstatus-Symbole können an bestehende Symbolleisten angehängt oder in neue Symbolleisten platziert werden.
Standardwert: 1

© 2018-2024 Altova GmbH

4.2.1.75 Authentic.ToolbarRows

Deklaration: ToolbarRows als AuthenticToolbarRows[149]

Beschreibung
Enthält eine Sammlung aller Symbolleisten, die angezeigt werden sollen. Informationen zum Löschen, Hinzufügen oder Ändern von Symbolleisten finden Sie unter AuthenticToolbarRows[149].

4.2.1.76 Authentic.ToolbarsEnabled

Deklaration: ToolbarsEnabled als Boolean

Beschreibung
True, wenn Authentic Browser-Symbolleisten aktiviert sind.
Diese Eigenschaft kann zum Aktivieren oder Deaktivieren aller Symbolleisten verwendet werden.
Standardwert: true

4.2.1.77 Authentic.ToolbarTooltipsEnabled

Deklaration: ToolbarTooltipsEnabled als Boolean

Beschreibung
True, wenn die Tooltipps für die Authentic Browser-Symbolleisten aktiviert sind.
Standardwert: true

4.2.1.78 Authentic.UICommands

Deklaration: UICommands als AuthenticCommands[99]

Beschreibung
Enthält eine Sammlung aller verfügbaren Symbolleistenbefehle (mit Beschreibung).
Für diese Befehle können in den verschiedenen Symbolleisten Schaltflächen platziert werden.
Schreibgeschützt.

Beispiel
Holt alle verfügbaren Befehle, Befehlsgruppen, Beschreibungen und zeigt Sie in einem Nachrichtenfeld an.

```
dim str
for each UICommand in objPlugin.UICommands
str = str & UICommand.CommandID & " | " & UICommand.Group & " | " &
UICommand.ShortDescription & chr(13)
next
msgbox str
```

4.2.1.79 Authentic.ValidateDocument

Siehe auch

Deklaration: ValidateDocument(*showResults* als Boolean) als Boolean

Rückgabewert
Ergebnis der Validierung

Beschreibung
Überprüft die aktuellen XML-Daten auf Ihre Gültigkeit hinsichtlich der XML-Schemadaten. Wenn der Parameter showResults FALSE ist, werden die Validierungsfehler nicht angezeigt, ansonsten werden sie angezeigt.

4.2.1.80 Authentic.validationBadData

Siehe auch

Deklaration: validationBadData als XMLData[180]

Beschreibung
Diese Eigenschaft kann zusätzliche Informationen über den letzten Validierungsfehler liefern. Sie wird nach Aufrufen von `ValidateDocument()` gesetzt und ist entweder Null oder sie enthält eine Referenz auf das fehlerverursachende XML-Element.

4.2.1.81 Authentic.validationMessage

Siehe auch

Deklaration: validationMessage als String

Beschreibung
Wenn die Validierung fehlgeschlagen ist (nach Aufruf von ValidateDocument), speichert diese Eigenschaft einen String mit der Fehlermeldung.

4.2.1.82 Authentic.XMLDataLoadObject

Siehe auch

Deklaration: XMLDataLoadObject als AuthenticLoadObject[110]

© 2018-2024 Altova GmbH

Beschreibung
Das XMLDataLoadObject enthält eine Referenz auf das XML-Dokument, das gerade bearbeitet wird. Das XML-Dokument wird normalerweise mit Hilfe von XMLSpy definiert und anhand einer Datenbank oder eines anderes Geschäftsprozesses generiert.

Beispiel dazu finden Sie unter SchemaLoadObject[92].

4.2.1.83 Authentic.XMLDataSaveUrl

Siehe auch

Deklaration: XMLDataSaveUrl als String

Beschreibung
Wenn die XML-Daten geändert wurden, können diese über eine URL wieder auf einem Server gespeichert werden. Bei Speichern der XML-Daten mittels eines `HTTP PUT` mit der Methode `Authentic.Save` wird mit dieser Eigenschaft der Pfad definiert, unter dem die XML-Daten gespeichert werden. Wenn Sie die Daten mit der `Authentic.SavePOST` Methode über ein `HTTP POST`absenden, wird mit dieser Eigenschaft der Pfad des serverseitigen Scripts bzw. der serverseitigen Applikation definiert, das/die die `POST`-Daten verarbeiten soll. Bei Untrusted Versionen können Sie auch den vollständigen lokalen Pfad verwenden.

Siehe auch die Methoden Authentic.Save[90] und Authentic.SavePOST[91].

4.2.1.84 Authentic.XMLRoot

Siehe auch

Deklaration: XMLRoot als XMLData[180]

Beschreibung
XMLRoot ist das Parent-Element der aktuell angezeigten XML-Struktur. Über die XMLData[180] Schnittstelle haben Sie vollen Zugriff auf den kompletten Dateiinhalt.

Weitere Informationen dazu siehe auch Using XMLData[60].

4.2.1.85 Authentic.XMLTable

Deklaration: XMLTable als AuthenticXMLTableCommands[172]

Beschreibung
Holt einen Satz aller XML-Tabellenbefehle.
Schreibgeschützt.

4.2.2 AuthenticCommand

Methoden

Eigenschaften

4.2.2.1 AuthenticCommand.CommandID

Deklaration: CommandID als SPYAuthenticCommand 191

Beschreibung
Enthält die CommandId des Befehls
Mögliche Werte: siehe AuthenticToolbarButton
Schreibgeschützt

Beispiel
Siehe Beispiel unter Authentic.UICommands 95

4.2.2.2 AuthenticCommand.Group

Deklaration: Group als SPYAuthenticCommandGroup 193

Beschreibung
Die CommandGroup, zu der der Befehl gehört.
Schreibgeschützt

Beispiel
Siehe Beispiel unter Authentic.UICommands 95

4.2.2.3 AuthenticCommand.ShortDescription

Deklaration: ShortDescription als String

Beschreibung
Kurzbeschreibung des Befehls (z.B. der Tooltipp-Text)
Schreibgeschützt

Beispiel
Siehe Beispiel unter Authentic.UICommands 95

© 2018-2024 Altova GmbH

4.2.2.4 AuthenticCommand.Name

Deklaration: Name als String

Beschreibung
Für zukünftige Verwendungszwecke

4.2.3 AuthenticCommands

Methoden
Item 99

Eigenschaften
Count 99

4.2.3.1 AuthenticCommands.Count

Deklaration: Count als long

Beschreibung
Anzahl der verfügbaren Benutzerschnittstellen-Befehle.
Schreibgeschützt

4.2.3.2 AuthenticCommands.Item

Deklaration: Item (nPosition als long) als AuthenticCommand 98

Beschreibung
Liefert den Befehl in Position nPosition. nPosition beginnt mit 1.

4.2.4 AuthenticContextMenu

Über die ContextMenu-Schnittstelle kann der Benutzer die in Authentic angezeigten Kontextmenüs anpassen. Die Schnittstelle hat die in diesem Abschnitt aufgelisteten Methoden.

4.2.4.1 CountItems

Methode: `CountItems()`

Rückgabewert
Gibt die Anzahl der Menüoptionen zurück.

Fehler
2501 Ungültiges Objekt.

4.2.4.2 DeleteItem

Methode: `DeleteItem(position als integer)`

Rückgabewert
Löscht einen bestehenden Menüeintrag.

Fehler
2501 Ungültiges Objekt
2502 Ungültiger Index

4.2.4.3 GetItemText

Methode: `GetItemText(position als integer)` Menübefehlname als String

Rückgabewert
Ruft den Namen des Menübefehls ab.

Fehler
2501 Ungültiges Objekt
2502 Ungültiger Index

4.2.4.4 InsertItem

Methode: `InsertItem(position als integer, menu item name als string, macro name als string)`

Rückgabewert
Fügt einen benutzerdefinierten Menübefehl ein, mit dem ein Makro gestartet wird, daher muss ein gültiger Makroname angegeben werden.

Errors
2501 Ungültiges Objekt
2502 Ungültiger Index

© 2018-2024 Altova GmbH

2503 Kein Makro dieses Namens vorhanden
2504 Interner Fehler

4.2.4.5 SetItemText

Methode: `SetItemText(position als integer, menu item name als string)`

Rückgabewert
Definiert den Namen des Menübefehls.

Fehler
2501 Ungültiges Objekt
2502 Ungültiger Index

4.2.5 AuthenticDataTransfer

Methoden
getData 101

Eigenschaften
dropEffect 101
ownDrag 102
type 102

Beschreibung
AuthenticDataTransfer-Schnittstelle.

4.2.5.1 AuthenticDataTransfer.dropEffect

Siehe auch

Deklaration: dropEffect als long

Beschreibung
Die Eigenschaft speichert das Drop-Verhalten aus dem Standard-Ereignishandler. Sie können das Drop-Verhalten definieren, wenn Sie diesen Wert ändern und AuthenticEvent.cancelBubble 104 auf TRUE setzen.

4.2.5.2 AuthenticDataTransfer.getData

Siehe auch

Deklaration: getData als Variant

Beschreibung
getData holt die aktuellen Daten, die mit diesem dataTransfer-Objekt verknüpft sind. Siehe auch AuthenticDataTransfer.type[102].

4.2.5.3 AuthenticDataTransfer.ownDrag

Siehe auch

Deklaration: ownDrag als Boolean

Beschreibung
Die Eigenschaft ist TRUE, wenn die Quelle des Drag-Vorgangs von innerhalb der Authentic-Ansicht kommt.

4.2.5.4 AuthenticDataTransfer.type

Siehe auch

Deklaration: type als String

Beschreibung
Enthält den Datentyp, den Sie mit der AuthenticDataTransfer.getData[101] Methode holen.

Derzeit werden die folgenden Datentypen unterstützt:

OWN	Daten aus dem Plug-In selbst
TEXT	Nur Text
UNICODETEXT	Nur Text als UNICODE
IUNKNOWN	Unbekannte Referenz auf IDataObject-Instanz

4.2.6 AuthenticEvent

Eigenschaften
altKey[103]
altLeft[103]
ctrlKey[104]
ctrlLeft[105]
shiftKey[106]
shiftLeft[106]
keyCode[105]
repeat[106]
button[103]
clientX[104]
clientY[104]
dataTransfer[105]

© 2018-2024 Altova GmbH

srcElement 107
fromElement 105
propertyName 106
cancelBubble 104
returnValue 106
type 107

Beschreibung
AuthenticEvent-Schnittstelle.

4.2.6.1 AuthenticEvent.altKey

Siehe auch

Deklaration: altKey als Boolean

Beschreibung
True, wenn Sie die rechte ALT-Taste drücken.

4.2.6.2 AuthenticEvent.altLeft

Siehe auch

Deklaration: altLeft als Boolean

Beschreibung
True, wenn die linke ALT-Taste gedrückt wird.

4.2.6.3 AuthenticEvent.button

Siehe auch

Deklaration: button als long

Beschreibung
Gibt an, welche Maustaste gedrückt wird:

0	Keine Taste
1	Linke Taste
2	Rechte Taste
3	Sowohl die linke als auch die rechte Taste
4	Mittlere Taste
5	Linke und mittlere Taste
6	Rechte und mittlere Taste
7	Alle drei Tasten

Die Ereignisse onbuttondown und onbuttonup setzen den Tastenwert auf verschiedene Arten. Das onbuttonup-Ereignis setzt immer nur den Wert für die Taste, die gerade losgelassen wurde und das Ereignis ausgelöst hat, unabhängig davon, welche anderen Tasten in diesem Moment ebenfalls gedrückt sind.

4.2.6.4 AuthenticEvent.cancelBubble

Siehe auch

Deklaration: cancelBubble als Boolean

Beschreibung
Setzt cancelBubble auf TRUE, wenn der Standard-Ereignishandler nicht aufgerufen werden soll.

4.2.6.5 AuthenticEvent.clientX

Siehe auch

Deklaration: clientX als long

Beschreibung
X-Wert der aktuellen Mausposition in den Client-Koordinaten.

4.2.6.6 AuthenticEvent.clientY

Siehe auch

Deklaration: clientY als long

Beschreibung
Y-Wert der aktuellen Mausposition in den Client-Koordinaten.

4.2.6.7 AuthenticEvent.ctrlKey

Siehe auch

Deklaration: ctrlKey als Boolean

Beschreibung
True, wenn die rechte Strg-Taste gedrückt wird.

© 2018-2024 Altova GmbH

4.2.6.8 AuthenticEvent.ctrlLeft

Siehe auch

Deklaration: ctrlLeft als Boolean

Beschreibung
True, wenn die linke Strg-Taste gedrückt wird.

4.2.6.9 AuthenticEvent.dataTransfer

Siehe auch

Deklaration: dataTransfer als Variant

Beschreibung
Eigenschaft dataTransfer

4.2.6.10 AuthenticEvent.fromElement

Siehe auch

Deklaration: fromElement als Variant

Beschreibung
Derzeit wird diese Eigenschaft von keinem Ereignis gesetzt.

4.2.6.11 AuthenticEvent.keyCode

Siehe auch

Deklaration: keyCode als long

Beschreibung
Tastaturcode der aktuell gedrückten Taste.

Diese Eigenschaft ist mit Lese- und Schreibzugriff.

4.2.6.12 AuthenticEvent.propertyName

Siehe auch

Deklaration: propertyName als String

Beschreibung
Derzeit wird diese Eigenschaft von keinem Ereignis gesetzt.

4.2.6.13 AuthenticEvent.repeat

Siehe auch

Deklaration: repeat als Boolean

Beschreibung
True, wenn sich das onkeydown-Ereignis wiederholt.

4.2.6.14 AuthenticEvent.returnValue

Siehe auch

Deklaration: returnValue als Variant

Beschreibung
Verwenden Sie returnValue, um einen Rückgabewert für Ihren Ereignishandler zu setzen.

4.2.6.15 AuthenticEvent.shiftKey

Siehe auch

Deklaration: shiftKey als Boolean

Beschreibung
True, wenn die rechte Umschalttaste gedrückt wird.

4.2.6.16 AuthenticEvent.shiftLeft

Siehe auch

Deklaration: shiftLeft als Boolean

© 2018-2024 Altova GmbH

Beschreibung
True, wenn die linke Umschalttaste gedrückt wird.

4.2.6.17 AuthenticEvent.srcElement

Siehe auch

Deklaration: srcElement als Variant

Beschreibung
Element, das das aktuelle Ereignis auslöst.
Dies ist normalerweise ein XMLData[180]-Objekt.

Seit Version 3.0.0.0:

Die Eigenschaft kann auch eine Referenz auf ein AuthenticCommand[98]-Objekt enthalten, wenn Sie von einem ondoceditcommand-Ereignis aus gesetzt wurde.

4.2.6.18 AuthenticEvent.type

Siehe auch

Deklaration: type als String

Beschreibung
Derzeit wird diese Eigenschaft von keinem Ereignis gesetzt.

4.2.7 AuthenticEventContext

Über die `EventContext` Schnittstellen haben Sie Zugriff auf viele Eigenschaften des Kontexts, in dem ein Makro ausgeführt wird.

4.2.7.1 EvaluateXPath

Methode: `EvaluateXPath` (`string` expression) als string

Rückgabewert
Diese Methode wertet den XPath-Ausdruck im Kontext des Node aus, in dem das Event ausgelöst wurde, und gibt einen String zurück.

Beschreibung
`EvaluateXPath()` führt einen XPath-Ausdruck im angegebenen Event-Kontext aus. Das Ergebnis wird als String zurückgegeben. Im Fall einer Sequenz ist der String durch Leerzeichen getrennt.

Fehler
2201 Ungültiges Objekt.
2202 Kein Kontext.
2209 Ungültiger Parameter.
2210 Interner Fehler.
2211 XPath-Fehler.

4.2.7.2 GetEventContextType

Methode: `GetEventContextType ()` als `EventContextType` Enumeration

Rückgabewert
Gibt den Kontext-Node-Typ zurück.

Beschreibung
Mit Hilfe von `GetEventContextType` kann der Benutzer feststellen, ob sich das Makro in einem XML-Node oder in einem atomaren XPath Element-Kontext befindet. Die Enumeration `AuthenticEventContextType` ist folgendermaßen definiert:

```
authenticEventContextXML,
authenticEventContextAtomicItem,
authenticEventContextOther
```

Wenn der Kontext ein normaler XML-Node ist, erhalten Sie mit der Funktion `GetXMLNode()` Zugriff darauf (falls nicht, wird `NULL` zurückgegeben).

Fehler
2201 Ungültiges Objekt.
2202 Kein Kontext.
2209 Ungültiger Parameter.

4.2.7.3 GetNormalizedTextValue

Methode: `GetNormalizedTextValue()` Wert als String

Rückgabewert
Gibt den Wert des aktuellen Node als String zurück.

Fehler
2201 Ungültiges Objekt.
2202 Kein Kontext.
2203 Ungültiger Kontext
2209 Ungültiger Parameter.

© 2018-2024 Altova GmbH

4.2.7.4 GetVariableValue

Methode: `GetVariableValue(name als string, value als string)`

Rückgabewert
Ruft den Wert der benannten Variablen ab.

Beschreibung
`GetVariableValue` ruft den Wert der Variablen im Geltungsbereich des Kontexts ab.

```
nZoom = parseInt( AuthenticView.EventContext.GetVariableValue( 'Zoom' ) );
if ( nZoom > 1 )
{
	AuthenticView.EventContext.SetVariableValue( 'Zoom', nZoom - 1 );
}
```

Fehler

2201	Ungültiges Objekt.
2202	Kein Kontext.
2204	Keine Variable dieses Namens im Geltungsbereich
2205	Die Variable kann nicht ausgewertet werden
2206	Die Variable gibt eine Sequenz zurück
2209	Ungültiger Parameter

4.2.7.5 GetXMLNode

Methode: `GetXMLNode()` XMLData Objekt

Rückgabewert
Gibt den XML-Kontext-Node oder `NULL` zurück

Fehler

2201	Ungültiges Objekt.
2202	Kein Kontext.
2203	Ungültiger Kontext
2209	Ungültiger Parameter.

4.2.7.6 IsAvailable

Methode: `IsAvailable()`

Rückgabewert
Gibt "true" zurück, wenn `EventContext` definiert ist.

Fehler

2201	Ungültiges Objekt.

4.2.7.7 SetVariableValue

Methode: `SetVariableValue(name als string, value als string)`

Rückgabewert
Definiert den Wert der benannten Variablen.

Beschreibung
`SetVariableValue` definiert den Wert der Variablen im Geltungsbereich des Kontexts.

```
nZoom = parseInt( AuthenticView.EventContext.GetVariableValue( 'Zoom' ) );
if ( nZoom > 1 )
{
	AuthenticView.EventContext.SetVariableValue( 'Zoom', nZoom - 1 );
}
```

Fehler

2201	Ungültiges Objekt.
2202	Kein Kontext.
2204	Keine Variable dieses Namens im Geltungsbereich
2205	Die Variable kann nicht ausgewertet werden
2206	Die Variable gibt eine Sequenz zurück
2207	Die Variable ist schreibgeschützt
2208	Keine Änderung zulässig

4.2.8 AuthenticLoadObject

Eigenschaften
String 110
URL 111

Beschreibung
Das XMLSpyXMLLoadSave-Objekt dient dazu, die Quelle für die Dateien festzulegen, die geladen werden müssen. Sie können den Inhalt entweder direkt über die String-Eigenschaft festlegen oder als externen Pfad über die URL-Eigenschaft.

Nähere Informationen zur Verwendung der Eigenschaften finden Sie auch unter Authentic.SchemaLoadObject 92, Authentic.DesignDataLoadObject 75 und Authentic.XMLDataLoadObject 96 .

4.2.8.1 AuthenticLoadObject.String

Siehe auch

Deklaration: String als String

© 2018-2024 Altova GmbH

Beschreibung
Mit Hilfe dieser Eigenschaft können Sie die XML-Struktur von einem String aus setzen. Wenn Sie diese Eigenschaft verwenden möchten, muss die URL-Eigenschaft des Objekts leer sein.

4.2.8.2 AuthenticLoadObject.URL

Siehe auch

Deklaration: URL als String

Beschreibung
Die Eigenschaft sollte eine gültige URL für den Lade- oder Speichervorgang enthalten. Derzeit werden die HTTP-Protokolle http, https, ftp und gopher unterstützt.

4.2.9 AuthenticRange

Siehe auch

In der ersten Tabelle werden die Eigenschaften und Methoden von AuthenticRange aufgelistet, mit Hilfe derer Sie durch das Dokument navigieren und bestimmte Abschnitte auswählen können.

Eigenschaften
Application 113
FirstTextPosition 119
FirstXMLData 120
FirstXMLDataOffset 121
LastTextPosition 133
LastXMLData 134
LastXMLDataOffset 135
Parent 138

Clone 115
CollapsToBegin 115
CollapsToEnd 116
ExpandTo 119
Goto 125
GotoNext 125
GotoPrevious 126
IsEmpty 130
IsEqual 131

Methoden
MoveBegin 136
MoveEnd 136
NextCursorPosition 126
PreviousCursorPosition 127
Select 140
SelectNext 140
SelectPrevious 141
SetFromRange 143

In der folgenden Tabelle, werden die Methoden zum Ändern des Inhalts aufgelistet. Viele davon können durch Rechtsklick mit der Maus aufgerufen werden.

Eigenschaften
Text 143

Bearbeitungsoperationen
Copy 116
Cut 116
Delete 117
Paste 138

Operationen für dynamische Tabellen
AppendRow 113
DeleteRow 117
DuplicateRow 118
InsertRow 129
IsInDynamicTable 131
MoveRowDown 137
MoveRowUp 137

Die folgenden Methoden stellen die Funktionalitäten der Authentic-Eingabehilfenfenster für Range (Bereichs)-Objekte bereit.

Operationen der Eingabehilfenfenster

Elemente	Attribute	Entities
CanPerformActionWith 114	GetElementAttributeValue 123	GetEntityNames 124
CanPerformAction 114	GetElementAttributeNames 122	InsertEntity 128
PerformAction 139	GetElementHierarchy 123	
	HasElementAttribute 127	
	SetElementAttributeValue 142	

Beschreibung
Bei AuthenticRange-Objekten der Automation-Schnittstelle handelt es sich um die Selektion. Sie können damit den Cursor auf jede Cursorposition in der Authentic-Ansicht setzen oder einen Abschnitt des Dokuments auswählen. Die für AuthenticRange-Objekte verfügbaren Operationen haben dann auf diese Auswahl dieselbe Wirkung wie die entsprechenden Operationen der Benutzeroberfläche auf die aktuelle Auswahl in der Authentic-Ansicht. Der Hauptunterschied ist, dass Sie eine beliebige Anzahl von AuthenticRange-Objekten gleichzeitig verwenden können, während es bei der Benutzeroberfläche nur genau eine Selektion gibt.

Um zunächst einmal ein Bereichsobjekt zu erhalten, verwenden Sie AuthenticView.Selection 169, um einen Bereich zu erhalten, der der aktuellen Selektion auf der Benutzeroberfläche entspricht. Alternativ dazu können einige einfache Bereiche auch über die schreibgeschützten Eigenschaften AuthenticView.DocumentBegin 164, AuthenticView.DocumentEnd 164 und AuthenticView.WholeDocument 171 aufgerufen werden. Die flexibelste Methode ist AuthenticView.Goto 167, mit Hilfe derer Sie in nur einem Aufruf zu einem bestimmten Bereich im Dokument navigieren können. Für komplexere Auswahlvorgänge können Sie die obigen Methoden kombinieren. Die verschiedenen Navigationsmethoden zu Bereichsobjekten sind in der ersten Tabelle auf dieser Seite aufgelistet.

Ein andere Methode, einen Dokumentenabschnitt auszuwählen, ist die Verwendung der Positionseigenschaften des Bereichsobjekts. Es stehen zwei Systeme zur Positionierung zur Verfügung, die beliebig kombiniert werden können:

- **Absolute** Textcursor-Positionen, beginnend mit Position 0 am Dokumentanfang, können für den Anfang und das Ende eines Bereichs gesetzt und abgerufen werden. Nähere Informationen dazu finden Sie unter FirstTextPosition 119 und LastTextPosition 133. Zur Verwendung dieser Methode sind komplexe interne Berechnungen erforderlich, daher sollte sie mit Vorsicht verwendet werden.

- Das **XMLData-**Element und eine Textposition innerhalb dieses Elements können für den Anfang und das Ende eines Bereichs gesetzt und abgerufen werden. Nähere Informationen dazu finden Sie unter FirstXMLData 120, FirstXMLDataOffset 121, LastXMLData 134 und LastXMLDataOffset 135. Diese Methode ist äußerst effizient, erfordert aber Kenntnisse über die zugrunde liegende Dokumentenstruktur. Sie können damit XMLData-Objekte suchen und Operationen daran durchführen, die sonst über die Benutzeroberfläche nicht zur Verfügung stehen.

Änderungen am Dokumenteninhalt können durch verschiedene Methoden vorgenommen werden:

- mit Hilfe der Text 143-Eigenschaft können Sie den durch das Bereichsobjekt ausgewählten Dokumententext abrufen. Ist sie gesetzt, wird der ausgewählte Dokumententext durch den neuen Text ersetzt.
- die Standard-Dokumentbearbeitungsfunktionen Cut 116, Copy 116, Paste 138 und Delete 117.
- Tabellenoperationen für Tabellen, die dynamisch wachsen können.

© 2018-2024 Altova GmbH

- Methoden, die die Funktionalitäten der Authentic-Eingabehilfenfenster abbilden.
- Zugriff auf die XMLData-Objekte des zugrunde liegenden Dokuments, damit diese direkt geändert werden können.

4.2.9.1 AuthenticRange.AppendRow

Siehe auch

Methode: AppendRow () als Boolean

Beschreibung
Wenn sich der Anfang des Bereichs innerhalb einer dynamischen Tabelle befindet, wird mit der Methode eine neue Zeile am Ende der ausgewählten Tabelle eingefügt. Die Auswahl des Bereichs wird geändert, sodass sie auf den Beginn der neuen Zeile zeigt. Die Funktion gibt den Wert *true* zurück, wenn das Anhängen erfolgreich war. Andernfalls wird *false* zurückgegeben.

Fehler

2001	Das Authentic Range-Objekt oder das dazugehörige View-Objekt ist nicht mehr gültig.
2005	Für den zurückgegebenen Parameter wurde eine ungültige Adresse angegeben.

Beispiele

```
' ------------------------------------------------------
'                        VBScript
' Append row at end of current dynamically growable table
' ------------------------------------------------------
Dim objRange
Set objRange = objPlugin.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
```

4.2.9.2 AuthenticRange.Application

Siehe auch

Eigenschaft: Application als Authentic [69] (read-only)

Beschreibung
Ruft das Authentic Desktop Applikationsobjekt auf.

Fehler

2001	Das Authentic Range-Objekt oder das dazugehörige View-Objekt ist nicht mehr gültig.
2005	Für den zurückgegebenen Parameter wurde eine ungültige Adresse angegeben.

4.2.9.3 AuthenticRange.CanPerformAction

Siehe auch

Methode: CanPerformAction (*eAction* als SPYAuthenticActions, *strElementName* als String) als Boolean

Beschreibung
CanPerformAction und die dazugehörigen Methoden gestatten den Zugriff auf die Eingabehilfenfunktionen von Authentic. Mit Hilfe dieser Funktion können Sie den Inhalt eines Dokuments einfach und auf konsistente Weise bearbeiten, ohne wissen zu müssen, wo genau die Änderung vorgenommen wird. Der Anfang des Bereichsobjekts dient dazu, die nächste gültige Position zu finden, an der die angegebene Aktion durchgeführt werden kann. Wird diese Position gefunden, gibt die Methode *True* zurück, andernfalls *False*.

HINWEIS: Um alle gültigen Elementnamen für eine bestimmte Aktion zu finden, verwenden Sie CanPerformActionWith[114].

Fehler

2001	Das Authentic Range-Objekt oder das dazugehörige View-Objekt ist nicht mehr gültig.
2005	Für den zurückgegebenen Parameter wurde eine ungültige Adresse angegeben.
2007	Es wurde eine ungültige Aktion definiert.

Beispiele
Siehe PerformAction[139].

4.2.9.4 AuthenticRange.CanPerformActionWith

Siehe auch

Methode: CanPerformActionWith (*eAction* als SPYAuthenticActions, *out_arrElementNames* als Variant)

Beschreibung
CanPerformActionWith und die dazugehörigen Methoden ermöglichen den Zugriff auf die Eingabehilfenfunktionen von Authentic. Mit Hilfe dieser Funktion können Sie den Inhalt eines Dokuments einfach und auf konsistente Weise bearbeiten, ohne wissen zu müssen, wo genau die Änderung vorgenommen wird.

© 2018-2024 Altova GmbH

Diese Methode gibt einen Array der Elementnamen zurück, an denen die angegebene Aktion ausgeführt werden kann.

HINWEIS: Verwenden Sie PerformActionWith[139], um die Aktion anzuwenden.

Fehler

2001	Das Authentic Range-Objekt oder das dazugehörige View-Objekt ist nicht mehr gültig.
2005	Für den zurückgegebenen Parameter wurde eine ungültige Adresse angegeben.
2007	Es wurde eine ungültige Aktion definiert.

Beispiele

Siehe PerformAction[139].

4.2.9.5 AuthenticRange.Clone

Siehe auch

Methode: Clone () als AuthenticRange[111]

Beschreibung

Gibt eine Kopie des Bereichsobjekts zurück.

Fehler

2001	Das Authentic Range-Objekt oder das dazugehörige View-Objekt ist nicht mehr gültig.
2005	Für den zurückgegebenen Parameter wurde eine ungültige Adresse angegeben.

4.2.9.6 AuthenticRange.CollapsToBegin

Siehe auch

Methode: CollapsToBegin () als AuthenticRange[111]

Beschreibung

Setzt das Ende des Bereichsobjekts and seinen Beginn. Die Methode gibt das geänderte Bereichsobjekt zurück.

Fehler

2001	Das Authentic Range-Objekt oder das dazugehörige View-Objekt ist nicht mehr gültig.
2005	Für den zurückgegebenen Parameter wurde eine ungültige Adresse angegeben.

4.2.9.7 AuthenticRange.CollapsToEnd

Siehe auch

Methode: CollapsToEnd () als AuthenticRange[111]

Beschreibung
Setzt den Beginn des Bereichsobjekts an seinen Anfang. Die Methode gibt das geänderte Bereichsobjekt zurück.

Fehler

2001	Das Authentic Range-Objekt oder das dazugehörige View-Objekt ist nicht mehr gültig.
2005	Für den zurückgegebenen Parameter wurde eine ungültige Adresse angegeben.

4.2.9.8 AuthenticRange.Copy

Siehe auch

Methode: Copy () als Boolean

Beschreibung
Gibt *False* zurück, wenn der Bereich keine Dokumentabschnitte enthält, die kopiert werden können.
Gibt *True* zurück, wenn Text sowie - im Falle von vollständig selektierten XML-Elementen - die Elemente selbst in den Kopier-/Einfügepuffer kopiert wurden.

Fehler

2001	Das Authentic Range-Objekt oder das dazugehörige View-Objekt ist nicht mehr gültig.
2005	Für den zurückgegebenen Parameter wurde eine ungültige Adresse angegeben.

4.2.9.9 AuthenticRange.Cut

Siehe auch

Methode: Cut () als Boolean

Beschreibung
Gibt *False* zurück, wenn der Bereich Dokumentabschnitte enthält, die nicht gelöscht werden dürfen.
Gibt den Wert *True* zurück, wenn Text sowie - im Falle von vollständig selektierten XML-Elementen - die Elemente selbst aus dem Dokument gelöscht wurden und im Kopier-/Einfügepuffer gespeichert wurden.

© 2018-2024 Altova GmbH

Fehler

2001 Das Authentic Range-Objekt oder das dazugehörige View-Objekt ist nicht mehr gültig.
2005 Für den zurückgegebenen Parameter wurde eine ungültige Adresse angegeben.

4.2.9.10 AuthenticRange.Delete

Siehe auch

Methode: Delete () als Boolean

Beschreibung
Gibt *False* zurück, wenn der Bereich Dokumentabschnitte enthält, die nicht gelöscht werden dürfen. Gibt den Wert *True* zurück, wenn Text sowie - im Falle von vollständig selektierten XML-Elementen - die Elemente selbst aus dem Dokument gelöscht wurden.

Fehler

2001 Das Authentic Range-Objekt oder das dazugehörige View-Objekt ist nicht mehr gültig.
2005 Für den zurückgegebenen Parameter wurde eine ungültige Adresse angegeben.

4.2.9.11 AuthenticRange.DeleteRow

Siehe auch

Methode: DeleteRow () als Boolean

Beschreibung
Wenn sich der Anfang des Bereichs innerhalb einer dynamischen Tabelle befindet, löscht diese Methode die ausgewählte Zeile. Die Auswahl des Bereichs wird geändert und zeigt auf das nächste Element nach der gelöschten Zeile. Die Funktion gibt *true* zurück, wenn der Löschvorgang erfolgreich war, andernfalls wird *false* zurückgegeben.

Fehler

2001 Das Authentic Range-Objekt oder das dazugehörige View-Objekt ist nicht mehr gültig.
2005 Für den zurückgegebenen Parameter wurde eine ungültige Adresse angegeben.

Beispiele

```
' ------------------------------------------------------
'                      VBScript
' Delete selected row from dynamically growing table
' ------------------------------------------------------
Dim objRange
Set objRange = objPlugin.ActiveDocument.AuthenticView.Selection
```

```
' check if we are in a table
If objRange.IsInDynamicTable Then
     objRange.DeleteRow
End If
```

4.2.9.12 AuthenticRange.DuplicateRow

Siehe auch

Methode: DuplicateRow () als Boolean

Beschreibung
Wenn sich der Anfang des Bereichs innerhalb einer dynamischen Tabelle befindet, fügt diese Methode eine Kopie der aktuellen Zeile hinter der ausgewählten Zeile ein. Die Funktion gibt *true* zurück, wenn der Kopiervorgang erfolgreich war, andernfalls wird *false* zurückgegeben.

Fehler

2001	Das Authentic Range-Objekt oder das dazugehörige View-Objekt ist nicht mehr gültig.
2005	Für den zurückgegebenen Parameter wurde eine ungültige Adresse angegeben.

Beispiele

```
' --------------------------------------------------------
'                       VBScript
' dupplicate row in current dynamically growable table
' --------------------------------------------------------
Dim objRange
Set objRange = objPlugin.ActiveDocument.AuthenticView.Selection

' check if we can insert soemthing
If objRange.IsInDynamicTable Then
     objRange.DuplicateRow
     ' objRange points to begining of new row
     objRange.Select
End If
```

4.2.9.13 AuthenticRange.EvaluateXPath

Methode: `EvaluateXPath` (string expression) als string

Rückgabewert
Die Methode gibt einen String zurück

© 2018-2024 Altova GmbH

Beschreibung
`EvaluateXPath()` führt einen XPath-Ausdruck aus, wobei der Kontext-Node der Anfang der Bereichsauswahl ist. Das als String zurückgegebene Ergebnis ist im Fall einer Sequenz ein durch Leerzeichen getrennter String. Wenn der XML-Node nicht relevant ist, kann der Benutzer jeden Node angeben, wie z.B. `AuthenticView.XMLDataRoot`.

Fehler

2001	Ungültiges Objekt
2005	Ungültiger Parameter
2008	Interner Fehler
2202	Fehlender Kontext-Node
2211	XPath-Fehler

4.2.9.14 AuthenticRange.ExpandTo

Siehe auch

Methode: ExpandTo (*eKind* als SPYAuthenticElementKind) als AuthenticRange[111]

Beschreibung
Wählt das gesamte Element vom Typ *eKind* aus, das an der ersten Cursorposition des Bereichs beginnt bzw. diese enthält. Die Methode gibt das geänderte Bereichsobjekt zurück.

Fehler

2001	Das Authentic Range-Objekt oder das dazugehörige View-Objekt ist nicht mehr gültig.
2003	Die Erweiterung des Bereichs würde über das Ende des Dokuments hinausgehen.
2005	Für den zurückgegebenen Parameter wurde eine ungültige Adresse angegeben.

4.2.9.15 AuthenticRange.FirstTextPosition

Siehe auch

Eigenschaft: FirstTextPosition als Long

Beschreibung
Setzt oder holt den am weitesten links befindlichen Textpositionsindex des Bereichsobjekts. Dieser Index ist immer kleiner oder gleich der LastTextPosition[133]. Die Indizierung beginnt am Anfang des Dokuments mit 0 und erhöht sich schrittweise mit jeder neuen Position, an der sich der Textcursor befinden kann. Eine Erhöhung der Textposition um 1 hat dieselbe Wirkung wie die Rechtspfeil-Taste. Eine Verringerung der Textposition um den Wert 1 hat dieselbe Wirkung wie die Linkspfeil-Taste.

Wenn Sie *FirstTextPosition* auf einen Wert setzen, der größer ist als die aktuelle LastTextPosition[133], wird LastTextPosition[133] auf den Wert der neuen *FirstTextPosition* gesetzt.

HINWEIS: Verwenden Sie Textcursorpositionen mit Vorsicht, da dies im Vergleich zur Cursorpositionierung auf Basis von XMLData ein aufwändiger Vorgang ist.

Fehler

2001	Das Authentic Range-Objekt oder das dazugehörige View-Objekt ist nicht mehr gültig.
2005	Für den zurückgegebenen Parameter wurde eine ungültige Adresse angegeben.
2006	Es wurde eine Textposition außerhalb des Dokuments angegeben

Beispiele

```
  ' ------------------------------------
  '               VBScript
  ' ------------------------------------
  Dim objAuthenticView
  ' we assume that the active document is open in authentic view mode
  Set objAuthenticView = objPlugin.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
```

4.2.9.16 AuthenticRange.FirstXMLData

Siehe auch

Eigenschaft: FirstXMLData als XMLData

Beschreibung
Setzt oder holt das erste XMLData-Element im zugrunde liegenden Dokument, das zum Teil oder zur Gänze vom Bereich ausgewählt ist. Der genaue Beginn der Auswahl wird durch das FirstXMLDataOffset[121] Attribut definiert.

Wenn FirstXMLData auf ein neues Datenobjekt gesetzt wird, wird FirstXMLDataOffset[121] auf die erste Cursorposition innerhalb dieses Elements gesetzt. Es dürfen nur XMLData-Objekte verwendet werden, für die eine Cursorposition möglich ist. Wenn Sie *FirstXMLData* setzen / wählt FirstXMLDataOffset[121] eine Position

© 2018-2024 Altova GmbH

aus, die größer ist als der aktuelle Wert von LastXMLData[134] / LastXMLDataOffset[135]. Das zweite wird auf die neue Anfangsposition verschoben.

HINWEIS: Sie können mit Hilfe der Eigenschaften FirstXMLData[120] und LastXMLData[134] direkt auf das zugrunde liegende XML-Dokument zugreifen und dieses manipulieren, wenn die mit dem AuthenticRange[111]-Objekt verfügbaren Methoden nicht ausreichen.

Fehler

2001	Das Authentic Range-Objekt oder das dazugehörige View-Objekt ist nicht mehr gültig.
2005	Für den zurückgegebenen Parameter wurde eine ungültige Adresse angegeben.
2008	Interner Fehler
2009	Das XMLData-Objekt kann nicht aufgerufen werden.

Beispiele

```
' ------------------------------------------------
'                 VBScript
' show name of currently selected XMLData element
' ------------------------------------------------
Dim objAuthenticView
Set objAuthenticView = objPlugin.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
```

4.2.9.17 AuthenticRange.FirstXMLDataOffset

Siehe auch

Eigenschaft: FirstXMLDataOffset als Long

Beschreibung
Setzt oder holt den Cursorpositions-Offset im FirstXMLData[120]-Element für den Anfang des Bereichs. Die Offset-Positionen basieren auf den von der Text[143]-Eigenschaft zurückgegebenen Zeichen und beginnen mit dem Wert 0. Verwenden Sie beim Setzen eines neuen Offset den Wert -1, um den Offset an die letzte mögliche Position im Element zu setzen. Seien Sie besonders in den folgenden Fällen vorsichtig:

- Die Textform von Einträgen in Auswahllisten, Kontrollkästchen und ähnlichen Steuerelementen unterscheidet sich unter Umständen von dem, was auf dem Bildschirm angezeigt wird. Der Daten-Offset basiert zwar auf diesem Text, doch gibt es nur zwei gültige Offset-Positionen, eine am Anfang und eine am Ende des Eintrags. Wenn Sie versuchen, den Offset in die Mitte des Eintrags zu positionieren, wird der Offset an das Ende des Eintrags gesetzt.

- Die Textform von XML Entities kann sich in ihrer Darstellung von der auf dem Bildschirm dargestellten Länge unterscheiden. Der Offset basiert auf der Textform der Entities.

Wenn *FirstXMLData* / FirstXMLDataOffset[121] eine Position hinter dem aktuellen LastXMLData[134] / LastXMLDataOffset[135] auswählt, wird letzteres an die neue Anfangsposition verschoben.

Fehler

2001	Das Authentic Range-Objekt oder das dazugehörige View-Objekt ist nicht mehr gültig.
2005	Es wurde ein ungültiger Offset-Wert angegeben. Für den zurückgegebenen Parameter wurde eine ungültige Adresse angegeben.

Beispiele

```
' ------------------------------------------
'               VBScript
' Select the complete text of an XMLData element
' using XMLData based selection and ExpandTo
' ------------------------------------------
Dim objAuthenticView
Set objAuthenticView = objPlugin.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
```

4.2.9.18 AuthenticRange.GetElementAttributeNames

Siehe auch

Methode: GetElementAttributeNames (*strElementName* als String, *out_arrAttributeNames* als Variant)

Beschreibung
Liefert die Namen aller Attribute für das einschließende Element mit dem angegebene Namen. Verwenden Sie die Element- / Attribut-Paare, um den Attributwert mit Hilfe der Methoden GetElementAttributeValue[123] und SetElementAttributeValue[142] zu setzen oder zu holen.

© 2018-2024 Altova GmbH

Fehler

2001 Das Authentic Range-Objekt oder das dazugehörige View-Objekt ist nicht mehr gültig.

2005 Es wurde ein ungültiger Elementname angegeben.
Für den zurückgegebenen Parameter wurde eine ungültige Adresse angegeben.

Beispiele

Siehe SetElementAttributeValue[142].

4.2.9.19 AuthenticRange.GetElementAttributeValue

Siehe auch

Methode: GetElementAttributeValue (*strElementName* als String, *strAttributeName* als String) als String

Beschreibung

Liefert für das mit *strElementName* definierte Element den Wert des in *strAttributeName* definierten Attributs. Wenn das Attribut unterstützt wird, ihm jedoch kein Wert zugewiesen wurde, wird ein leerer String zurückgegeben. Um herauszufinden, welche Attribute von einem Element unterstützt werden, verwenden Sie GetElementAttributeNames[122] oder HasElementAttribute[127].

Fehler

2001 Das Authentic Range-Objekt oder das dazugehörige View-Objekt ist nicht mehr gültig.

2005 Es wurde ein ungültiger Elementname angegeben.
Es wurde ein ungültiger Attributname angegeben.
Für den zurückgegebenen Parameter wurde eine ungültige Adresse angegeben.

Beispiele

Siehe SetElementAttributeValue[142].

4.2.9.20 AuthenticRange.GetElementHierarchy

Siehe auch

Methode: GetElementHierarchy (*out_arrElementNames* als Variant)

Beschreibung

Liefert die Namen aller XML-Elemente, die Parent-Elemente der aktuellen Auswahl sind. Innere Elemente werden vor den diese einschließenden Elementen aufgelistet. Wenn sich die aktuelle Auswahl nicht innerhalb eines einzelnen XMLData-Elements befindet, wird eine leere Liste zurückgegeben.

Zusammen mit dem Bereichsobjekt identifizieren die Namen der Elementhierarchie XMLData-Elemente eindeutig. Die Attribute dieser Elemente können direkt durch GetElementAttributeNames[122] und die dazugehörigen Methoden aufgerufen werden.

Fehler

2001	Das Authentic Range-Objekt oder das dazugehörige View-Objekt ist nicht mehr gültig.
2005	Für den zurückgegebenen Parameter wurde eine ungültige Adresse angegeben.

Beispiele

Siehe SetElementAttributeValue[142].

4.2.9.21 AuthenticRange.GetEntityNames

Siehe auch

Methode: GetEntityNames (*out_arrEntityNames* als Variant)

Beschreibung

Liefert die Namen aller definierten Entities. Die Liste der abgerufenen Entities ist unabhängig von der aktuellen Auswahl oder der aktuellen Position. Verwenden Sie einen dieser Namen mit der InsertEntity[128]-Funktion.

Fehler

2001	Das Authentic Range-Objekt oder das dazugehörige View-Objekt ist nicht mehr gültig.
2005	Für den zurückgegebenen Parameter wurde eine ungültige Adresse angegeben.

Beispiele

Siehe InsertEntity[128].

4.2.9.22 AuthenticRange.GetVariableValue

Methode: `GetVariableValue(name als string, value als string)`

Rückgabewert

Ruft den Wert der benannten Variable ab.

Fehler

2001	Ungültiges Objekt.
2202	Kein Kontext.
2204	Es befindet sich keine Variable dieses Namens im Geltungsbereich
2205	Die Variable kann nicht ausgewertet werden
2206	Die Variable gibt eine Sequenz zurück
2209	Ungültiger Parameter

© 2018-2024 Altova GmbH

4.2.9.23 AuthenticRange.Goto

Siehe auch

Methode: Goto (*eKind* als SPYAuthenticElementKind, *nCount* als Long, *eFrom* als SPYAuthenticDocumentPosition) als AuthenticRange[111]

Beschreibung
Setzt den Bereich an den Anfang des *nCount*-Elements vom Typ *eKind*. Die Anfangsposition wird durch den Parameter *eFrom* definiert.

Verwenden Sie zum Navigieren zum Ende des Dokuments positive Positionswerte für *nCount*. Verwenden Sie negative Werte, um zum Anfang des Dokuments zu navigieren. Die Methode gibt das geänderte Range-Objekt zurück.

Fehler

2001	Das Authentic Range-Objekt oder das dazugehörige View-Objekt ist nicht mehr gültig.
2003	Das Ziel befindet sich hinter dem Ende des Dokuments.
2004	Das Ziel befindet sich vor dem Anfang des Dokuments.
2005	Es wurde ein ungültiger Elementtyp definiert. Es wurde eine ungültige Anfangsposition definiert. Für den zurückgegebenen Parameter wurde eine ungültige Adresse angegeben.

4.2.9.24 AuthenticRange.GotoNext

Siehe auch

Methode: GotoNext (*eKind* als SPYAuthenticElementKind) als AuthenticRange[111]

Beschreibung
Setzt den Bereich an den Anfang des nächsten Elements vom Typ *eKind*. Die Methode gibt das geänderte Bereichsobjekt zurück.

Fehler

2001	Das Authentic Range-Objekt oder das dazugehörige View-Objekt ist nicht mehr gültig.
2003	Das Ziel befindet sich hinter dem Ende des Dokuments.
2005	Es wurde ein ungültiger Elementtyp definiert. Für den zurückgegebenen Parameter wurde eine ungültige Adresse angegeben.

Beispiele

```
' ----------------------------------------
'               VBScript
' Scan through the whole document word-by-word
' ----------------------------------------
Dim objAuthenticView
Set objAuthenticView = objPlugin.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
```

4.2.9.25 AuthenticRange.GotoNextCursorPosition

Siehe auch

Methode: GotoNextCursorPosition () als AuthenticRange[111]

Beschreibung
Setzt den Bereich auf die nächste Cursorposition hinter dem Ende der aktuellen Endposition. Gibt das geänderte Objekt zurück.

Fehler

2001	Das Authentic Range-Objekt oder das dazugehörige View-Objekt ist nicht mehr gültig.
2003	Das Ziel befindet sich hinter dem Ende des Dokuments.
2005	Für den zurückgegebenen Parameter wurde eine ungültige Adresse angegeben.

4.2.9.26 AuthenticRange.GotoPrevious

Siehe auch

Methode: GotoPrevious (*eKind* als SPYAuthenticElementKind) als AuthenticRange[111]

Beschreibung
Setzt den Bereich an den Anfang des Elements vom Typ *eKind*, welches sich vor dem Anfang des aktuellen Bereichs befindet. Die Methode gibt das geänderte Bereichsobjekt zurück.

Fehler

2001	Das Authentic Range-Objekt oder das dazugehörige View-Objekt ist nicht mehr gültig.
2004	Das Ziel befindet sich vor dem Anfang des Dokuments.
2005	Es wurde ein ungültiger Elementtyp definiert.

© 2018-2024 Altova GmbH

Für den zurückgegebenen Parameter wurde eine ungültige Adresse angegeben.

Beispiele

```
' ----------------------------------------
'                  VBScript
' Scan through the whole document tag-by-tag
' ----------------------------------------
Dim objAuthenticView
Set objAuthenticView = objPlugin.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
```

4.2.9.27 AuthenticRange.GotoPreviousCursorPosition

Siehe auch

Methode: GotoPreviousCursorPosition () als AuthenticRange[111]

Beschreibung
Setzt den Bereich der Cursorposition unmittelbar vor die aktuelle Position. Gibt das geänderte Objekt zurück.

Fehler

2001	Das Authentic Range-Objekt oder das dazugehörige View-Objekt ist nicht mehr gültig.
2004	Das Ziel befindet sich vor dem Anfang des Dokuments.
2005	Für den zurückgegebenen Parameter wurde eine ungültige Adresse angegeben.

4.2.9.28 AuthenticRange.HasElementAttribute

Siehe auch

Methode: HasElementAttribute (*strElementName* als String, *strAttributeName* als String) als Boolean

Beschreibung
Testet, ob das einschließende Element mit dem Namen *strElementName* das in *strAttributeName* definierte Attribut unterstützt.

Fehler

2001	Das Authentic Range-Objekt oder das dazugehörige View-Objekt ist nicht mehr gültig.
2005	Es wurde ein ungültiger Elementname definiert. Für den zurückgegebenen Parameter wurde eine ungültige Adresse angegeben.

4.2.9.29 AuthenticRange.InsertEntity

Siehe auch

Methode: InsertEntity (*strEntityName* als String)

Beschreibung
Ersetzt die Bereichsauswahl durch die angegebene Entity. Die Entity muss einer der Entity-Namen sein, die von GetEntityNames[124] zurückgegeben werden.

Fehler

2001	Das Authentic Range-Objekt oder das dazugehörige View-Objekt ist nicht mehr gültig.
2005	Es wurde ein ungültiger Name für den Eintrag angegeben.

Beispiele

```
' ------------------------------------------------------
'                        VBScript
' Insert the first entity in the list of availabel entities
' ------------------------------------------------------
Dim objRange
Set objRange = objPlugin.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)
     objRange.Select()
Else
     MsgBox "Sorry, no entities are available for this document"
End If
```


© 2018-2024 Altova GmbH

4.2.9.30 AuthenticRange.InsertRow

Siehe auch

Methode: InsertRow () als Boolean

Beschreibung
Wenn sich der Anfang des Bereichs innerhalb einer dynamischen Tabelle befindet, wird mit der Methode eine neue Zeile vor der aktuellen eingefügt. Die Auswahl des Bereichs wird geändert, sodass sie auf den Beginn der neu eingefügten Zeile zeigt. Die Funktion gibt den Wert *true* zurück, wenn das Einfügen erfolgreich war. Andernfalls wird *false* zurückgegeben.

Fehler

2001	Das Authentic Range-Objekt oder das dazugehörige View-Objekt ist nicht mehr gültig.
2005	Für den zurückgegebenen Parameter wurde eine ungültige Adresse angegeben.

Beispiele

```
' ------------------------------------------------------------
'                      VBScript
' Insert row at beginning of current dynamically growing table
' ------------------------------------------------------------
Dim objRange
Set objRange = objPlugin.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
```

4.2.9.31 AuthenticRange.IsCopyEnabled

Siehe auch

Eigenschaft: `IsCopyEnabled` als Boolean (schreibgeschützt)

Beschreibung
Überprüft, ob die Kopieroperation für diesen Bereich unterstützt wird.

Fehler

2001	Das Authentic-Bereichsobjekt oder das damit in Zusammenhang stehende Ansichtsobjekt ist nicht mehr gültig.
2005	Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

4.2.9.32 AuthenticRange.IsCutEnabled

Siehe auch

Eigenschaft: `IsCutEnabled` als Boolean (schreibgeschützt)

Beschreibung
Überprüft, ob die Ausschneideoperation für diesen Bereich unterstützt wird.

Fehler
2001 Das Authentic-Bereichsobjekt oder das damit in Zusammenhang stehende Ansichtsobjekt ist nicht mehr gültig.
2005 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

4.2.9.33 AuthenticRange.IsDeleteEnabled

Siehe auch

Eigenschaft: `IsDeleteEnabled` als Boolean (schreibgeschützt)

Beschreibung
Überprüft, ob die Löschoperation für diesen Bereich unterstützt wird.

Fehler
2001 Das Authentic-Bereichsobjekt oder das damit in Zusammenhang stehende Ansichtsobjekt ist nicht mehr gültig.
2005 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

4.2.9.34 AuthenticRange.IsEmpty

Siehe auch

Methode: IsEmpty () als Boolean

Beschreibung
Testet, ob die erste und die letzte Position des Bereichs identisch sind.

Fehler
2001 Das Authentic Range-Objekt oder das dazugehörige View-Objekt ist nicht mehr gültig.
2005 Für den zurückgegebenen Parameter wurde eine ungültige Adresse angegeben.

© 2018-2024 Altova GmbH

4.2.9.35 AuthenticRange.IsEqual

Siehe auch

Methode: IsEqual (*objCmpRange* als AuthenticRange[111]) als Boolean

Beschreibung
Testet, ob der Beginn und das Ende der beiden Bereiche identisch sind.

Fehler
2001 Eines der beiden verglichenen Bereichsobjekte ist ungültig.
2005 Für den zurückgegebenen Parameter wurde eine ungültige Adresse angegeben.

4.2.9.36 AuthenticRange.IsFirstRow

Siehe auch

Eigenschaft: `IsFirstRow()` als Boolean (schreibgeschützt)

Beschreibung
Überprüft, ob der Bereich in der ersten Zeile einer Tabelle ist. Welche Tabelle berücksichtigt wird, ist vom Ausmaß des Bereichs abhängig. Wenn die Auswahl über eine einzige Tabellenzeile hinausgeht, wird überprüft, ob diese Tabelle das erste Element in einer einbettenden Tabelle ist. Nähere Informationen dazu finden Sie im Benutzerhandbuch unter Eingabehilfen.

Fehler
2001 Das Authentic-Bereichsobjekt oder das damit in Zusammenhang stehende Ansichtsobjekt ist nicht mehr gültig.
2005 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

4.2.9.37 AuthenticRange.IsInDynamicTable

Siehe auch

Methode: IsInDynamicTable () als Boolean

Beschreibung
Testet, ob sich der Anfang eines Bereichs innerhalb einer Tabelle befindet, die andere Zeilenoperationen unterstützt.

Fehler
2001 Das Authentic Range-Objekt oder das dazugehörige View-Objekt ist nicht mehr gültig.
2005 Für den zurückgegebenen Parameter wurde eine ungültige Adresse angegeben.

4.2.9.38 AuthenticRange.IsLastRow

Siehe auch

Eigenschaft: `IsLastRow()` als Boolean (schreibgeschützt)

Beschreibung
Überprüft, ob der Bereich in der letzten Zeile einer Tabelle ist. Welche Tabelle berücksichtigt wird, hängt vom Ausmaß des Bereichs ab. Wenn die Auswahl über eine einzige Tabellenzeile hinausgeht, wird überprüft, ob diese Tabelle das letzte Element in einer einbettenden Tabelle ist. Nähere Informationen dazu finden Sie im Benutzerhandbuch unter Eingabehilfen.

Fehler
2001 Das Authentic-Bereichsobjekt oder das damit in Zusammenhang stehende Ansichtsobjekt ist nicht mehr gültig.
2005 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

4.2.9.39 AuthenticRange.IsPasteEnabled

Siehe auch

Eigenschaft: `IsPasteEnabled` als Boolean (schreibgeschützt)

Beschreibung
Überprüft, ob die Einfügeoperatione für diesen Bereich unterstützt wird.

Fehler
2001 Das Authentic-Bereichsobjekt oder das damit in Zusammenhang stehende Ansichtsobjekt ist nicht mehr gültig.
2005 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

4.2.9.40 AuthenticRange.IsSelected

Eigenschaft: `IsSelected` als Boolean

Beschreibung
Gibt true() zurück, wenn die Auswahl vorhanden ist. Der Auswahlbereich kann dennoch leer sein. Dies passiert z.B. wenn der Cursor gesetzt ist.

© 2018-2024 Altova GmbH

4.2.9.41 AuthenticRange.IsTextStateApplied

Siehe auch

Methode: `IsTextStateApplied` (*`i_strElementName`* als String) als Boolean

Beschreibung
Überprüft, ob der gesamte ausgewählte Text in ein XML-Element mit dem Namen `i_strElementName` eingebettet ist. Ein typisches Beispiel für den Parameter `i_strElementName` ist "strong", "bold" oder "italic".

Fehler

2001	Das Authentic-Bereichsobjekt oder das damit in Zusammenhang stehende Ansichtsobjekt ist nicht mehr gültig.
2005	Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

4.2.9.42 AuthenticRange.LastTextPosition

Siehe auch

Eigenschaft: LastTextPosition als Long

Beschreibung
Setzt oder holt den am weitesten rechts befindlichen Textpositionsindex des Bereichsobjekts. Dieser Index ist immer größer oder gleich der FirstTextPosition[119]. Die Indizierung beginnt am Anfang des Dokuments mit 0 und erhöht sich schrittweise mit jeder neuen Position, an der sich der Textcursor befinden kann. Eine Erhöhung der Textposition um 1 hat dieselbe Wirkung wie die Rechtspfeil-Taste. Eine Verringerung der Textposition um den Wert 1 hat dieselbe Wirkung wie die Linkspfeil-Taste.

Wenn Sie *LastTextPosition* auf einen Wert setzen, der kleiner ist als die aktuelle FirstTextPosition[119], wird FirstTextPosition[119] auf den Wert der neuen *LastTextPosition* gesetzt.

HINWEIS: Verwenden Sie Textcursorpositionen mit Vorsicht, da dies im Vergleich zur Cursorpositionierung auf Basis von XMLData ein aufwändiger Vorgang ist.

Fehler

2001	Das Authentic Range-Objekt oder das dazugehörige View-Objekt ist nicht mehr gültig.
2005	Für den zurückgegebenen Parameter wurde eine ungültige Adresse angegeben.
2006	Es wurde eine Textposition außerhalb des Dokuments angegeben

Beispiele

```
' --------------------------------------
'                VBScript
' --------------------------------------
Dim objAuthenticView
Set objAuthenticView = objPlugin.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
```

4.2.9.43 AuthenticRange.LastXMLData

Siehe auch

Eigenschaft: LastXMLData als XMLData

Beschreibung
Setzt oder holt das letzte XMLData-Element im zugrunde liegenden Dokument, das zum Teil oder zur Gänze vom Bereich ausgewählt ist. Das genaue Ende der Auswahl wird durch das LastXMLDataOffset[135] Attribut definiert.

Wenn *LastXMLData* auf ein neues Datenobjekt gesetzt wird, wird LastXMLDataOffset[135] auf die letzte Cursorposition innerhalb dieses Elements gesetzt. Es dürfen nur XMLData-Objekte verwendet werden, für die es eine Cursorposition gibt. Wenn Sie *LastXMLData* / LastXMLDataOffset[135] setzen, wählen Sie eine Position aus, die kleiner ist als der aktuelle Wert von FirstXMLData[120] / FirstXMLDataOffset[121]. Letzteres wird auf die neue Endposition verschoben.

HINWEIS: Sie können mit Hilfe der Eigenschaften FirstXMLData[120] und LastXMLData[134] direkt auf das zugrunde liegende XML-Dokument zugreifen und dieses manipulieren, wenn die mit dem AuthenticRange[111]-Objekt verfügbaren Methoden nicht ausreichen.

Fehler

2001	Das Authentic Range-Objekt oder das dazugehörige View-Objekt ist nicht mehr gültig.
2005	Für den zurückgegebenen Parameter wurde eine ungültige Adresse angegeben.
2008	Interner Fehler
2009	Das XMLData-Objekt kann nicht aufgerufen werden.

© 2018-2024 Altova GmbH

4.2.9.44 AuthenticRange.LastXMLDataOffset

Siehe auch

Eigenschaft: LastXMLDataOffset als Long

Beschreibung
Setzt oder holt die Cursorposition im LastXMLData[134]-Element für das Ende des Bereichs.

Die Offset-Positionen basieren auf den von der Text[143]-Eigenschaft zurückgegebenen Zeichen und beginnen mit dem Wert 0. Verwenden Sie beim Setzen eines neuen Offset den Wert -1, um den Offset an die letzte mögliche Position im Element zu setzen. Seien Sie besonders in den folgenden Fällen vorsichtig:

- Die Textform von Einträgen in Auswahllisten, Kontrollkästchen und ähnlichen Steuerelementen unterscheidet sich unter Umständen von dem, was auf dem Bildschirm angezeigt wird. Der Daten-Offset basiert zwar auf diesem Text, doch gibt es nur zwei gültige Offset-Positionen, eine am Anfang und eine am Ende des Eintrags. Wenn Sie versuchen, den Offset in die Mitte des Eintrags zu positionieren, wird der Offset an das Ende des Eintrags gesetzt.

- Die Textform von XML Entities kann sich in ihrer Darstellung von der auf dem Bildschirm dargestellten Länge unterscheiden. Der Offset basiert auf der Textform der Entities z.B. &.

Wenn LastXMLData[134] / LastXMLDataOffset[135] eine Position vor dem aktuellen FirstXMLData[120] / FirstXMLDataOffset[121] auswählt, wird letzteres an die neue Endposition verschoben.

Fehler

2001	Das Authentic Range-Objekt oder das dazugehörige View-Objekt ist nicht mehr gültig.
2005	Es wurde ein ungültiger Offset-Wert angegeben. Für den zurückgegebenen Parameter wurde eine ungültige Adresse angegeben.

Beispiele

```
' ------------------------------------------
'                 VBScript
' Select the complete text of an XMLData element
' using XMLData based selection and ExpandTo
' ------------------------------------------
Dim objAuthenticView
Set objAuthenticView = objPlugin.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
```

4.2.9.45 AuthenticRange.MoveBegin

Siehe auch

Methode: MoveBegin (*eKind* als SPYAuthenticElementKind, *nCount* als Long) als AuthenticRange[111]

Beschreibung
Verschiebt den Anfang des Bereichs an den Anfang des *nCount* Elements vom Typ *eKind*. Die Zählung beginnt am aktuellen Beginn des Bereichsobjekts.

Mit Hilfe von positiven Zahlen für *nCount* können Sie die Position zum Ende des Dokuments hin verschieben, mit Hilfe von negativen Zahlen können Sie die Position zum Beginn des Dokuments hin verschieben. Das Ende des Bereichs wird nicht verschoben, es sei denn, der neue Anfang des Bereichs wäre größer als der Wert. In diesem Fall wird das Ende auf den neuen Anfang verschoben. Die Methode gibt das geänderte Bereichsobjekt zurück.

Fehler

2001	Das Authentic Range-Objekt oder das dazugehörige View-Objekt ist nicht mehr gültig.
2003	Das Ziel befindet sich hinter dem Ende des Dokuments.
2004	Das Ziel befindet sich vor dem Anfang des Dokuments.
2005	Es wurde ein ungültiger Elementtyp definiert. Für den zurückgegebenen Parameter wurde eine ungültige Adresse angegeben.

4.2.9.46 AuthenticRange.MoveEnd

Siehe auch

Methode: MoveEnd (*eKind* als SPYAuthenticElementKind, *nCount* als Long) als AuthenticRange[111]

Beschreibung
Verschiebt das Ende des Bereichs an den Anfang des *nCount* Elements vom Typ *eKind*. Die Zählung beginnt am aktuellen Beginn des Bereichsobjekts.

Mit Hilfe von positiven Zahlen für *nCount* können Sie die Position zum Ende des Dokuments hin verschieben, mit Hilfe von negativen Zahlen können Sie die Position zum Beginn des Dokuments hin verschieben. Das Ende des Bereichs wird nicht verschoben, es sei denn, der neue Anfang des Bereichs wäre größer als der Wert. In

© 2018-2024 Altova GmbH

diesem Fall wird das Ende an den neuen Anfang verschoben. Die Methode gibt das geänderte Bereichsobjekt zurück.

Fehler

2001	Das Authentic Range-Objekt oder das dazugehörige View-Objekt ist nicht mehr gültig.
2003	Das Ziel befindet sich hinter dem Ende des Dokuments.
2004	Das Ziel befindet sich vor dem Anfang des Dokuments.
2005	Es wurde ein ungültiger Elementtyp definiert. Für den zurückgegebenen Parameter wurde eine ungültige Adresse angegeben.

4.2.9.47 AuthenticRange.MoveRowDown

Siehe auch

Methode: MoveRowDown () als Boolean

Beschreibung
Wenn sich der Anfang eines Bereichs innerhalb einer dynamischen Tabelle befindet und eine Zeile ausgewählt wird, die nicht die letzte Zeile in dieser Tabelle ist, vertauscht diese Methode diese Zeile mit der Zeile unmittelbar danach. Die Auswahl des Bereichs verschiebt sich mit der Zeile, ändert sich aber sonst nicht. Die Funktion gibt den Wert *true* zurück, wenn das Verschieben erfolgreich war, ansonsten wird der Wert *false* zurückgegeben.

Fehler

2001	Das Authentic Range-Objekt oder das dazugehörige View-Objekt ist nicht mehr gültig.
2005	Für den zurückgegebenen Parameter wurde eine ungültige Adresse angegeben.

4.2.9.48 AuthenticRange.MoveRowUp

Siehe auch

Methode: MoveRowUp () als Boolean

Beschreibung
Wenn sich der Anfang eines Bereichs innerhalb einer dynamischen Tabelle befindet und eine Zeile ausgewählt wird, die nicht die erste Zeile in dieser Tabelle ist, vertauscht diese Methode diese Zeile mit der Zeile unmittelbar oberhalb davon. Die Auswahl des Bereichs verschiebt sich mit der Zeile, ändert sich aber sonst nicht. Die Funktion gibt den Wert *true* zurück, wenn das Verschieben erfolgreich war, ansonsten wird der Wert *false* zurückgegeben.

Fehler

2001	Das Authentic Range-Objekt oder das dazugehörige View-Objekt ist nicht mehr gültig.

2005 Für den zurückgegebenen Parameter wurde eine ungültige Adresse angegeben.

Beispiele

Siehe JScript - Bubble Sort Dynamic Tables [30].

4.2.9.49 AuthenticRange.Parent

Siehe auch

Eigenschaft: Parent als AuthenticView [150] (read-only)

Beschreibung

Ruft die Ansicht auf, die dieses Bereichsobjekt enthält.

Fehler

2001 Das Authentic Range-Objekt oder das dazugehörige View-Objekt ist nicht mehr gültig.

2005 Für den zurückgegebenen Parameter wurde eine ungültige Adresse angegeben.

4.2.9.50 AuthenticRange.Paste

Siehe auch

Methode: Paste () als Boolean

Beschreibung

Gibt *False* zurück, wenn der Kopier-/Einfügepuffer leer ist oder die aktuelle Auswahl nicht durch den Inhalt dieses Puffers ersetzt werden kann.

Andernfalls wird die aktuelle Auswahl gelöscht, der Inhalt dieses Puffers eingefügt und der Wert *true* zurückgegeben.

Fehler

2001 Das Authentic Range-Objekt oder das dazugehörige View-Objekt ist nicht mehr gültig.

2005 Für den zurückgegebenen Parameter wurde eine ungültige Adresse angegeben.

© 2018-2024 Altova GmbH

4.2.9.51 AuthenticRange.PerformAction

Siehe auch

Methode: PerformAction (*eAction* als SPYAuthenticActions, *strElementName* als String) als Boolean

Beschreibung
PerformAction und die dazugehörigen Methoden verleihen Zugriff auf die Eingabehilfenfunktionen von Authentic. Mit Hilfe dieser Funktion können Sie Dokumentinhalt schnell und auf konsistente Weise ändern, ohne wissen zu müssen, wo genau die Änderung vorgenommen wird. Der Anfang des Bereichsobjekts dient dazu, die nächste gültige Position zu finden, an der die angegebene Aktion durchgeführt werden kann. Wenn es keine solche Position gibt, gibt die Methode den Wert *false* zurück. Andernfalls wird das Dokument geändert und der Bereich zeigt auf den Anfang der Änderung.

HINWEIS: Um Elementnamen zu finden, die als der zweite Parameter übergeben werden können, verwenden Sie CanPerformActionWith 114.

Fehler

2001	Das Authentic Range-Objekt oder das dazugehörige View-Objekt ist nicht mehr gültig.
2005	Für den zurückgegebenen Parameter wurde eine ungültige Adresse angegeben.
2007	Es wurde eine ungültige Aktion definiert.

Beispiele

```
' ------------------------------------------
'             VBScript
' Insert the innermost element
' ------------------------------------------
Dim objRange
Set objRange = objPlugin.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
```

4.2.9.52 AuthenticRange.Select

Siehe auch

Methode: Select ()

Beschreibung
Macht diesen Bereich zur aktuellen Auswahl der Ansicht. Dasselbe Ergebnis erzielen Sie bei Verwendung von: '*objRange.Parent.Selection = objRange*'

Fehler

2001	Das Authentic Range-Objekt oder das dazugehörige View-Objekt ist nicht mehr gültig.

Beispiele

```
' ---------------------------------------
'                  VBScript
' ---------------------------------------

' set current selection to end of document
objPlugin.objAuthenticView.DocumentEnd.Select()
```

4.2.9.53 AuthenticRange.SelectNext

Siehe auch

Methode: SelectNext (*eKind* als SPYAuthenticElementKind) als AuthenticRange[111]

Beschreibung
Wählt das Element vom Typ *eKind* nach dem aktuellen Bereichsende aus. Die Methode gibt das geänderte Bereichsobjekt zurück.

Fehler

2001	Das Authentic Range-Objekt oder das dazugehörige View-Objekt ist nicht mehr gültig.
2003	Das Ziel befindet sich hinter dem Ende des Dokuments.
2005	Es wurde ein ungültiger Elementtyp definiert. Für den zurückgegebenen Parameter wurde eine ungültige Adresse angegeben.

Beispiele

```
' ----------------------------------------
'                  VBScript
' Scan through the whole document word-by-word
' ----------------------------------------
Dim objAuthenticView
Set objAuthenticView = objPlugin.AuthenticView

Dim objRange
Set objRange = objAuthenticView.DocumentBegin
```


© 2018-2024 Altova GmbH

```
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
```

4.2.9.54 AuthenticRange.SelectPrevious

Siehe auch

Methode: GotoPrevious (*eKind* als SPYAuthenticElementKind) als AuthenticRange[111]

Beschreibung
Wählt das Element vom Typ *eKind* vor dem Beginn es aktuelle Bereichs aus. Die Methode gibt das geänderte Bereichsobjekt zurück.

Fehler

2001	Das Authentic Range-Objekt oder das dazugehörige View-Objekt ist nicht mehr gültig.
2004	Das Ziel befindet vor dem Beginn des Dokuments.
2005	Es wurde ein ungültiger Elementtyp definiert. Für den zurückgegebenen Parameter wurde eine ungültige Adresse angegeben.

Beispiele

```
' ----------------------------------------
'               VBScript
' Scan through the whole document tag-by-tag
' ----------------------------------------
Dim objAuthenticView
Set objAuthenticView = objPlugin.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
```

4.2.9.55 AuthenticRange.SetElementAttributeValue

Siehe auch

Methode: SetElementAttributeValue (*strElementName* als String, *strAttributeName* als String, *strAttributeValue* als String)

Beschreibung
Liefert den Wert des in *strAttributeName* Attributs für das mit *strElementName* bezeichnete Element. Wenn das Attribut unterstützt wird, aber keinen zugewiesenen Wert hat, wird ein leerer String zurückgegeben. Um die Namen der von einem Element unterstützten Attribute zu ermitteln, verwenden Sie GetElementAttributeNames 122 oder HasElementAttribute 127.

Fehler

2001	Das Authentic Range-Objekt oder das dazugehörige View-Objekt ist nicht mehr gültig.
2005	Es wurde ein ungültiger Elementname angegeben. Es wurde ein ungültiger Attributname angegeben. Es wurde ein ungültiger Attributwert angegeben.

Beispiele

```
' ----------------------------------------
'            VBScript
' Get and set element attributes
' ----------------------------------------
Dim objRange
Set objRange = objPlugin.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
```

 © 2018-2024 Altova GmbH

```
                    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
```

4.2.9.56 AuthenticRange.SetFromRange

Siehe auch

Methode: SetFromRange (*objSrcRange* als AuthenticRange[111])

Beschreibung
Setzt das Bereichsobjekt auf dieselbe Anfangs- und Endposition wie *objSrcRange*.

Fehler

2001	Eines der beiden Bereichsobjekte ist ungültig.
2005	Es wurde kein Objekt als Quellobjekt definiert.

4.2.9.57 AuthenticRange.SetVariableValue

Methode: `SetVariableValue(name als string, value als string)`

Rückgabewert
Definiert den Wert der benannten Variablen.

Fehler

2201	Ungültiges Objekt.
2202	Kein Kontext.
2204	Es befindet sich keine Variable dieses Namens im Geltungsbereich.
2205	Die Variable kann nicht ausgewertet werden.
2206	Die Variable gibt eine Sequenz zurück.
2207	Die Variable ist schreibgeschützt.
2208	Es ist keine Änderung zulässig.

4.2.9.58 AuthenticRange.Text

Siehe auch

Eigenschaft: Text als String

Beschreibung
Setzt oder holt den vom Bereichsobjekt ausgewählten Textinhalt.

Die Anzahl der abgerufenen Zeichen muss nicht unbedingt übereinstimmen, da es zwischen dem Anfang und dem Ende des ausgewählten Bereichs Textcursor-Positionen gibt. Die meisten Dokumentelemente unterstützen eine Bereichsendeposition des Cursors, die nicht mit der Bereichsanfangsposition des folgenden Elements übereinstimmt. Dropdown-Listen verfügen nur über eine Cursorposition, können aber Strings beliebiger Länge auswählen. Bei Optionsfeldern und Kontrollkästchen enthält der Texteigenschaftswert den String des entsprechenden XML-Elements.

Wenn mit dem Bereich mehr als ein Element auswählt wird, besteht der Text aus einer Verkettung der Einzeltexte. XML Entities werden erweitert, so dass statt '&' '&' erscheint.

Wenn der Text auf den leeren String gesetzt wird, wird kein XML-Element gelöscht. Verwenden Sie stattdessen Cut[116], Delete[117] oder PerformAction[139].

Fehler

2001	Das Authentic Range-Objekt oder das dazugehörige View-Objekt ist nicht mehr gültig.
2005	Für den zurückgegebenen Parameter wurde eine ungültige Adresse angegeben.

4.2.10 AuthenticSelection

Siehe auch

Eigenschaften
Start[145]
StartTextPosition[145]
End[144]
EndTextPosition[145]

4.2.10.1 AuthenticSelection.End

Siehe auch

Deklaration: End als XMLData[180]

Beschreibung
XML-Element, an dem die aktuelle Auswahl endet.

© 2018-2024 Altova GmbH

4.2.10.2 AuthenticSelection.EndTextPosition

Siehe auch

Deklaration: EndTextPosition als long

Beschreibung
Position in Authentic.End[144], an der die Auswahl endet.

4.2.10.3 AuthenticSelection.Start

Siehe auch

Deklaration: Start als XMLData[180]

Beschreibung
XML-Element, an dem die aktuelle Auswahl beginnt.

4.2.10.4 AuthenticSelection.StartTextPosition

Siehe auch

Deklaration: StartTextPosition als long

Beschreibung
Position in Authentic.Start[145], an der die Auswahl beginnt.

4.2.11 AuthenticToolbarButton

Methoden

Eigenschaften
CommandID[145]

4.2.11.1 AuthenticToolbarButton.CommandID

Deklaration: CommandID als SPYAuthenticCommand[191]

Beschreibung
Die CommandId der Symbolleisten-Schaltfläche.

4.2.12 AuthenticToolbarButtons

Methoden
Item 146
NewButton 146
NewCustomButton 147
NewSeparator 148
Remove 148

Eigenschaften
Count 146

4.2.12.1 AuthenticToolbarButtons.Count

Deklaration: Count als long

Beschreibung
Holt die tatsächliche Anzahl der Schaltflächen der Symbolleiste.
Schreibgeschützt.

4.2.12.2 AuthenticToolbarButtons.Item

Deklaration: Item(n als long) als AuthenticToolbarButton 145

Beschreibung
Holt die nte Schaltfläche der aktuellen Symbolleiste. n beginnt mit 1.

Beispiel
Siehe Beispiel unter AuthenticToolbarButtons.NewButton 146

4.2.12.3 AuthenticToolbarButtons.NewButton

Deklaration: NewButton(nPosition als long, nCommandId als SPYAuthenticCommand 191)

Beschreibung
Fügt eine neue Schaltfläche für den Befehl "nCommandId" an der Position "nPosition" der Symbolleiste ein. nPosition beginnt mit 1.

Beispiel
Fügt eine neue Symbolleiste hinzu, ordnet sie am unteren Rand an und fügt neue Symbolleisten-Schaltflächen hinzu.

```
objPlugIn.ToolbarRows.NewRow(3)                    // add a new Toolbar (Row 3)
```


© 2018-2024 Altova GmbH

```
set ToolbarRow  = objPlugIn.ToolbarRows.Item(3)
set Buttons = ToolbarRow.Buttons
ToolbarRow.Alignment    = 2          // align Toolbar to bottom
Buttons.NewButton       1, 2         // add Print Button
Buttons.NewButton       1, 3         // add PrintPreview Button
Buttons.NewSeparator    2            // add Separator
Buttons.NewButton       1, 4         // add Validate Button
```

Bei Aufruf von StartEditing oder ReloadToolbars werden die geänderten Symbolleisteneinstellungen verwendet.

4.2.12.4 AuthenticToolbarButtons.NewCustomButton

Deklaration: NewCustomButton(nPosition als long, strName als String, strTooltip als String, strBitmapURL als String)

Beschreibung
Fügt in der Symbolleiste an der Position nPosition eine neue benutzerdefinierte Schaltfläche mit dem Namen strName ein. nPosition beginnt bei 1.
strTooltip wird als Text für den Tooltipp verwendet.

strBitmapURL ist der Pfad zur Bitmap-Datei für die neue Schaltfläche. Dieser Pfad ist relativ zum Pfad, der mit der TextStateBmpURL[94]-Eigenschaft definiert wurde.

Beispiel

Angenommen, Sie haben eine benutzerdefinierte Schaltfläche mit dem Namen "MyFunction" in Ihre Symbolleisten eingefügt. Im folgenden Event Handler für doceditcommand[50] sehen Sie, was geschehen muss, wenn der Benutzer auf Ihre Schaltfläche geklickt hat und wie Sie den ein/aus-Status dafür festlegen.

```
<SCRIPT LANGUAGE=javascript FOR=objPlugIn EVENT=doceditcommand>
	// event.type is set to "command" if the user clicked the button
	if(objPlugIn.event.type == "command")
	{
		if(objPlugIn.event.srcElement.Name == "MyFunction")
			window.alert("You pressed my custom button!");
	}

	// event.type is set to "update" if the button state must be set
	if(objPlugIn.event.type == "update")
	{
		if(objPlugIn.event.srcElement.Name == "MyFunction")
		{
			// we enable the button if only one element is selected

if(objPlugIn.CurrentSelection.Start.IsSameNode(objPlugIn.CurrentSelection.End))
				objPlugIn.event.returnValue = 1;
			else
				objPlugIn.event.returnValue = 0;

			objPlugIn.event.cancelBubble = true;
```

```
            }
      }
</SCRIPT>
```

4.2.12.5 AuthenticToolbarButtons.NewSeparator

Deklaration: NewSeparator(nPosition als long)

Beschreibung
Fügt ein Trennzeichen an der Position nPosition der Symbolleiste ein. nPosition beginnt mit 1.

Beispiel
Siehe Beispiel unter AuthenticToolbarButtons.NewButton 146

4.2.12.6 AuthenticToolbarButtons.Remove

Deklaration: Remove(nPosition als long)

Beschreibung
Entfernt die Schaltfläche oder das Trennzeichen an Position nPosition der Symbolleiste. nPosition beginnt bei 1.

4.2.13 AuthenticToolbarRow

Methoden
Alignment 148

Eigenschaften
Buttons 149

4.2.13.1 AuthenticToolbarRowAlignment

Deklaration: Alignment(nAlign als SPYAuthenticToolbarAllignment 195)

Beschreibung
Liefert oder setzt die Anordnung der Symbolleiste im Plug-In.

Beispiel
Anordnen aller Symbolleisten am unteren Rand:

```
for each ToolbarRow in objPlugin.ToolbarRows
      ToolbarRow.Alignment = 2
next
```


© 2018-2024 Altova GmbH

4.2.13.2 AuthenticToolbarRowButtons

Deklaration: Buttons als AuthenticToolbarButtons 146

Beschreibung
Liefert alle Schaltflächen der Symbolleiste

Beispiel
Siehe Beispiel unter AuthenticToolbarButtons.NewButton 146

4.2.14 AuthenticToolbarRows

Methoden
Item 149
RemoveRow 150
NewRow 150

Eigenschaften
Count 149

4.2.14.1 AuthenticToolbarRows.Count

Deklaration: Count als long

Beschreibung
Liefert die aktuelle Anzahl der definierten Symbolleisten.
Schreibgeschützt.

4.2.14.2 AuthenticToolbarRows.Item

Deklaration: Item(nPosition als long) als AuthenticToolbarRow 148

Beschreibung
Liefert die Symbolleiste in Position nPosition. nPosition beginnt mit "1".
Schreibgeschützt.

Beispiel
Siehe Beispiel unter AuthenticToolbarButtons.NewButton 146

4.2.14.3 AuthenticToolbarRows.RemoveRow

Deklaration: RemoveRow(nPosition als long)

Beschreibung
Entfernt die Symbolleiste an Position nPosition von der Benutzeroberfläche. nPosition beginnt mit 1.

4.2.14.4 AuthenticToolbarRows.NewRow

Deklaration: NewRow(nPosition als long)

Beschreibung
Fügt eine neue Symbolleistenzeile ein. nPosition beginnt bei 1.

Beispiel
Siehe Beispiel unter AuthenticToolbarButtons.NewButton 146

4.2.15 AuthenticView

Siehe auch

Eigenschaften	Methoden	Ereignisse
Application 162	Goto 167	
DocumentBegin 164	Print 169	
DocumentEnd 164	Redo 169	
MarkupVisibility 168	Undo 170	
Parent 168	UpdateXMLInstanceEntities 171	
Selection 169		
WholeDocument 171		

Beschreibung
AuthenticView und sein Child-Objekt AuthenticRange 111 stellen die Schnittstelle für die Authentic-Ansicht bereit, über die Sie Dokument-Inhalte einfach und auf konsistente Art bearbeiten können. Die Funktionsüberschneidung mit bereits bestehenden Methoden und Eigenschaften im Authentic-Objekt ist beabsichtigt. Die Funktionen des Authentic-Objekts zur Bearbeitung von Inhalt werden dadurch nicht mehr benötigt. Es wird empfohlen, unbedingt die neue AuthenticView-Schnittstelle zu verwenden.

AuthenticView verschafft Ihnen einfachen Zugriff auf bestimmte Funktionen wie z.B. Drucken, mehrfaches Rückgängigmachen und die aktuelle Cursor-Auswahl oder -Position.

AuthenticView verwendet Objekte vom Typ AuthenticRange 111, um das einfache Navigieren im Dokument und die problemlose Auswahl logischer Textelemente zu ermöglichen. Verwenden Sie für einfache Auswahlvorgänge die Eigenschaften DocumentBegin 164, DocumentEnd 164 oder WholeDocument 171 und für komplexere Auswahlvorgänge die Goto 167-Methode. Informationen zum Navigieren relativ zu einem

© 2018-2024 Altova GmbH

vorgegebenen Dokumentenbereich finden Sie unter den Methoden und Eigenschaften des AuthenticRange[111]-Objekts.

4.2.15.1 Events

4.2.15.1.1 OnBeforeCopy

Siehe auch

Event: `OnBeforeCopy()` als Boolean

XMLSpy Skripting-Umgebung - VBScript:

```
Function On_AuthenticBeforeCopy()
     ' On_AuthenticBeforeCopy = False  ' to disable operation
End Function
```

XMLSpy Skripting-Umgebung - JScript:

```
function On_AuthenticBeforeCopy()
{
     // return false;  /* to disable operation */
}
```

XMLSpy IDE Plugin:

```
IXMLSpyPlugIn.OnEvent (21, ...)  //  nEventId = 21
```

Beschreibung
Dieses Event wird ausgelöst, bevor eine Kopieroperation an einem Dokument durchgeführt wird. Bei Rückgabe von *True* (oder nichts) wird die Kopieroperation gestattet, bei Rückgabe von *False* wird das Kopieren deaktiviert.

4.2.15.1.2 OnBeforeCut

Siehe auch

Event: `OnBeforeCut()` als Boolean

XMLSpy Skripting-Umgebung - VBScript:

```
Function On_AuthenticBeforeCut()
     ' On_AuthenticBeforeCut = False  ' to disable operation
End Function
```

XMLSpy Skripting-Umgebung - JScript:

```
function On_AuthenticBeforeCut()
{
     // return false;  /* to disable operation */
}
```

XMLSpy IDE Plugin:

```
IXMLSpyPlugIn.OnEvent (20, ...)  //  nEventId = 20
```

Beschreibung

Dieses Event wird ausgelöst, bevor eine Ausschneideoperation an einem Dokument durchgeführt wird. Geben Sie *True* (oder nichts) zurück, um den Vorgang zu gestatten und *False* um den Vorgang zu deaktivieren.

4.2.15.1.3 OnBeforeDelete

Siehe auch

Event: `OnBeforeDelete()` als Boolean

XMLSpy Skripting-Umgebung - VBScript:

```
Function On_AuthenticBeforeDelete()
    ' On_AuthenticBeforeDelete = False  ' to disable operation
End Function
```

XMLSpy Skripting-Umgebung - JScript:

```
function On_AuthenticBeforeDelete()
{
    // return false;  /* to disable operation */
}
```

XMLSpy IDE Plugin:

```
IXMLSpyPlugIn.OnEvent (22, ...)  //  nEventId = 22
```

Beschreibung

Dieses Event wird ausgelöst, bevor eine Löschoperation an einem Dokument durchgeführt wird. Geben Sie *True* (oder nichts) zurück, um den Vorgang zu gestatten und *False* um den Vorgang zu deaktivieren.

4.2.15.1.4 OnBeforeDrop

Siehe auch

Event: `OnBeforeDrop` (*`i_nXPos`* als Long, *`i_nYPos`* als Long, *`i_ipRange`* als `AuthenticRange`, *`i_ipData`* als `cancelBoolean`

XMLSpy Skripting-Umgebung - VBScript:

```
Function On_AuthenticBeforeDrop(nXPos, nYPos, objRange, objData)
    ' On_AuthenticBeforeDrop = False  ' to disable operation
End Function
```

XMLSpy Skripting-Umgebung - JScript:

```
function On_AuthenticBeforeDrop(nXPos, nYPos, objRange, objData)
{
```


© 2018-2024 Altova GmbH

```
    // return false;  /* to disable operation */
}
```

XMLSpy IDE Plugin:

```
IXMLSpyPlugIn.OnEvent (11, ...)  //  nEventId = 11
```

Beschreibung

Dieses Event wird ausgelöst, nachdem ein Objekt mit der Maus in das Applikationsfenster gezogen wurde. Alle mit dem Event in Zusammehang stehenden Informationen werden als Parameter übergeben.

Die ersten beiden Parameter definieren die Mausposition zum Zeitpunkt des Auftretens des Event. Der Parameter *objRange* übergibt ein Bereichsobjekt, das das XML-Element unterhalb der Mausposition auswählt. Der Wert dieses Parameters kann *NULL* sein. Überprüfen Sie dies, bevor Sie das Bereichsobjekt aufrufen. Mit Hilfe des Parameters *objData* können Sie Informationen über das gezogene Objekt aufrufen.

Geben Sie *False* zurück, um den Drop-Vorgang abzubrechen. Geben Sie *True* (oder nichts) zurück, um mit einem normalen Vorgang fortzusetzen.

Beispiele

```
' ------------------------------------------------------------------------
' VB code snippet - connecting to object level events
' ------------------------------------------------------------------------
' access XMLSpy (without checking for any errors)
Dim objSpy As XMLSpyLib.Application
Set objSpy = GetObject("", "XMLSpy.Application")

' this is the event callback routine connected to the OnBeforeDrop
' event of object objView
Private Function objView_OnBeforeDrop(ByVal i_nXPos As Long, ByVal i_nYPos As Long,
                                      ByVal i_ipRange As IAuthenticRange,
                                      ByVal i_ipData As IAuthenticDataTransfer) As
Boolean

    If (Not i_ipRange Is Nothing) Then
        MsgBox ("Dropping on content is prohibited");
        Return False;
    Else
        Return True;
    End If
End Function

' use VBA keyword WithEvents to connect to object-level event
Dim WithEvents objView As XMLSpyLib.AuthenticView
Set objView = objSpy.ActiveDocument.AuthenticView

' continue here with something useful ...
' and serve the windows message loop
```

4.2.15.1.5 OnBeforePaste

Siehe auch

Event: `OnBeforePaste` (*`objData`* als Variant, *`strType`* als String) als Boolean

XMLSpy Skripting-Umgebung - VBScript:

```
Function On_AuthenticBeforePaste(objData, strType)
    ' On_AuthenticBeforePaste = False  ' to disable operation
End Function
```

XMLSpy Skripting-Umgebung - JScript:

```
function On_AuthenticBeforePaste(objData, strType)
{
    // return false;  /* to disable operation */
}
```

XMLSpy IDE Plugin:

```
IXMLSpyPlugIn.OnEvent (19, ...)    // nEventId = 19
```

Beschreibung

Dieses Event wird ausgelöst, bevor eine Einfügeoperation am Dokument ausgeführt wird. Der Parameter *`strType`* ist entweder "TEXT", "UNICODETEXT" oder "IUNKNOWN". In den ersten beiden Fällen enthält *`objData`* eine String-Darstellung des Objekts, das eingefügt wird. Im letzteren Fall enthält *`objData`* einen Pointer auf eine IUnknown COM-Schnittstelle.

Geben Sie *True* (oder nichts) zurück, um die Einfügeoperation zu gestatten. Geben Sie *False* zurück, um die Operation zu deaktivieren.

4.2.15.1.6 OnBeforeSave

Event: `OnBeforeSave` (SaveAs flag) als Boolean

Beschreibung: Mit `OnBeforeSave` haben Sie die Möglichkeit den Benutzer vor einem Überschreiben des bestehenden Dokuments zu warnen oder das Dokument mit einem Schreibschutz zu versehen, wenn bestimmte Bedingungen nicht erfüllt werden. Das Event wird ausgelöst, bevor das Dialogfeld "Datei" angezeigt wird. (Beachten Sie bitte, dass das Event beim Speichern des Dokuments und nicht beim Speichern des SPS-Designs in StyleVision ausgelöst wird.)

4.2.15.1.7 OnDragOver

Siehe auch

© 2018-2024 Altova GmbH

Event: OnDragOver (*nXPos* als Long, *nYPos* als Long, *eMouseEvent* als SPYMouseEvent, *objRange* als AuthenticRange, *objData* als AuthenticDataTransfer) als Boolean

XMLSpy Skripting-Umgebung - VBScript:

```
Function On_AuthenticDragOver(nXPos, nYPos, eMouseEvent, objRange, objData)
    ' On_AuthenticDragOver = False  ' to disable operation
End Function
```

XMLSpy Skripting-Umgebung - JScript:

```
function On_AuthenticDragOver(nXPos, nYPos, eMouseEvent, objRange, objData)
{
    // return false;  /* to disable operation */
}
```

XMLSpy IDE Plugin:

```
IXMLSpyPlugIn.OnEvent (10, ...)  // nEventId = 10
```

Beschreibung

Dieses Event wird ausgelöst, wenn ein Objekt, von innerhalb oder außerhalb der Authentic-Ansicht mit der Maus über das Applikationsfenster gezogen wird. Alle mit dem Event in Zusammenhang stehenden Informationen werden als Parameter übergeben.

Die ersten drei Parameter definieren die Mausposition, den Status der Maustaste und den Status virtueller Schaltflächen zum Zeitpunkt, zu dem das Event auftritt. Der Parameter *objRange* übergibt ein Bereichsobjekt, das das XML-Element unterhalb der Mausposition auswählt. Der Wert dieses Parameters kann *NULL* sein. Überprüfen Sie dies, bevor Sie das Bereichsobjekt aufrufen. Mit Hilfe des Parameters *objData* können Sie Informationen über das gezogene Objekt aufrufen.

Geben Sie *False* zurück, um den Drag-Vorgang abzubrechen. Geben Sie *True* (oder nichts) zurück, um mit einem normalen Vorgang fortzusetzen.

Beispiele

```
' ------------------------------------------------------------------------
' VB code snippet - connecting to object level events
' ------------------------------------------------------------------------
' access XMLSpy (without checking for any errors)
Dim objSpy As XMLSpyLib.Application
Set objSpy = GetObject("", "XMLSpy.Application")

' this is the event callback routine connected to the OnDragOver
' event of object objView
Private Function objView_OnDragOver(ByVal i_nXPos As Long, ByVal i_nYPos As Long,
                                    ByVal i_eMouseEvent As SPYMouseEvent,
                                    ByVal i_ipRange As IAuthenticRange,
                                    ByVal i_ipData As IAuthenticDataTransfer) As Boolean

    If (((i_eMouseEvent And spyShiftKeyDownMask) <> 0) And
              (Not i_ipRange Is Nothing)) Then
        MsgBox ("Floating over element " & i_ipRange.FirstXMLData.Parent.Name);
    End If

    Return True;
End Function
```

```
' use VBA keyword WithEvents to connect to object-level event
Dim WithEvents objView As XMLSpyLib.AuthenticView
Set objView = objSpy.ActiveDocument.AuthenticView

' continue here with something useful ...
' and serve the windows message loop
```

4.2.15.1.8 OnKeyboardEvent

Siehe auch

Event: `OnKeyboardEvent` (*eKeyEvent* als SPYKeyEvent, *nKeyCode* als Long, *nVirtualKeyStatus* als Long) als Boolean

XMLSpy Skripting-Umgebung - VBScript:

```
Function On_AuthenticKeyboardEvent(eKeyEvent, nKeyCode, nVirtualKeyStatus)
    ' On_AuthenticKeyboardEvent = True  ' to cancel bubbling of event
End Function
```

XMLSpy Skripting-Umgebung - JScript:

```
function On_AuthenticKeyboardEvent(eKeyEvent, nKeyCode, nVirtualKeyStatus)
{
    // return true;  /* to cancel bubbling of event */
}
```

XMLSpy IDE Plugin:

```
IXMLSpyPlugIn.OnEvent (30, ...)  //  nEventId = 30
```

Beschreibung

Dieses Event wird für *WM_KEYDOWN*, *WM_KEYUP* und *WM_CHAR* Windows-Meldungen ausgelöst.

Der eigentliche Meldungstyp steht im *eKeyEvent* Parameter zur Verfügung. Der Status virtueller Tasten ist im Parameter *nVirtualKeyStatus* kombiniert. Verwenden Sie die im Enumerations-Datentyp `SPYVirtualKeyMask` definierten Bit-Masken, um eine Überprüfung der unterschiedlichen Tasten und ihrer Kombinationen durchzuführen.

ANMERKUNG: Die folgenden Events aus der Skripting-Umgebung und dem IDE Plug-in von XMLSpy werden weiterhin unterstützt, werden aber bei diesem Event nicht mehr unterstützt:

```
On_AuthenticKeyUp()          IXMLSpyPlugIn.OnEvent (13, ...)    //  nEventId = 13
On_AuthenticKeyDown()        IXMLSpyPlugIn.OnEvent (12, ...)    //  nEventId = 12
On_AuthenticKeyPressed()     IXMLSpyPlugIn.OnEvent (14, ...)    //  nEventId = 14
```

Beispiele

```
' ----------------------------------------------------------------------
' VB code snippet - connecting to object level events
' ----------------------------------------------------------------------
' access XMLSpy (without checking for any errors)
Dim objSpy As XMLSpyLib.Application
Set objSpy = GetObject("", "XMLSpy.Application")
```


© 2018-2024 Altova GmbH

```
' this is the event callback routine connected to the OnKeyboard
' event of object objView
Private Function objView_OnKeyboardEvent(ByVal i_keyEvent As Long, ByVal io_pnKeyCode As
Long, ByVal i_nVirtualKeyStatus As Long) As Boolean
    If ((i_keyEvent = XMLSpyLib.spyKeyUp) And ((i_nVirtualKeyStatus And
XMLSpyLib.spyCtrlKeyMask) <> 0)) Then
        MsgBox ("Ctrl " & io_pnKeyCode & " pressed")
        objView_OnKeyboardEvent = True
    Else
        objView_OnKeyboardEvent = False
    End If
End Function

' use VBA keyword WithEvents to connect to object-level event
Dim WithEvents objView As XMLSpyLib.AuthenticView
Set objView = objSpy.ActiveDocument.AuthenticView

' continue here with something useful ...
' and serve the windows message loop
```

4.2.15.1.9 OnLoad

Event: `OnLoad` ()

Beschreibung: Mit `OnLoad` können einige Funktionalitäten der Authentic-Ansicht eingeschränkt werden, wie im Beispiel unten gezeigt:

```
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 );
}
```

Im Beispiel wird der Status der Symbolleisten-Schaltflächen "Kleine Markup-Symbole", "Große Markup-Symbole" und "Gemischte Markup-Symbole" mit Hilfe der Schaltflächen-Identifier bearbeitet. Siehe vollständige Liste 160.

4.2.15.1.10 OnMouseEvent

Siehe auch

Event: `OnMouseEvent` (*nXPos* als Long, *nYPos* als Long, *eMouseEvent* als `SPYMouseEvent`, *objRange* als `AuthenticRange`) als Boolean

XMLSpy Skripting-Umgebung - VBScript:

```
Function On_AuthenticMouseEvent(nXPos, nYPos, eMouseEvent, objRange)
    ' On_AuthenticMouseEvent = True  ' to cancel bubbling of event
End Function
```

XMLSpy Skripting-Umgebung - JScript:

```
function On_AuthenticMouseEvent(nXPos, nYPos, eMouseEvent, objRange)
{
    // return true;  /* to cancel bubbling of event */
}
```

XMLSpy IDE Plugin:

```
IXMLSpyPlugIn.OnEvent (31, ...)  // nEventId = 31
```

Beschreibung

Dieses Event wird für jede Mausbewegung und jede Mausschaltflächen-Windows-Meldung ausgelöst.

Der eigentliche Meldungstyp und der Status der Maustaste stehen im *eMouseEvent* Parameter zur Verfügung. Verwenden Sie die im Enumerations-Datentyp `SPYMouseEvent` definierten Bit-Masken, um eine Überprüfung auf die unterschiedlichen Meldungen, Tastenstatus und ihre Kombinationen durchzuführen.

Der Parameter *objRange* definiert den Teil des Dokuments, der sich an der aktuellen Mauscursorposition befindet. Das Bereichsobjekt wählt immer einen vollständigen Tag des Dokuments aus. (Dies kann sich in zukünftigen Versionen ändern, wenn ein genauerer Positionierungsmechanismus zur Verfügung steht). Wenn sich an der aktuellen Position kein auswählbarer Teil des Dokuments befindet, ist das Bereichsobjekt *Null*.

ANMERKUNG: Die folgenden Events aus der Skripting-Umgebung und dem IDE Plug-in von XMLSpy werden weiterhin unterstützt, werden aber bei diesem Event nicht mehr unterstützt:

```
On_AuthenticKeyUp()          IXMLSpyPlugIn.OnEvent (13, ...)   // nEventId = 13
On_AuthenticKeyDown()        IXMLSpyPlugIn.OnEvent (12, ...)   // nEventId = 12
On_AuthenticKeyPressed()     IXMLSpyPlugIn.OnEvent (14, ...)   // nEventId = 14
```

Beispiele

```
' ---------------------------------------------------------------------
' VB code snippet - connecting to object level events
' ---------------------------------------------------------------------
' access XMLSpy (without checking for any errors)
Dim objSpy As XMLSpyLib.Application
Set objSpy = GetObject("", "XMLSpy.Application")

' this is the event callback routine connected to the OnMouseEvent
' event of object objView. If you click with the left mouse button
' while pressing a control key, the current selection will be set
```


© 2018-2024 Altova GmbH

```
' to the tag below the current mouse cursor position
Private Function objView_OnMouseEvent(ByVal i_nXPos As Long, ByVal i_nYPos As Long, ByVal
i_eMouseEvent As XMLSpyLib.SPYMouseEvent, ByVal i_pRange As XMLSpyLib.IAuthenticRange) As
 Boolean
    If (i_eMouseEvent = (XMLSpyLib.spyLeftButtonDownMask Or
XMLSpyLib.spyCtrlKeyDownMask)) Then
        On Error Resume Next
        i_pRange.Select
        objView_OnMouseEvent = True
    Else
        objView_OnMouseEvent = False
    End If
End Function

' use VBA keyword WithEvents to connect to object-level event
Dim WithEvents objView As XMLSpyLib.AuthenticView
Set objView = objSpy.ActiveDocument.AuthenticView

' continue here with something useful ...
' and serve the windows message loop
```

4.2.15.1.11 OnSelectionChanged

Event: `OnSelectionChanged` (*`objNewSelection`* als `AuthenticRange`)

XMLSpy Skripting-Umgebung - VBScript:

```
Function On_AuthenticSelectionChanged (objNewSelection)
End Function
```

XMLSpy Skripting-Umgebung - JScript:

```
function On_AuthenticSelectionChanged (objNewSelection)
{
}
```

XMLSpy IDE Plugin:

```
IXMLSpyPlugIn.OnEvent (23, ...)  // nEventId = 23
```

Beschreibung
Dieses Event wird ausgelöst, wenn die sich die Auswahl auf der Benutzeroberfläche ändert.

Beispiele

```
' ----------------------------------------------------------------------
' VB code snippet - connecting to object level events
' ----------------------------------------------------------------------
' access XMLSpy (without checking for any errors)
Dim objSpy As XMLSpyLib.Application
Set objSpy = GetObject("", "XMLSpy.Application")

' this is the event callback routine connected to the OnSelectionChanged
' event of object objView
```

```
Private Sub objView_OnSelectionChanged (ByVal i_ipNewRange As XMLSpyLib.IAuthenticRange)
    MsgBox ("new selection: " & i_ipNewRange.Text)
End Sub

' use VBA keyword WithEvents to connect to object-level event
Dim WithEvents objView As XMLSpyLib.AuthenticView
Set objView = objSpy.ActiveDocument.AuthenticView

' continue here with something useful ...
' and serve the windows message loop
```

4.2.15.1.12 OnToolbarButtonClicked

Event: `OnToolbarButtonClicked` (Button identifier)

Beschreibung: `OnToolbarButtonClicked` wird ausgelöst, wenn der Benutzer auf eine Symbolleisten-Schaltfläche klickt. Über den Parameter Schaltflächen-Identifier, kann ermittelt werden, auf welche Schaltfläche geklickt wurde. Unten sehen Sie die Liste der vordefinierten Schaltflächen-Identifier:

- `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`

© 2018-2024 Altova GmbH

- AuthenticTableCenterCellVertically
- AuthenticTableAlignCellContentBottom
- AuthenticTableAlignCellContentLeft
- AuthenticTableCenterCellContent
- AuthenticTableAlignCellContentRight
- AuthenticTableJustifyCellContent
- AuthenticTableInsertTable
- AuthenticTableDeleteTable
- AuthenticTableProperties
- AuthenticAppendRow
- AuthenticInsertRow
- AuthenticDuplicateRow
- AuthenticMoveRowUp
- AuthenticMoveRowDown
- AuthenticDeleteRow
- AuthenticDefineEntities

- AuthenticXMLSignature

Der Benutzer kann für seine benutzerdefinierten Schaltflächen seine eigenen Identifier hinzufügen. Beachten Sie bitte, dass der Benutzer aufpassen muss, da die Identifier nicht auf Eindeutigkeit überprüft werden. Dieselben Identifier können zur Kennzeichnung von Schaltflächen in `Set/GetToolbarState()` COM API-Aufrufen verwendet werden. Durch Hinzufügen von Code für unterschiedliche Schaltflächen hat der Benutzer die Möglichkeit, das Verhalten der Symbolleiste der Authentic-Ansicht vollständig neu zu definieren, indem er seine eigenen Methoden zur Tabellenbearbeitung usw. hinzufügt.

4.2.15.1.13 OnToolbarButtonExecuted

Event: `OnToolbarButtonExecuted` (Button identifier)

Beschreibung: `OnToolbarButtonClicked` wird ausgelöst, wenn ein Benutzer auf eine Symbolleisten-Schaltfläche klickt. Über den Parameter Schaltflächen-Identifier, kann ermittelt werden, auf welche Schaltfläche geklickt wurde. Siehe Liste der vordefinierten vordefinierten Schaltflächen-Identifier 160.

`OnToolbarButtonExecuted` wird ausgelöst, nachdem die Symbolleisten-Aktion ausgeführt wurde und eignet sich z.B. zum Hinzufügen von aktualisiertem Code, wie im Beispiel unten gezeigt:

```
//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 diesem Fall ist `UpdateOwnToolbarButtonStates` eine in den globalen Deklarationen definierte Benutzerfunktion.

4.2.15.1.14 OnUserAddedXMLNode

Event: `OnUserAddedXMLNode` (XML node)

Beschreibung: `OnUserAddedXMLNode` wird ausgelöst, wenn der Benutzer einen XML-Node als Primäraktion hinzufügt. Dies geschieht in Situationen, in denen der Benutzer eine der folgenden Aktionen ausführt:

- Klicken auf automatisch hinzugefügte Hyperlinks (siehe Beispiel OnUserAddedXMLNode.sps)
- Auswahl der Kontextmenübefehle "Einfügen...", "Einfügen nach...", "Einfügen vor..."
- Klicken auf "Zeile anhängen", beim Einfügen neuer Symbolleisten-Schaltflächen
- Ausführen von "Einfügen nach...", "Einfügen vor..." Aktionen in Elementeingabehilfen (außerhalb von StyleVision)

Das Event wird **nicht** bei "Zeile duplizieren" oder, wenn der Node extern (z.B. über die COM API) hinzugefügt wurde, oder bei "Anwenden" (z.B. Textstatus-Symbole) oder in XML-Tabellenoperationen bzw. in Datenbankoperationen ausgelöst.

Der Event-Parameter ist das XML-Node-Objekt, das hinzugefügt wurde, wodurch der Benutzer die Möglichkeit hat, den hinzugefügten XML-Node zu bearbeiten. Ein ausführliches Beispiel für einen Event Handler finden Sie in der Datei `OnUserAddedXMLNode.sps`.

4.2.15.2 AuthenticView.Application

Siehe auch

Eigenschaft: Application als Authentic [69] (Schreibgeschützt)

Beschreibung
Ruft das -Applikationsobjekt auf.

Fehler

2000	Das Authentic View-Objekt ist nicht mehr gültig.
2005	Für den zurückgegebenen Parameter wurde eine ungültige Adresse angegeben.

4.2.15.3 AuthenticView.AsXMLString

Siehe auch

Eigenschaft: `AsXMLString` als String

Beschreibung
Gibt den Dokumentinhalt als XML-String zurück oder definiert ihn als solchen. Wenn der Inhalt auf einen neuen Wert gesetzt wird, wird die gerade verwendete Schema- oder sps-Datei nicht geändert. Wenn der neue `XMLString` der aktuellen Schemadatei nicht entspricht, wird Fehler 2011 zurückgegeben.

© 2018-2024 Altova GmbH

Fehler

2000 Das Authentic-Ansichtsobjekt ist nicht mehr gültig.
2011 `AsXMLString` wurde auf einen Wert gesetzt, der gemäß der aktuellen Schemadatei kein gültiger XML-Code ist.

4.2.15.4 AuthenticView.ContextMenu

Eigenschaft: `ContextMenu` als `ContextMenu`

Beschreibung
Über die Eigenschaft `ContextMenu` erhält der Benutzer Zugriff auf das Kontextmenü. Am besten lässt sich dies über den Event Handler `OnContextMenuActivated` durchführen.

Fehler

2000 Ungültiges Objekt.
2005 Ungültiger Parameter.

4.2.15.5 AuthenticView.CreateXMLNode

Methode: `CreateXMLNode` (*`nKind`* als `SPYXMLDataKind`) als `XMLData`

Rückgabewert
Die Methode gibt das neue XMLData Objekt zurück.

Beschreibung
Um ein neues `XMLData` Objekt zu erstellen, verwenden Sie die Methode `CreateXMLNode()`. Siehe auch "Verwendung von XMLData".

Fehler

2000 Ungültiges Objekt.
2012 XML-Node kann nicht erstellt werden.

4.2.15.6 AuthenticView.DisableAttributeEntryHelper

Methode: `DisableAttributeEntryHelper()`

Beschreibung
`DisableAttributeEntryHelper()` deaktiviert die Attribut-Eingabehilfe in XMLSpy, Authentic Desktop und dem Authentic Browser Plug-in.

Fehler

2000 Ungültiges Objekt.

4.2.15.7 AuthenticView.DisableElementEntryHelper

Methode: `DisableElementEntryHelper()`

Beschreibung
`DisableElementEntryHelper()` deaktiviert die Element-Eingabehilfe in XMLSpy, Authentic Desktop und dem Authentic Browser Plug-in.

Fehler
2000 Ungültiges Objekt.

4.2.15.8 AuthenticView.DisableEntityEntryHelper

Methode: `DisableEntityEntryHelper()`

Beschreibung
`DisableEntityEntryHelper()` deaktiviert die Entity-Eingabehilfe in XMLSpy, Authentic Desktop und dem Authentic Browser Plug-in.

Fehler
2000 Ungültiges Objekt.

4.2.15.9 AuthenticView.DocumentBegin

Siehe auch

Eigenschaft: `DocumentBegin` als AuthenticRange[111] (schreibgeschützt)

Beschreibung
Ruft ein Bereichsobjekt auf, das auf den Anfang des Dokuments zeigt.

Fehler
2000 Das Authentic-Ansichtsobjekt ist nicht mehr gültig.
2005 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

4.2.15.10 AuthenticView.DocumentEnd

Siehe auch

Eigenschaft: DocumentEnd als AuthenticRange[111] (Schreibgeschützt)

Beschreibung
Ruft ein Bereichsobjekt ab, das auf das Ende des Dokuments zeigt.

© 2018-2024 Altova GmbH

Fehler

2000 Das Authentic View-Objekt ist nicht mehr gültig.
2005 Für den zurückgegebenen Parameter wurde eine ungültige Adresse angegeben.

4.2.15.11 AuthenticView.DoNotPerformStandardAction

Methode `DoNotPerformStandardAction ()`

Beschreibung
`DoNotPerformStandardAction()` dient als Abbrech-Bubble für Makros und bricht die weitere Ausführung ab, nachdem das Makro fertig ausgeführt wurde.

Fehler

2000 Ungültiges Objekt.

4.2.15.12 AuthenticView.EvaluateXPath

Methode: `EvaluateXPath` (XMLData, expression) als string

Rückgabewert
Diese Methode gibt einen String zurück.

Beschreibung
`EvaluateXPath()` führt einen XPath-Ausdruck im angegebenen XML-Kontext-Node aus. Das Ergebnis wird als String zurückgegeben. Im Fall einer Sequenz ist der String durch Leerzeichen getrennt.

Fehler

2000 Ungültiges Objekt.
2005 Ungültiger Parameter.
2008 Interner Fehler.
2013 XPath-Fehler.

4.2.15.13 AuthenticView.Event

Siehe auch

Eigenschaft: `Event` als `AuthenticEvent` (schreibgeschützt)

Beschreibung
Über diese Eigenschaft erhalten Sie auf dieselbe Art wie bei `OldAuthenticView.event` Zugriff auf die Parameter des letzten Event. Da nun alle Events für die Skripting-Umgebung und externe Clients mit Parameters verfügbar sind, sollte diese `Event` Eigenschaft nur von IDE-Plug-ins aus verwendet werden.

Fehler
2000 Das Authentic-Ansichtsobjekt ist nicht mehr gültig.
2005 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

4.2.15.14 AuthenticView.EventContext

Eigenschaft: `EventContext` als `EventContext`

Beschreibung
Über die Eigenschaft `EventContext` erhalten Sie Zugriff auf den Kontext des laufenden Makros. Nähere Informationen dazu finden Sie unter der Beschreibung zur `EventContext` Schnittstelle.

Fehler
2000 Ungültiges Objekt.

4.2.15.15 AuthenticView.GetToolbarButtonState

Methode: `GetToolbarButtonState` (`ButtonIdentifier` als `string`) als `AuthenticToolbarButtonState`

Rückgabewert
Die Methode gibt `AuthenticToolbarButtonState` zurück.

Beschreibung
`GetToolbarButtonState` fragt den Status der Symbolleisten-Schaltfläche ab und gestattet dem Benutzer, die durch ihren Schaltflächen-Identifier gekennzeichnete Schaltfläche (siehe Liste oben 160) zu aktivieren bzw. zu deaktivieren. Auf diese Art können die Symbolleisten-Schaltflächen z.B. permanent deaktiviert werden. Eine andere Verwendung der Methode ist `SetToolbarButtonState` in den `OnSelectionChanged` Event Handler zu setzen, da die Symbolleisten-Schaltflächen regelmäßig aktualisiert werden, wenn sich die Auswahl im Dokument ändert.

Der Status der Symbolleisten-Schaltflächen wird unter der Liste der Enumerationen 195 aufgelistet.

Der Standardzustand bedeutet, dass die Aktivierung/Deaktivierung der Schaltfläche über AuthenticView erfolgt. Wenn der Benutzer den Zustand auf aktiviert oder deaktiviert setzt, bleibt die Schaltfläche so lange in diesem Zustand, solange der Benutzer den Zustand nicht ändert.

Fehler
2000 Ungültiges Objekt.
2005 Ungültiger Parameter.
2008 Interner Fehler.
2014 Ungültiger Schaltflächen-Identifier.

© 2018-2024 Altova GmbH

4.2.15.16 AuthenticView.Goto

Siehe auch

Methode: Goto (*eKind* als SPYAuthenticElementKind, *nCount* als Long, *eFrom* als SPYAuthenticDocumentPosition) als AuthenticRange[111]

Beschreibung
Ruft ein Bereichsobjekt auf, das auf den Anfang des *nCount*-Elements vom Typ *eKind* zeigt. Die Anfangsposition wird durch den Parameter *eFrom* definiert. Verwenden Sie positive Werte für *nCount*, um zum Ende des Dokuments zu navigieren. Verwenden Sie negative Werte, um zum Anfang des Dokuments zu navigieren.

Fehler

2000	Das Authentic View-Objekt ist nicht mehr gültig.
2003	Das Ziel befindet sich hinter dem Ende des Dokuments.
2004	Das Ziel befindet vor den Anfang des Dokuments.
2005	Es wurde ein ungültiger Elementtyp angegeben. Die Anfangsposition ist keine Position aus *spyAuthenticDocumentBegin* oder *spyAuthenticDocumentEnd*. Für den zurückgegebenen Parameter wurde eine ungültige Adresse angegeben.

Beispiele

```
Dim objAuthenticView
Set objAuthenticView = objPlugin.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
```

4.2.15.17 AuthenticView.IsRedoEnabled

Siehe auch

Eigenschaft: `IsRedoEnabled` als Boolean (schreibgeschützt)

Beschreibung
True, wenn Wiederhol-Schritte verfügbar sind und `Redo` möglich ist.

Fehler

2000	Das Authentic-Ansichtsobjekt ist nicht mehr gültig.
2005	Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

4.2.15.18 AuthenticView.IsUndoEnabled

Siehe auch

Eigenschaft: `IsUndoEnabled` als Boolean (schreibgeschützt)

Beschreibung
True, wenn Rückgängig-Schritte verfügbar sind und `Undo` möglich ist.

Fehler
2000 Das Authentic-Ansichtsobjekt ist nicht mehr gültig.
2005 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

4.2.15.19 AuthenticView.MarkupVisibility

Siehe auch

Eigenschaft: MarkupVisibility als SPYAuthenticMarkupVisibility

Beschreibung
Setzt oder liefert den aktuellen Sichtbarkeitsstatus von Markup.

Fehler
2000 Das Authentic View-Objekt ist nicht mehr gültig.
2005 Es wurde ein ungültiger Enumerationswert angegeben.
Für den zurückgegebenen Parameter wurde eine ungültige Adresse angegeben.

4.2.15.20 AuthenticView.Parent

Siehe auch

Eigenschaft: Parent als Authentic 69 (schreibgeschützt)

Beschreibung
Liefert eine Referenz auf die Applikation.

Fehler
2000 Das Authentic View-Objekt ist nicht mehr gültig.
2005 Für den zurückgegebenen Parameter wurde eine ungültige Adresse angegeben.

© 2018-2024 Altova GmbH

4.2.15.21 AuthenticView.Print

Siehe auch

Methode: Print (*bWithPreview* als Boolean, *bPromptUser* als Boolean)

Beschreibung
Druckt das in dieser Ansicht angezeigte Dokument. Wenn *bWithPreview* auf *True* gesetzt ist, wird das Dialogfeld "Druckvorschau" angezeigt. Wenn *bPromptUser* auf *True* gesetzt ist, wird das Dialogfeld "Drucken" angezeigt. Wenn beide Parameter auf *False* gesetzt sind, wird das Dokument ohne weiteren Eingriff durch den Benutzer gedruckt.

Fehler

2000	Das Authentic View-Objekt ist nicht mehr gültig.

4.2.15.22 AuthenticView.Redo

Siehe auch

Methode: Redo () als Boolean

Beschreibung
Stellt die letzte mit dem Befehl "Rückgängig" rückgängig gemachte Aktion wieder her.

Fehler

2000	Das Authentic View-Objekt ist nicht mehr gültig.
2005	Für den zurückgegebenen Parameter wurde eine ungültige Adresse angegeben.

4.2.15.23 AuthenticView.Selection

Siehe auch

Eigenschaft: Selection als AuthenticRange[111]

Beschreibung
Setzt oder holt die aktuelle Textauswahl auf der Benutzeroberfläche.

Fehler

2000	Das Authentic View-Objekt ist nicht mehr gültig.
2002	Es ist keine Selektion aktiv.
2005	Für den zurückgegebenen Parameter wurde eine ungültige Adresse angegeben.

Beispiele

```
' ------------------------------------
```

```
'              VBScript
' -------------------------------------
Dim objAuthenticView
Set objAuthenticView = objPlugin.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
```

4.2.15.24 AuthenticView.SetToolbarButtonState

Methode: `SetToolbarButtonState` (`ButtonIdentifier` als string, `AuthenticToolbarButtonState` state)

Beschreibung
`SetToolbarButtonState` fragt den Status der Symbolleisten-Schaltfläche ab und gestattet dem Benutzer, die durch ihren Schaltflächen-Identifier gekennzeichnete Schaltfläche (siehe Liste oben 160) zu aktivieren bzw. zu deaktivieren. Auf diese Art können die Symbolleisten-Schaltflächen z.B. permanent deaktiviert werden. Eine andere Verwendung der Methode ist `SetToolbarButtonState` in den `OnSelectionChanged` Event Handler zu setzen, da die Symbolleisten-Schaltflächen regelmäßig aktualisiert werden, wenn sich die Auswahl im Dokument ändert.

Der Status der Symbolleisten-Schaltflächen wird unter der Liste der Enumerationen 195 aufgelistet.

Der Standardzustand bedeutet, dass die Aktivierung/Deaktivierung der Schaltfläche über AuthenticView erfolgt. Wenn der Benutzer den Zustand auf aktiviert oder deaktiviert setzt, bleibt die Schaltfläche so lange in diesem Zustand, solange der Benutzer den Zustand nicht ändert.

Fehler

2000	Ungültiges Objekt.
2008	Interner Fehler.
2014	Ungültiger Schaltflächen-Identifier.

4.2.15.25 AuthenticView.Undo

Siehe auch

Methode: Undo () als Boolean

Beschreibung
Macht die letzte in dieser Ansicht vorgenommene Änderung am Dokument rückgängig.

© 2018-2024 Altova GmbH

Fehler

2000 Das Authentic View-Objekt ist nicht mehr gültig.
2005 Für den zurückgegebenen Parameter wurde eine ungültige Adresse angegeben.

4.2.15.26 AuthenticView.UpdateXMLInstanceEntities

Siehe auch

Methode: UpdateXMLInstanceEntities ()

Beschreibung
Aktualisiert die interne Darstellung der deklarierten Entities und befüllt die Eingabehilfen erneut. Zusätzlich dazu wird der Validator neu geladen, damit die XML-Datei korrekt validiert werden kann. Beachten Sie, dass dadurch eventuell auch Schema-Dateien neu geladen werden.

Fehler
Die Methode gibt nie einen Fehler zurück.

Beispiel

```
// ---------------------------------------
//               JavaScript
// ---------------------------------------
var objDocType;
objDocType = objPlugin.XMLRoot.GetFirstChild(10);

if(objDocType)
{
     var objEntity = objPlugin.CreateChild(14);
     objEntity.Name = "child";
     objEntity.TextValue = "SYSTEM \"child.xml\"";
     objDocType.AppendChild(objEntity);

     objPlugin.AuthenticView.UpdateXMLInstanceEntities();
}
```

4.2.15.27 AuthenticView.WholeDocument

Siehe auch

Eigenschaft: WholeDocument als AuthenticRange[111] (Schreibgeschützt)

Beschreibung
Ruft ein Bereichsobjekt ab, das das gesamte Dokument auswählt.

Fehler

2000 Das Authentic View-Objekt ist nicht mehr gültig.

2005 Für den zurückgegebenen Parameter wurde eine ungültige Adresse angegeben.

4.2.15.28 AuthenticView.XMLDataRoot

Siehe auch

Eigenschaft: `XMLDataRoot` als `XMLData` (schreibgeschützt)

Beschreibung
Gibt das am höchsten gestufte XMLData-Element des aktuellen Dokuments zurück oder definiert dieses. Dieses Element beschreibt normalerweise die Dokumentstruktur und wäre von der Art spyXMLDataXMLDocStruct, spyXMLDataXMLEntityDocStruct oder spyXMLDataDTDDocStruct.

Fehler
2000 Das Authentic-Ansichtsobjekt ist nicht mehr gültig.
2005 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

4.2.16 AuthenticXMLTableCommands

Methoden
Insert 175
Delete 175
AppendRow 174
InsertRow 176
DeleteRow 175
AppendColumn 174
InsertColumn 176
DeleteColumn 175
JoinLeft 176
JoinRight 176
JoinUp 176
JoinDown 176
SplitHorizontal 180
SplitVertical 180
AlignVerticalTop 174
AlignVerticalCenter 174
AlignVerticalBottom 174
AlignHorizontalLeft 173
AlignHorizontalRight 173
AlignHorizontalCenter 173
AlignHorizontalJustify 173
EditProperties 175

Eigenschaften
MayInsert 178
MayDelete 177
MayAppendRow 177

© 2018-2024 Altova GmbH

4.2.16.1 AuthenticXMLTableCommands.AlignHorizontalCenter

Deklaration: AlignHorizontalCenter()

Beschreibung
Setzt die horizontale Ausrichtung auf "zentriert" oder setzt das Attribut zurück, wenn es zuvor auf "zentriert" gesetzt war.

4.2.16.2 AuthenticXMLTableCommands.AlignHorizontalJustify

Deklaration: AlignHorizontalJustify()

Beschreibung
Setzt die horizontale Ausrichtung auf "Blocksatz" oder setzt das Attribut zurück, wenn es zuvor auf "Blocksatz" gesetzt war.

4.2.16.3 AuthenticXMLTableCommands.AlignHorizontalLeft

Deklaration: AlignHorizontalLeft()

Beschreibung
Setzt die horizontale Ausrichtung auf "links" oder setzt das Attribut zurück, wenn es zuvor auf "links" gesetzt war.

4.2.16.4 AuthenticXMLTableCommands.AlignHorizontalRight

Deklaration: AlignHorizontalRight()

Beschreibung
Setzt die horizontale Ausrichtung auf "rechts" oder setzt das Attribut zurück, wenn es zuvor auf "rechts" gesetzt war.

4.2.16.5 AuthenticXMLTableCommands.AlignVerticalBottom

Deklaration: AlignVerticalBottom()

Beschreibung
Setzt die vertikale Ausrichtung auf "unten" oder setzt das Attribut zurück, wenn es zuvor auf "unten" gesetzt war.

4.2.16.6 AuthenticXMLTableCommands.AlignVerticalCenter

Deklaration: AlignVerticalCenter()

Beschreibung
Setzt die vertikale Ausrichtung auf "Mitte" oder setzt das Attribut zurück, wenn es zuvor auf "Mitte" gesetzt war.

4.2.16.7 AuthenticXMLTableCommands.AlignVerticalTop

Deklaration: AlignVerticalTop()

Beschreibung
Setzt die vertikale Ausrichtung auf "oben" oder setzt das Attribut zurück, wenn es zuvor auf "oben" gesetzt war.

4.2.16.8 AuthenticXMLTableCommands.AppendCol

Deklaration: AppendCol()

Beschreibung
Hängt eine Spalte an die aktuelle Tabelle an.

4.2.16.9 AuthenticXMLTableCommands.AppendRow

Deklaration: AppendRow()

Beschreibung
Hängt eine Zeile an die aktuelle Tabelle an.

© 2018-2024 Altova GmbH

4.2.16.10 AuthenticXMLTableCommands.Delete

Deklaration: Delete()

Beschreibung
Wenn der Benutzer den Vorgang bestätig hat, löscht die Methode die aktuell ausgewählte Tabelle.

4.2.16.11 AuthenticXMLTableCommands.DeleteCol

Deklaration: DeleteCol()

Beschreibung
Die Methode löscht die aktuell ausgewählte Spalte.
Wenn es nur mehr eine Spalte gibt und der Benutzer den Vorgang bestätigt, wird die gesamte Tabelle gelöscht.

4.2.16.12 AuthenticXMLTableCommands.DeleteRow

Deklaration: DeleteRow()

Beschreibung
Die Methode löscht die aktuell ausgewählte Zeile.
Wenn es nur mehr eine Zeile gibt und der Benutzer den Vorgang bestätigt, wird die gesamte Tabelle gelöscht.

4.2.16.13 AuthenticXMLTableCommands.EditProperties

Deklaration: EditProperties()

Beschreibung
Zeigt das Dialogfeld "Eigenschaften" für die ausgewählte Tabelle/Zelle an.

4.2.16.14 AuthenticXMLTableCommands.Insert

Deklaration: Insert()

Beschreibung
Ruft ein Dialogfeld auf und fügt eine neue Tabelle in die bestehende XML-Datei ein.

4.2.16.15 AuthenticXMLTableCommands.InsertCol

Deklaration: InsertCol()

Beschreibung
Fügt eine neue Spalte vor der aktuelle ausgewählten Spalte ein.

4.2.16.16 AuthenticXMLTableCommands.InsertRow

Deklaration: InsertRow()

Beschreibung
Fügt eine Zeile vor der aktuell ausgewählten Zeile ein.

4.2.16.17 AuthenticXMLTableCommands.JoinDown

Deklaration: JoinDown()

Beschreibung
Verbindet die aktuelle Zelle mit der Zelle unmittelbar darunter.

4.2.16.18 AuthenticXMLTableCommands.JoinLeft

Deklaration: JoinLeft()

Beschreibung
Verbindet die aktuelle Zelle mit der Zelle unmittelbar links davon.

4.2.16.19 AuthenticXMLTableCommands.JoinRight

Deklaration: JoinRight()

Beschreibung
Verbindet die aktuelle Zelle mit der Zelle unmittelbar rechts davon.

4.2.16.20 AuthenticXMLTableCommands.JoinUp

Deklaration: JoinUp()

© 2018-2024 Altova GmbH

Beschreibung
Verbindet die aktuelle Zelle mit der Zelle unmittelbar darüber.

4.2.16.21 AuthenticXMLTableCommands.MayAlignHorizontal

Deklaration: MayAlignHorizontal als Boolean

Beschreibung
True, wenn die Attribute für die horizontale Ausrichtung für die aktuelle Zelle gesetzt werden können.

4.2.16.22 AuthenticXMLTableCommands.MayAlignVertical

Deklaration: MayAlignVertical als Boolean

Beschreibung
True, wenn die Attribute für die vertikale Ausrichtung für die aktuelle Zelle gesetzt werden können.

4.2.16.23 AuthenticXMLTableCommands.MayAppendCol

Deklaration: MayAppendCol als Boolean

Beschreibung
True, wenn eine Spalte angehängt werden kann.

4.2.16.24 AuthenticXMLTableCommands.MayAppendRow

Deklaration: MayAppendRow als Boolean

Beschreibung
True, wenn eine Zeile angehängt werden kann.

4.2.16.25 AuthenticXMLTableCommands.MayDelete

Deklaration: MayDelete als Boolean

Beschreibung
Die Eigenschaft ist "true", wenn eine Tabelle ausgewählt ist.

4.2.16.26 AuthenticXMLTableCommands.MayDeleteCol

Deklaration: MayDeleteCol als Boolean

Beschreibung
True, wenn die aktuelle Spalte gelöscht werden kann.

4.2.16.27 AuthenticXMLTableCommands.MayDeleteRow

Deklaration: MayDeleteRow als Boolean

Beschreibung
True, wenn die aktuelle Zeile gelöscht werden kann.

4.2.16.28 AuthenticXMLTableCommands.MayEditProperties

Deklaration: MayEditProperties als Boolean

Beschreibung
Die Eigenschaft ist "true", wenn das Dialogfeld "Eigenschaften" für die ausgewählte Tabelle/Zelle zur Verfügung steht.

4.2.16.29 AuthenticXMLTableCommands.MayInsert

Deklaration: MayInsert als Boolean

Beschreibung
Die Eigenschaft ist "true", wenn an der aktuellen Auswahl eine neue Tabelle eingefügt werden kann.

4.2.16.30 AuthenticXMLTableCommands.MayInsertCol

Deklaration: MayInsertCol als Boolean

Beschreibung
True, wenn eine Spalte eingefügt werden kann.

© 2018-2024 Altova GmbH

4.2.16.31 AuthenticXMLTableCommands.MayInsertRow

Deklaration: MayInsertRow als Boolean

Beschreibung
True, wenn eine Zeile eingefügt werden kann.

4.2.16.32 AuthenticXMLTableCommands.MayJoinDown

Deklaration: MayJoinDown als Boolean

Beschreibung
Die Eigenschaft ist "true", wenn die JoinDown-Operation für die aktuell ausgewählte Zelle möglich ist.

4.2.16.33 AuthenticXMLTableCommands.MayJoinLeft

Deklaration: MayJoinLeft als Boolean

Beschreibung
Die Eigenschaft ist true, wenn die JoinLeft-Operation für die aktuell ausgewählte Zelle möglich ist.

4.2.16.34 AuthenticXMLTableCommands.MayJoinRight

Deklaration: MayJoinRight als Boolean

Beschreibung
Die Eigenschaft ist true, wenn die JoinRight-Operation für die aktuell ausgewählte Zelle möglich ist.

4.2.16.35 AuthenticXMLTableCommands.MayJoinUp

Deklaration: MayJoinUp als Boolean

Beschreibung
Die Eigenschaft ist "true", wenn die JoinUp-Operation für die aktuell ausgewählte Zelle möglich ist.

4.2.16.36 AuthenticXMLTableCommands.MaySplitHorizontal

Deklaration: MaySplitHorizontal als Boolean

Beschreibung
True, wenn die aktuelle Zelle horizontal geteilt werden kann.

4.2.16.37 AuthenticXMLTableCommands.MaySplitVertical

Deklaration: MaySplitVertical als Boolean

Beschreibung
True, wenn die aktuelle Zelle vertikal geteilt werden kann.

4.2.16.38 AuthenticXMLTableCommands.SplitHorizontal

Deklaration: SplitHorizontal()

Beschreibung
Die Methode teilt die aktuelle Zelle horizontal.

4.2.16.39 AuthenticXMLTableCommands.SplitVertical

Deklaration: SplitVertical()

Beschreibung
Die Methode teilt die aktuelle Zelle vertikal.

4.2.17 XMLData

Methoden

 © 2018-2024 Altova GmbH

HasChildrenKind[187]
CountChildren[181]
CountChildrenKind[182]

Eigenschaften
Name[189]
TextValue[190]
HasChildren[187]
MayHaveChildren[189]
Kind[189]
Parent[190]

Beschreibung

Mit Hilfe der XMLData-Schnittstelle können Sie den Inhalt der gerade angezeigten XML-Datei manipulieren. Die Schnittstelle ist ein vereinfachtes Pendant zur COM-Schnittstelle, die im Plug-In und in XMLSpy selbst implementiert ist.

Verwenden Sie zum Erstellen eines neuen XMLData-Objekts die CreateChild()-Methode der Plug-In-Schnittstelle.

4.2.17.1 XMLData.AppendChild

Siehe auch

Deklaration: AppendChild(*pNewData* als XMLData[180])

Beschreibung
AppendChild hängt pNewData als letztes Child an das XMLData-Objekt an. Siehe auch "Using XMLData[60]".

Beispiel

```
Dim objCurrentParent
Dim objNewChild

Set objNewChild = objPlugIn.CreateChild(spyXMLDataElement)
Set objCurrentParent = objPlugIn.XMLRoot

objCurrentParent.AppendChild objNewChild

Set objNewChild = Nothing
```

4.2.17.2 XMLData.CountChildren

Siehe auch

Deklaration: CountChildren als long

Beschreibung
`CountChildren` holt die Anzahl der Children.

Verfügbar mit TypeLibrary Version 1.6

Fehler
1500 Das XMLData-Objekt ist nicht mehr gültig.

4.2.17.3 XMLData.CountChildrenKind

Siehe auch

Deklaration: CountChildrenKind (*nKind* as SPYXMLDataKind[195]) als long

Beschreibung
`CountChildrenKind` holt die Anzahl der Children der jeweiligen Art.

Verfügbar mit TypeLibrary Version 1.6

Fehler
1500 Das XMLData-Objekt ist nicht mehr gültig.

4.2.17.4 XMLData.EraseAllChildren

Siehe auch

Deklaration: EraseAllChildren

Beschreibung
EraseAllChildren löscht alle verknüpften Children des XMLData-Objekts.

Beispiel

In diesem Beispiel werden alle Elemente des aktiven Dokuments gelöscht.

```
Dim objCurrentParent

Set objCurrentParent = objPlugIn.XMLRoot
objCurrentParent.EraseAllChildren
```


© 2018-2024 Altova GmbH

4.2.17.5 XMLData.EraseChild

Methode: `EraseChild` (Child-Node und neuer Node als XMLData[180])

Beschreibung
Löscht den angegebenen Child Node.

Fehler
1500 Ungültiges Objekt.
1506 Ungültige XML-Input-Datei.
1510 Ungültiger Parameter.

4.2.17.6 XMLData.EraseCurrentChild

Siehe auch

Deklaration: EraseCurrentChild

Beschreibung
EraseCurrentChild löscht das aktuelle XMLData Child-Objekt. Bevor Sie EraseCurrentChild aufrufen, müssen Sie mit XMLData.GetFirstChild[185] einen internen Iterator initialisieren.

Beispiel

In diesem JavaScript-Beispiel werden alle Elemente mit dem Namen "EraseMe" gelöscht. Sie sehen im Code, dass innerhalb eines Schleifendurchlaufs EraseCurrentChild und GetNextChild aufgerufen werden können, sodass die Child-Elemente der Reihe nach abgearbeitet werden können.

```
function DeleteXMLElements(objXMLData)
{
	if(objXMLData == null)
		return;

	if(objXMLData.HasChildren) {
		var objChild;
		objChild = objXMLData.GetFirstChild(-1);

		while(objChild)	{
			DeleteXMLElements(objChild);

			try	{
				if(objChild.Name == "EraseMe")
					objXMLData.EraseCurrentChild();

				objChild = objXMLData.GetNextChild();
			}
			catch(Err)	{
				objChild = null;
```

```
                    }
                }
            }
        }
```

4.2.17.7 XMLData.GetChild

Siehe auch

Deklaration: `GetChild` (*`position`* as long) als XMLData[180]

Rückgabewert
Gibt ein XML-Element als `XMLData`-Objekt zurück.

Beschreibung
`GetChild()` gibt eine Referenz auf das Child des jeweiligen Index (nullbasiert) zurück.

Verfügbar mit TypeLibrary Version 1.6

Fehler
1500 Das XMLData Objekt ist nicht mehr gültig.
1510 Es wurde eine ungültige Adresse für den Rückgabeparameter angegeben.

4.2.17.8 XMLData.GetChildAttribute

Methode: `GetChildAttribute` (Name als string) als XMLData Objekt (`NULL` bei Fehler)

Beschreibung
Ruft das Attribut mit dem angegebenen Namen auf.

Fehler
1500 Ungültiges Objekt.
1510 Ungültiger Parameter.

4.2.17.9 XMLData.GetChildElement

Methode: `GetChildElement` (Name als string, position als integer) als XMLData-Objekt (`NULL` bei Fehler)

Beschreibung
Ruft das nte Child Element des angegebenen Namens auf.

Fehler
1500 Ungültiges Objekt.
1510 Ungültiger Parameter.

© 2018-2024 Altova GmbH

4.2.17.10 XMLData.GetChildKind

Siehe auch

Deklaration: GetChildKind (*position* as long, *nKind* as SPYXMLDataKind[195]) als XMLData[180]

Rückgabewert
Gibt ein XML-Element als `XMLData`-Objekt zurück.

Beschreibung
`GetChildKind()` gibt eine Referenz auf das Child dieser Art des jeweiligen Index (nullbasiert) zurück. Der Positionsparameter ist relativ zur Anzahl der Children der ngegebenen Art und nicht zu alle Children des Objekts.

Verfügbar mit TypeLibrary Version 1.6

Fehler
1500 Das XMLData Objekt ist nicht mehr gültig.
1510 Es wurde eine ungültige Adresse für den Rückgabeparameter angegeben.

4.2.17.11 XMLData.GetCurrentChild

Siehe auch

Deklaration: GetCurrentChild als XMLData[180]

Rückgabewert
Gibt ein XML-Element als XMLData-Objekt zurück.

Beschreibung
GetCurrentChild holt das aktuelle Child. Bevor Sie GetCurrentChild aufrufen, müssen Sie mit XMLData.GetFirstChild[185] einen internen Iterator initialisieren.

4.2.17.12 XMLData.GetFirstChild

Siehe auch

Deklaration: GetFirstChild(*nKind* als SPYXMLDataKind[195]) als XMLData[180]

Rückgabewert
Gibt ein XML-Element als XMLData-Objekt zurück.

Beschreibung
GetFirstChild initialisiert einen neuen Iterator und gibt das erste Child zurück. Setzen Sie nKind = -1, um einen Iterator für alle Arten von Children zu holen.

Beispiel

Siehe Beispiel unter XMLData.GetNextChild[186].

4.2.17.13 XMLData.GetNamespacePrefixForURI

Methode: `GetNamespacePrefixForURI` (URI als string) Präfix als string

Beschreibung
Gibt das Namespace-Präfix der vorgegebenen URI zurück.

Fehler
1500 Ungültiges Objekt.
1510 Ungültiger Parameter.

4.2.17.14 XMLData.GetNextChild

Siehe auch

Deklaration: GetNextChild als XMLData[180]

Rückgabewert
Gibt ein XML-Element als XMLData-Objekt zurück.

Beschreibung
GetNextChild geht zum nächsten Child dieses Elements. Bevor Sie GetNextChild aufrufen, müssen Sie mit XMLData.GetFirstChild[185] einen internen Iterator initialisieren.

Überprüft, wann das letzte Child des Elements erreicht wurde (siehe Beispiel unten).

Beispiel

```
On Error Resume Next
Set objParent = objPlugIn.XMLRoot

'get elements of all kinds
Set objCurrentChild = objParent.GetFirstChild(-1)

Do
	'do something useful with the child

	'step to next child
```


© 2018-2024 Altova GmbH

```
        Set objCurrentChild = objParent.GetNextChild
    Loop Until (Err.Number - vbObjectError = 1503)
```

4.2.17.15 XMLData.GetTextValueXMLDecoded

Methode: `GetTextValueXMLDecoded` als string

Beschreibung
Ruft den dekodierten Textwert der XML-Datei ab.

Fehler
1500 Ungültiges Objekt.
1510 Ungültiger Parameter.

4.2.17.16 XMLData.HasChildren

Siehe auch

Deklaration: HasChildren als Boolean

Beschreibung
Die Eigenschaft ist "true", wenn das Objekt ein Parent eines anderen XMLData-Objekts ist.

Diese Eigenschaft ist schreibgeschützt.

4.2.17.17 XMLData.HasChildrenKind

Siehe auch

Dekaration: HasChildrenKind (*nKind* as SPYXMLDataKind[195]) al Boolean

Beschreibung
Die Methode gibt "true" zurück, wenn das Objekt der Parent anderer `XMLData` Objekte der angegebenen Art ist.

Verfügbar mit TypeLibrary Version 1.6

Fehler
1500 Das XMLData Objekt ist nicht mehr gültig.
1510 Es wurde eine ungültige Adresse für den Rückgabeparameter angegeben.

4.2.17.18 XMLData.InsertChild

Siehe auch

Deklaration: InsertChild(*pNewData* als XMLData[180])

Beschreibung
InsertChild fügt das neue Child vor dem aktuellen Child ein (Siehe auch XMLData.GetFirstChild[185], XMLData.GetNextChild[186] zu Setzen des aktuellen Child).

4.2.17.19 XMLData.InsertChildAfter

Methode: `InsertChildAfter` (Child Node und neuer Node als XMLData)

Beschreibung
Fügt einen neuen XML-Node nach dem angegebenen Node ein.

Errors

1500	Ungültiges Objekt.
1506	Ungültige XML-Input-Datei
1507	Es sind keine Child-Nodes zulässig
1510	Ungültiger Parameter.
1512	Das Child wurde bereits hinzugefügt
1514	Ungültige Art an dieser Stelle

4.2.17.20 XMLData.InsertChildBefore

Methode: `InsertChildBefore` (Child Node und neuer Node als XMLData)

Beschreibung
Fügt einen neuen XML-Node vor dem angegebenen Node ein.

Errors

1500	Ungültiges Objekt.
1506	Ungültige XML-Input-Datei
1507	Es sind keine Child-Nodes zulässig
1510	Ungültiger Parameter.
1512	Das Child wurde bereits hinzugefügt
1514	Ungültige Art an dieser Stelle

© 2018-2024 Altova GmbH

4.2.17.21 XMLData.IsSameNode

Siehe auch

Deklaration: IsSameNode(*pNodeToCompare* als XMLData[180]) als Boolean

Beschreibung
Gibt "true" zurück, wenn pNodeToCompare denselben Node referenziert wie das Objekt selbst.

4.2.17.22 XMLData.Kind

Siehe auch

Deklaration: Kind als SPYXMLDataKind[195]

Beschreibung
Typ dieses XMLData-Objekts.

Diese Eigenschaft ist schreibgeschützt.

4.2.17.23 XMLData.MayHaveChildren

Siehe auch

Deklaration: MayHaveChildren als Boolean

Beschreibung
Gibt an, ob es zulässig ist, Children zu diesem XMLData-Objekt hinzuzufügen.

Diese Eigenschaft ist schreibgeschützt.

4.2.17.24 XMLData.Name

Siehe auch

Deklaration: Name als String

Beschreibung
Wird verwendet, um den Namen des XMLData-Objekts zu holen und zu ändern.

4.2.17.25 XMLData.Parent

Siehe auch

Deklaration: Parent als XMLData[180]

Rückgabewert
Parent als XMLData-Objekt.
Kein Wert (oder NULL), wenn es kein Parent-Element gibt.

Beschreibung
Parent von diesem Element.

Diese Eigenschaft ist schreibgeschützt.

4.2.17.26 XMLData.SetTextValueXMLDecoded

Methode: `SetTextValueXMLDecoded` (string-Wert)

Beschreibung
Definiert den kodierten Textwert der XML-Datei.

Fehler
1500 Ungültiges Objekt.
1513 Keine Änderung zulässig.

4.2.17.27 XMLData.TextValue

Siehe auch

Deklaration: TextValue als String

Beschreibung
Wird verwendet, um den Textwert von diesem XMLData-Objekt zu ändern und zu holen. Siehe auch "Using XMLData[60]".

© 2018-2024 Altova GmbH

4.3 Enumerationen

Dieser Abschnitt enthält eine Liste und Beschreibung der Authentic Browser Enumerationen.

4.3.1 SPYAuthenticActions

Beschreibung
Aktionen, die an AuthenticRange[111]-Objekten ausgeführt werden können.

Mögliche Werte:

spyAuthenticInsertAt	= 0
spyAuthenticApply	= 1
spyAuthenticClearSurr	= 2
spyAuthenticAppend	= 3
spyAuthenticInsertBefore	= 4
spyAuthenticRemove	= 5

4.3.2 SPYAuthenticCommand

Beschreibung
Enumeration aller verfügbaren Befehle.

Mögliche Werte:

// CommandGroupMain

k_CommandSeparator	= 0
k_CommandSave	= 1
k_CommandPrint	= 2
k_CommandPrintPreview	= 3
k_CommandValidate	= 4
k_CommandUndo	= 5
k_CommandRedo	= 6

// CommandGroupEdit

k_CommandEditCut	= 7
k_CommandEditCopy	= 8
k_CommandEditPaste	= 9
k_CommandEditFind	= 10

k_CommandEditRepeat = 11
k_CommandEditReplace = 12

// CommandGroupMarkup
k_CommandMarkupHide = 13
k_CommandMarkupLarge = 14

// CommandGroupRow
k_CommandRowAppend = 15
k_CommandRowInsert = 16
k_CommandRowDuplicate = 17
k_CommandRowMoveUp = 18
k_CommandRowMoveDown = 19
k_CommandRowDelete = 20

// CommandGroupXMLTables
k_CommandXMLTableInsert = 21
k_CommandXMLTableDelete = 22
k_CommandXMLTableAppendRow = 23
k_CommandXMLTableInsertRow = 24
k_CommandXMLTableDeleteRow = 25
k_CommandXMLTableAppendCol = 26
k_CommandXMLTableInsertCol = 27
k_CommandXMLTableDeleteCol = 28
k_CommandXMLTableJoinRight = 29
k_CommandXMLTableJoinLeft = 30
k_CommandXMLTableJoinUp = 31
k_CommandXMLTableJoinDown = 32
k_CommandXMLTableSplitHorz = 33
k_CommandXMLTableSplitVert = 34
k_CommandXMLTableVAlignTop = 35
k_CommandXMLTableVAlignCenter = 36
k_CommandXMLTableVAlignBottom = 37
k_CommandXMLTableAlignLeft = 38
k_CommandXMLTableAlignCenter = 39
k_CommandXMLTableAlignRight = 40
k_CommandXMLTableAlignJustify = 41
k_CommandXMLTableEditProperties = 42

© 2018-2024 Altova GmbH

// since TypeLib 1.2

k_CommandCheckSpelling	= 43
k_CommandAbout	= 44
k_CommandPackageManagement	= 45

4.3.3 SPYAuthenticCommandGroup

Beschreibung

Gruppen, zu denen ein Befehl gehören kann.

Mögliche Werte:

k_CommandGroupMain	= 0
k_CommandGroupEdit	= 1
k_CommandGroupMarkup	= 2
k_CommandGroupRow	= 3
k_CommandGroupXMLTables	= 4

4.3.4 SPYAuthenticDocumentPosition

Beschreibung

Relative und absolute Position zum Navigieren mit AuthenticRange[111]-Objekten.

Mögliche Werte:

spyAuthenticDocumentBegin	= 0
spyAuthenticDocumentEnd	= 1
spyAuthenticRangeBegin	= 2
spyAuthenticRangeEnd	= 3

4.3.5 SPYAuthenticElementActions

Beschreibung

Aktionen, die mit GetAllowedElements[81] verwendet werden können.

Mögliche Werte:

k_ActionInsertAt	= 0
k_ActionApply	= 1
k_ActionClearSurr	= 2
k_ActionAppend	= 3

k_ActionInsertBefore	= 4
k_ActionRemove	= 5

4.3.6 SPYAuthenticElementKind

Beschreibung

Enumeration der verschiedenen zur Navigation und Auswahl innerhalb der Objekte AuthenticRange[111] und AuthenticView[150] verwendeten Elementarten.

Mögliche Werte:

spyAuthenticChar	= 0
spyAuthenticWord	= 1
spyAuthenticLine	= 3
spyAuthenticParagraph	= 4
spyAuthenticTag	= 6
spyAuthenticDocument	= 8
spyAuthenticTable	= 9
spyAuthenticTableRow	= 10
spyAuthenticTableColumn	= 11

4.3.7 SPYAuthenticEntryHelperWindows

Beschreibung

Identifizierung der verfügbaren Eingabehilfenfenster

Mögliche Werte:

k_Elements	= 1
k_Attributes	= 2
k_Entities	= 4

4.3.8 SPYAuthenticMarkupVisibility

Beschreibung

Enumerationswerte zum Aus- oder Einblenden des Markups.

Mögliche Werte:

spyAuthenticMarkupHidden	= 0
spyAuthenticMarkupSmall	= 1
spyAuthenticMarkupLarge	= 2
spyAuthenticMarkupMixed	= 3

© 2018-2024 Altova GmbH

4.3.9 SPYAuthenticToolbarAllignment

Beschreibung

Werte zur Definition der Symbolleistenposition.

Mögliche Werte:

k_ToolbarAlignTop	= 0
k_ToolbarAlignLeft	= 1
k_ToolbarAlignBottom	= 2
k_ToolbarAlignRight	= 3

4.3.10 SPYAuthenticToolbarButtonState

Beschreibung

Die Status der Authentic-Symbolleisten-Schaltflächen werden durch die folgende Enumeration definiert:

Possible values:

authenticToolbarButtonDefault	= 0
authenticToolbarButtonEnabled	= 1
authenticToolbarButtonDisabled	= 2

4.3.11 SPYXMLDataKind

Beschreibung

Die verschiedenen Typen von XMLData-Elementen, die für XML-Dokumente zur Verfügung stehen

Mögliche Werte:

spyXMLDataXMLDocStruct	= 0
spyXMLDataXMLEntityDocStruct	= 1
spyXMLDataDTDDocStruct	= 2
spyXMLDataXML	= 3
spyXMLDataElement	= 4
spyXMLDataAttr	= 5
spyXMLDataText	= 6
spyXMLDataCData	= 7
spyXMLDataComment	= 8
spyXMLDataP	= 9
spyXMLDataDefDoctype	= 10

spyXMLDataDefExternalID	= 11
spyXMLDataDefElement	= 12
spyXMLDataDefAttlist	= 13
spyXMLDataDefEntity	= 14
spyXMLDataDefNotation	= 15
spyXMLDataKindsCount	= 16

© 2018-2024 Altova GmbH

5 Lizenzinformationen

Dieser Anhang enthält die folgenden Informationen:

- die Lizenzvereinbarung zu diesem Software-Produkt

Lesen Sie die Informationen bitte sorgfältig - sie sind rechtlich bindend, da Sie sich bei der Installation dieses Software-Produkts damit einverstanden erklärt haben.

Den Inhalt aller Altova-Lizenzenvereinbarungen finden Sie auf der Altova Website unter Rechtliches.

5.1 Altova Endbenutzer-Lizenzvereinbarung zu Authentic

- Die Altova-Endbenutzer-Lizenzvereinbarung für Authentic kann unter https://www.altova.com/de/legal/authentic-eula eingesehen werden.
- Die Altova-Datenschutzbestimmungen finden Sie unter https://www.altova.com/de/privacy.

© 2018-2024 Altova GmbH

Index

A

© 2018-2024 Altova GmbH

B

© 2018-2024 Altova GmbH

C

D

E

H

I

L

M

N

O

P

© 2018-2024 Altova GmbH

R

S

T

V

X

Z

© 2018-2024 Altova GmbH