User Manual and Programmers' Reference

Copyright © 1998-2011. Altova GmbH. All rights reserved. Use of this software is governed by and subject to an Altova software license agreement. XMLSpy, MapForce, StyleVision, SemanticWorks, SchemaAgent, UModel, DatabaseSpy, DiffDog, Authentic, AltovaXML, MissionKit, and ALTOVA as well as their logos are trademarks and/or registered trademarks of Altova GmbH. Protected by US Patent 7,739,292.

ALTOVA®

XML, XSL, XHTML, and W3C are trademarks (registered in numerous countries) of the World Wide Web Consortium; marks of the W3C are registered and held by its host institutions, MIT, INRIA, and Keio. UNICODE and the Unicode Logo are trademarks of Unicode Inc. This software contains 3rd party software or material that is protected by copyright and subject to other terms and conditions as detailed on the Altova website at http://www.altova.com/legal_3rdparty.html

Altova XMLSpy 2011 User & Reference Manual

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: 2011

© 2011 Altova GmbH

Table of Contents

Welcome to XMLSpy 3

User Manual 6

1 Introduction 8

1.1 The Graphical User Interface (GUI) 9
- 1.1.1 Main Window....10
- 1.1.2 Project Window....11
- 1.1.3 Info Window....13
- 1.1.4 Entry Helpers....13
- 1.1.5 Output Window: Messages....14
- 1.1.6 Menu Bar, Toolbars, Status Bar....15

1.2 The Application Environment 16
- 1.2.1 Settings and Customization....16
- 1.2.2 Tutorials, Projects, Examples....16
- 1.2.3 XMLSpy Features and Help, and Altova Products....17

2 XMLSpy Tutorial 18

2.1 XMLSpy Interface 19

2.2 XML Schemas 20

2.3 XML Documents 22
- 2.3.1 Creating a New XML File....22
- 2.3.2 Specifying the Type of an Element....24
- 2.3.3 Entering Data in Text View....25
- 2.3.4 Validating the Document....29
- 2.3.5 Adding Elements and Attributes....32

2.4 XSLT Transformations 34
- 2.4.1 Assigning an XSLT File....34
- 2.4.2 Transforming the XML File....35
- 2.4.3 Modifying the XSL File....36

2.5 Project Management 38
- 2.5.1 Benefits of Projects....38

2.5.2 Building a Project 38
2.6 That's It 40

3 Editing Views 41

3.1 Text View 42
3.1.1 Formatting in Text View 42
3.1.2 Displaying the Document 44
3.1.3 Editing in Text View 47
3.1.4 Entry Helpers in Text View 49
3.2 Grid View 51
3.2.1 Entry Helpers in Grid View 52
3.3 Schema View 54
3.3.1 Schema Overview 54
3.3.2 Content Model View 58
3.3.3 Base Type Modification 70
3.3.4 Entry Helpers in Schema View 71
3.3.5 Identity Constraints 74
3.3.6 Smart Restrictions 78
3.3.7 xml:base, xml:id, xml:lang, xml:space 82
3.3.8 Back and Forward: Moving through Positions 84
3.4 Authentic View 85
3.4.1 Overview of the GUI 85
3.4.2 Authentic View Toolbar Icons 86
3.4.3 Authentic View Main Window 88
3.4.4 Authentic View Entry Helpers 90
3.4.5 Authentic View Context Menus 94
3.5 Browser View 97

4 XML 98

4.1 Creating, Opening, and Saving XML Documents 99
4.2 Assigning Schemas and Validating 101
4.3 Editing XML in Text View 103
4.4 Editing XML in Grid View 105
4.5 Editing XML in Authentic View 106
4.6 Entry Helpers for XML Documents 108
4.7 Processing with XSLT and XQuery 110
4.8 Additional Features 112

5 DTDs and XML Schemas 113

5.1 DTDs 114
5.2 XML Schemas 115

5.3 Catalogs in XMLSpy 116

6 XSLT and XQuery 120

6.1 XSLT 121
6.1.1 XSLT Documents 121
6.1.2 XSLT Processing 123
6.2 XQuery 125
6.2.1 XQuery Documents 126
6.2.2 XQuery Entry Helpers 127
6.2.3 XQuery Syntax Coloring 127
6.2.4 XQuery Intelligent Editing 129
6.2.5 XQuery Validation and Execution 130

7 Authentic 132

7.1 Authentic View Tutorial 134
7.1.1 Opening an XML Document in Authentic View 135
7.1.2 The Authentic View Interface 136
7.1.3 Node Operations 138
7.1.4 Entering Data in Authentic View 141
7.1.5 Entering Attribute Values 143
7.1.6 Adding Entities 143
7.1.7 Printing the Document 144
7.2 Editing in Authentic View 146
7.2.1 Basic Editing 146
7.2.2 Tables in Authentic View 150
– SPS Tables 150
– CALS/HTML Tables 151
– CALS/HTML Table Editing Icons 155
7.2.3 Editing a DB 157
– Navigating a DB Table 157
– DB Queries 158
– Modifying a DB Table 162
7.2.4 Working with Dates 163
– Date Picker 163
– Text Entry 164
7.2.5 Defining Entities 164
7.2.6 Images in Authentic View 166
7.2.7 Keystrokes in Authentic View 167
7.3 Authentic Scripting 168

8 HTML, CSS, JSON 170

8.1 HTML 171
8.2 CSS 173

9 Altova Global Resources 177

9.1 Defining Global Resources 178
9.1.1 Files 180
9.1.2 Folders 182
9.1.3 Copying Configurations 183
9.2 Using Global Resources 184
9.2.1 Assigning Files and Folders 184
9.2.2 Changing Configurations 187

10 Projects 188

10.1 Creating and Editing Projects 189
10.2 Using Projects 192

11 Source Control 194

11.1 Supported Source Control Systems 196
11.2 Installing Source Control Systems 201
11.3 SCSs and Altova DiffDog Differencing 208

12 User Reference 214

12.1 File Menu 215
12.1.1 New 215
12.1.2 Open 218
12.1.3 Reload 222
12.1.4 Encoding 222
12.1.5 Close, Close All, Close All But Active 222
12.1.6 Save, Save As, Save All 223
12.1.7 Send by Mail 228
12.1.8 Print 229
12.1.9 Print Preview, Print Setup 229
12.1.10 Recent Files, Exit 230
12.2 Edit Menu 231
12.2.1 Undo, Redo 231
12.2.2 Cut, Copy, Paste, Delete 231
12.2.3 Insert 231
12.2.4 Copy XPath 234
12.2.5 Copy XPointer 234
12.2.6 Pretty-Print XML Text 235

12.2.7 Select All 235
12.2.8 Find, Find Next 235
12.2.9 Replace 236
12.2.10 Bookmark Commands 236
12.2.11 Comment In/Out 236
12.3 Project Menu 238
12.3.1 New Project 239
12.3.2 Open Project 240
12.3.3 Reload Project 240
12.3.4 Close Project 240
12.3.5 Save Project, Save Project As 240
12.3.6 Source Control 240
– Open from Source Control 241
– Enable Source Control 242
– Get Latest Version 243
– Get 243
– Get Folders 244
– Check Out 245
– Check In 246
– Undo Check Out 247
– Add to Source Control 248
– Remove from Source Control 249
– Share from Source Control 249
– Show History 250
– Show Differences 251
– Show Properties 252
– Refresh Status 253
– Source Control Manager 253
– Change Source Control 253
12.3.7 Add Files to Project 254
12.3.8 Add Global Resource to Project 254
12.3.9 Add URL to Project 254
12.3.10 Add Active File to Project 255
12.3.11 Add Active And Related Files to Project 255
12.3.12 Add Project Folder to Project 255
12.3.13 Add External Folder to Project 255
12.3.14 Add External Web Folder to Project 258
12.3.15 Properties 261
12.3.16 Most Recently Used Projects 263
12.4 XML Menu 264
12.4.1 Table 264
– Display as Table 264
– Ascending Sort 265
– Descending Sort 265
12.4.2 Check Well-Formedness 266

12.4.3 Validate XML 267
12.4.4 Update Entry Helpers 270
12.5 DTD/Schema Menu 271
12.5.1 Assign DTD 271
12.5.2 Assign Schema 271
12.5.3 Go to DTD 271
12.5.4 Go to Schema 272
12.5.5 Go to Definition 272
12.5.6 Generate XML from DB, Excel, EDI with MapForce 272
12.5.7 Design HTML/PDF/Word Output with StyleVision 272
12.5.8 Generate Sample XML File 272
12.5.9 Flush Memory Cache 274
12.6 Schema Design Menu 275
12.6.1 Schema Settings 275
12.6.2 Configure View 276
12.6.3 Zoom 279
12.6.4 Display All Globals 280
12.6.5 Display Diagram 280
12.7 XSL/XQuery Menu 281
12.7.1 XSL Transformation 281
12.7.2 XSL-FO Transformation 282
12.7.3 XSL Parameters / XQuery Variables 283
12.7.4 XQuery Execution 286
12.7.5 Assign XSL 287
12.7.6 Assign XSL-FO 287
12.7.7 Assign Sample XML File 287
12.7.8 Go to XSL 288
12.8 Authentic Menu 289
12.8.1 New Document 289
12.8.2 Edit Database Data 290
12.8.3 Assign/Edit a StyleVision Stylesheet 291
12.8.4 Select New Row with XML Data for Editing 291
12.8.5 Define XML Entities 292
12.8.6 Hide Markup, Show Small/Large/Mixed Markup 294
12.8.7 Append/Insert/Duplicate/Delete Row 294
12.8.8 Move Row Up/Down 295
12.9 View Menu 296
12.9.1 Text View 296
12.9.2 Enhanced Grid View 296
12.9.3 Schema Design View 296
12.9.4 Authentic View 297
12.9.5 Browser View 297
12.9.6 Expand 297
12.9.7 Collapse 297
12.9.8 Expand Fully 298

12.9.9 Collapse Unselected ... 298
12.9.10 Optimal Widths ... 298
12.9.11 Word Wrap ... 298
12.9.12 Go to Line/Character ... 298
12.9.13 Go to File ... 299
12.9.14 Text View Settings ... 299
12.10 Browser Menu ... 301
12.10.1 Back ... 301
12.10.2 Forward ... 301
12.10.3 Stop ... 301
12.10.4 Refresh ... 301
12.10.5 Fonts ... 301
12.10.6 Separate Window ... 301
12.11 Tools Menu ... 303
12.11.1 Global Resources ... 303
12.11.2 Active Configuration ... 304
12.11.3 Customize ... 304
– Commands ... 304
– Toolbars ... 305
– Keyboard ... 306
– Menu ... 310
– Options ... 312
12.11.4 Restore Toolbars and Windows ... 312
12.11.5 Options ... 313
– File ... 313
– File Types ... 314
– Editing ... 315
– View ... 316
– Grid Fonts ... 316
– Schema Fonts ... 316
– Text Fonts ... 317
– Colors ... 318
– Encoding ... 319
– XSL ... 319
– Source Control ... 321
12.12 Window Menu ... 322
12.12.1 Cascade ... 322
12.12.2 Tile Horizontally ... 322
12.12.3 Tile Vertically ... 322
12.12.4 Project Window ... 322
12.12.5 Info Window ... 322
12.12.6 Entry Helpers ... 322
12.12.7 Output Windows ... 322
12.12.8 Project and Entry Helpers ... 323
12.12.9 All On/Off ... 323

12.12.10 Currently Open Window List....323
12.13 Help Menu....324
12.13.1 Table of Contents....324
12.13.2 Index....324
12.13.3 Search....324
12.13.4 Keyboard Map....325
12.13.5 Activation, Order Form, Registration, Updates....325
12.13.6 Support Center, FAQ, Downloads....326
12.13.7 On the Internet....327
12.13.8 About....327
12.14 Command Line....328

Appendices 330

1 Engine Information 331

1.1 XSLT 1.0 Engine: Implementation Information....332
1.2 XSLT 2.0 Engine: Implementation Information....334
1.2.1 General Information....334
1.2.2 XSLT 2.0 Elements and Functions....336
1.3 XQuery 1.0 Engine: Implementation Information....337
1.4 XPath 2.0 and XQuery 1.0 Functions....340
1.4.1 General Information....340
1.4.2 Functions Support....341
1.5 Extensions....344
1.5.1 Altova Extension Functions....344
– General Functions....344

2 Technical Data 348

2.1 OS and Memory Requirements....349
2.2 Altova XML Parser....350
2.3 Altova XSLT and XQuery Engines....351
2.4 Unicode Support....352
2.4.1 Windows XP....352
2.4.2 Right-to-Left Writing Systems....353
2.5 Internet Usage....354

3 License Information 355

3.1 Electronic Software Distribution....356

3.2 Software Activation and License Metering 357
3.3 Intellectual Property Rights 358
3.4 Altova End User License Agreement 359

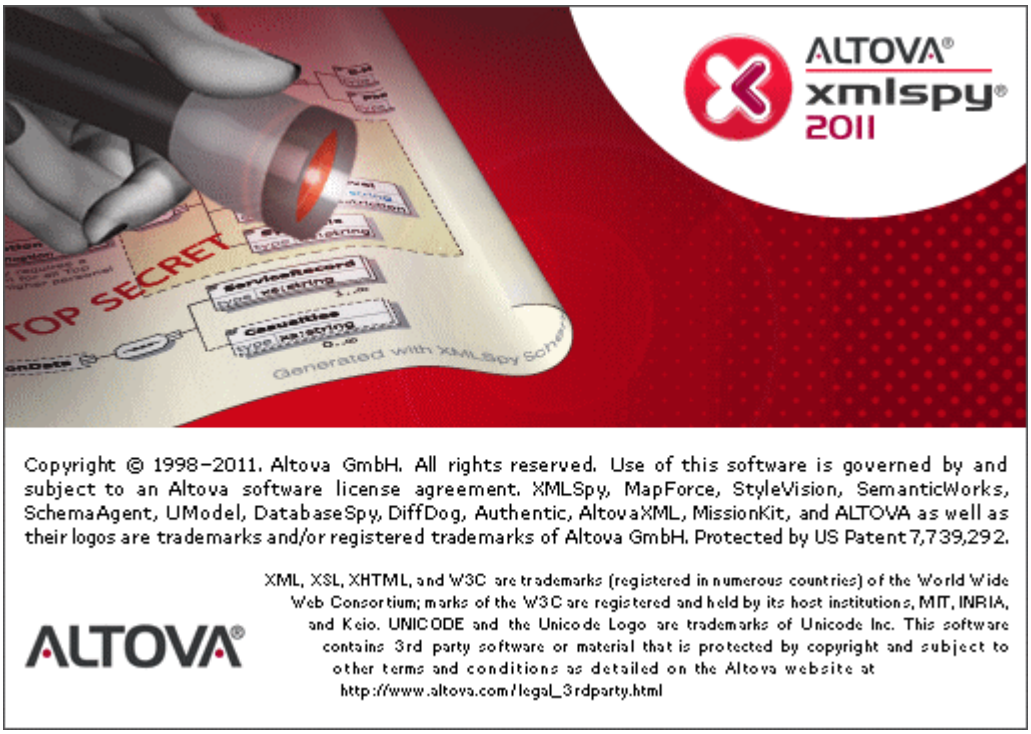
Index **371**

Altova XMLSpy 2011

Welcome to XMLSpy

Welcome to XMLSpy

Altova XMLSpy® 2011 Standard Edition is an entry-level XML editor for viewing, validating and editing XML documents. It is the perfect tool for users who need to view XML, DTD, XML Schema, XSLT and XQuery files and perform light-weight editing tasks.



Copyright © 1998-2011. Altova GmbH. All rights reserved. Use of this software is governed by and subject to an Altova software license agreement. XMLSpy, MapForce, StyleVision, SemanticWorks, SchemaAgent, UModel, DatabaseSpy, DiffDog, Authentic, AltovaXML, MissionKit, and ALTOVA as well as their logos are trademarks and/or registered trademarks of Altova GmbH. Protected by US Patent 7,739,292.

ALTOVA®

XML, XSL, XHTML, and W3C are trademarks (registered in numerous countries) of the World Wide Web Consortium; marks of the W3C are registered and held by its host institutions, MIT, INRIA, and Keio. UNICODE and the Unicode Logo are trademarks of Unicode Inc. This software contains 3rd party software or material that is protected by copyright and subject to other terms and conditions as detailed on the Altova website at http://www.altova.com/legal_3rdparty.html

This documentation is organized into the following sections:

- User Manual
- Appendices

It is available in the following formats:

- As the built-in Help system of XMLSpy (Help menu or **F1**)
- In HTML and PDF formats, and for purchase as a book, these formats being available via the Altova website

Last updated: 07/08/2011

Altova XMLSpy 2011

User Manual

User Manual

The User Manual part of this documentation contains all the information you need to get started using XMLSpy and to learn the various XMLSpy features. It is organized into four broad parts: (i) an Introduction; (ii) a Tutorial; (iii) a Working With part; and (iv) a User Reference.

We suggest that you start by reading the Introduction in order to get a feel for the GUI and to understand key application settings. If you are new to XML, the XMLSpy tutorial will help you not only to get to know XMLSpy but also to easily create and use your first XML documents. After that, you should read the Editing Views section and then the sections in the Working With part that are of most interest to you. The User Reference part can be used thereafter as a reference.

Introduction
The introduction describes the GUI, important settings in the Options dialog, and the application environment.

Tutorial
The XMLSpy tutorial helps you get started and shows you how to use the most common XMLSpy features.

Working With
The Working With part of the documentation describes the functionality that XMLSpy offers for working with various XML and XML-related technologies. It starts with a description of XMLSpy 's multiple editing views. The sections that immediately follow explain the functionality for each technology separately. The Working With part concludes with a series of descriptions of application-wide XMLSpy features, such as Altova Global Resources and projects. The sections in the Working With part are:

- Editing Views: Describes the various editing views in XMLSpy
- XML: Explains the various features available for working with XML documents in XMLSpy
- DTDs and XML Schemas: Shows how schemas (DTDs and XML Schemas) can be edited and leveraged in XMLSpy
- XSLT and XQuery: Presents the range of features available for XSLT and XQuery development
- Authentic: Explains the utility of Altova's graphical XML editor, and shows how it is used
- HTML, CSS, and JSON: Explores XMLSpy's support for HTML and CSS
- Global Resources: Describes a unique Altova mechanism that can be used to boost development efficiency when using Altova products, especially as a suite of products
- Projects: Explains XMLSpy's project mechanism, which enables increased efficiency and additional development options

User Reference
The User Reference part is organized according to the menus in XMLSpy and describes each menu command in detail.

File paths in Windows XP, Windows Vista, and Windows 7
File paths given in this documentation will not be the same for all operating systems. You should note the following correspondences:

© 2011 Altova GmbH

- *(My) Documents folder:* The My Documents folder of Windows XP is the Documents folder of Windows Vista and Windows 7. It is located by default at the following respective locations. Example files are usually located in a sub-folder of the (My) Documents folder.

Windows XP	C:/Documents and Settings/<username>/My Documents
Windows Vista, Windows 7	C:/Users/<username>/Documents

- *Application folder:* The Application folder is the folder where your Altova application is located. The path to the Application folder is, by default, the following.

Windows XP	C:/Program Files/Altova
Windows Vista, Windows 7	C:/Program Files/Altova
32-bit package on 64-bit Windows OS (XP, Vista, 7)	C:/Program Files (x86)/Altova

1 Introduction

This introduction describes:

- The application GUI, and
- The application environment.

The GUI section starts off by presenting an overview of the GUI and then goes on to describe each of the the various GUI windows in detail. It also shows you how to re-size, move, and otherwise work with the windows and the GUI.

The Application Environment section points out the various settings that control how files are displayed and can be edited. It also explains how and where you can customize your application. In this section, you will learn where important example and tutorial files have been installed on your machine, and, later in the section, you are linked to the Altova website, where you can explore the feature matrix of your application, learn about the multiple formats of your user manual, find out about the various support options available to you, and discover other products in the Altova range.

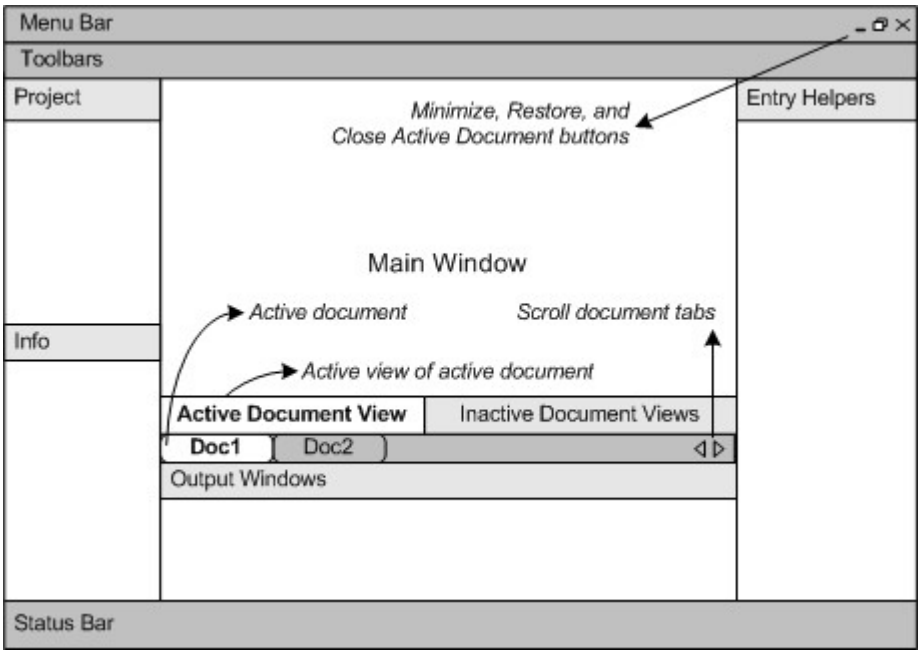
© 2011 Altova GmbH

1.1 The Graphical User Interface (GUI)

The Graphical User Interface (GUI) consists of a Main Window and several sidebars (*see illustration below*). By default, the sidebars are located around the Main Window and are organized into the following groups:

- Project Window
- Info Window
- Entry Helpers: Elements, Attributes, Entities, etc (depending upon the type of document currently active)
- Output Windows: Messages

The main window and sidebars are described in the sub-sections of this section.



The GUI also contains a menu bar, status bar, and toolbars, all of which are described in a subsection of this section.

Switching on and off the display of sidebars
Sidebar groups (Project Window, Info Window, Entry Helpers, Output Windows) can be displayed or hidden by toggling them on and off via the commands in the **Window** menu. A displayed sidebar (or a group of tabbed sidebars) can also be hidden by right-clicking the title bar of the displayed sidebar (or tabbed-sidebar group) and selecting the command **Hide**.

Floating and docking the sidebars
An individual sidebar window can either float free of the GUI or be docked within the GUI. When a floating window is docked, it docks into its last docked position. A window can also be docked as a tab within another window.

A window can be made to float or dock using one of the following methods:

- Right-click the title bar of a window and choose the required command (**Floating** or **Docking**).

- Double-click the title bar of the window. If docked, the window will now float. If floating, the window will now dock in the last position in which it was docked.
- Drag the window (using its title bar as a handle) out of its docked position so that it floats. Drag a floating window (by its title bar) to the location where it is to be docked. Two sets of blue arrows appear. The outer set of four arrows enables docking relative to the application window (along the top, right, bottom, or left edge of the GUI). The inner set of arrows enables docking relative to the window over which the cursor is currently placed. Dropping a dragged window on the button in the center of the inner set of arrows (or on the title bar of a window) docks the dragged window as a tabbed window within the window in which it is dropped.

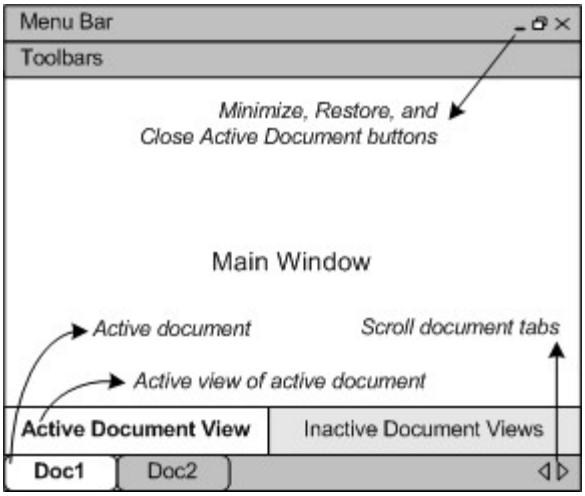
To float a tabbed window, double-click its tab. To drag a tabbed window out of a group of tabbed windows, drag its tab.

Auto-hiding sidebars
The Auto-hide feature enables you to minimize docked sidebars to buttons along the edges of the application window. This gives you more screen space for the Main Window and other sidebars. Scrolling over a minimized sidebar rolls out that sidebar.

To auto-hide and restore sidebars click the drawing pin icon in the title bar of the sidebar window (or right-click the title bar and select **Auto-Hide**).

1.1.1 Main Window

The Main Window (*screenshot below*) is where you view and edit documents.



Files in the Main Window

- Any number of files can be opened and edited at once.
- Each open document has its own window and a tab with its name at the bottom of the Main Window. To make an open document active, click its tab.
- If several files are open, some document tabs might not be visible for lack of space in the document tabs bar. Document tabs can be brought into view by: (i) using the scroll buttons at the right of the document tab bar, or (ii) selecting the required document from the list at the bottom of the **Window** menu.
- When the active document is maximized, its **Minimize**, **Restore**, and **Close** buttons are located at the right side of the Menu Bar. When a document is cascaded, tiled, or

 © 2011 Altova GmbH

minimized, the **Maximize, Restore**, and **Close** buttons are located in the title bar of the document window.
- When you maximize one file, all open files are maximized.
- Open files can be cascaded or tiled using commands in the **Window** menu.
- You can also activate open files in the sequence in which they were opened by using **Ctrl+Tab** or **Ctrl+F6**.
- Right-clicking a document tab opens a context-menu with a selection of **File** commands, such as **Print** and **Close**.

Views in the Main Window
The active document can be displayed and edited in multiple views. The available views are displayed in a bar above the document tabs (*see illustration above*), and the active view is highlighted. A view is selected by clicking the required view button or by using the commands in the **View** menu.

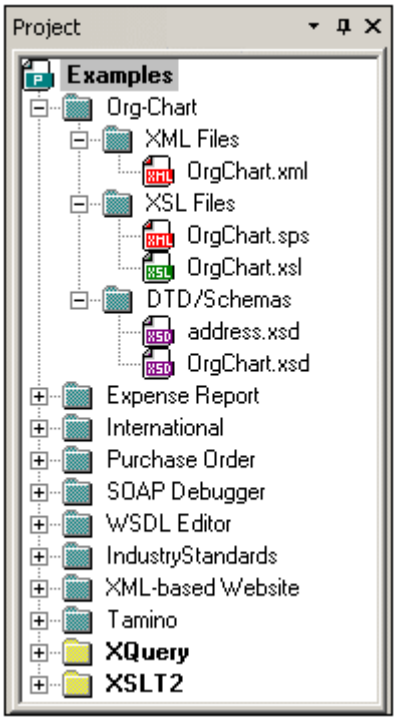
The available views are either editing or browser views:

- **Text View:** An editing view with syntax-coloring for source-level work.
- **Grid View:** For structured editing. The document is displayed as a structured grid, which can be manipulated graphically. This view also contains an embedded**Table View**, which shows repeating elements in a tabular format. In Standard Edition, Grid View is read-only. Editing is available in Enterprise and Professional Editions.
- **Schema View:** For viewing and editing XML Schemas. In Standard Edition, Schema View is read-only. Editing is available in Enterprise and Professional Editions.
- **Authentic View:** For editing XML documents that are based on StyleVision Power Stylesheets
- **Browser View:** An integrated browser view that supports both CSS and XSL stylesheets.

Note: The default view for individual file extensions can be customized in the **Tools | Options** dialog: in the Default View pane of the File Types tab.

1.1.2 Project Window

A project is a collection of files that are related to each other in some way you determine. For example, in the screenshot below, a project named `Examples` collects the files for various examples in separate example folders, each of which can be further organized into sub-folders. Within the `Examples` project, for instance, the `OrgChart` example folder is further organized into sub-folders for XML, XSL, and Schema files.



Projects thus enable you to gather together files that are used together and to access them quicker. Additionally, you can define schemas and XSLT files for individual folders, thus enabling the batch processing of files in a folder.

Project operations

Commands for folder operations are available in the **Project** menu, and some commands are available in the context menus of the project and its folders (right-click to access).

- One project is open at a time in the Project Window. When a new project is created or an existing project opened, it replaces the project currently open in the Project Window.
- After changes have been made to a project, the project must be saved (by clicking the **Project | Save Project** command).
- The project has a tree structure composed of folders, files, and other resources. Such resources can be added at any level and to an unlimited depth.
- Project folders are *semantic* folders that represent a logical grouping of files. They **do not need** to correspond to any hierarchical organization of files on your hard disk.
- Folders can correspond to, and have a direct relationship to, physical directories on your file system. We call such folders *external folders*, and they are indicated in the Project Window by a yellow folder icon (as opposed to normal project folders, which are green). External project folders must be explicitly synchronized by using the **Refresh** command.
- A folder can contain an arbitrary mix of file-types. Alternatively, you can define file-type extensions for each folder (in the Properties dialog of that folder) to keep common files in one convenient place. When a file is added to the parent folder, it is automatically added to the sub-folder that has been defined to contain files of that file extension.
- In the Project Window, a folder can be dragged to another folder or to another location within the same folder, while a file can be dragged to another folder but cannot be moved within the same folder (within which files are arranged alphabetically). Additionally, files and folders can be dragged from Windows File Explorer to the Project Window.
- Each folder has a set of properties that are defined in the Properties dialog of that

 © 2011 Altova GmbH

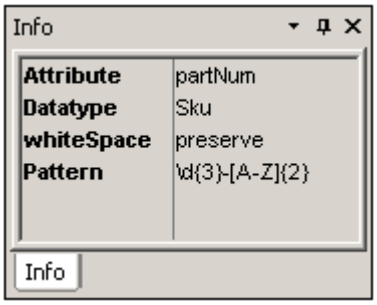
folder. These properties include file extensions for the folder, the schema by which to validate XML files, the XSLT file with which to transform XML files, etc.
- Batch processing of files in a folder is done by right-clicking the folder and selecting the relevant command from the context menu.

For a more detailed description of projects, see the section Projects.

Note: The display of the Project Window can be turned on and off in the **Window** menu.

1.1.3 Info Window

The Info Window (*screenshot below*) shows information about the element or attribute in which the cursor is currently positioned.



Note: The display of the Info Window can be turned on and off in the **Window** menu.

1.1.4 Entry Helpers

Entry helpers are an intelligent editing feature that helps you to create valid XML documents quickly. When you are editing a document, the entry helpers display structural editing options according to the current location of the cursor. The entry helpers get the required information from the underlying DTD, XML Schema, and/or StyleVision Power Stylesheet. If, for example, you are editing an XML data document, then elements, attributes, and entities that can be inserted at the current cursor position are displayed in the relevant entry helpers windows.

The entry helpers that are available depend upon:

1. *The kind of document being edited*. For example, XML documents will have different entry helpers than XQuery documents: elements, attributes, and entities entry helpers in the former case, but XQuery keywords, variables, and functions entry helpers in the latter case. The available entry helpers for each document type are described in the description of that document type.
2. *The current view*. Since the editing mechanisms in the different views are different, the entry helpers are designed so as to be compatible with the editing mechanism in the relevant views. For example: In Text View, an element can only be inserted at the cursor location point, so the entry helper is designed to insert an element when the element is double-clicked. But in Grid View, an element can be inserted before the selected node, appended after it, or added as a child node, so the Elements entry helper in Grid View has three tabs for Insert, Append, and Add as Child, with each tab containing the elements available for that particular operation.

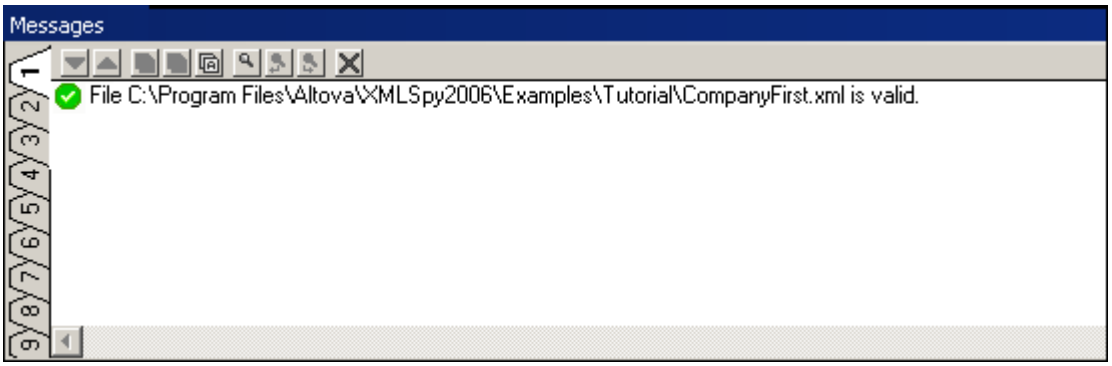
A general description of entry helpers in each type of view is given in Editing Views. Further document-type-related differences within a view are noted in the description of the individual document types, for example XML entry helpers and XQuery entry helpers.

Note the following:

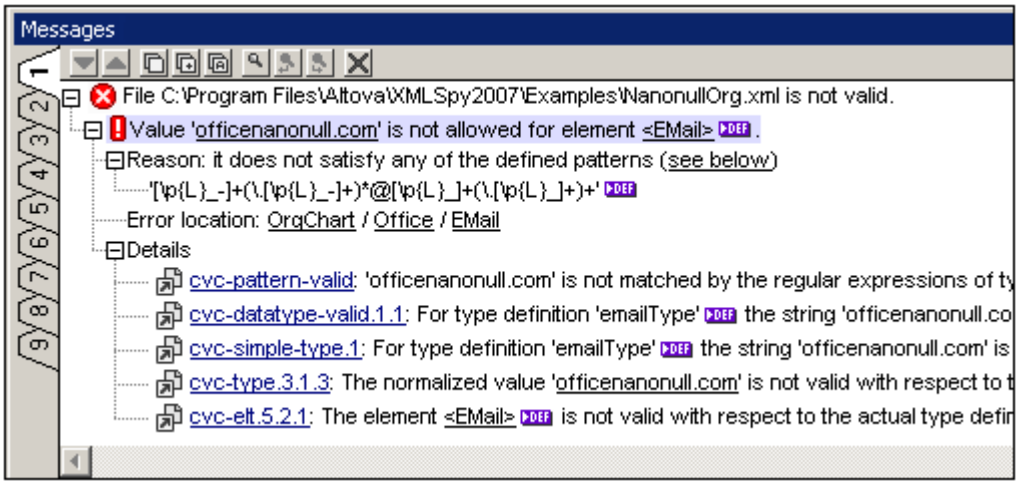
- You can turn the display of entry helpers on or off with the menu option **Window | Entry Helpers**.

1.1.5 Output Window: Messages

The Messages Window displays messages about actions carried out in XMLSpy as well as errors and other output. For example, if an XML, XML Schema, DTD, or XQuery document is validated and is valid, a successful validation message (*screenshot below*) is displayed in the Messages Window:



Otherwise, a message that describes the error (*screenshot below*) is displayed. Notice that there are links (black link text) to nodes and node content in the XML document, as well as links (blue link text) to the sections in the relevant specification on the Internet that describe the rule in question. Clicking the purple `Def` buttons, opens the relevant schema definition in Schema View.



The Messages Window is enabled in all views, but clicking a link to content in an XML document highlights that node in the XML document in Text View.

Note: The **Validate** command (in the XML menu) is normally applied to the active document. But you can also apply the command to a file, folder, or group of files in the active project. Select the required file or folder in the Project Window (by clicking on it), and click **XML | Validate** or **F8**. Invalid files in a project will be opened and made active in the Main Window, and the File Is Invalid error message will be displayed.

 © 2011 Altova GmbH

1.1.6 Menu Bar, Toolbars, Status Bar

Menu Bar
The menu bar (*see illustration*) contains the various application menus. The following conventions apply:

- If commands in a menu are **not** applicable in a view or at a particular location in the document, they are unavailable.
- Some menu commands pop up a submenu with a list of additional options. Menu commands with submenus are indicated with a right-pointing arrowhead to the right of the command name.
- Some menu commands pop up a dialog that prompts you for further information required to carry out the selected command. Such commands are indicated with an ellipsis (...) after the name of the command.
- To access a menu command, click the menu name and then the command. If a submenu is indicated for a menu item, the submenu opens when you mouseover the menu item. Click the required sub-menu item.
- A menu can be opened from the keyboard by pressing the appropriate key combination. The key combination for each menu is **Alt+*KEY***, where ***KEY*** is the underlined letter in the menu name. For example, the key combination for the **File** menu is **Alt+F**.
- A menu command (that is, a command in a menu) can be selected by sequentially selecting (i) the menu with its key combination (see previous point), and then (ii) the key combination for the specific command (**Alt+*KEY***, where ***KEY*** is the underlined letter in the command name). For example, to create a new file (**File | New**), press **Alt+F** and then **Alt+N**.
- Some menu commands can be selected **directly** by pressing a special **shortcut** key or key combination (**Ctrl+*KEY***). Commands which have shortcuts associated with them are indicated with the shortcut key or key combination listed to the right of the command. For example, you can use the shortcut key combination **Ctrl+N** to create a new file; the shortcut key **F8** to validate an XML file. You can create your own shortcuts in the Keyboard tab of the Customize dialog (**Tools | Customize**).

Toolbars
The toolbars (*see illustration*) contain icons that are shortcuts for selecting menu commands. The name of the command appears when you place your mouse pointer over the icon. To execute the command, click the icon.

Toolbar buttons are arranged in groups. In the **Tools | Customize | Toolbars** dialog, you can specify which toolbar groups are to be displayed. These settings apply to the current view. To make a setting for another view, change to that view and then make the setting in the **Tools | Customize | Toolbars**. In the GUI, you can also drag toolbar groups by their handles (or title bars) to alternative locations on the screen. Double-clicking the handle causes the toolbar to undock and to float; double-clicking its title bar causes the toolbar to dock at its previous location.

Status Bar
The Status Bar is located at the bottom of the application window (*see illustration*) and displays (i) status information about the loading of files, and (ii) information about menu commands and command shortcuts in the toolbars when the mouse cursor is placed over these.

1.2 The Application Environment

In this section we describe various aspects of the application that are important for getting started. Reading through this section will help you familiarize yourself with XMLSpy and get you off to a confident start. It contains important information about settings and customization, which you should read for a general idea of the range of settings and customization options available to you and how these can be changed.

This section is organized as follows:

- *Settings and Customization:* Describes how and where important settings and customization options can be defined.
- *Tutorials, Projects, Examples:* notes the location of the various non-program files included in the application package.
- *Product features and documentation, and Altova products:* provides links to the Altova website, where you can find information about product features, additional Help formats, and other Altova products.

1.2.1 Settings and Customization

In XMLSpy, there are several settings and customization options that you can select. In this section, we point you to these options. This section is organized into the following parts.

- Settings
- Customization

Settings
Several important XMLSpy settings are defined in different tabs in the Options dialog. You should look through the various options to familiarize yourself with what's available.

Customization
You can also customize various aspects of XMLSpy, including the appearance of the GUI. These customization options are available in the Customize dialog (accessed via the menu command **Tools | Customize**).

The various customization options are described in the **User Reference** section.

1.2.2 Tutorials, Projects, Examples

The XMLSpy installation package contains tutorials, projects, and example files.

Location of tutorials, projects, and example files
The XMLSpy tutorials, projects, and example files are installed in the folder:

```
C:\Documents and Settings\<username>\My Documents\
Altova\XMLSpy2011\Examples\
```

The `My Documents\Altova\XMLSpy2011` folder will be installed for each user registered on a PC within that user's `<username>` folder. Under this installation system, therefore, each user will have his or her own `Examples` folder in a separate working area.

Note about the master XMLSpy folder

 © 2011 Altova GmbH

When XMLSpy is installed on a machine, a master `Altova\XMLSpy2011` folder is created at the following folder location:

`C:\Documents and Settings\`**`All Users\Application Data\`**

When a user on that machine starts XMLSpy for the first time, XMLSpy creates a copy of this master folder in the user's `<username>\My Documents\` folder. It is therefore important not to use the master folder when working with tutorial or example files, otherwise these edited files will be copied to the user folder of a user subsequently using XMLSpy for the first time.

Location of tutorial, project, and examples files
All tutorial, project, and example files are located in the `Examples` folder. Specific locations are as follows:

- *XMLSpy tutorial:* `Tutorial` folder.
- *Authentic View tutorial:* `Examples` folder.
- *Project file:* The Examples project with which XMLSpy opens is defined in the file `Examples.spp`, which is located in the `Examples` folder.
- *Examples files:* are in the `Examples` folder and in sub-folder of the `Examples` folder.

1.2.3 XMLSpy Features and Help, and Altova Products

The Altova website, www.altova.com, has a wealth of XMLSpy-related information and resources. Among these are the following.

XMLSpy feature listing
The Altova website carries an up-to-date list of XMLSpy features, which also compares the support of various features across XMLSpy editions (Enterprise, Professional, and Standard). On the website, you can also obtain a listing of features that are new since any previous release.

XMLSpy Help
This documentation is the Altova-supplied Help for XMLSpy. It is available as the built-in Help system of XMLSpy, which is accessible via the **Help** menu or by pressing **F1**. Additionally, the user manuals for all Altova products are available in the following formats:

- Online HTML manuals, accessed via the Support page at the Altova website
- Printable PDFs, which you can download from the Altova website and print locally
- Printed books that you can buy via a link at the Altova website

Support options
If you require additional information to what is available in the user manual (this documentation) or have a query about Altova products, visit our Support Center at the Altova website. Here you will find:

- Links to our FAQ pages
- Discussion forums on Altova products and general XML subjects
- Online Support Forms that enable you to make support requests, should you have a support package. Your support request will be processed by our support team.

Altova products
For a list of all Altova products, see the Altova website.

2 XMLSpy Tutorial

This tutorial provides an overview of XML and takes you through a number of key XML tasks. In the process you will learn how to use some of XMLSpy's most powerful features.

The tutorial is divided into the following parts:

- Creating an XML Schema. You will be introduced to XML Schemas and the various views available in XMLSpy for viewing and editing XML Schemas.
- Creating an XML document. You will learn how to assign a schema for an XML document, edit an XML document in Grid View and Text View, and validate XML documents using XMLSpy's built-in validator.
- Transforming an XML file using an XSLT stylesheet. This involves assigning an XSLT file and carrying out the transformation using XMLSpy's built-in XSLT engines.
- Working with XMLSpy projects, which enable you to easily organize your XML documents.

Installation and configuration

This tutorial assumes that you have successfully installed XMLSpy on your computer and received a free evaluation key-code, or are a registered user. The evaluation version of XMLSpy is fully functional but limited to a 30-day period. You can request a regular license from our secure web server or through any one of our resellers.

Tutorial example files

The tutorial files are available in the application folder:

`C:\Documents and Settings\<username>\My Documents\Altova\XMLSpy2011\Examples\Tutorial`

The `Examples` folder contains various XML files for you to experiment with, while the `Tutorial` folder contains all the files used in this tutorial.

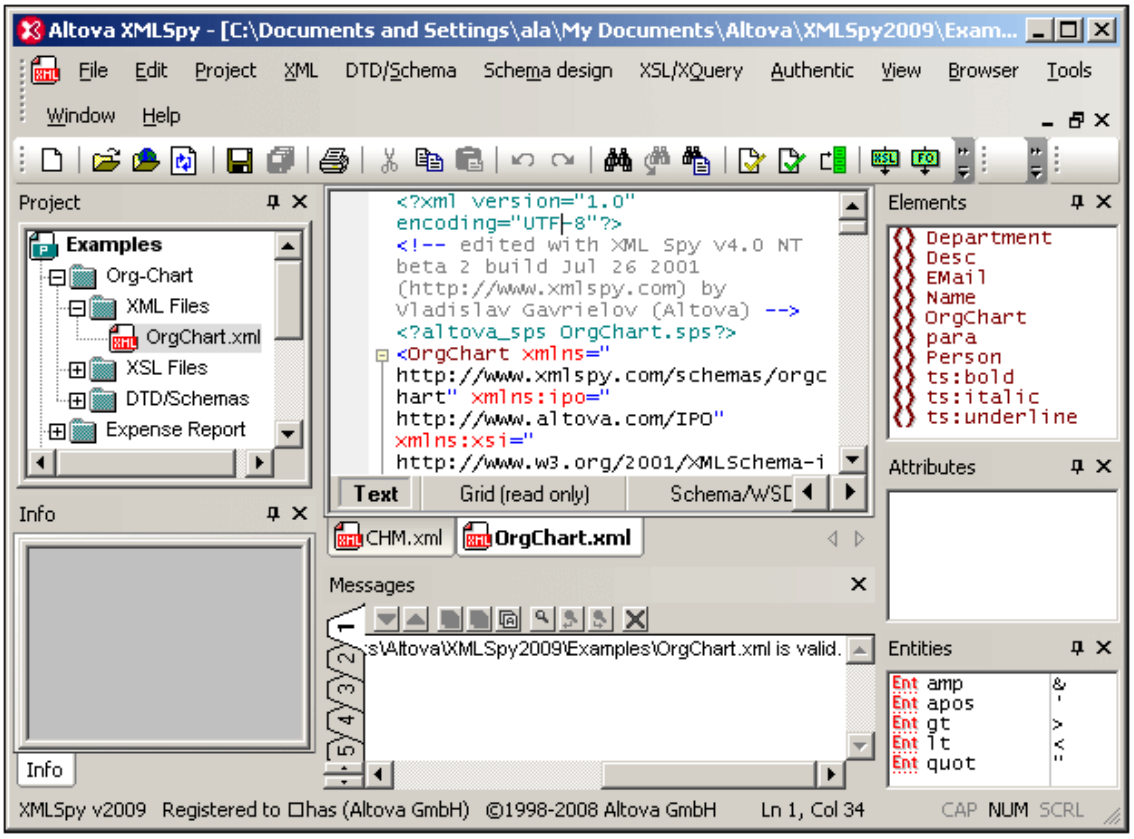
The `Template` folder in the application folder (typically in `c:\Program Files\Altova`) contains all the XML template files that are used whenever you select the menu option **File | New**. These files supply the necessary data (namespaces and XML declarations) for you to start working with the respective XML document immediately.

© 2011 Altova GmbH

2.1 XMLSpy Interface

The XMLSpy interface is structured into three vertical areas. The central area provides you with multiple views of your XML document. The areas on either side of this central area contain windows that provide information, editing help, and file management features.

- The left area consists of the **Project** and **Info** windows.
- The central area, called the **Main** window, is where you edit and view all types of XML documents. You can switch between different views: Text View, Schema View, Authentic View, and Browser View. In Standard Edition, Grid View and Schema View are read-only views; they are fully functional editing views in the Enterprise and Professional Editions. These views are described in detail in the individual sections about them in the User Manual.
- The right-hand area contains the three **Entry Helper** windows, which enable you to insert or append elements, attributes, and entities. What entries are displayed in the Entry Helper windows depends on the current selection or cursor location in the XML file.



The details of the interface are explained as we go along. Note that the interface changes dynamically according to the document that is active in the Main Window and according to the view selected.

2.2 XML Schemas

An XML Schema describes the structure of an XML document. An XML document can be validated against an XML Schema to check whether it conforms to the requirements specified in the schema. If it does, it is said to be **valid**; otherwise it is **invalid**. XML Schemas enable document designers to specify the allowed structure and content of an XML document and to check whether an XML document is valid.

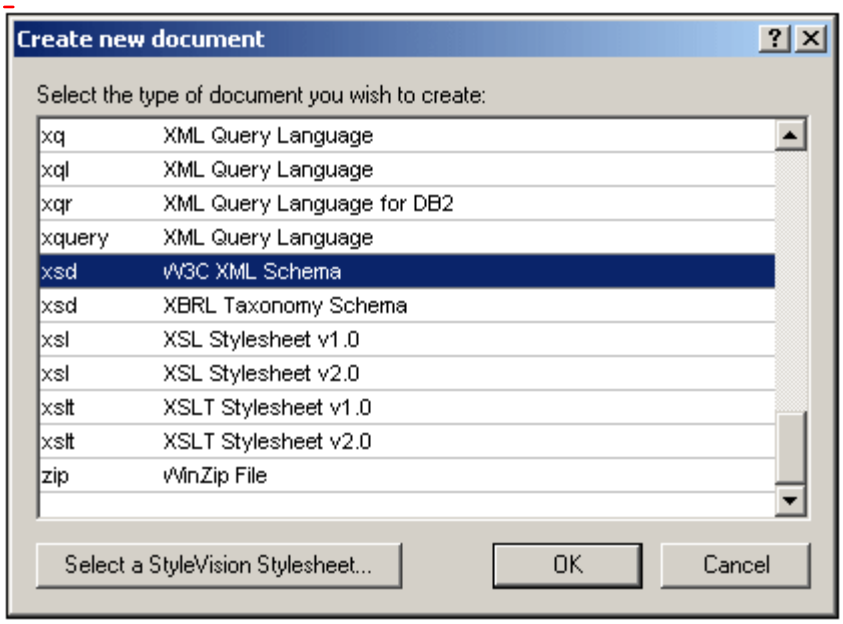
Schema editing views in XMLSpy

The structure and syntax of an XML Schema document is complex, and being an XML document itself, an XML Schema must be valid according to the rules of the XML Schema specification. In XMLSpy, Schema View enables you to easily build valid XML Schemas by using graphical drag-and-drop techniques. The XML Schema document you construct is also editable in Text View and Grid View, but is much easier to create and modify in Schema View. In the Standard Edition, XML Schema documents can be viewed in Text View, Schema View and Grid View, but can be edited only in Text View. Editing in Schema View and Grid View is available in the Enterprise and Professional editions.

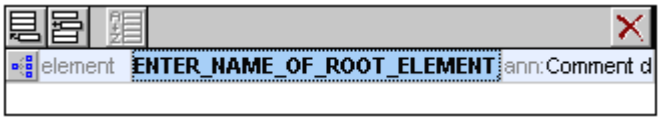
Creating a new XML Schema document

To create a new XML Schema file in XMLSpy, you must first start XMLSpy and then create a new XML Schema (. `xsd`) document. Create the document as follows:

1. Select the menu option **File | New**. The Create new document dialog opens.



2. In the dialog, select the `xsd` entry (the document description and the list in the window might vary from that in the screenshot) and confirm with **OK**. An empty schema file appears in the Main Window in Schema View (*screenshot below*). In Standard Edition, you cannot edit XML Schema documents in Schema View, so you must switch to Text View to edit the document.



 © 2011 Altova GmbH

3. The schema you will use for the rest of this tutorial is `AddressLast.xsd`, which is located in the `C:\Documents and Settings\<username>\My Documents\Altova\XMLSpy2011\\Examples\Tutorial` folder: Open this file, and explore it in Text View and Schema View. Note that in Standard Edition you cannot edit this document in Schema View. Schema View is a drag-and-drop editing view in the Enterprise and Professional editions, in which you can edit an overview of the schema's global components, and then edit each global component in a separate view (that component's content model view).

The XML file you create in the next part of the tutorial will be based on the `AddressLast.xsd` schema, so make sure that you do not modify the `AddressLast.xsd` schema that is supplied withy our installation.

2.3 XML Documents

In this section you will learn how to create and work with XML documents in XMLSpy. You will also learn how to use the various intelligent editing features of XMLSpy.

Objective
The objectives of this section are to learn how to do the following:

- Create a new XML document based on the `AddressLast.xsd` schema.
- Specify the type of an element so as to make an extended content model for that element available to the element during validation.
- Insert elements and attributes and enter content for them in Text View using intelligent entry helpers.
- Validate the XML document.

Commands used in this section
In this section of the tutorial, you will mostly use the Grid View and Text View, and in one section Schema View. The following commands are used:

File | New. Creates a new type of XML file.

View | Text View. Switches to Text View.

F7. Checks for well-formedness.

F8. Validates the XML document against the associated DTD or Schema.

Opens the associated DTD or XML Schema file.

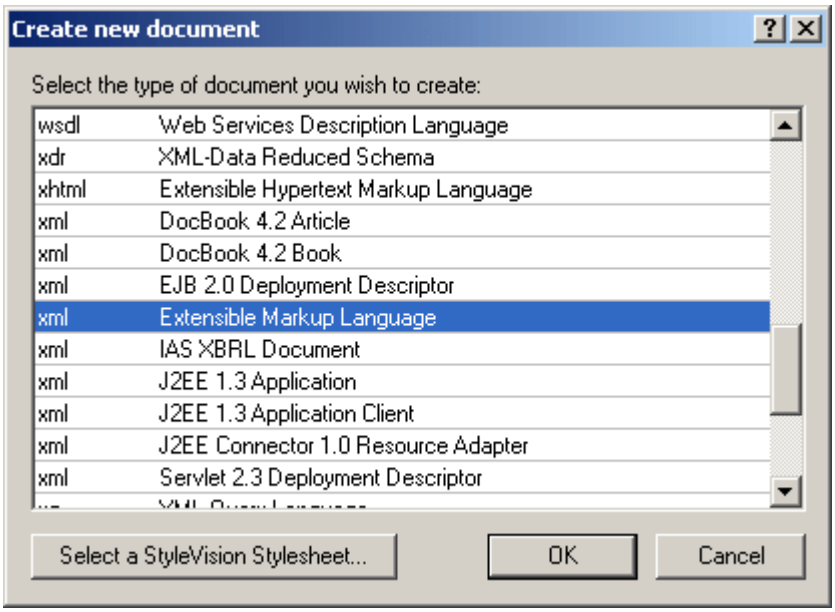
2.3.1 Creating a New XML File

When you create a new XML file in XMLSpy, you are given the option of basing it on a schema (DTD or XML Schema) or not. In this section you will create a new file that is based on the `AddressLast.xsd` schema you created earlier in the tutorial.

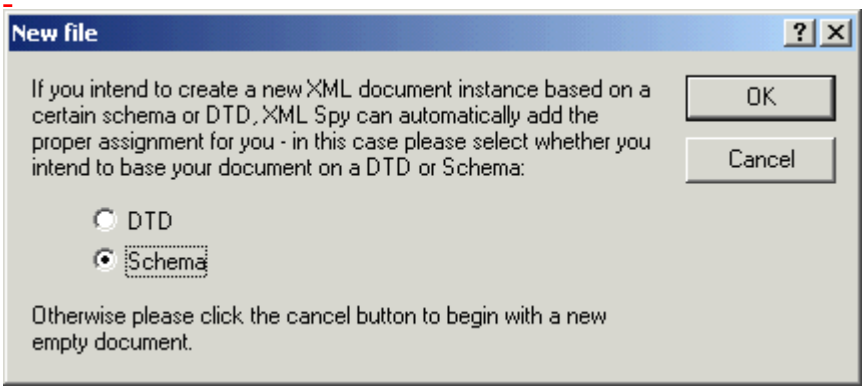
To create the new XML file:

1. Select the menu option **File | New**. The Create new document dialog opens.

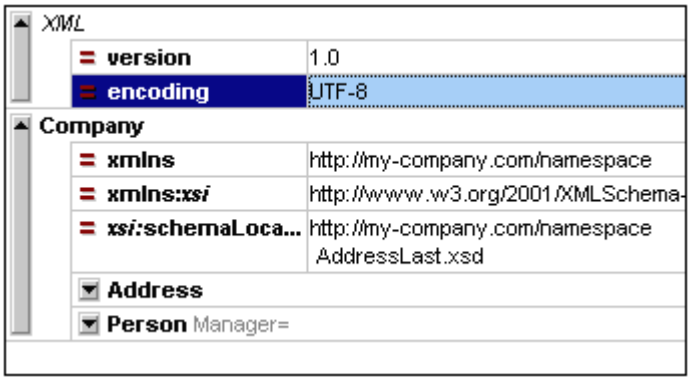
© 2011 Altova GmbH



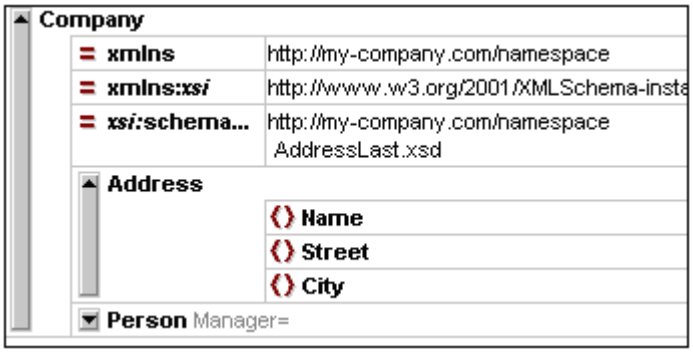
2. Select the `Extensible Markup Language` entry (or generic XML document entry) from the dialog, and confirm with **OK**. A prompt appears, asking if you want to base the XML document on a DTD or Schema.



3. Click the Schema radio button, and confirm with **OK**. A further dialog appears, asking you to select the schema file your XML document is to be based on.
4. Use the Browse or Window buttons to find the schema file. The Window button lists all files open in XMLSpy and projects. Select `AddressLast.xsd` (see Tutorial introduction for location), and confirm with **OK**. An XML document containing the main elements defined by the schema opens in the main window. Notice the structure of the document in Text View.
5. Click the **Grid** tab to select Grid View.
6. In Grid View, notice the structure of the document. Click on any element to reduce selection to that element. Your document should look something like this:

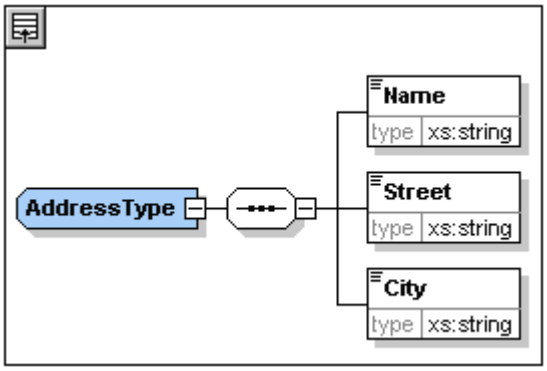


7. Click on the icon next to `Address`, to view the child elements of `Address`. Your document should look like this:



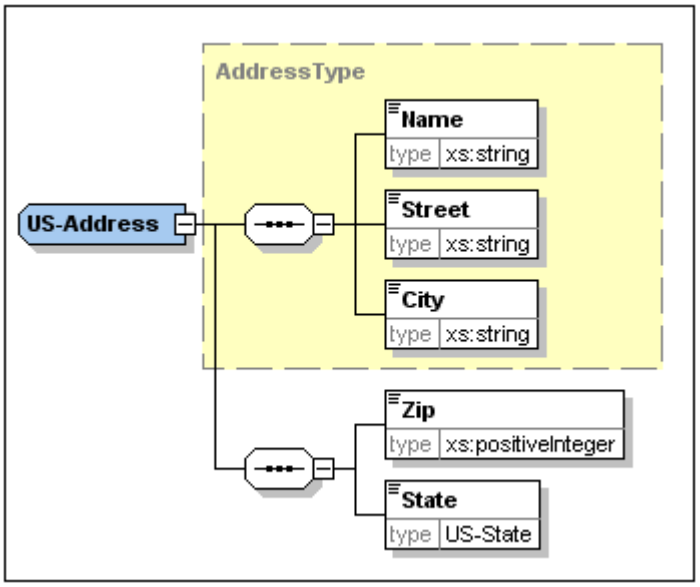
2.3.2 Specifying the Type of an Element

The child elements of `Address` are those defined for the global complex type `AddressType` (*the content model of which is defined in the XML Schema* `AddressLast.xsd` *shown in the Grid View screenshot below*).



We would, however, like to use a specific US or UK address type rather than the generic address type. You will recall that, in the `AddressLast.xsd` schema, we created global complex types for `US-Address` and `UK-Address` by extending the `AddressType` complex type. The content model of `US-Address` is shown below.

 © 2011 Altova GmbH



In the XML document, in order to specify that the `Address` element must conform to one of the extended `Address` types (`US-Address` or `UK-Address`) rather than the generic `AddressType`, we must specify the required extended complex type as an attribute of the `Address` element.

Do this as follows. In the XML document, on the `Address` element, enter an `xsi:type` attribute with a value of `US-Address` (*screenshot below*).

```
<Address xsi:type="US-Address">
  <Name>US dependency</Name>
  <Street>Noble Ave.</Street>
  <City>Dallas</City>
  <Zip>04812</Zip>
  <State>Texas</State>
</Address>
```

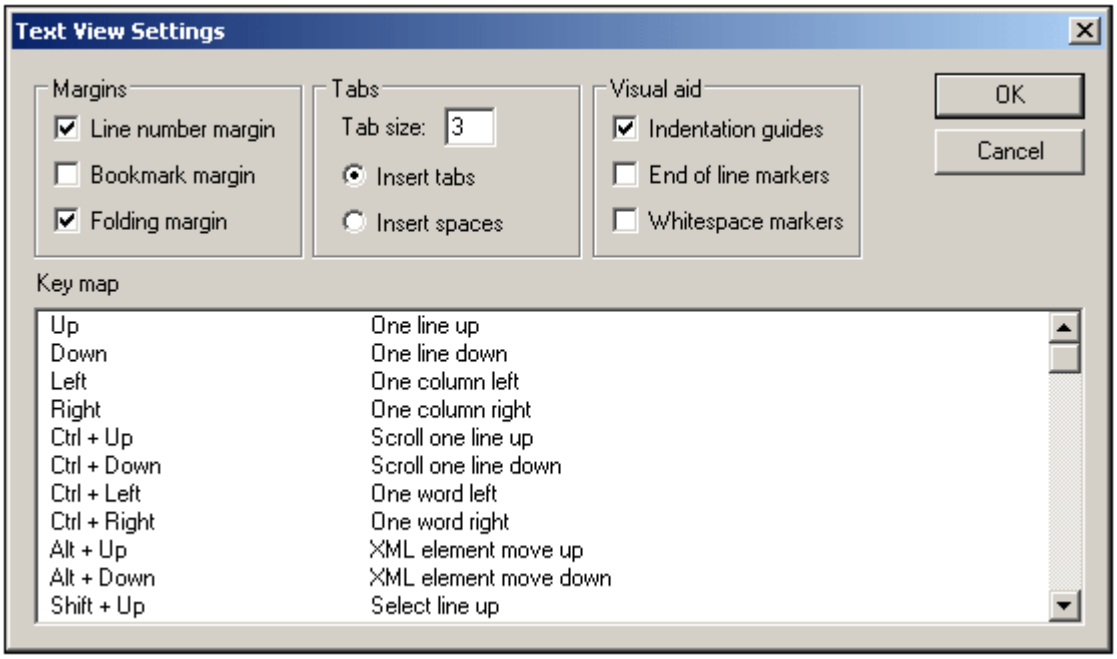
You can now enter data for the `Address` element. Enter the values shown in the screenshot above. Then delete the `Person` element (it will be added in the next section of the tutorial).

Please note: The `xsi` prefix allows you to use special XML Schema related commands in your XML document instance. Notice that the the namespace for the `xsi` prefix was automatically added to the document element when you assigned a schema to your XML file. In the above case, you have specified a type for the `Address` element. See the XML Schema specification for more information.

2.3.3 Entering Data in Text View

Text View is ideal for editing the actual data and markup of XML files because of its DTD/XML Schema-related intelligent editing features.

Structural layout features
In addition, Text View has a number of viewing and structural editing features that make editing large sections of text easy. These features can be switched on and off in the Text View Settings dialog (**View | Text View Settings**, *screenshot below*).



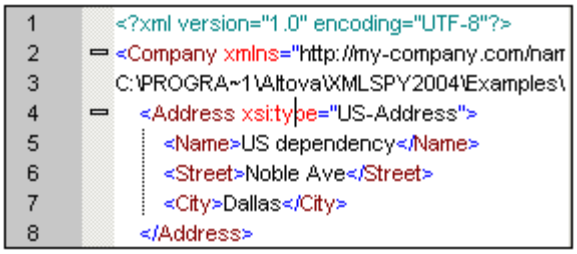
The following margins in Text View can be switched on and off:

- Line number margins
- Bookmark margins, in which individual lines can be highlighted with a marker
- Source folding margins, which contain icons to expand and collapse the display of elements

Additionally, visual aids such as indentation guides, end-of-line markers, and whitespace markers can be switched on and off, by checking and unchecking, respectively, their check boxes in the *Visual Aid* pane of the Text View Settings dialog (*see screenshot above*).

The bookmark feature is useful for setting up markers in your document. To insert a bookmark, use the command **Edit | Insert/Remove Bookmark**. Once bookmarks have been inserted you can navigate these bookmarks using commands in the **Edit** menu.

The screenshot below shows the current XML file in Text View with all structural editing features enabled. For the sake of clarity, none of the line numbers, indentation guides, etc, will be shown in Text View in rest of this tutorial. Please see the User Manual for more information on Text View.

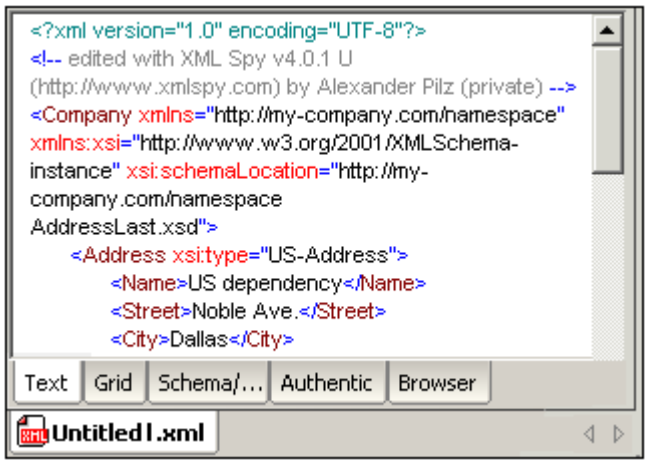


Editing in Text View
In this section, you will enter and edit data in Text View in order to become familiar with the features of Text View.

 © 2011 Altova GmbH

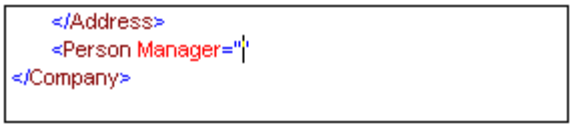
Do the following:

1. Select the menu item **View | Text view**, or click on the **Text** tab. You now see the XML document in its text form, with syntax coloring.

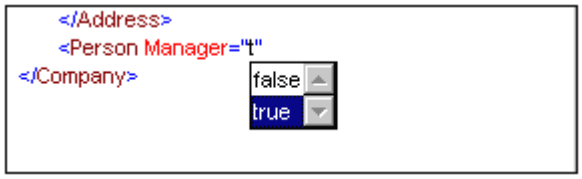


2. Place the text cursor after the end tag of the Address element, and press **Enter** to add a new line.
3. Enter the less-than angular bracket **<** at this position. A dropdown list of all elements allowed at that point (according to the schema) is displayed. Since only the `Person` element is allowed at this point, it will be the only element displayed in the list.

4. Select the `Person` entry. The `Person` element, as well as its attribute `Manager`, are inserted, with the cursor inside the value-field of the `Manager` attribute.

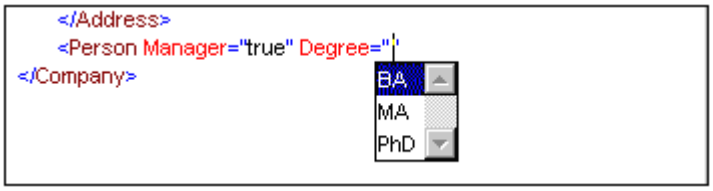


5. From the dropdown list for the `Manager` attribute, select `true`.

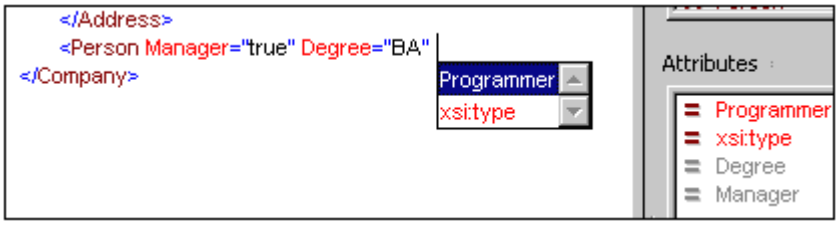


   Press **Enter** to insert the value `true` at the cursor position.
6. Move the cursor to the end of the line (using the **End** key if you like), and press the space bar. This opens a dropdown list, this time containing a list of attributes allowed at that point. Also, in the Attributes Entry Helper, the available attributes are listed in red. The Manager attribute is grayed out because it has already been used.

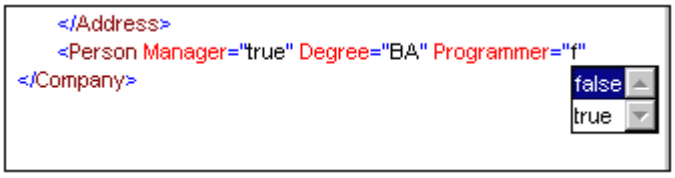
7. Select `Degree` with the Down arrow key, and press **Enter**. This opens another list box, from which you can select one of the predefined enumerations (`BA`, `MA`, or `PhD`).



8. Select `BA` with the Down arrow key and confirm with **Enter**. Then move the cursor to the end of the line (with the **End** key), and press the space bar. `Manager` and `Degree` are now grayed out in the Attributes Entry Helper.



9. Select `Programmer` with the Down arrow key and press **Enter**.



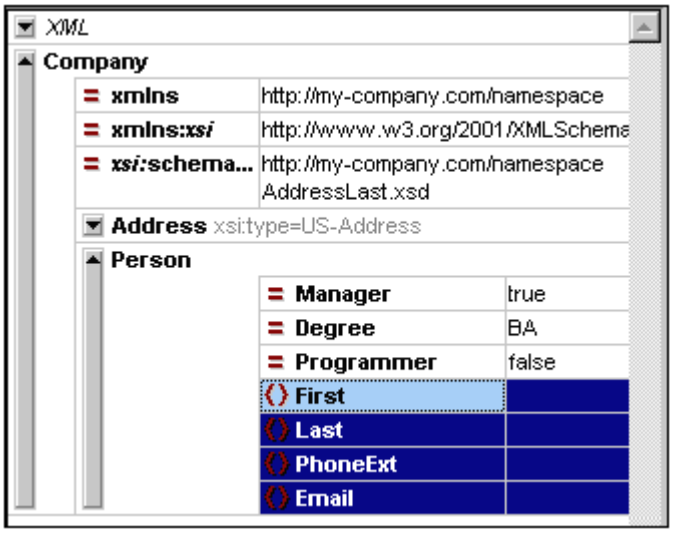
10. Enter the letter `"f"` and press **Enter**.
11. Move the cursor to the end of the line (with the **End** key), and enter the greater-than angular bracket **>**. XMLSpy automatically inserts all the required child elements of `Person`. (Note that the optional `Title` element is not inserted.) Each element has start and end tags but no content.

© 2011 Altova GmbH

```
<?xml version="1.0" encoding="UTF-8"?>
<Company xmlns="http://my-company.com/namespace"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://my-company.com/namespace
AddressLast.xsd">
    <Address xsi:type="US-Address">
        <Name>US dependency</Name>
        <Street>Noble Ave</Street>
        <City>Dallas</City>
    </Address>
    <Person Manager="true" Degree="BA" Programmer="false">
        <First></First>
        <Last></Last>
        <PhoneExt></PhoneExt>
        <Email></Email>
    </Person>
</Company>
```

You could now enter the `Person` data in Text View, but let's move to Grid View to see the flexibility of moving between views when editing a document.

Switching to Grid View
To switch to Grid View, select the menu item **View | Grid View,** or click the **Grid** tab. The newly added child elements of `Person` are highlighted.



Now let us validate the document and correct any errors that the validation finds.

2.3.4 Validating the Document

XMLSpy provides two evaluations of the XML document:

- A well-formedness check
- A validation check

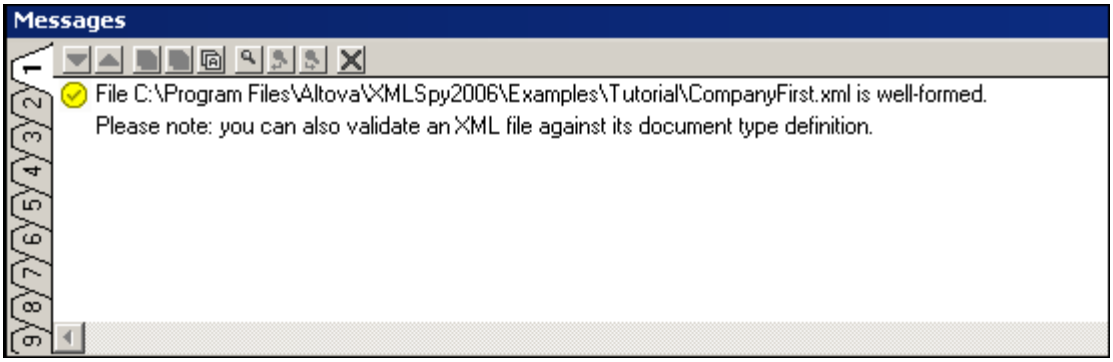
If either of these checks fails, we will have to modify the document appropriately.

Checking well-formedness
An XML document is well-formed if starting tags match closing tags, elements are nested

correctly, there are no misplaced or missing characters (such as an entity without its semi-colon delimiter), etc.

You can do a well-formedness check in any editing view. Let us select Text View. To do a well-formedness check, select the menu option **XML | Check well-formedness**, or press the **F7** key, or click . A message appears in the Messages window at the bottom of the Main Window saying the document is well-formed.

Notice that the output of the Messages window has 9 tabs. The validation output is always displayed in the active tab. Therefore, you can check well-formedness in Tab1 for one schema file and keep the result by switching to Tab2 before validating the next schema document (otherwise Tab1 is overwritten with the validation result).



Please note: This check does not check the structure of the XML file for conformance with the schema. Schema conformance is evaluated in the validity check.

Checking validity

An XML document is valid according to a schema if it conforms to the structure and content specified in that schema.

To check the validity of your XML document, first select Text View, then select the menu option **XML | Validate**, or press the **F8** key, or click . An error message appears in the Messages window saying the file is not valid. Mandatory elements are expected after the `City` element in `Address`. If you check your schema, you will see that the `US-Address` complex type (which you have set this `Address` element to be with its `xsi:type` attribute) has a content model in which the City element must be followed by a `Zip` element and a `State` element.

Fixing the invalid document

The point at which the document becomes invalid is highlighted, in this case the `City` element.

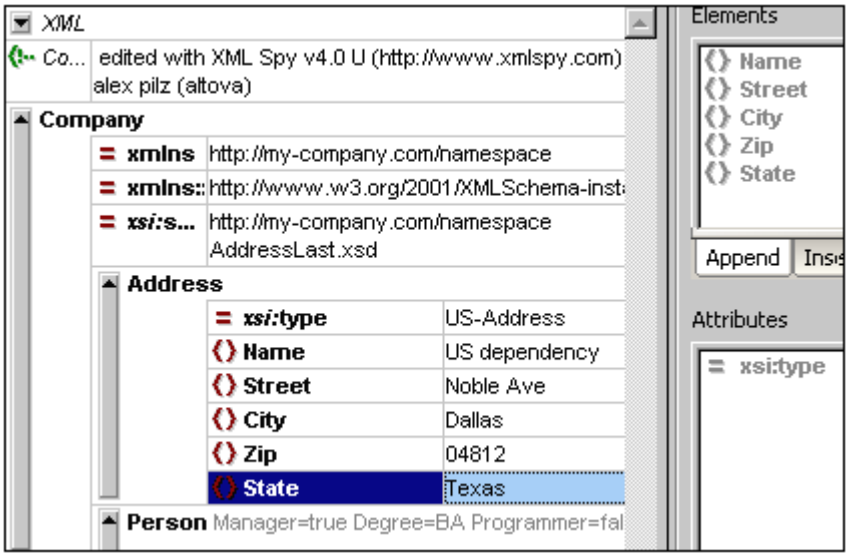
Now look at the Elements Entry Helper (at top right). Notice that the `Zip` element is prefixed with an exclamation mark, which indicates that the element is mandatory in the current context.

To fix the validation error:

1. Place the cursor after the `City` element and, in the Elements Entry Helper, double-click the `Zip` element.
2. Ensure the cursor is between the start and end tags of the `Zip` element, and enter the Zip Code of the State (`04812`), then confirm with **Enter**. The Elements Entry Helper now shows that the `State` element is mandatory (it is prefixed with an exclamation mark).

 © 2011 Altova GmbH

3. Place the cursor after the Zip element, and in the Elements Entry Helper, double-click the `State` element. Then enter the name of the state (`Texas`). Confirm with **Enter**. The Elements Entry Helper now contains only grayed-out elements. This shows that there are no more required child elements of `Address`. Switch to Grid View to view your changes (*screenshot below*).



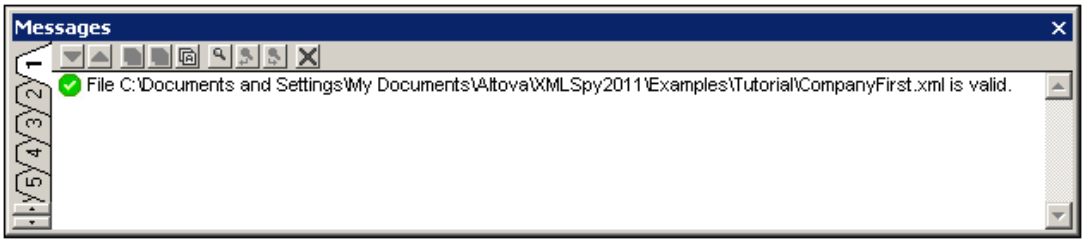
Completing the document and revalidating
Let us now complete the document (enter data for the `Person` element) before revalidating.

Do the following:

1. In the element `First`, enter a first name (say `Fred`). Then press **Enter**.
2. In the same way enter data for all the child elements of `Person`, that is, for `Last`, `PhoneExt`, and `Email`. Note that the value of `PhoneExt` must be an integer with a maximum value of 99 (since this is the range of allowed `PhoneExt` values you defined in your schema). Your XML document should then look something like this in Text View:

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- edited with XML Spy v4.0.1 U
(http://www.xmlspy.com) by Alexander Pilz
(private) -->
<Company xmlns="http://my-company.com/namespace
" xmlns:xsi="
http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="
http://my-company.com/namespace
AddressLast.xsd">
  <Address xsi:type="US-Address">
    <Name>US dependency</Name>
    <Street>Noble Ave.</Street>
    <City>Dallas</City>
    <Zip>04812</Zip>
    <State>Texas</State>
  </Address>
  <Person Manager="true" Degree="BA" Programmer
="false">
    <First>Fred</First>
    <Last>Smith</Last>
    <PhoneExt>22</PhoneExt>
    <Email>Smith@work.com</Email>
  </Person>
</Company>
```

3. Click again to check if the document is valid. A message appears in the Messages window stating that the file is valid. The XML document is now valid against its schema.



4. Select the menu option **File | Save** and give your XML document a suitable name (for example `CompanyFirst.xml`). Note that the finished tutorial file `CompanyFirst.xml` is in the `Tutorial` folder, so you may need to rename it before you give that name to the file you have created.

Please note: An XML document does not have to be valid in order to save it. Saving an invalid document causes a prompt to appear warning you that you are about to save an invalid document. You can select **Save anyway**, if you wish to save the document in its current invalid state.

2.3.5 Adding Elements and Attributes

At this point, there is only one `Person` element in the document.

To add a new `Person` element:

1. Place the cursor after the already created `Person` element.
2. Press Enter. This creates a new line, with the cursor positioned at the start of the new line. Notice that the `Person` element is now available in the Elements Entry Helper.
3. Double-click the `Person` element in the Elements Entry Helper. A new `Person` element with all mandatory child elements is appended (*screenshot below*). Notice that the optional `Title` child element of `Person` is not inserted.

```
<Person Manager="true" Degree="BA" Programmer="false">
  <First>Fred</First>
  <Last>Smith</Last>
  <PhoneExt>22</PhoneExt>
  <Email>Smith@work.com</Email>
</Person>
<Person Manager="">
  <First></First>
  <Last></Last>
  <PhoneExt></PhoneExt>
  <Email></Email>
</Person>
```

4. Place the cursor before the closing angular bracket of the opening tag. Then, in the **Append** tab of the Attributes Entry Helper, double-click the `Programmer` entry. This inserts an empty `Programmer` attribute after the `Manager` attribute. The `Programmer` attribute is now grayed out in the Attributes Entry Helper.

Select the menu option **File | Save As...** and save the file as `CompanyLast.xml`. (Remember to rename the original `CompanyLast.xml` file that is delivered with XMLSpy to something else, like `CompanyLast_orig.xml`).

Please note: The `CompanyLast.xml` file delivered with XMLSpy is in the in the `Tutorial` folder.

2.4 XSLT Transformations

Objective
To generate an HTML file from the XML file using an XSL stylesheet to transform the XML file. You should note that a "transformation" does not change the XML file into anything else; instead a new output file is generated. The word "transformation" is a convention.

Method
The method used to carry out the transformation is as follows:

- Assign a predefined XSL file, `Company.xsl`, to the XML document.
- Execute the transformation within the XMLSpy interface using one of the two built-in Altova XSLT engines. (*See note below.*)

Commands used in this section
The following XMLSpy commands are used in this section:

XSL/XQuery | Assign XSL, which assigns an XSL file to the active XML document.

XSL/XQuery | Go to XSL, opens the XSL file referenced by the active XML document.

XSL/XQuery | XSL Transformation (F10), or the toolbar icon , transforms the active XML document using the XSL stylesheet assigned to the XML file. If an XSL file has not been assigned then you will be prompted for one when you select this command.

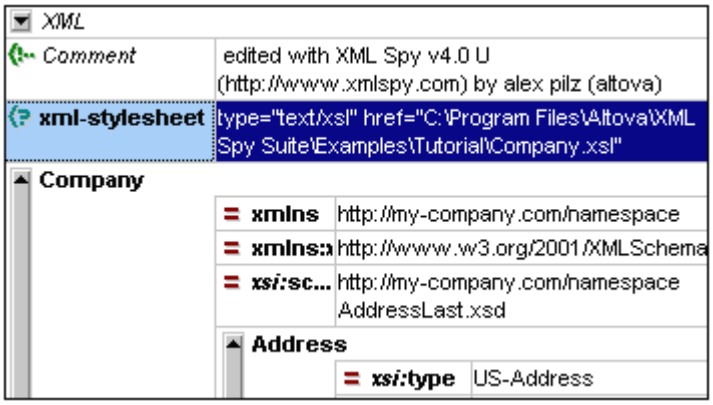
Please note: XMLSpy has two built-in XSLT engines, the Altova XSLT 1.0 Engine and Altova XSLT 2.0 Engine. The Altova XSLT 1.0 Engine is used to process XSLT 1.0 stylesheets. The Altova XSLT 2.0 Engine is used to process XSLT 2.0 stylesheets. The correct engine is automatically selected by XMLSpy on the basis of the version attribute in the `xsl:stylesheet` or `xsl:transform` element. In this tutorial transformation, we use XSLT 1.0 stylesheets. The Altova XSLT 1.0 Engine will automatically be selected for transformations with these stylesheets when the **XSL Transformation** command is invoked.

2.4.1 Assigning an XSLT File

To assign an XSLT file to the `CompanyLast.xml` file:

1. Click the `CompanyLast.xml` tab in the main window so that `CompanyLast.xml` becomes the active document, and switch to Text View.
2. Select the menu command **XSL/XQuery | Assign XSL**.
3. Click the **Browse** button, and select the `Company.xsl` file from the Tutorial folder. In the dialog, you can activate the option Make Path Relative to `CompanyLast.xml` if you wish to make the path to the XSL file (in the XML document) relative.4. Click **OK** to assign the XSL file to the XML document.
5. Switch to Grid View to see the assignment (*screenshot below*).

 © 2011 Altova GmbH

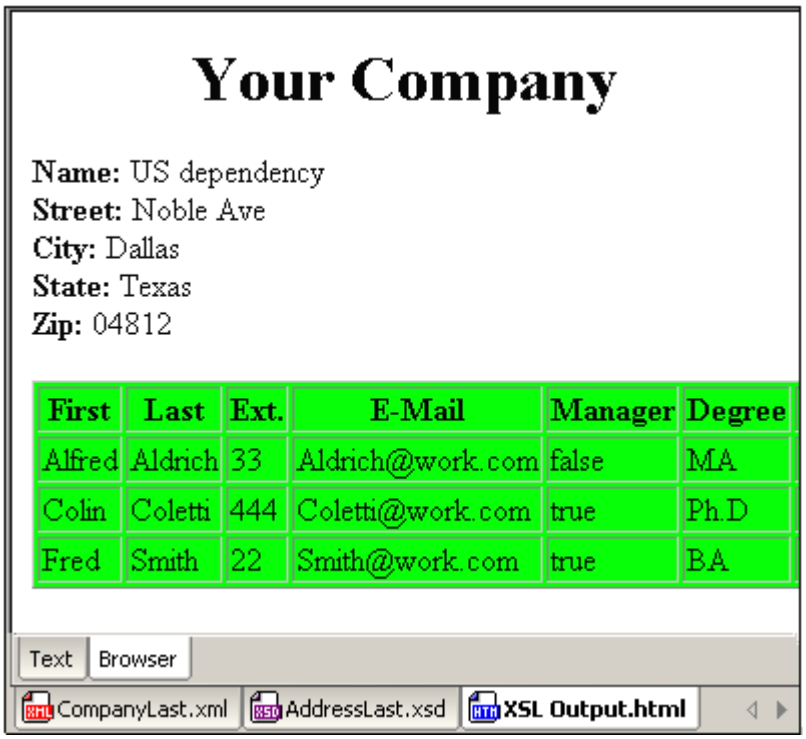


An `XML-stylesheet` processing instruction is inserted in the XML document that references the XSL file. If you activated the Make Path Relative to `CompanyLast.xml` check box, then the path is relative; otherwise absolute (as in the screenshot above).

2.4.2 Transforming the XML File

To transform the XML document using the XSL file you have assigned to it:

1. Ensure that the XML file is the active document.
2. Select the menu option **XSL/XQuery | XSL Transformation (F10)** or click the icon. This starts the transformation using the XSL stylesheet referenced in the XML document. (Since the `Company.xsl` file is an XSLT 1.0 document, the built-in Altova XSLT 1.0 Engine is automatically selected for the transformation.) The output document is displayed in Browser View; it has the name `XSL Output.html`. (If the HTML output file is not generated, ensure that, in the XSL tab of the Options dialog (**Tools | Options**), the default file extension of the output file has been set to `.html`.) The HTML document shows the Company data in one block down the left, and the Person data in tabular form below.



Please note: Should you only see a table header and no table data in the output file, make sure that you have defined the target namespace for your schema. The namespace must be **identical** in all three files (Schema, XML, and XSL).

2.4.3 Modifying the XSL File

You can change the output by modifying the XSL document. For example, let's change the background-color of the table in the HTML output from lime to yellow.

Do the following:

1. Click the `CompanyLast.xml` tab to make it the active document, and make sure you are in Grid View.
2. Select the menu option **XSL/XQuery | Go to XSL**.

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
xmlns:xsi ="http://www.w3.org/2000/10/XMLSchema-instance"
xmlns:my="http://my-company.com/namespace">

<xsl:template match="/">
    <html>
        <head> <title>Your company</title></head>
            <body>
                <h1><center>Your Company</center></h1>
                <xsl:apply-templates select="//my:Address"/>
                    <table border="1" bgcolor="lime">
                        <thead align="center">
                            <td><strong>First</strong></td>
                            <td><strong>Last</strong></td>
                            <td><strong>Ext.</strong></td>
```

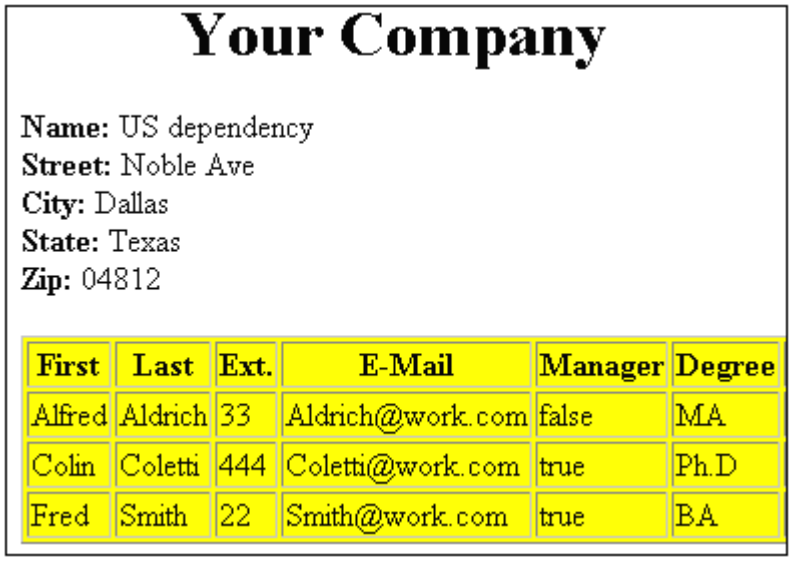
The command opens the `Company.xsl` file referenced in the XML document.

 © 2011 Altova GmbH

3. Find the line `<table border="1" bgcolor="lime">`, and change the entry `bgcolor="lime"` to `bgcolor="yellow"`.

```
<h1><center>Your Company</center></h1>
<xsl:apply-templates select="//my:Address"/>
    <table border="1" bgcolor="yellow">
        <thead align="center">
            <td><strong>First</strong></td>
            <td><strong>Last</strong></td>
```

4. Select the menu option **File | Save** to save the changes made to the XSL file.
5. Click the `CompanyLast.xml` tab to make the XML file active, and select **XSL/XQuery | XSL Transformation,** or press **F10**. A new `XSL Output.html` file appears in the XMLSpy GUI in Browser View. The background color of the table is yellow.



Your Company

Name: US dependency
Street: Noble Ave
City: Dallas
State: Texas
Zip: 04812

First	Last	Ext.	E-Mail	Manager	Degree
Alfred	Aldrich	33	Aldrich@work.com	false	MA
Colin	Coletti	444	Coletti@work.com	true	Ph.D
Fred	Smith	22	Smith@work.com	true	BA

6. Select the menu option **File | Save**, and save the document as `Company.html`.

2.5 Project Management

This section introduces you to the project management features of XMLSpy. After learning about the benefits of organizing your XML files into projects, you will organize the files you have just created into a simple project.

2.5.1 Benefits of Projects

The benefits of organizing your XML files into projects are listed below.

- Files and URLs can be grouped into folders by common extension or any other criteria.
- Batch processing can be applied to specific folders or the project as a whole.
- A DTD or XML Schema can be assigned to specific folders, allowing validation of the files in that folder.
- XSLT files can be assigned to specific folders, allowing transformations of the XML files in that folder using the assigned XSLT.
- The destination folders of XSL transformation files can be specified for the folder as a whole.

All the above project settings can be defined using the menu option **Project | Project Properties...**. In the next section, you will create a project using the Project menu.

Additionally, the following advanced project features are available:

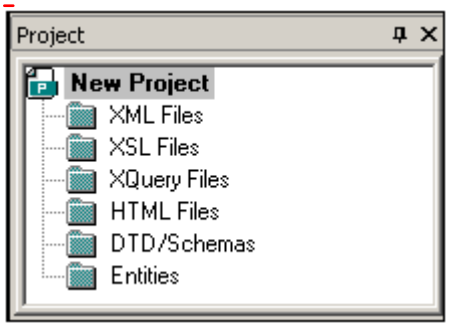
- XML files can be placed under source control using the menu option **Project | Source control | Add to source control...**. (Please see the Source Control section in the online help for more information.)
- Personal, network and web folders can be added to projects, allowing batch validation.

2.5.2 Building a Project

Having come to this point in the tutorial, you will have a number of tutorial-related files open in the Main Window. You can group these files into a tutorial project. First you create a new project and then you add the tutorial files into the appropriate sub-folders of the project.

Creating a basic project
To create a new project:

1. Select the menu option **Project | New Project**. A new project folder called `New Project` is created in the Project Window. The new project contains empty folders for typical categories of XML files in a project (*screenshot below*).



2. Click the `CompanyLast.xml` tab to make the `CompanyLast.xml` file the active file in the Main Window.
3. Select the menu option **Project | Add active and related files to project.** Two files are added to the project: `CompanyLast.xml` and `AddressLast.xsd`. Note that files

referenced with Processing instructions (such as XSLT files) do not qualify as related files.

4. Select the menu option **Project | Save Project** and save the project under the name `Tutorial`.

Adding files to the project
You can add other files to the project as well. Do this as follows:

1. Click on any open XML file (with the `.xml` file extension) other than `CompanyLast.xml` to make that XML file the active file. (If no other XML file is open, open one or create a new XML file.)
2. Select the menu option **Project | Add active file to project**. The XML file is added to the XML Files folder on the basis of its `.xml` file type.
3. In the same way, add an HTML file and XSD file (say, the `Company.html` and `AddressFirst.xsd` files) to the project. These files will be added to the HTML Files folder and DTD/Schemas folder, respectively.
4. Save the project, either by selecting the menu option **Project | Save Project** or by selecting any file or folder in the Project Window and clicking the Save icon in the toolbar (or **File | Save**).

Please note: Alternatively, you can right-click a project folder and select Add Active File to add the active file to that specific folder.

Other useful commands
Here are some other commonly used project commands:

- To add a new folder to a project, select **Project | Add Project folder to Project**, and insert the name of the project folder.
- To delete a folder from a project, right-click the folder and select **Delete** from the context menu.
  To delete a file from a project, select the file and press the **Delete** key.

2.6 That's It

If you have come this far congratulations, and thank you!

We hope that this tutorial has been helpful in introducing you to the basics of XMLSpy. If you need more information please use the context-sensitive online help system, or print out the PDF version of the tutorial, which is available as `tutorial.pdf` in your XMLSpy application folder.

© 2011 Altova GmbH

3 Editing Views

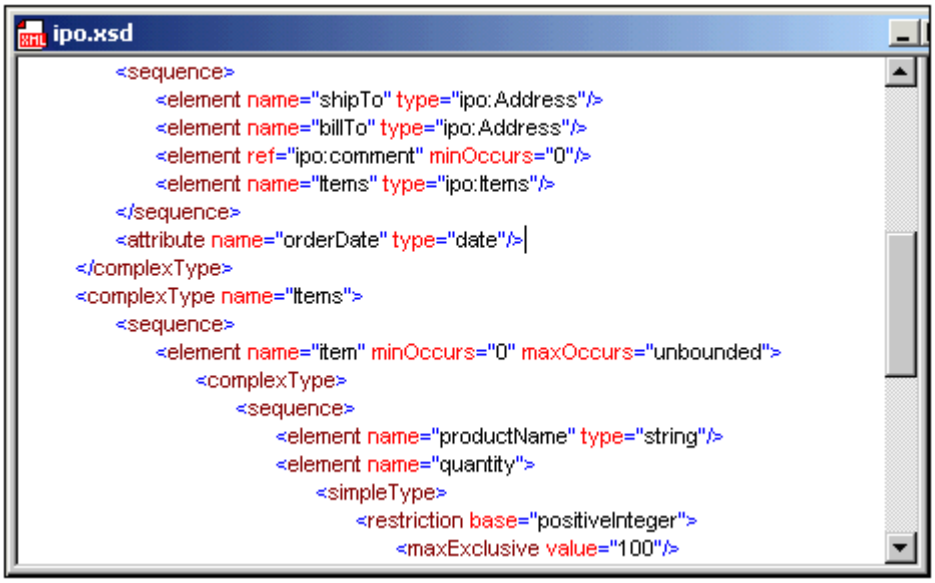
XMLSpy contains powerful editing views. In addition to a Text View with intelligent editing features, there are graphical views that greatly simplify the editing of documents. Depending on what type of document is currently active in XMLSpy, the Main Window will have one or more of XMLSpy's Editing Views. For example, when an HTML document is active, the Main Window will contain two editing views: Text View and Browser View. When an XML document is active, there will be five editing views: Text View, Grid View, Schema View, Authentic View, and Browser View. Of these views, Schema View will be enabled only for XML Schema documents.

In this section, we describe the various editing views available in XMLSpy:

- Text View
- Grid View
- Schema View
- Authentic View
- Browser View

3.1 Text View

In Text View (*screenshot below*), you can type in the text of your document—both, markup and content—directly. Any text file, including non-XML documents (such as XQuery and HTML documents) can be edited in Text View. A number of features help you to quickly and accurately type in your document.



In this section, we describe general Text View features that are available for all kinds of documents. Specific document types, such as XML, XQuery, and CSS have certain type-specific features, which are described in the respective sections for those document types. For example, additional XML-specific features of Text View are described in the section XML | Editing XML in Text View.

The general Text View features have been organized as follows:

- Formatting in Text View describes how the font properties, indentation, and word-wrapping of the document can be specified.
- Document display contains information about the line-numbering, bookmarking, expanding/collapsing of nodes, and other display-related features.
- Editing in Text View describes the features that are available while you edit, particularly the intelligent editing features.
- Entry helpers are the windows that provide context-sensitive data-entry options. For example, the elements or attributes that can be validly added at a given document location are displayed in an entry helper and any one of these options can be inserted by double-clicking it.

Switching to Text View
To open the Text View of a document, click the **Text** button at the bottom of the Document Window or select **View | Text View**.

3.1.1 Formatting in Text View

Text View offers a number of text formatting options. These are listed below.

 © 2011 Altova GmbH

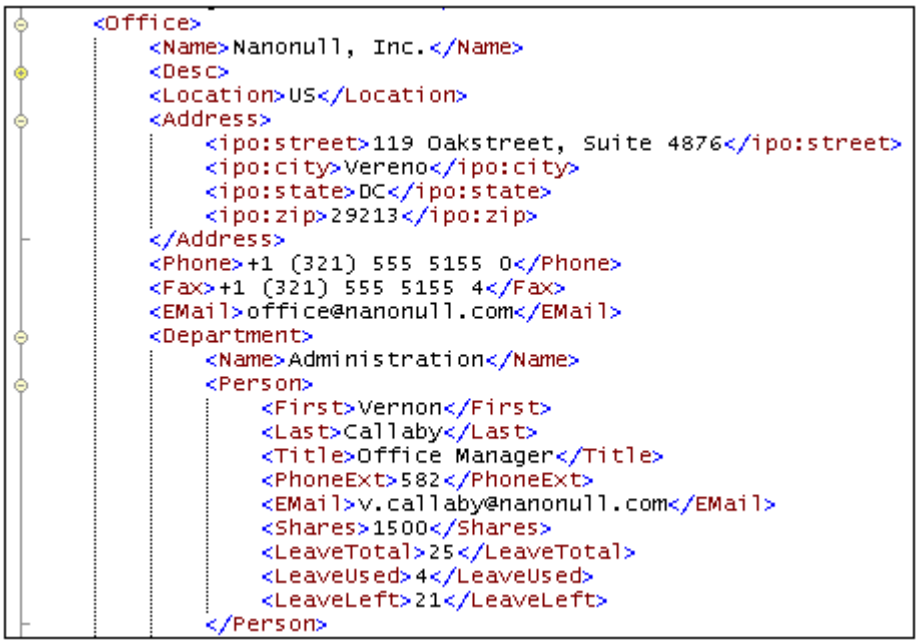
Fonts

The font-family, font-size, font-style, and text background-color can be customized separately for the following groups of documents: (i) generic XML documents (including HTML); (ii) XQuery documents; and (iii) CSS documents.

Text items in a document that have different semantics, can be colored differently. For example, you can color element names, attribute names, and element content differently. When you set different colors for different text items, the syntax-coloring feature is enabled. Text fonts are customized in the Text Fonts tab of the Options dialog, and how to do this is described in the section, User Reference | Options | Text Fonts section of this documentation.

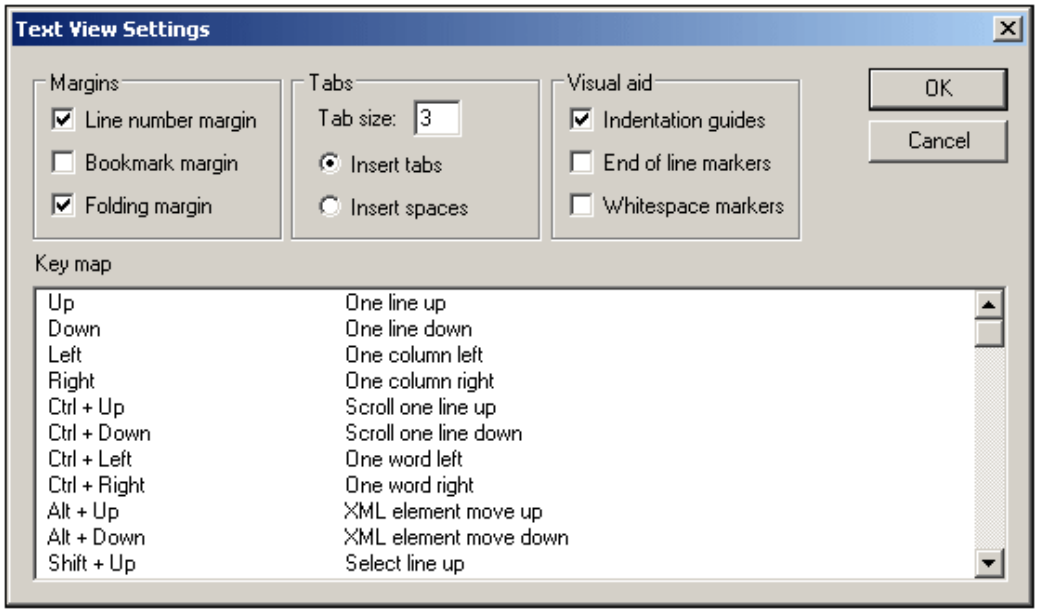
Indentation

Well-formed XML documents can be pretty-printed. This means that the document can be formatted so that the hierarchical structure of the document is displayed using new lines and indentations (*see screenshot below*).



To display the document in this way, you need to do the following:

1. In the View Tab of the Options dialog, check the Use Indentation option for pretty printing. This will cause the document to be pretty-printed with indents to indicate the hierarchical structure. Each deeper level will be displayed with a deeper indent than its parent element. If the Use Indentation option is not checked, every line in the document will start with a zero indent.
2. In the Text View Settings dialog (*screenshot below*), select either Insert Tabs or Insert Spaces. This determines whether tabs or spaces will be used for indentation when the document is pretty-printed. If spaces are specified, each deeper level of the hierarchy is indented with an additional number of spaces as specified in the Tab Size setting of the Text View Settings dialog.



3. Click the **Edit | Pretty-Print XML Text** command or the **Pretty Print** icon in the Text toolbar. This will cause the document text to be displayed (i) with or without indentation as specified in the View Tab of the Options dialog; and (ii) if indentation is specified in the View Tab of the Options dialog, then the the indentation is determined by the settings in the Tabs pane of the Text View Settings dialog. Clicking the Pretty Print command removes unnecessary leading or trailing whitespace.

Note: Pretty-printing is also used in the background when you save the document or switch views. If the document is not well-formed, you will get an error message to that effect. Correct the error and then pretty-print. The extent of indentation of a line is indicated by indentation guides, which are vertical dotted lines (*see screenshot at the start of this section*) that are toggled on and off with the Indentation Guides check box in the Visual Aid pane of the Text View Settings dialog (*see screenshot above*).

Using tabs and spaces for formatting
You can use tabs and spaces for formatting text, especially for non-XML documents, where the pretty-printing option is not available. When you press **Return** or **Shift+Return**, the cursor will jump to a position on the next line that corresponds to the starting position of the previous line.

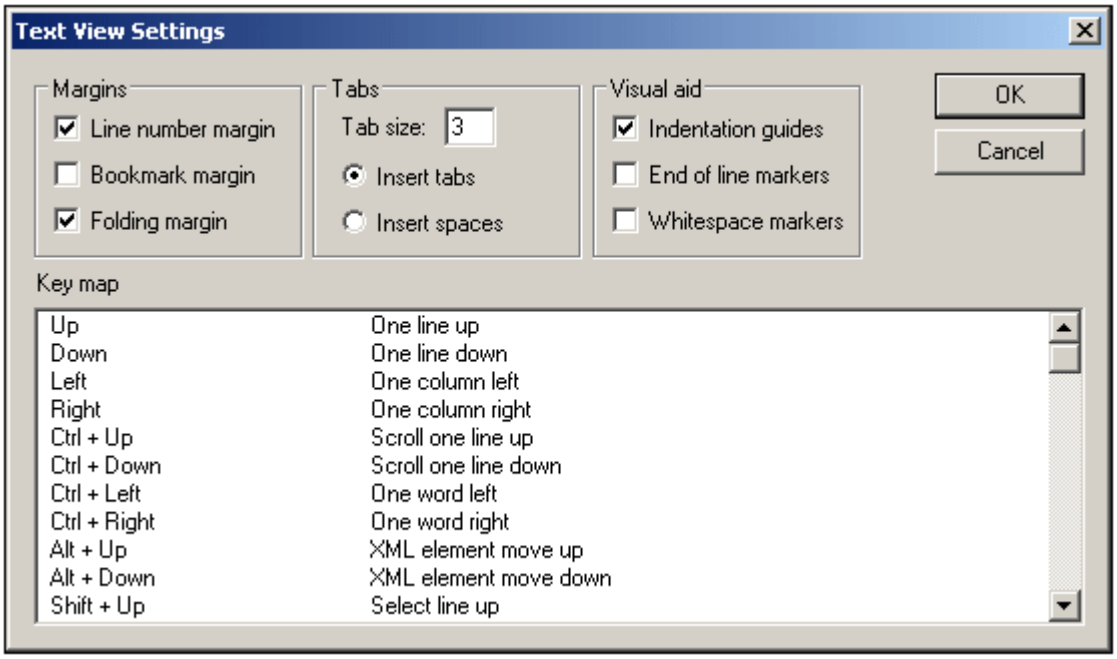
Word-wrapping
Lines of text that are longer than the breadth of the Main Window can be made to wrap by toggling the **View | Word Wrap** command on; the corresponding icon is in the Text toolbar.

3.1.2 Displaying the Document

Text View has visual features to make the display and editing of large sections of text easier. Some very useful features are: (i) Line Numbers, (ii) Bookmarks, (iii) Source Folding (expanding and collapsing the display of nodes), (iv) Indentation Guides, and (v) End-of-Line and Whitespace Markers. These commands are available in the Text View Settings dialog (*first screenshot below*) and the Text toolbar (*second screenshot below*).

The Text View Settings dialog is accessed via the **View | Text View Settings** command or the **Text View Settings** button in the Text toolbar. Settings in the Text View Settings dialog apply to

the entire application—not only to the active document.



Other useful features are the Zooming and Go-to-Line/Character features.

Line numbers
Line numbers are displayed in the line numbers margin (*screenshot below*), which can be toggled on and off In the Text View Settings dialog (see screenshot above). When a section of text is collapsed, the line numbers of the collapsed text are also hidden. A related command is the Go-to-Line/Character command.

Bookmarks
Lines in the document can be separately bookmarked for quick reference and access. If the bookmarks margin is toggled on, bookmarks are displayed in the bookmarks margin; otherwise, bookmarked lines are highlighted in cyan.

The bookmarks margin can be toggled on or off in the Text View Settings dialog (*screenshot above*).

You can edit and navigate bookmarks using commands in the **Edit** menu and Text toolbar. Bookmarks can be inserted with the **Edit | Insert/Remove Bookmark** command, enabling you to mark a line in the document for reference. A bookmark can be removed by selecting the bookmarked line and then selecting the **Edit | Insert/Remove Bookmark** command. To navigate through the bookmarks in a document, use the **Edit | Next Bookmark** and **Edit | Previous Bookmark** commands. These bookmark commands are also available as icons in the Text toolbar (*screenshot above*).

Source folding

Source folding refers to the ability to expand and collapse nodes (*screenshot below*) and is displayed in the source folding margin. The margin can be toggled on and off in the Text View Settings dialog (*see screenshot above*). In the screenshot below, notice how the line numbering at Lines 14 and 24 has been collapsed together with the collapsed nodes.

The **Toggle All Folds** command in the Text toolbar toggles **all** nodes to their expanded forms or collapses all nodes to the top-level document element. After a node has been expanded, the following options are available when clicking on the node's toggle icon (+/- icon):

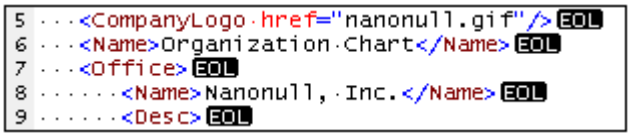
- ***Plain click:*** Collapses the node. Clicking on it again expands the node so that descendant nodes are shown expanded or collapsed according to how they were before the node was collapsed.
- ***Shift+Click:*** Collapses all descendant nodes up to the level of the children nodes. The clicked node itself is open and shows the collapsed children nodes. This will work for all collapsible nodes in a branch of the hierarchy except the last collapsible node in that branch.
- ***Ctrl+Click:*** If a node or any of its descendants is collapsed, **Ctrl+Click** expands all collapsed descendant nodes. In short, this means that all descendant nodes of the control-clicked node are expanded.

Indentation guides

Indentation guides are vertical dotted lines that indicate the extent of a line's indentation (*see screenshot above*). They can be toggled on and off in the Text View Settings dialog.

End-of-line markers, whitespace markers

End-of-line (EOL) markers and whitespace markers can be toggled on in the Text View Settings dialog. The screenshot below shows these markers in the document display; each dot represents a whitespace.



 © 2011 Altova GmbH

Zooming in and out
You can zoom in and out of Text View by scrolling (with the scroll-wheel of the mouse) while keeping the **Ctrl** key pressed. This enables you to magnify and reduce the size of text in Text View. If you wish to increase the size of fonts, do this in the Options dialog.

Go to line/character
This command in the **View** menu and Text toolbar enables you to go to a specific line and character in the document text.

3.1.3 Editing in Text View

The following text editing features are available in Text View generally for all document types. These features are in addition to common features of editing applications, such as **Cut**, **Copy**, **Paste**, **Delete**, and **Select All** (which are available as commands in the **Edit** menu).

- Syntax coloring
- Start-tag and end-tag matching
- Intelligent editing
- Auto-completion
- Moving siblings relative to each other
- Selecting an entire element
- Drag-and-drop and context menus
- Find and replace
- Unlimited undo
- Spelling check

For some document types (such as XML and XQuery) additional specialized features are available, and these are described, respectively, in the sections that deal with those document types.

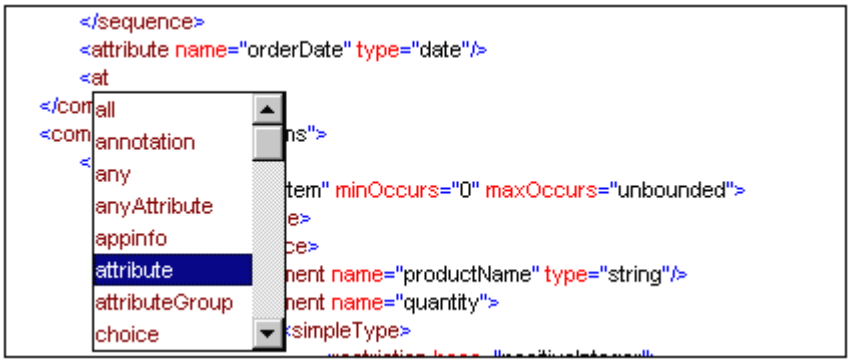
Note: For large files, Auto-completion and entry helpers can be disabled, thus enabling faster loading and editing. The threshold file size is specified by the user. For more details, see the reference section, Options | Editing.

Syntax coloring
Syntax coloring is applied according to the semantic value of the text. For example, in XML documents, depending on whether the XML node is an element, attribute, content, CDATA section, comment, or processing instruction, the node name (and in some cases the node's content) is colored differently. Four groups of document type are distinguished: (i) generic XML (which includes HTML); (ii) XQuery; (iii) CSS; and (iv) JSON. The text properties (including color) of each group can be set in the Text Fonts tab of the Options dialog (**Tools | Options**).

Start-tag and end-tag matching
When you place the cursor inside a start or end tag of a markup element, pressing **Ctrl+E** highlights the other member of the pair. Pressing **Ctrl+E** repeatedly enables you to switch between the start and end tags. This is an excellent aid to locating the start and end tags of an XML element.

Intelligent Editing

If you are working with an XML document based on a schema, XMLSpy provides you with various intelligent editing capabilities in Text View. These allow you to quickly insert the correct element, attribute, or attribute value according to the content model defined for the element you are currently editing. For example, when you start typing the start tag of an element, the names of all the elements allowed by the schema at this point are shown in a pop-up (*screenshot below*). Selecting the required element name and pressing **Enter** inserts that element name in the start tag.



Popup windows also appears in the following cases:

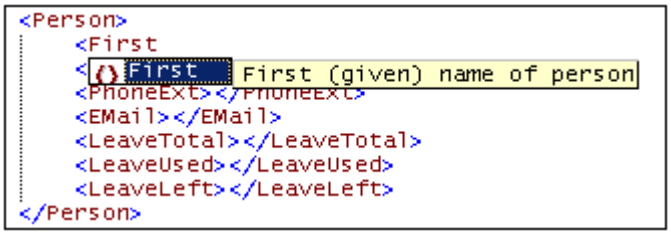
- When the cursor is inside the start tag of an element that has an attribute defined for it and the space bar is pressed. The popup will contain all available attributes.
- When the cursor is within the double-quotes delimiting an attribute value that has enumerated values. The popup will contain the enumerated values.
- When you type `</` (which signifies the start of a closing tag), the name of the element to be closed appears in the popup.
- When you wish to write an empty element as a single tag or convert an empty element of two tags to an empty element of one tag, type in the closing slash after the element name: `<element/`. An empty element with a single tag is created; if a close tag exists, it is removed: `<element/>`.

Auto-completion

Editing in Text View can easily result in XML and other marked-up documents (such as HTML) that are not well-formed. For example, closing tags may be missing, mis-spelled, or structurally mismatched. XMLSpy automatically completes the start and end tags of elements, as well as inserts all required attributes as soon as you finish entering the element name on your keyboard. The cursor is also automatically positioned between the start and end tags of the element, so that you can immediately continue to add child elements or contents: `<img src="" alt="">  </img>`

XMLSpy makes use of the XML rules for well-formedness and validity to support auto-completion. The information about the structure of the document is obtained from the schema on which the document is based. (In the case of well-used schemas, such as HTML and XSLT, the schema information is built into XMLSpy.) Auto-completion uses not only information about the structure of the document, but also the values stored in the document. For example, enumerations and schema annotations in an XML Schema are actively used by the Auto-Completion feature. If, in the schema, values are enumerated for a particular node, then those enumerations will be displayed as auto-completion options when that node comes to be edited. Similarly, if, for a node, annotations exist in the schema, then these annotations are displayed when the node name is being typed in the document (*screenshot below*). (*First (given) name of person* is the schema annotation of the `First` element.)

 © 2011 Altova GmbH



Auto-completion can be switched on and off in the Editing tab of the Options dialog (**Tools | Options | Editing**).

Moving sibling elements relative to each other
When the cursor is within an element, pressing **Alt+ArrowUp** or **Alt+ArrowDown** moves the selected element up or down relative to its siblings.

Drag-and-Drop and Context Menus
You can also use drag-and-drop to move a text block to a new location, as well as right-click to directly access frequently used editing commands (such as Cut, Copy, Paste, Delete, Send by Mail, and Go to line/char) in a context menu.

Find and Replace
You can use the Find and Replace commands to quickly locate and change text. These commands also take regular expressions as input, thereby giving you powerful search capabilities. (See Edit | Find for details.)

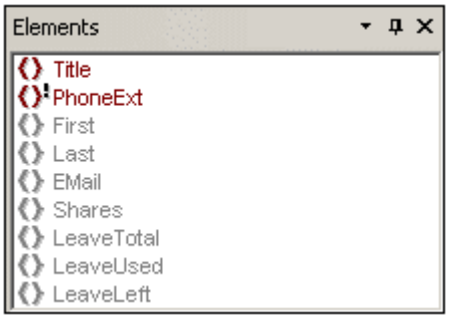
Unlimited Undo
XMLSpy offers unlimited levels of Undo and Redo for all editing operations.

Spelling check
In Text View, documents can be spellchecked with any of the built-in language dictionaries. A user dictionary can also be created and edited to allow words not contained in the language dictionary. For details, see the descriptions of the Spelling and Spelling Options commands.

3.1.4 Entry Helpers in Text View

What entry helpers are available in Text View depends upon the type of document being edited. A list of entry helpers is given below for the most common document types. The general use of entry helpers is described below. Additional features for specific document types, if any, are described in the sections describing the respective document types.

- *XML:* Elements (*screenshot below*), Attributes, Entities



- *HTML:* Elements, Attributes, Entities
- *CSS:* CSS Outline, CSS Properties, HTML Elements
- *DTD:* None
- *XQuery:* XQuery Keywords, XQuery Variables, XQuery Functions
- *Text:* Entities

Note that several document types, such as XSD, XSLT, XHTML, and RDF, are essentially XML documents and will therefore have the Elements, Attributes, and Entities entry helpers.

Display and use of entry helper items
Different items in the various entry helpers are variously color-coded. These color codes are explained in the Entry Helpers documentation of the respective document types. In general, the following points should be noted about entry helpers:

- The entry helpers are context-sensitive and display items that may be inserted at that point.
- If the item has already been inserted at the selected (or at another equivalent and valid location) and may not be inserted again at that location (for example, an XML attribute), it is displayed in gray.
- If the item is mandatory, an exclamation mark icon is displayed next to it.
- To insert an entry helper item at the cursor selection point in the text, double-click the entry-helper item.
- When an element is inserted via the Elements entry helper, its start and end tags are inserted in the document text. Mandatory elements are also inserted if this option has been specified in the **Options** dialog (**Tools | Options | Editing**).
- When an attribute is inserted via the Attributes entry helper, the attribute is inserted at the cursor point together with an equals-to sign and quotes to delimit the attribute value. The cursor is placed between the quotes, so you can start typing in the attribute value directly.

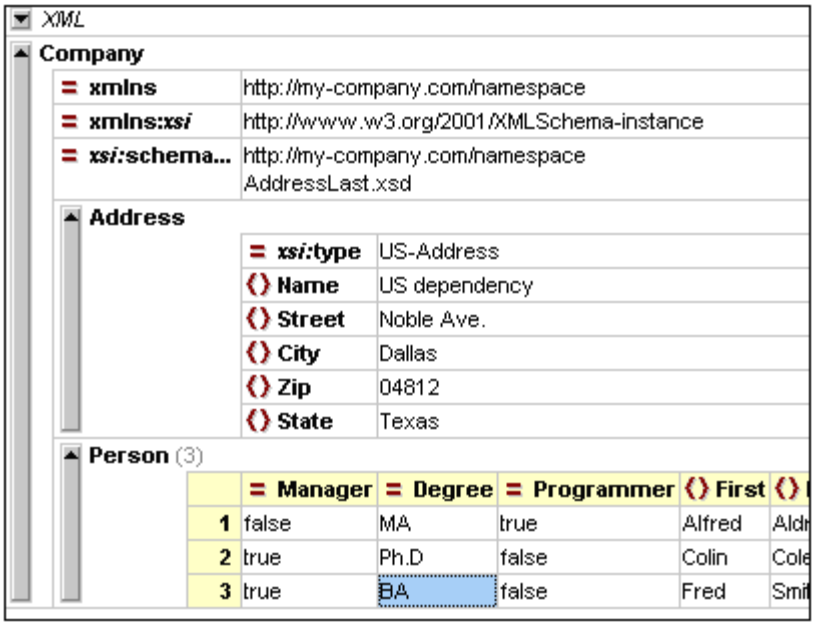
Note: For large files, Auto-completion and entry helpers can be disabled, thus enabling faster loading and editing. The threshold file size is specified by the user. For more details, see the reference section, Options | Editing.

 © 2011 Altova GmbH

3.2 Grid View

Grid View (*screenshot below*) shows the hierarchical structure of XML documents and DTDs through a set of nested containers, that can be easily expanded and collapsed to get a clear picture of the document's structure. In Grid View contents and structure can both be easily manipulated.

Note: In Standard Edition, Grid View is available as a read-only view. Editing in Grid View is available in the Enterprise and Professional editions.



A hierarchical item (such as the XML declaration, document type declaration, or element containing child elements) is represented with a gray bar on the left side containing a small upwards-pointing arrow at the top. Clicking the side bar expands or collapses the item. An element is denoted with the icon, an attribute with the icon.

Display and navigation

- The contents of an item depend on its kind. For example, in the case of elements, the contents will typically be attributes, character data, comments, and child elements. Attributes are always listed and are ordered as in the input file. Following the attributes, items appear exactly in the order found in the source file. This order can be changed using drag-and-drop, and the change will also be implemented in the source data.
- If an element contains only character data, the data will be shown in the same line as the element and the element will not be considered hierarchical by nature. The character data for any other element will be shown indented with the attributes and potential child elements and will be labeled as "Text".
- If an element is collapsed, its attributes and their values are shown in the same line in gray. This attribute preview is especially helpful, when editing XML documents that contain a large number of elements of the same name that differ only in their contents and attributes (for example, database-like applications).
- The arrow keys move the selection bar in the grid view
- The + and – keys on the numeric keypad allow you to expand and collapse items.

Customizing Grid View

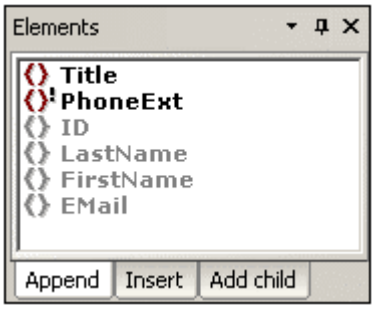
- To resize columns, place the cursor over the appropriate border and drag so as to achieve the desired width.
- To resize a column to the width of its largest entry, double-click on the grid line to the right of that column.
- To adjust column widths to display all content, select the menu item View | Optimal widths command, or click on the Optimal widths icon .
- The heights of cells are determined by their contents. They can be adjusted with the menu option **Tools | Options | View | Grid View,** "Limit cell height to xx lines".

Please note: If you mark data in Grid View and switch to Text View, that data will be marked also in Text View.

3.2.1 Entry Helpers in Grid View

Note: In Standard Edition, Grid View is available as a read-only view. Editing in Grid View is available in the Enterprise and Professional editions.

Elements Entry Helper
In Grid View, the Elements Entry Helper has three tabs: Append, Insert, and Add Child. The **Append** tab displays elements that can be appended after all the siblings of the current element; the **Insert** tab displays all elements that can be inserted before the current element; and the **Add Child** tab displays those elements you can insert as a child of the current element.

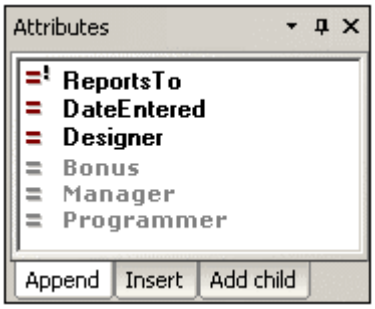


To insert an element, select the appropriate tab and double-click the required element. Note that mandatory elements are indicated with an exclamation mark. Siblings of allowed elements that cannot themselves be inserted at the cursor point are unavailable.

Please note: In the **Options** dialog (**Tools | Options | Editing**), you can specify that mandatory child elements are inserted when an element is inserted.

Attributes Entry Helper
The Attributes Entry Helper displays a list of available attributes for the element you are currently editing. Mandatory attributes are indicated with an exclamation mark "**!**" before the name of the attribute. If an attribute has already been entered for that element, that attribute is shown in gray.

 © 2011 Altova GmbH



- To use the attributes in the Append and Insert tabs, select, in Grid View, an existing attribute or an element that is a child of the attribute's parent element, and double-click the required attribute in the Entry Helper.
- To use the attributes in the Add Child tab, select the attribute's parent element in Grid View and double-click the required attribute in the Entry Helper.

Please note: Existing attributes, which cannot legally be added to the current element a second time, are shown in gray.

Entities Entry Helper
The Entities Entry Helper displays all parsed or unparsed entities that are declared inline or in an external DTD. If a text node or attribute node is selected in Grid View, the Add Child tab will appear empty—because, by definition, such nodes cannot have any children.

To use the cursor to insert an entity in Grid View, place the cursor at the required position in a text field or select the required field; then select the appropriate tab, and double-click the entity. Note the following rules:

- If the cursor is placed within a text field (including an attribute value field), the entity is inserted at the cursor insertion point.
- If an element with a text-only child (i.e. `#PCDATA` or `simpleType`) is selected but the cursor is not placed in the text field, then any existing text content **is replaced** by the entity.
- If a non-text field is selected, then the entity is created as text at a location corresponding to the Entry Helper tab that you select.

Please note: If you add an internal entity, you will need to save and reopen your document before the entity appears in the Entities Entry Helper.

3.3 Schema View

XML Schemas can be viewed and edited in Schema View. Schema View itself has two views:

- ***Schema Overview*** displays all global components, such as global elements and complex types, in a simple tabular list (*see screenshot*)
- ***Content Model View*** shows the content models of individual global components (*see screenshot*)

You can switch from Schema Overview to the content model of a specific global component by clicking the icon of that global component. To return to Schema Overview, click the Show Globals icon in Content Model View.

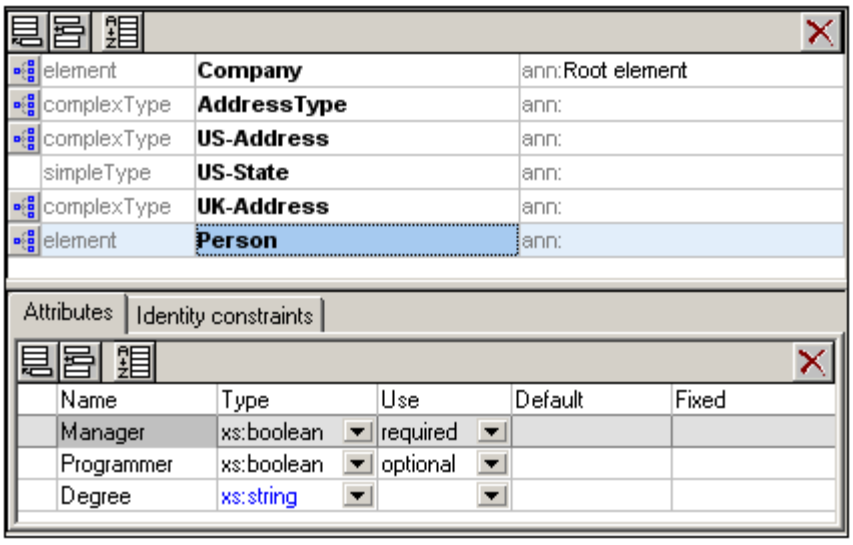
Note: In Standard Edition, Schema View is available as a read-only view. Editing in Schema View is available in the Enterprise and Professional editions.

Organization of this section
This section has been organized into the following sub-sections

- Schema Overview describes how global components are displayed in Schema Overview and how they are edited
- Content Model View describes how the content models of individual global components are edited
- Entry Helpers explains the organization of the entry helpers in Schema View
- Identity Constraints describes how identity constraints are displayed and edited in Schema View
- Smart Restrictions is a XMLSpy editing feature that facilitates the graphical creation and editing of derived types from their base types; this section describes how the Smart Restrictions feature is used

3.3.1 Schema Overview

Schema Overview displays a list of all the global components of the schema (global elements, complex types, etc).



You can insert, append, or delete global components, as well as modify their properties. To

insert, append, or delete, use the respective buttons at the top of Schema Overview. To modify properties, select the required component in the Schema Overview list, and edit its properties in either the entry helpers (at right of view) or the Attributes/Identity Constraints pane (at bottom of view).

You can edit the content model of a global component in Content Model View, which is accessed by clicking the Content Model View icon at the left of the global component.

Note the following editing features of Schema Overview:

- You can reposition components in the Schema Overview list using drag-and-drop.
- You can navigate using the arrow keys of your keyboard.
- You can copy or move global components, attributes, and identity constraints to a different position and from one schema to another using cut/copy-and-paste.
- Right-clicking a component opens a context menu that allows you to cut, copy, paste, delete, or edit the annotation data of that component.

Note: In Standard Edition, Grid View is available as a read-only view. Editing in Grid View is available in the Enterprise and Professional editions.

Global components in Schema Overview

At the top level of an XML Schema document (i.e., at the level of children of the `schema` element), the following five basic components can be defined:

- Annotation
- Type definition (simple or complex)
- Declaration (element or attribute)
- Attribute group
- Model group

These components are called **global components**. The **Schema Overview** displays a list of all global components in your schema in a tabular form. Some global components (such as complex types, element declarations, and model groups) can have a content model which describes the component's structure and contents. Other global components (such as annotations, simple types, and attribute groups) do not have a content model. Those components for which content models are possible have a icon to the left of the component name. Clicking on this icon opens the Content Model View for that global component.

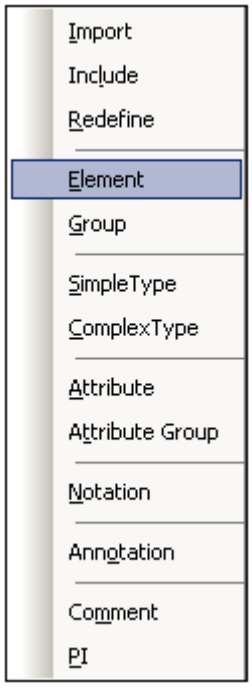
Key terms

- *Simple type and complex type.* A simple type is used to define all attributes and elements that contain only text and that have no associated attribute. A simple type, therefore, has no content model—only text (which can be restricted by the datatype). A complex type is one that has at least one child element or attribute. Declaring a child element on an element automatically assigns the element a type of complex.
- *Global and local components.* A global component can be any of the five listed above. A global component can be defined in Schema Overview, and it then immediately appears in the list of global components in Schema Overview. If the global component is a complex type, an element declaration, or a model group, you can subsequently define its content model by editing it in Content Model View. Once a global component has been defined, it can be referenced by local components. A local component is created directly within the content model of some component. Note that, in the Content Model View, a local component can be converted into a global component (via the right-click context menu).

Comments and processing instructions
In XML Schema documents, comments and processing instructions within simple types and complex types are collected and moved to the end of the enclosing object. In such cases, it is therefore recommended that you use annotations instead of comments.

Creating global components
To create a global component in Schema Overview:

1. Click the Insert or Append icon at the top of the Schema Overview. This pops up a menu listing the various component types (element, simple type, complex type, model group, etc).



2. Select the type of component you want. An entry of that type is created in the list of global components.
3. Enter the name of the component in the entry, and press **Enter**. The name of the new global component is added to the appropriate list/s (`Elm`, `Grp`, `Com`, `Sim`, etc.) in the Component Navigator entry helper. You can edit the content model of the new global component either by double-clicking the component name in the Component Navigator or by clicking the icon to the left of the new component's name in the list of global components.

Please note:

- You can also create a global component while editing in Content Model View. Right-click anywhere in the window and select **New Global | Element**.
- While editing in Content Model View, you can make a local element a global element—or even a complex type if the element has an element or attribute child. Select the local element, right-click anywhere in the window, and select **Make Global | Element** or **Make Global | Complex type**.

 © 2011 Altova GmbH

Deleting global components

To delete a global component:

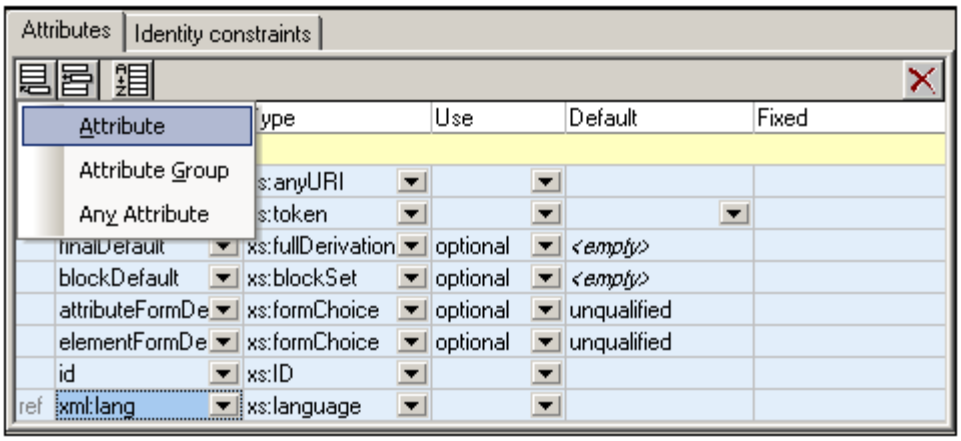
1. Select the global component in the list of global components in the Schema Overview.
2. Press the **Delete** key, or click the **Delete** icon at the top of the Schema Overview.

Attributes and identity constraints of components

You can define attributes and identity constraints for components in either Schema Overview or Content Model View. In Schema Overview, the attributes and identity constraints of a component are displayed in the Attributes/Identity Constraints pane at the bottom of the Schema Overview window and can be edited there. In Content Model View, attributes and identity constraints can appear either in the Attributes/Identity Constraints pane at the bottom of the Content Model Window or within the Content Model diagram itself, where it can be edited. You can select how to display attributes and identity constraints in Content Model View with a setting in the Schema Display Configuration dialog, which is accessed with the **Schema design | Configure view** menu command.

Defining attributes for a component

To define attributes for a component, you use the Attributes/Identity Constraints pane, which is at the bottom of the Schema Overview window.



To define attributes for a global component for which attributes are allowed:

1. Select the global component in the global components list.
2. In the Attributes/Identity Constraints pane, select the Attributes tab.
3. Click the Append or Insert icon at the top left of the Attribute tab.
4. From the popup that appears, select the attribute type you want to append or insert. An entry is created in the Attribute list.
5. In the newly created entry, enter the attribute's properties.

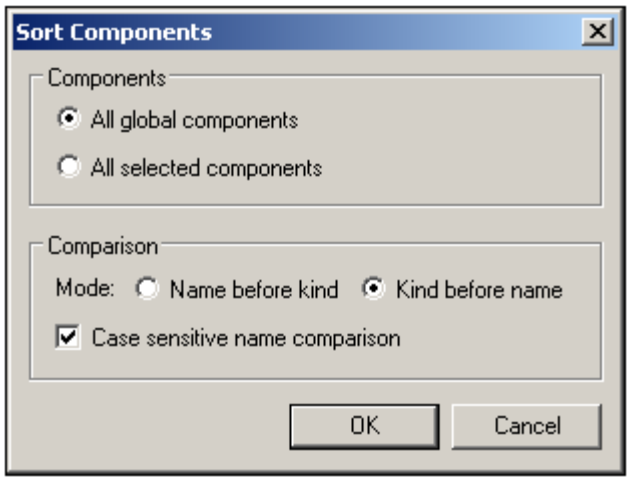
Please note: You can also define attributes for global components in Content Model View: Select the global component, and then define attributes as described above.

Defining identity constraints for a component

To define identity constraints for a component, you use the Attributes/Identity Constraints pane, which is at the bottom of the Schema Overview window. See Identity Constraints for a description of how to work with identity constraints.

Sorting global components, attributes, and identity constraints
Global components can be sorted alphabetically, in ascending order, by local name and kind. Do this as follows:

1. If you wish to sort a selection of global components, then select the global components you wish to sort by clicking them with the **Ctrl** key pressed. (You can also select a range by pressing the **Shift** key and clicking the first and last components of the range.) The global components you select will be highlighted. If you wish to sort all the global components you can select one or more global components. An option for sorting all the components will be available.
2. Clicking the **Sort** icon at the top of the Schema Overview pane. This pops up the Sort Components dialog (*screenshot below*).



3. In the *Components* pane, select whether all the components or only the selection is to be sorted. The *Selected Components* option is enabled only if a the number of selected global components is greater than one.
4. In the *Comparison* pane, you can select whether the sort is performed first on the criterion of local name or component kind. If there is more than one component of the same local name or same component kind (whichever is the first sort criterion), then these are sorted on the secondary criterion.
5. For case-sensitive name comparison, select the check box with this label. If this option is selected, lowercase will be listed before uppercase.

Sorting attributes and identity constraints works in the same way as sorting global components. The **Sort** icon is located in the toolbar of the Attributes/Identity Constraints pane, and it is enabled if the pane contains at least two named sibling components (such as attributes or keys).

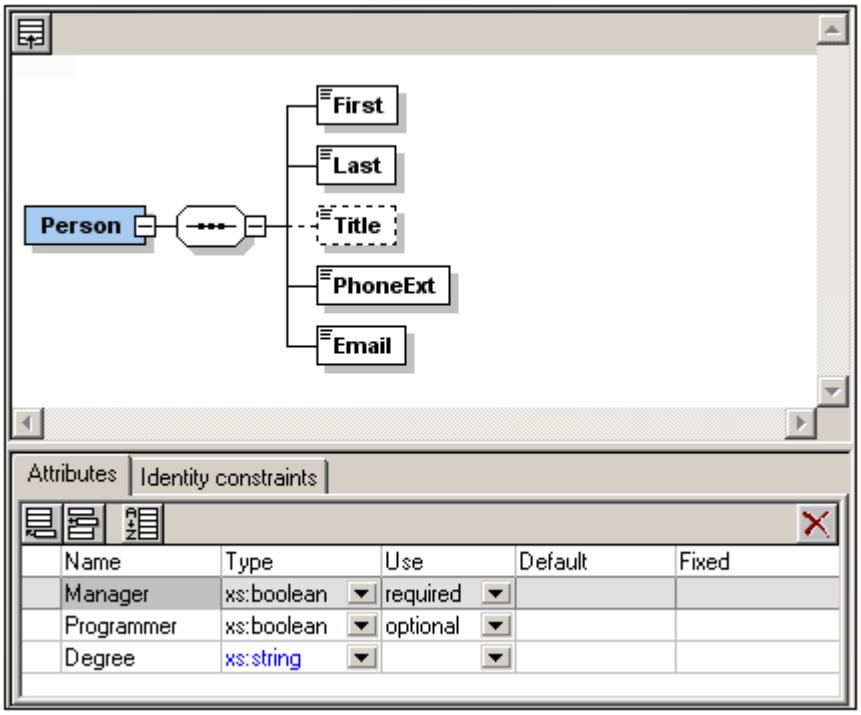
3.3.2 Content Model View

A content model is a description of the structure and contents of an element. Global components which can have a content model (for example, elements, complex types, and model groups; but not, for example, simple types) are indicated in the Schema Overview list with a icon to the left of the component name. Clicking on this icon opens the Content Model View for that global component. Alternatively, (i) select a component and then select the menu option **Schema design | Display Diagram,** or (ii) double-click on a component's name in the Component Navigator (which is the entry helper located, by default, at top right). Note that only one content model in the schema can be open at a time. When a content model is open, you

 © 2011 Altova GmbH

can jump to the content model of a component within the current content model by holding down **Ctrl** and double-clicking the required component.

The content model is displayed in the Content Model View as a tree (*see screenshot below*). You can configure the appearance of the tree in the Schema Display Configuration dialog (menu item **Schema design | Configure view**).



Note the following editing features of Content Model View:

- Each level (of elements or element groups) in the tree is joined to adjacent levels with a compositor.
- Drag-and-drop functionality enables you to move tree objects (compositors, elements, element groups) around.
- You can use keyboard shortcuts to copy (**Ctrl-c**) and paste (**Ctrl-v**) tree objects
- You can add objects (compositors, elements, and element groups) via the context menu (right-click an object).
- You can sort sibling components by selecting the sibling components you wish to sort, right-clicking, and selecting the **Sort Components** command from the context menu. You can prioritize two sort criteria: (i) local name, and (ii) component kind.
- You can edit the properties of an object in the Details entry helper (compositors, elements, element groups) and the Attributes/Identity Constraints pane.
- The Attributes and Identity Constraints of a component are displayed in a pane at the bottom of the Main Window. Attributes and Identity Constraints can also be displayed in the Content Model diagram instead of in the Attributes/Identity Constraints pane. This viewing option can be set in the Schema Display Configuration dialog.

These features are explained in detail in the subsections of this section and in the tutorial.

To return to the Schema Overview, click the **Show Globals** icon or select the menu option **Schema design | Display All Globals**.

Note: In Standard Edition, Grid View is available as a read-only view. Editing in Grid View is

available in the Enterprise and Professional editions.

Editing in Content Model View

Content Model View enables you to quickly define the content model of the following three component types graphically with a few mouse clicks:

- Complex types
- Element declarations
- Model groups

All other schema components (annotations, attribute declarations, simple types, etc.) do not have a content model.

In Content Model View, the various parts of the content model are represented graphically, These parts are organized into two broad groups: **compositors** and **components**. Typically a compositor is added and then the desired child components.

Compositors

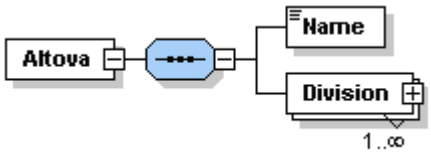
A **compositor** defines the order in which child elements occur. There are three compositors: `sequence`, `choice`, and `all`.

To insert a compositor:

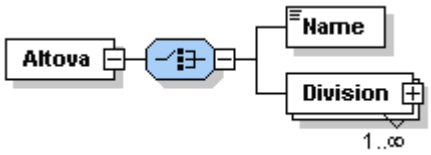
1. Right-click the element to which you wish to add child elements
2. Select **Add Child | Sequence** (or **Choice** or **All**).

The compositor is added, and will look as below:

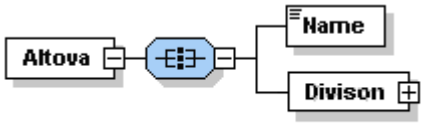
- **Sequence**



- **Choice**



- **All**



To change the compositor, right-click the compositor and select **Change Model | Sequence** (or **Choice** or **All**). After you have added the compositor, you will need to add child element/s or a model group.

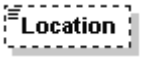
 © 2011 Altova GmbH

Components in the Content Model
Given below is a list of components that are used in content models. The graphical representation of each provides detailed information about the component's type and structural properties.

- Mandatory single element

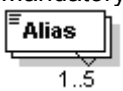
Details: The rectangle indicates an element and the solid border indicates that the element is required. The absence of a number range indicates a single element (i.e. `minOcc=1` and `maxOcc=1`). The name of the element is `Country`. The blue color indicates that the element is currently selected; (a component is selected by clicking it). When a component is not selected, it is white.

- Single optional element



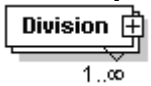
Details: The rectangle indicates an element and the dashed border means the element is optional. The absence of a number range indicates a single element (i.e. `minOcc=0` and `maxOcc=1`). Element name is `Location`.
Note: The context menu option **Optional** converts a mandatory element into an optional one.

- Mandatory multiple element

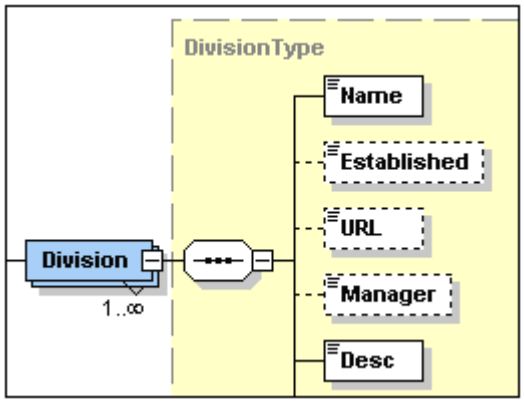


Details: The rectangle indicates an element and the solid border indicates that the element is required. The number range 1..5 means that `minOcc=1` and `maxOcc=5`. Element name is `Alias`.

- Mandatory multiple element containing child elements



Details: The rectangle indicates an element and the solid border indicates that the element is required. The number range 1..infinity means that `minOcc=1` and `maxOcc=unbounded`. The plus sign means complex content (i.e. at least one element or attribute child). Element name is `Division`.
Note: The context menu option **Unbounded** changes `maxOcc` to `unbounded`. Clicking on the + sign of the element expands the tree view and shows the child elements.



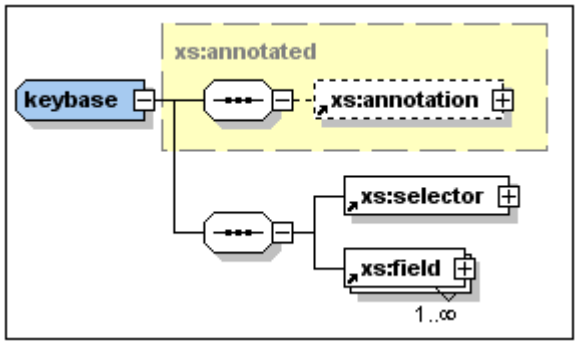
- Element referencing global element

Details: The arrow in the bottom-left means the element references a global element. The rectangle indicates an element and the solid border indicates that the element is required. The number range 1..infinity means that `minOcc=1` and `maxOcc=unbounded`. The plus sign indicates complex content (i.e. at least one element or attribute child). Element name is `xs:field`.
Note: A global element can be referenced from within simple and complex type definitions, thus enabling you to re-use a global declaration at multiple locations in your schema. You can create a reference to a global element in two ways: (i) by entering a name for the local element that is the same as that of the global element; and (ii) by right-clicking the local element and selecting the option **Reference** from the context menu. You can view the definition of a global element by holding down **Ctrl** and double-clicking the element. Alternatively, right-click, and select **Go to Definition**. If you create a reference to an element that does not exist, the element name appears in red as a warning that there is no definition to refer to.

- Complex type

Details: The irregular hexagon with a plus sign indicates a complex type. The complex type shown here has the name `keybase`. This symbol indicates a global complex type. A global complex type is declared in the Schema Overview, and its content model is typically defined in Content Model View. A global complex type can be used either as (i) the data type of an element, or (ii) the base type of another complex type by assigning it to the element or complex type, respectively, in the Details entry helper (in either Content Model View or in Schema Overview).

 © 2011 Altova GmbH



The `keybase` complex type shown above was declared in Schema Overview with a base type of `xs:annotated`. The base type is displayed as a rectangle with a dashed gray border and a yellow background color. Then, in Content Model View, the child elements `xs:selector` and `xs:field` were created. (Note the tiny arrows in the bottom left corner of the `xs:selector` and `xs:field` rectangles. These indicate that both elements reference global elements of those names.)

A local complex type is defined directly in Content Model View by creating a child element or attribute for an element. There is no separate symbol for local complex types.

Please note: The base type of a content model is displayed as a rectangle with a dashed gray border and a yellow background color. You can go to the content model of the base type by double-clicking its name.

- Model group

Details: The irregular octagon with a plus sign indicates a model group. A model group allows you to define and reuse element declarations.
Note: When the model group is declared (in Schema Overview) it is given a name. You subsequently define its content mode (in Content Model View) by assigning it a child compositor that contains the element declarations. When the model group is used, it is inserted as a child, or inserted or appended within the content model of some other component (in Content Model View).

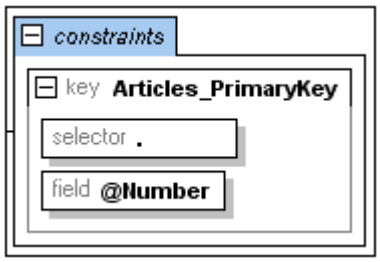
- Wildcards

Details: The irregular octagon with `any` at left indicates a wildcard.
Note: Wildcards are used as placeholders to allow elements not specified in the schema or from other namespaces. `##other` = elements can belong to any namespace other than the target namespace defined in the schema; `##any` = elements can belong to any namespace; `##targetNamespace` = elements must belong to the target namespace defined in the schema; `##local` = elements cannot belong to any namespace; *`anyURI`* = elements belong to the namespace you specify.

- Attributes

Details: Indicated with the word *'attributes'* in italics in a rectangle that can be expanded. Each attribute is shown in a rectangle with a (i) dashed border if the attribute is options, or (ii) a solid border if the attribute is required (mandatory).
Note: Attributes can be edited in the Details Entry Helper. Attributes can be displayed in the Content Model View diagram or in a pane below the Content Model View. You can toggle between these two views by clicking the Display Attributes icon. When attributes are displayed in the Content Model View diagram, all attributes of a single element are displayed in a rectangle with a dashed border. To change the order of attributes of an element, drag the attribute outside the containing box and drop when the arrow appears at the required location.

- Identity constraints

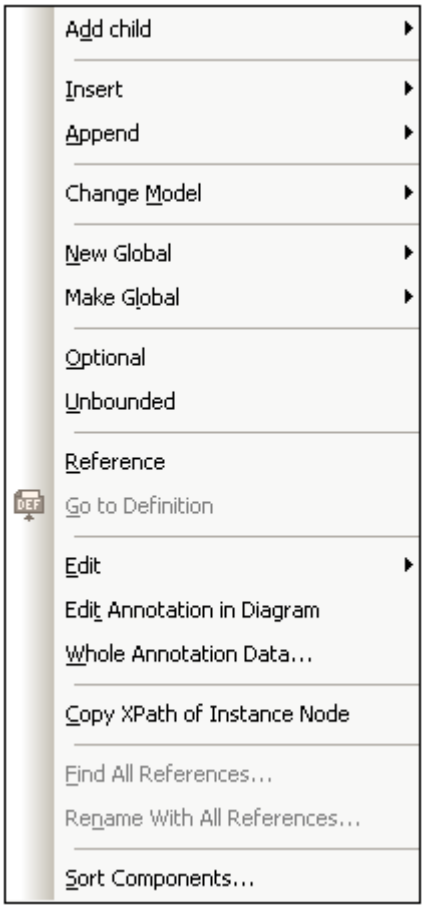


Details: Indicated with the word '*constraints*' in italics in a rectangle that can be expanded.
Note: The identity constraints listed in the content model of a component show constraints as defined with the `key` and `keyref` elements, and with the `unique` element. Identity constraints defined using the `ID` datatype are not shown in the content model diagram, but in the Details Entry Helper. Identity constraints can be displayed and edited in the Content Model View or in the Identity Constraints tab of Schema Overview. In Content Model View, you can toggle the Constraints box on and off with the Display Constraints icon..

Please note:

- Property descriptor lines you have defined in the Schema Display Configuration dialog can be turned on and off by clicking the Add Predefined Details toolbar icon.
- You can toggle Attributes and Identity Constraints to appear either in the diagram of the content model itself or in a pane below the Content Model View by clicking the Display in Diagram icons for attributes and constraints, and respectively.
- In Content Model View, you can jump to the content model view of any global component within the current content model by holding down the **CTRL** key and double-clicking the required component. You can go to the content model of a base type by double-clicking the name of the base type.

Other editing operations in Content Model View
Editing operations in Content Model View are carried out via the context menu (*see screenshot*) that appears when you right-click within Content Model View. A description of the operations are given below.



Adding child compositors/components and inserting/appending compositors/components

1. Right-click the compositor or component. This opens the context menu (with only the allowed operations enabled).
2. Select the required operation from the context menu.

Changing a compositor

1. Right-click the compositor you want to change.
2. Select the context menu option **Change Model** and, from the sub-menu, select the compositor to which you want to change. (The currently selected compositor is checked.) If a compositor is not allowed at that point, it is disabled.

Creating global components

- To create a new global component, right-click anywhere in Content Model View, select **New Global**, and, from the sub-menu, the required component.
- To make a local element a global element or global complex type, right-click the local element, select **Make Global**, and, from the sub-menu, select either **Element** or **Complex type**. If any of these components cannot legally be created, then it is disabled.

Changing the occurrence definition
You can toggle the minimum and maximum occurrences values of a compositor between `0` and `1` (for `minOccurs`) and `1` and `unbounded` (for `maxOccurs`), respectively.

Do this as follows:

1. Right-click the compositor or component for which the occurrence value has to be changed.
2. Select the context menu option **Optional** to set `minOccurs` to `0`, deselect **Optional** to set `minOccurs` to `1`. Select the context menu option **Unbounded** to set `maxOccurs` to `unbounded`, deselect **Unbounded** to set `maxOccurs` to `1`. A check mark appears to the left of the respective menu item when that menu item is selected.

Toggling between local definition and global definition
If a global element exists that has the same name as a local element, then you can toggle between referencing the global definition and using the local definition.

Do this as follows:

1. Right-click the element.
2. Select the context menu option **Reference**. If the global element is referenced, then the menu item is checked. If the local definition is used, the **Reference** item in the menu is not checked.

Jumping to another definition
When you are within a content model, you can jump to the definition of any global component that is contained in that content model.

Do this as follows:

1. Right-click the global component. The global component could be the yellow rectangle of a base type; an element that references a global element; or a model group.
2. Select the context menu option **Go to Definition**. This opens the Content Model View of that global component.

Alternatively, double-click the name of the base type, or press **Ctrl** and double-click the referencing element or the model group.

Editing element names and types

1. Right-click the element.
2. Select the context menu option **Edit | Name** and edit the name, or select **Edit | Type** and edit the type.

Alternatively, double-click the element name, and type in the change.

Creating and editing documentation for a compositor or component
You can add documentation to individual compositors and components as a guide for schema editors.

Do this as follows:

1. Right-click the compositor or component.

 © 2011 Altova GmbH

2. Select the context menu option **Edit Annotation**. This highlights the documentation space below the compositor/component, in which you can enter descriptive text about the compositor or component. In Text View, the `annotation` and `annotation/documentation` elements will have been created and the `documentation` element will contain the descriptive text you enter.

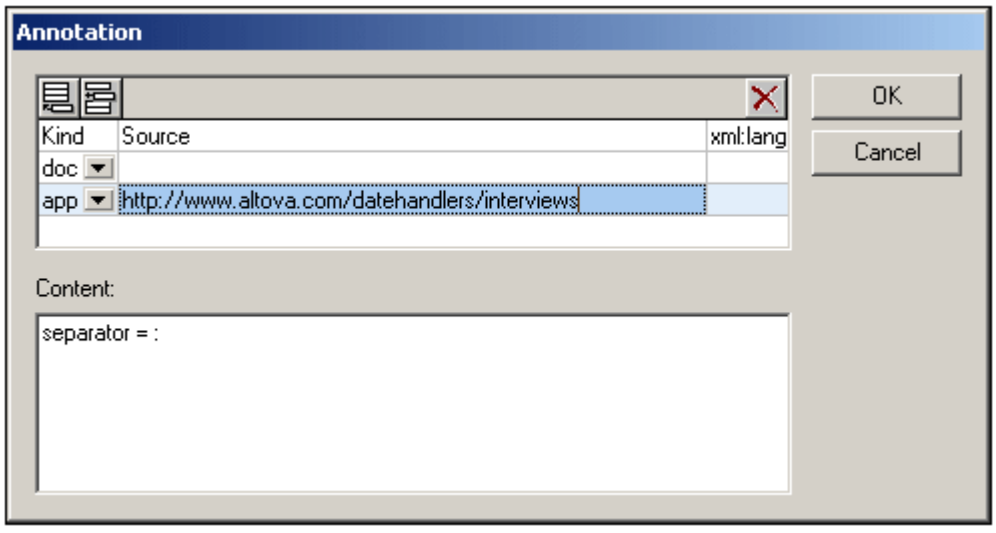
Alternatively, you can right-click the compositor or component and select Whole Annotation Data. In the Annotation dialog that opens, you can append or insert a documentation item and enter content for it.

In order to edit pre-existing documentation text, you can use either of the two methods described above, but a quicker method is to double-click the annotation in the diagram and edit directly.

Creating and editing application info for a compositor or component

1. Right-click the compositor or component.

2. Select the context menu option **Whole Annotation Data**. The Annotation dialog box opens (*see screenshot below*). If annotation (either documentation or appinfo) exists for that element, then this is indicated by a corresponding row in the dialog.



3. To create an `appinfo` element, click the Append or Insert icon at top left to append or insert a new row, respectively.
4. In the Kind field of the new row, select the `app` option from the dropdown menu.
5. In the Content pane of the dialog, enter the script or info that you want to have processed by a processing application.
6. Optionally, in the Source field, you can enter a source URI where further information can be made available to the processing application.

Copy XPath of Instance Node

The context menu command, **Copy XPath of Instance Node**, is enabled for elements and attributes defined within a global element or complex type. It copies to the clipboard an XPath expression that locates the selected node.

Find All References and Rename with All References

The two context menu commands, **Find All References** and **Rename with All References**, is enabled for global elements. These, respectively, find and rename the selected global component in the active document and, optionally, in all schema files related to the active document. See Finding and Renaming Globals for details.

Sorting declarations and references

The **Sort** command is enabled when a node (element or attribute) has multiple declarations or references. The sorting works in the same way as the sorting described in the section, Schema Overview.

Using keyboard shortcuts in Content Model View

You can copy and paste elements in Content Model View using the shortcuts **Ctrl-c** and **Ctrl-v**.

To copy and paste elements:

1. Select the elements you want to copy. To select one element, click on it. To select more than one element, click on the first element and use **SHIFT** and the down arrow key to select further elements.
2 Press **Ctrl**-**c**. The elements are copied to the clipboard.
3. Select the compositor you want to copy the elements to.
4. Press **Ctrl**-**v**. The elements are pasted as child elements of the compositor.

To copy and paste elements in reverse order:

1. Click on the lowermost element you want to copy and use **SHIFT** and the up arrow key to select further elements.
2. Press **Ctrl**-**c**. The elements are copied to the clipboard.
3. Select the compositor you want to copy the elements to.
4. Press **Ctrl**-**v**. The elements are pasted such that the element that was the uppermost element is now the lowermost element.

About XML Schema annotations

XML Schema annotations are held in the `annotation` element. There are two types of annotation, both of which are elements of the annotation element:

- compositor or component documentation, which contains information that could be useful for editors of the schema and is contained in the `documentation` child element of `annotation`.
- application information, which allows you to insert a script or information that a processing application may use; this information is contained in the `appinfo` child element of `annotation`.

Given below is the text of an annotation element. It is based on the example in the description of creating documentation and application information given above.

```
<xs:element name="session_date" type="xs:dateTime" nillable="true">
   <xs:annotation>
      <xs:documentation>Date and time when interview was
held</xs:documentation>
```

 © 2011 Altova GmbH

```
      <xs:appinfo
source="http://www.altova.com/datehandlers/interviews">separator =
:</xs:appinfo>
   </xs:annotation>
</xs:element>
```

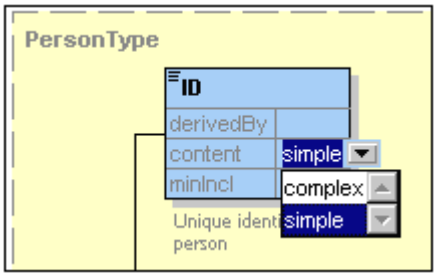
Comments and processing instructions
In XML Schema documents, comments and processing instructions within simple types and complex types are collected and moved to the end of the enclosing object. In such cases, it is therefore recommended that you use annotations instead of comments.

Configuring the content model view
You can configure the content model view for the entire schema in the Schema display configuration dialog (**Schema design | Configure view**). For details about configuration options, see the Configure view section later in the User Reference. Note that the settings you define here apply to the whole schema, and to the schema documentation output as well the printer output.

Changing component properties directly in the content model
If the Content Model View is configured so that components are displayed with property descriptor lines (additional information about components) in the component box, then you can edit this information and so change the properties of components. The property descriptor lines you have defined can be turned on and off by clicking the Add Predefined Details toolbar icon. You can toggle between a view containing the defined properties and a view not containing them.

To edit component properties:

1. Double-click the (component's) information field that you want to edit, and start entering or editing data. If a predefined option is available, then a drop-down list can be opened and the appropriate entry selected. Otherwise simply enter the required value.



2. Press **Enter** to confirm. The Details entry helpers will be updated to reflect your changes.

Alternatively, you can edit a component's properties in the Details entry helper, and changes will be reflected in the placeholder fields—if these are configured to be displayed.

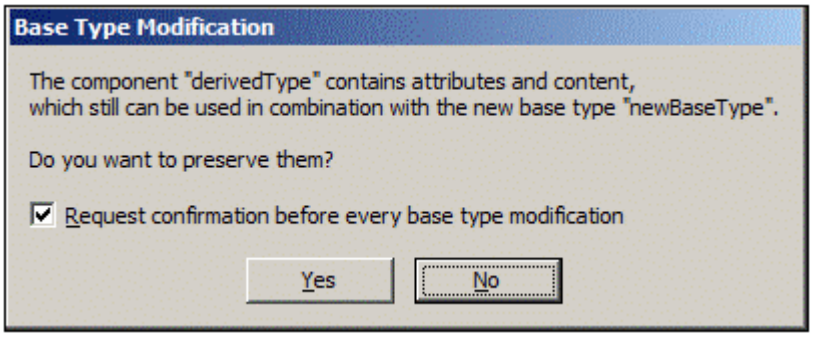
Attributes and Identity Constraints
Attributes and Identity constraints can appear in a pane below the Content Model View or as boxes in the Content Model View itself. You can toggle between these views by clicking the icon. For a description of how to insert and edit attributes and identity constraints, see Defining attributes for components and Defining identity constraints for components, respectively.

3.3.3 Base Type Modification

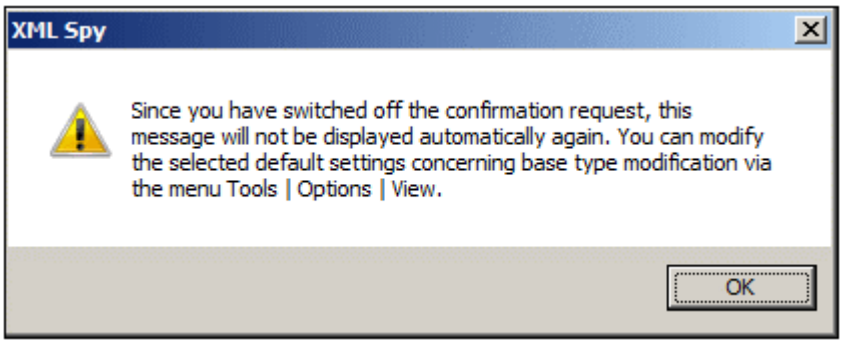
If the base type of a derived type is changed in Schema View, content, attributes, facets and sample values defined within the derived type can be handled in one of two ways:

- They can be preserved if they are still applicable in combination with the new base type.
- They can be removed automatically whether or not they are still applicable in combination with the new base type.

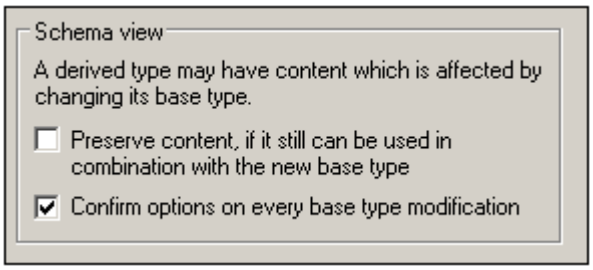
When changing the base type of a derived type which contains content, attributes, facets or sample values the Base Type Modification dialog (*screenshot below*) is displayed.



If the Request Confirmation check box is de-selected a pop-up (*screenshot below*) indicates that the confirmation can be turned on again in the View tab of the Options dialog (**Tools | Options | View**).



In the Schema View pane (*screenshot below*) of the View tab of the Options dialog (**Tools | Options | View**), you can specify whether content should be preserved and whether user confirmation is required for every base type modification.



Check the respective check boxes to preserve content and require confirmation if you wish

these to be the default options.

3.3.4 Entry Helpers in Schema View

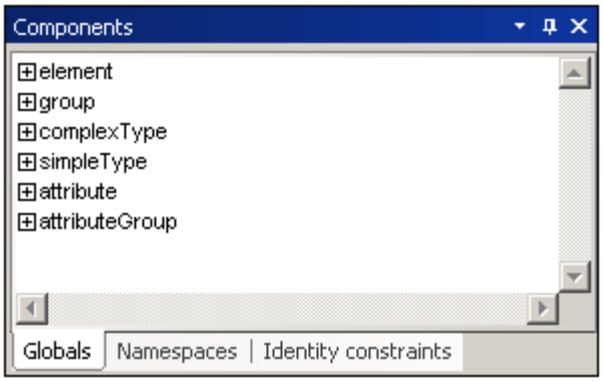
There are three Entry Helpers in Schema View: Component Navigator, Details Entry Helper, and Facets Entry Helper. The entry helpers are the same in both Schema Overview and Content Model View. They are described below.

Note: In Standard Edition, Schema View is available as a read-only view. Editing in Schema View is available in the Enterprise and Professional editions.

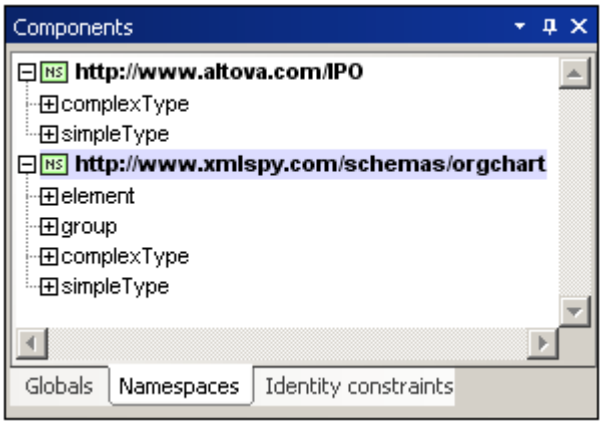
Component Navigator

The Component Navigator is an Entry Helper in **Schema View**. It serves three purposes:

- To organize global components in a tree view by component type and namespace *(see screenshots below)*. This provides organized overviews of all global components.
- To enable you to navigate to and display the Content Model View of a global component —if the component has a content model. If a component does not have a content model, the component is highlighted in the Schema Overview. Global components that are included or imported from other schemas are also displayed in the Component Navigator.
- To provide an overview of the identity constraints defined in the schema document. For a description of the Identity Constraints tab, see Identity Constraints Overview.



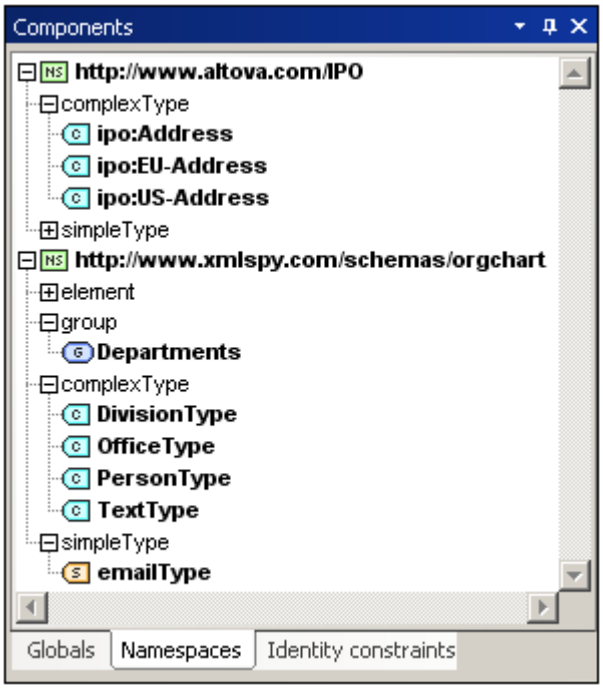
In the Globals tab *(above)* global components are grouped in a tree according to their component type. In the Namespace tab *(below)*, components are organized first according to namespace and then according to component type. Note that a component type is listed in a tree only if at least one component of that type exists in the schema.



In the tree display, global components are organized into the following six groups:

- Element Declarations (Elements)
- Model Groups (Groups)
- Complex Types
- Simple Types
- Attribute Declarations (Attributes)
- Attribute Groups

Expanding a component-type group in the tree displays all the components in that group *(see screenshot)*. This enables you to easily navigate to a required component.



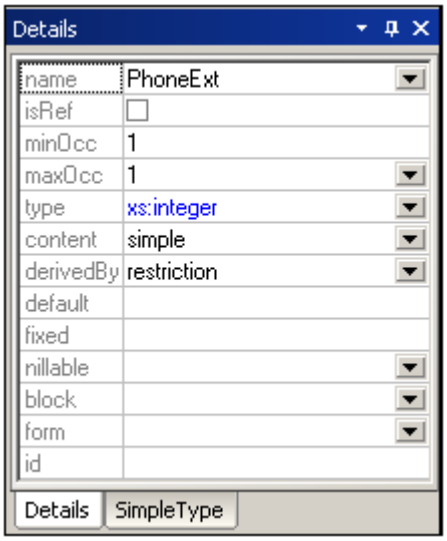
If a component has a content model (i.e., if it is an Element, Group, or Complex Type), double-clicking it will cause the content model of that component to be displayed in Content Model View (in the Main Window). If the component does not have a content model (i.e. if it is a Simple Type, Attribute, or Attribute Group), then the component is highlighted in the Schema Overview (in the Main Window).

 © 2011 Altova GmbH

Please note: If the component is in an included or imported schema, then the included/imported schema is opened (if it is not already open), and either the component's content model is displayed in Content Model View or the component is highlighted in Schema Overview.

Details Entry Helper

The Details Entry Helper is available in **Schema View**. It displays editable information about the compositor or component currently selected in the Main Window. If you are editing a schema file which contains database extensions, an additional tab with information about the DB extensions may be visible.

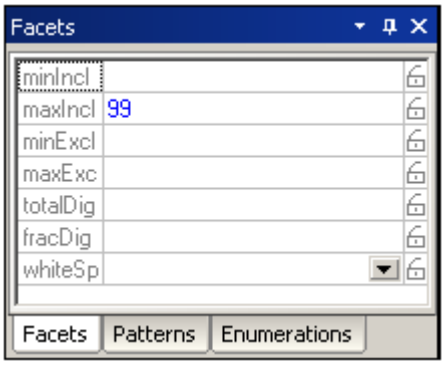


To change the properties of the currently selected compositor or component, double-click the field to be edited and edit or enter text directly. If a combo box is available in the field to be edited, select the desired value; this value is entered in the field.

Changes you make via the Details Entry Helper are immediately reflected in the content model diagram.

Facets Entry Helper

The Facets Entry Helper is available in the **Schema View**, and enables you to enter the values of facets, patterns, and enumerations.



To change facets, patterns, or enumerations in the Facets Entry Helper:

1. Select the required tab (Facets, Patterns, or Enumerations)
2. If a combo box is present, select a value from the drop-down menu. Alternatively, double-click a row, and edit or enter text directly.

Note:

- You can use the cut, copy and paste shortcuts (CTRL+X, CTRL+C, CTRL+V, respectively) to copy the patterns and enumerations of one component to another component. In the Facets Entry Helper, select the pattern/s or enumeration/s to copy, cut or copy the selection, then click in the Facets Entry Helper window of the target component, and paste.
- Sample values can be used when generating an XML file from the schema (with the menu command **DTD/Schema | Generate Sample XML File**).

3.3.5 Identity Constraints

Identity constraints can be defined for a global component via two entry points:

- In Schema Overview, select a global component and define identity constraints in the Identity Constraints tab (at the bottom of the view and below the main pane).
- In the Content Model View of a global component. Content Model View provides a graphical representation of the identity constraints and drag-and-drop editing functionality, neither of which is available when editing identity constraints in Schema Overview.

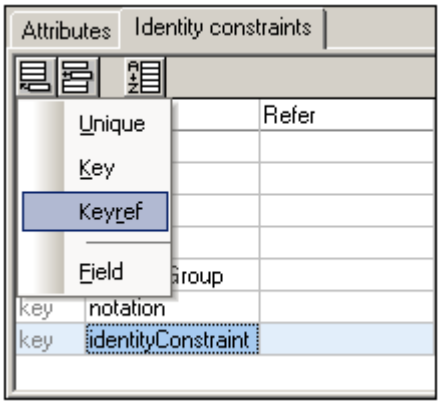
In this section, we describe these two mechanisms for creating and editing identity constraints. An overview of all identity constraints in the schema is available in the Identity Constraints tab of the Components entry helper; this is described further below.

Note: In Standard Edition, Schema View is available as a read-only view. Editing in Schema View is available in the Enterprise and Professional editions.

Identity constraints in Schema Overview

To define identity constraints for a global component in Schema Overview, do the following:

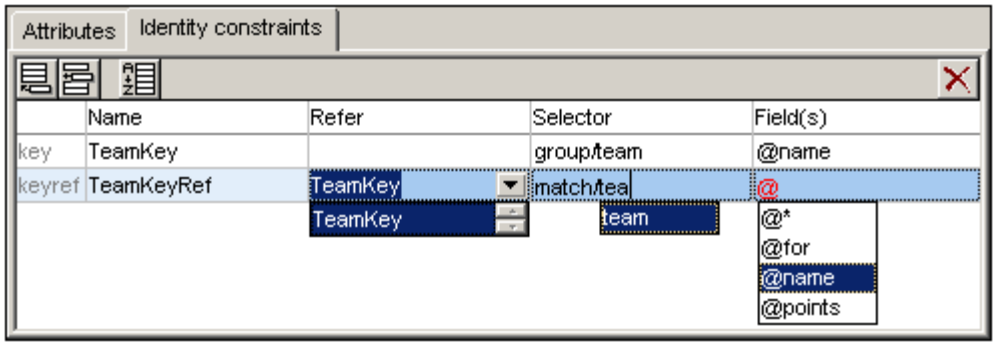
1. Select the global component in the global components list of Schema Overview.
2. In the Attributes/Identity Constraints pane, select the Identity Constraints tab.
3. Click the **Append** or **Insert** icon at the top left of the Identity Constraints tab.
4. From the popup that appears (*screenshot below*), select the type of identity constraint you want to append or insert: *Unique*, *Key*, or *Keyref*.



 © 2011 Altova GmbH

An entry is created in the Identity Constraints list.

5. In the newly created entry, enter the Identity Constraint's properties. Give the identity constraint a name (*see screenshot below*) and then define XPath expressions for the `selector` and `field` elements. For each location step in these XPath expression, a pop-up containing the available nodes appears (*see screenshot below*). For the `refer` element of the `keyref` kind of identity constraint, a dropdown list shows the available `key` elements (*see screenshot below*).



If you wish to define multiple field elements on an identity constraint, select the identity constraint, click the **Append** or **Insert** icon (*see Step 3 above*), and select **Field**. An empty Field entry box is added to the selected identity constraint. Enter the required XPath expression in the Field entry box.

Note: An ID can be assigned to any of these elements (that is, to the identity constraint, its selector, and/or field/s). To assign an ID, select the element and, in the Details entry helper, enter the ID in the `id` row.

Identity constraints in Content Model View

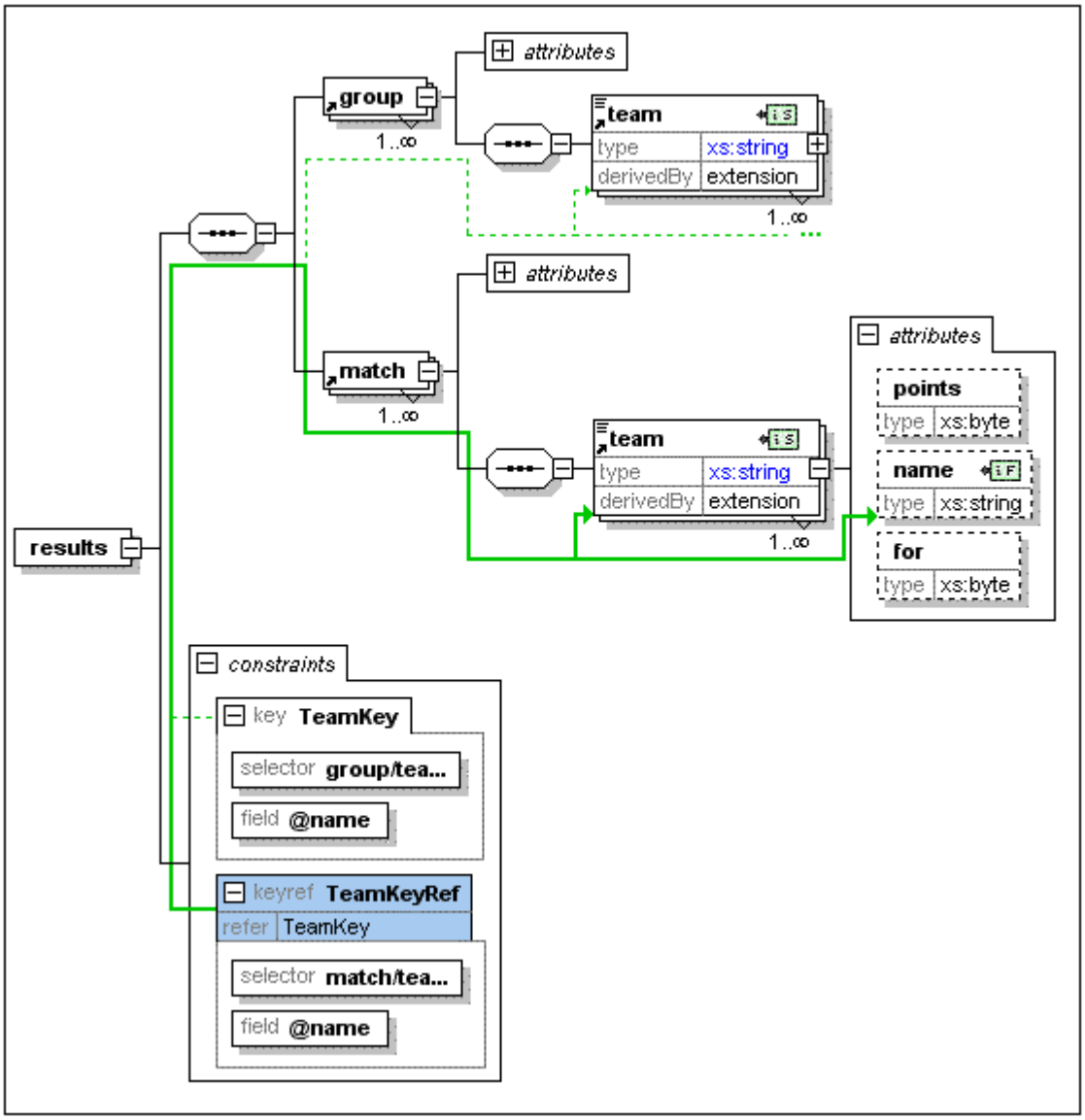
To view identity constraints in Content Model View, do the following:

1. In Schema Overview, select the global component for which you wish to create the identity constraint and switch to the Content Model View of that component.
2. Ensure that the display of identity constraints is toggled on in the Schema Display Configuration dialog (**Schema Design | Configure View**). This toggle switches on and off the display of the identity constraints box in the design (*see screenshot below*). You can also toggle on and off the display of the Constraints box with the Display Constraints icon in the Schema Design toolbar
3. Ensure that the Visualize ID Constraints icon in the Schema Design toolbar is toggled on. This ensures that relationship lines and ID constraint information are displayed in Content Model View (*see screenshot below*).

Note: The Visualize ID Constraints icon switches on ID Constraint editing and validating functionality in Schema View. If an XPath expression in an ID Constraint does not locate a node, an error or warning might be displayed. This error or warning is to alert you to the incorrect XPath expression; the XML Schema document itself is not invalid because of the incorrect XPath expression. To re-check the validity of the document without the XPath expression check, switch to Text View or Grid View and validate. You can also disable validation in Schema View by toggling the Visualize ID Constraints icon off.

The screenshot below shows the graphical display of identity constraints. There are two identity constraints in the identity constraints box: `TeamKey` and `TeamKeyRef`. The properties of `TeamKeyRef` are highlighted with a solid green line (since `TeamKeyRef` is selected). The

properties of `TeamKey` (unselected) are shown with a dotted green line. For each identity constraint the node selected by the XPath expression for `selector` and `field` are show with the icons and respectively. If a node is collapsed, as is the case with the `field` element of the `TeamKey` constraint, the relationship line ends with an ellipsis (*see screenshot below*).



To create an identity constraint in Content Model View, do the following:

1. In the Constraints box, right-click the Constraints entry or the name of an existing identity constraint.
2. Select **Insert** or **Append**, and then select the kind of identity constraint to insert (at the top of the list of existing constraints) or append (at the bottom of the list of existing constraints). The identity constraint will be created with an empty `selector` element and an empty `field` element.
3. The XPath expression can be entered in the `selector` and `field` element boxes in one of two ways: (i) By typing it in, or (ii) By dragging the target node into the `selector` or `field` element box and dropping it when the box becomes highlighted; the XPath expression will be created automatically.
4. To add an additional `field` element to an identity constraint, either right-click the

 © 2011 Altova GmbH

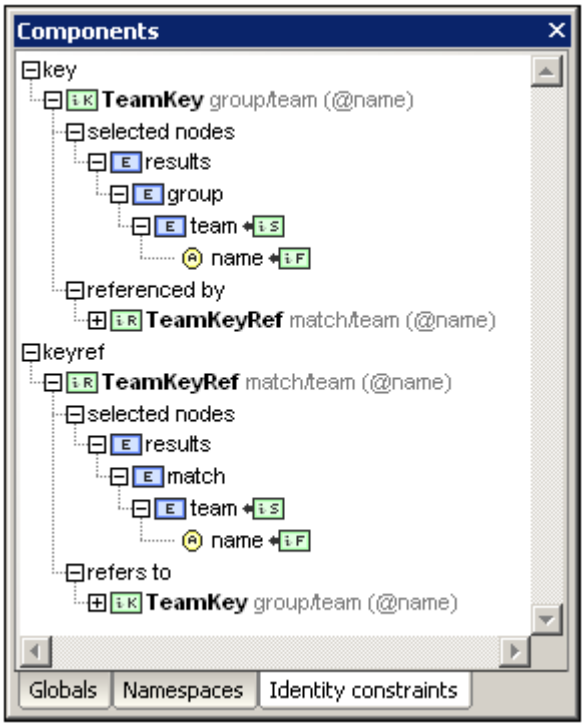
constraint name and select **Add child | Field** from the context menu, or right-click the `selector` or `field` element and select **Insert | Field** or **Append | Field** from the context menu.

5. To enter an ID attribute on any element in a constraint, select that element and enter a value in the `id` row of the Details entry helper.

Note: Identity constraint information can be toggled on and off with the Enable ID Constraints Information icon in the Schema Design toolbar. The display of the entire Constraint box can be toggled on and off in the Schema Display Configuration dialog (**Schema Design | Configure View**) or with the Display Constraints icon in the Schema Design toolbar.

Identity constraints overview

The Identity Constraints tab of the Components entry helper (*screenshot below*) provides an overview of a document's identity constraints. In this tab, identity constraints are listed by the kind of identity constraint (`unique`, `key`, `keyref`) and displayed as an expandable/collapsible tree.



Entries in bold are present in the current schema, while those in normal face are present in sub-schemas. Double-clicking an entry in the Identity Constraints tab selects that schema component in Content Model View.

The following context menu commands are available when an item in the Identity Constraints tab is selected:

- *Show in Diagram*: selects the schema component in Content Model View.
- *Show Selector/Field Target in Diagram*: selects, in Content Model View, the schema component targeted by the selector or field of the identity constraint. In the case of multiple fields, a dialog prompts the user for the required field.
- *Go to Identity Constraint*: selects the identity constraint in Schema Overview.

- *Expand/Collapse All*: expands or collapses the tree, respectively.

3.3.6 Smart Restrictions

When restricting a complex type, parts of the content model of the base type are rewritten in the derived type. This can be confusing if the content model is complex because while editing the derived type it might be hard to correctly remember exactly what the content model of the base type looks like.

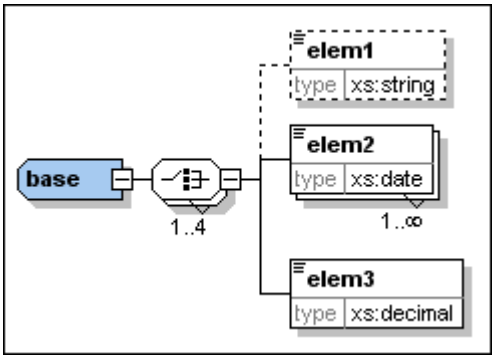
Smart Restrictions combine and correlate the two content models in the graphical view of the derived content model. In the derived complex type, all particles of the base complex type, and how they relate to the derived type, can be seen. Additionally, Smart Restrictions provide visual hints to show you all possible ways to restrict the base type. This makes it easy to correctly restrict the derived type.

To switch on Smart Restrictions:

- Click the Smart Restrictions icon . in the Schema Design toolbar.

The example that follows illustrates the features of Smart Restrictions. Note that in Standard Edition, Grid View is available as a read-only view. Editing in Schema View is available in the Enterprise and Professional editions.

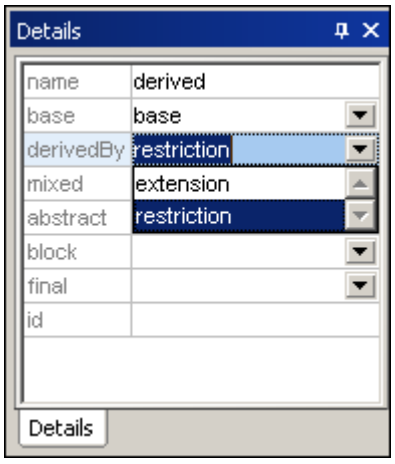
The following complex type is the base type used in this example:



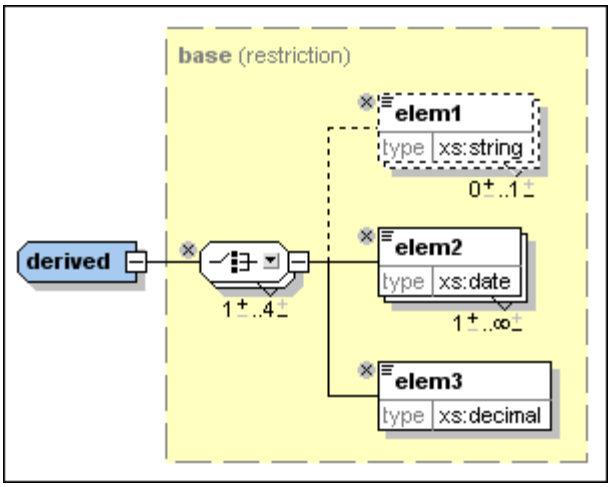
The complex type "derived" is derived from the "base" type as follows:

1. Create a new complex type in the schema and call it "derived".
2. In the Details Entry Helper select "base" from the **base** drop-down list and "restriction" from the **derivedBy** drop-down list.

 © 2011 Altova GmbH

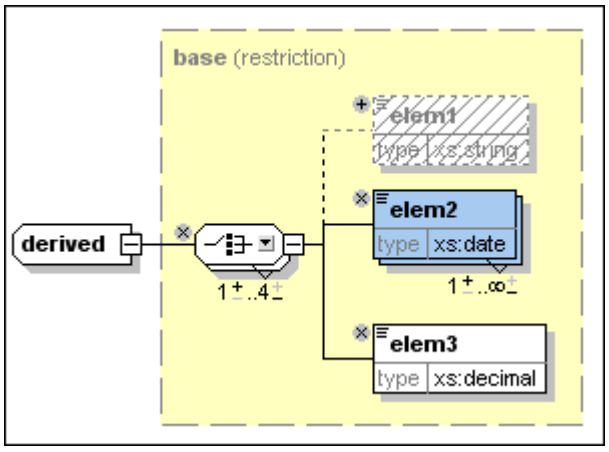


With Smart Restrictions switched on, the new derived type looks like this:



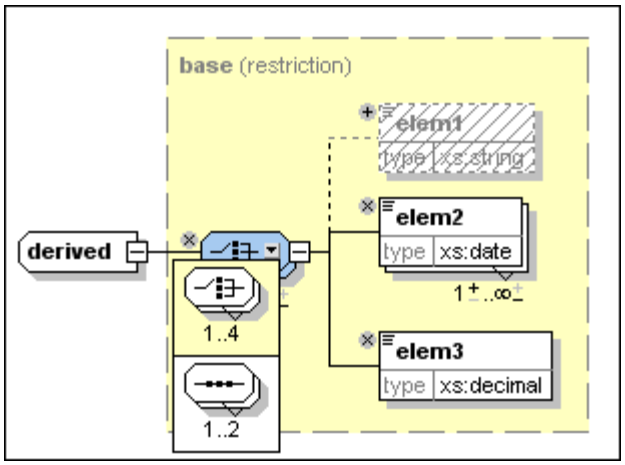
Notice the following controls that can be used to restrict the derived type in this example:

- Use this icon to remove elements that are in the base type from the derived type. Here, elem1 has been deleted. To add it again, click this icon .

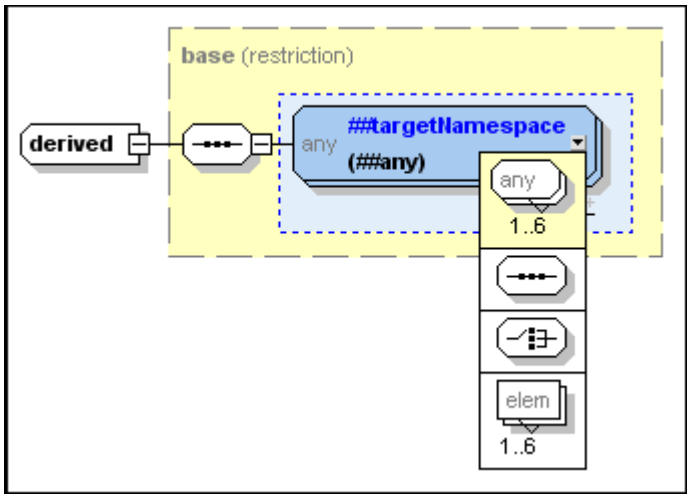


- Click the down arrow on the Choice compositor to get the following list, which allows

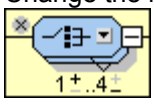
you to change the Choice model group to a Sequence model group:



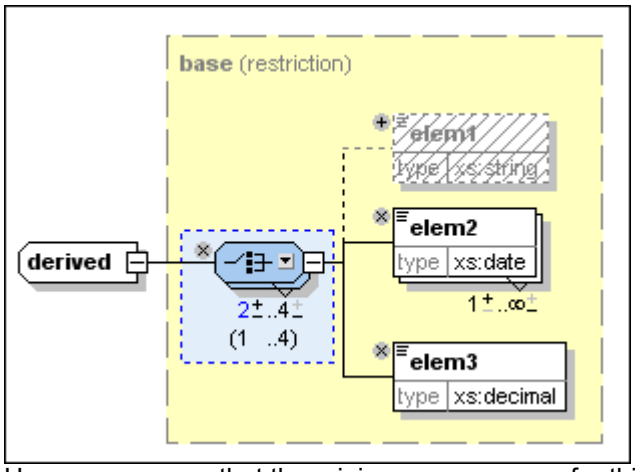
It is also possible to change wildcards in the same way, as seen in this example:



For a complete list of which particles can be replaced by which other particles, see the XML schema specification.

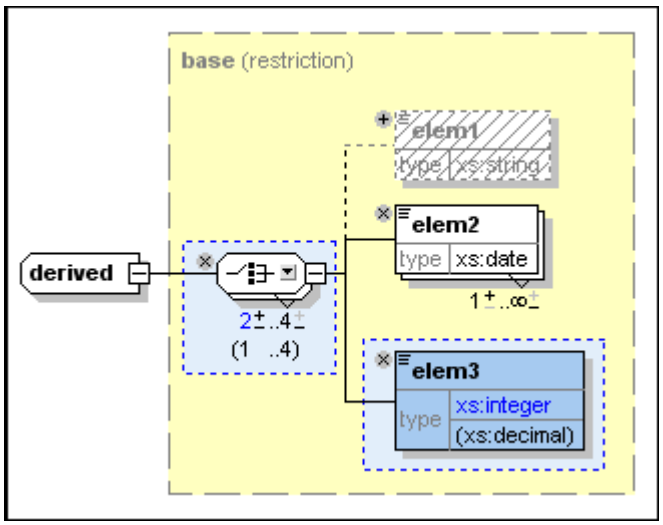
- Change the number of occurrences of the model group using the following control  to increase the minimum number of occurrences by clicking the plus sign over the "1", or to decrease the maximum number of occurrences by clicking the minus sign under "4". These controls are shown if the occurrence range in the base describes a real range (e.g., 2-5) and not a certain amount (e.g. 4-4). They are also displayed if the occurrence range is wrong.

© 2011 Altova GmbH

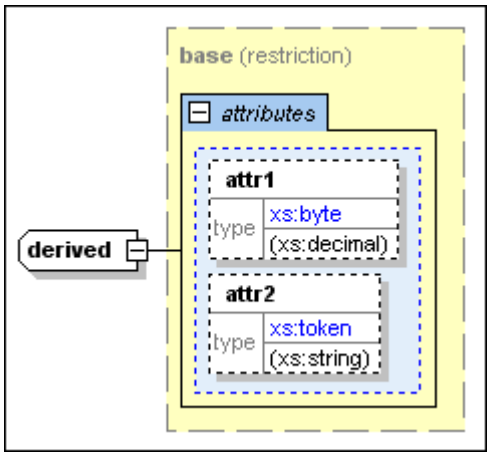


Here you can see that the minimum occurrence for this element has been changed to 2. Notice that the model group now has a blue background, which means that it is no longer the same as the model group in the base complex type. Also, the permitted occurrence range of the model group in the base particle is now displayed in parentheses.

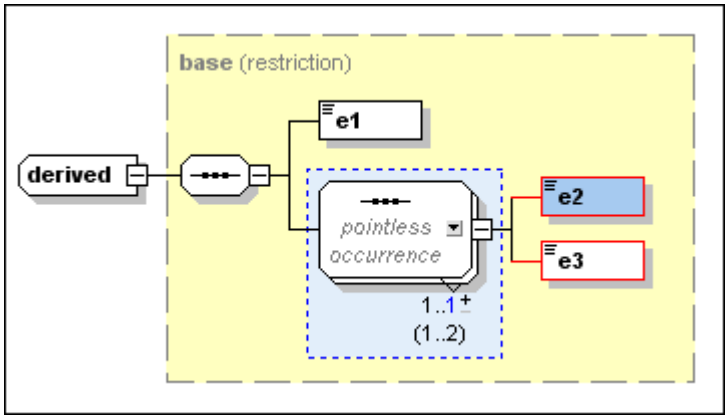
- It is possible to change the data types of attributes or elements if the new data type is a valid restriction of the base data type as defined in the XML schema specification. For example, you can change the data type of elem3 in the "derived" data type from decimal to integer. After you do this, the element has a blue background to show that is different from the element in the base type, and the type that the element has in the base type is displayed in parentheses:



This example shows attributes whose data types have been restricted in the derived complex type:



- Smart Restrictions alert you to *pointless occurrences* in the content model. A pointless occurrence happens, for example, when a sequence that is present in the content model is unnecessary. This example shows a pointless occurrence:



Please note: Pointless occurrences are only shown if the content model contains an error. It is possible for a content model to contain a pointless occurrence and be valid, in which case the pointless occurrence is not explicitly shown in order to avoid confusion.

See the XML schema specification for more information about pointless occurrences.

3.3.7 xml:base, xml:id, xml:lang, xml:space

The namespace `http://www.w3.org/XML/1998/namespace` is, according to the XML Namespaces specification, bound by definition to the `xml:` prefix. What this means is that this is the namespace that must be used with the `xml:` prefix and that is reserved for it. There are four attributes in this namespace that can be children of any XML element in any XML document (schema or instance):

- `xml:base` (for setting the base URI of an element)
- `xml:id` (for specifying the unique ID of an element)
- `xml:lang` (for identifying the language used within that element)
- `xml:space` (for specifying how whitespace in the element should be handled)

In Schema View, once the XML Namespaces namespace has been imported into the XML Schema document, these four `xml:` attributes can be referenced for use on any element in the

 © 2011 Altova GmbH

schema.

In order to declare one of these attributes on an element, do the following:

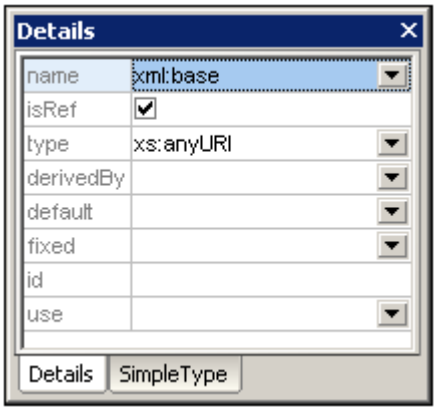
1. Declare the XML Namespaces namespace for that schema document and bind the namespace to the `xml:` prefix. When any of the four `xml:` attributes is used in the document, its name would then be expanded to include the correct namespace part.
2. Import the XML Namespaces namespace. XMLSpy's validator will recognize the namespace and make the four `xml:` attributes available as global attributes, which can be referenced within that schema.
3. Insert the required `xml:` attribute as the child of an element. The attribute is declared as a reference to the "imported" global attribute.

Declare the XML Namespaces namespace
You can declare the XML Namespaces namespace (`http://www.w3.org/XML/1998/namespace`) by entering it via the Schema Settings dialog, where all namespaces declared for that schema are stored and can be edited. The namespace must be bound to the `xml:` prefix. (Alternatively, you could declare the namespace (with the `xml:` prefix) on the `xs:schema` element in Text View.)

Import the XML Namespaces namespace
In Schema Overview, create a global import declaration for the XML Namespaces namespace. Do this by clicking the Insert or Append icon at the top of the Schema Overview window and selecting **Import** from the menu that pops up. Enter the XML Namespaces namespace as the namespace to be imported. In Text View, the import declaration should look like this:

```
<xs:import namespace="http://www.w3.org/XML/1998/namespace"
schemaLocation="http://www.w3.org/XML/1998/namespace"/>.
```

Adding the `xml:` attribute
In Schema Overview, select the element for which the `xml:` attribute is to be added, and add an attribute for it. In the Details entry helper (*screenshot below*), click the down arrow of the name combo box and select the required `xml:` attribute, for example `xml:base`. When you are prompted whether you wish to reference the global attribute, click **Yes**. The attribute is added as a reference.



XInclude and `xml:base`
When XInclude's `include` element is replaced by the XML file specified in the `href` attribute of

the include element, the top-level element of the parsed XML document is included with an `xml:base` attribute. If this XML document is going to be validated, then the schema must define an xml:base attribute on the relevant element/s.

3.3.8 Back and Forward: Moving through Positions

The **Back** and **Forward** commands in Schema View enable you to move through previously viewed positions in Schema View. This is useful because, while clicking through schema components in Schema View, you might wish to view a previously viewed component. Clicking the **Back** button once in the toolbar takes you to the previously viewed position. By repeatedly clicking the **Back** button, you can view up to 500 of the last visited positions. After moving back through previous positions, you can move forward through these positions by using the **Forward** button in the toolbar.

The shortcut keys for the two commands are:

- ***Back:*** **Alt + Left Arrow**
- ***Forward:*** **Alt + Right Arrow**

Back/Forward versus Undo/Redo
Note that the **Back** and **Forward** commands are not the same as the **Undo (Ctrl+Z)** and **Redo (Ctrl+Y)** commands. These two sets of commands make up two different series of steps. Clicking the **Back** command once takes you to the previously viewed component as previously displayed. Clicking the **Undo** command once undoes the last editing change regardless of when that editing change was made.

Additional notes
Note the following points:

- The **Back** button enables you to re-view the previous 500 positions.
- The Back/Forward feature is enabled across schemas. If a schema has since been closed or is currently open in another view, it will be opened in Schema View or switched to Schema View, respectively.
- If a component that was viewed in a previous position is deleted, then that component will not be able to be viewed. If such a component was part of a previous position, this position will be displayed without the deleted component. If the component comprised the entire position, the entire position will be unavailable, and clicking the **Back** button at this point in the Back series will take you to the position previous to the unavailable position.

 © 2011 Altova GmbH

3.4 Authentic View

Authentic View is enabled by clicking the Authentic tab of the active document. If no SPS has been assigned to the XML document, you are prompted to assign one. You can assign an SPS at any time via the **Authentic | Assign a Stylevision Stylesheet** command.

This section provides:

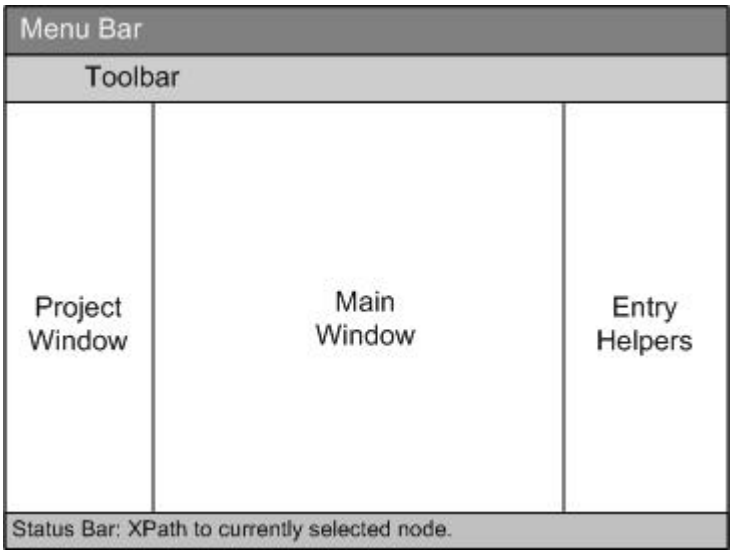
- An overview of the interface
- A description of the toolbar icons specific to Authentic View
- A description of viewing modes available in the main Authentic View window
- A description of the Entry Helpers and how they are to be used
- A description of the context menus available at various points in the Authentic View of the XML document

Additional sources of Authentic View information are:

- An Authentic View Tutorial, which shows you how to use the Authentic View interface. This tutorial is available in the documentation of the Altova XMLSpy and Altova Authentic Desktop products (see the Tutorials section), as well as online.
- For a detailed description of Authentic View menu commands, see the User Reference section of your product documentation.

3.4.1 Overview of the GUI

Authentic View has a menu bar and toolbar running across the top of the window, and three areas that cover the rest of the interface: the Project Window, Main Window, and Entry Helpers Window. These areas are shown below.



Menu bar
The menus available in the menu bar are described in detail in the User Reference section of your product documentation.

Toolbar
The symbols and icons displayed in the toolbar are described in the section, Authentic View

toolbar icons.

Project window
You can group XML, XSL, XML schema, and Entity files together in a project. To create and modify the list of project files, use the commands in the **Project** menu (described in the User Reference section of your product documentation). The list of project files is displayed in the Project window. A file in the Project window can be accessed by double-clicking it.

Main window
This is the window in which the XML document is displayed and edited. It is described in the section, Authentic View main window.

Entry helpers
There are three entry helper windows in this area: Elements, Attributes, and Entities. What entries appear in these windows (Elements and Attributes Entry Helpers) are context-sensitive, i.e. it depends on where in the document the cursor is. You can enter an element or entity into the document by double-clicking its entry helper. The value of an attribute is entered into the value field of that attribute in the Attributes Entry Helper. See the section Authentic View Entry Helpers for details.

Status Bar
The Status Bar displays the XPath to the currently selected node.

Context menus
These are the menus that appear when you right-click in the Main Window. The available commands are context-sensitive editing commands, i.e. they allow you to manipulate structure and content relevant to the selected node. Such manipulations include inserting, appending, or deleting a node, adding entities, or cutting and pasting content.

3.4.2 Authentic View Toolbar Icons

Icons in the Authentic View toolbar are command shortcuts. Some icons will already be familiar to you from other Windows applications or Altova products, others might be new to you. This section describes icons unique to Authentic View. In the description below, related icons are grouped together.

Show/hide XML markup
In Authentic View, the tags for all, some, or none of the XML elements or attributes can be displayed, either with their names (large markup) or without names (small markup). The four markup icons appear in the toolbar, and the corresponding commands are available in the **Authentic** menu.

Hide markup. All XML tags are hidden except those which have been collapsed. Double-clicking on a collapsed tag (which is the usual way to expand it) in Hide markup mode will cause the node's content to be displayed and the tags to be hidden.

Show small markup. XML element/attribute tags are shown without names.

Show large markup. XML element/attribute tags are shown with names.

 © 2011 Altova GmbH

Show mixed markup. In the StyleVision Power Stylesheet, each XML element or attribute can be specified to display (as either large or small markup), or not to display at all. This is called mixed markup mode since some elements can be specified to be displayed with markup and some without markup. In mixed markup mode, therefore, the Authentic View user sees a customized markup. Note, however, that this customization is created by the person who has designed the StyleVision Power Stylesheet. It cannot be defined by the Authentic View user.

Editing dynamic table structures

Rows in a **dynamic SPS table** are repetitions of a data structure. Each row represents an occurrence of a single element. Each row, therefore, has the same XML substructure as the next.

The dynamic table editing commands manipulate the rows of a dynamic SPS table. That is, you can modify the number and order of the element occurrences. You cannot, however, edit the columns of a dynamic SPS table, since this would entail changing the substructure of individual element occurrences.

The icons for dynamic table editing commands appear in the toolbar, and are also available in the **Authentic** menu.

Append row to table

Insert row in table

Duplicate current table row (i.e. cell contents are duplicated)

Move current row up by one row

Move current row down by one row

Delete the current row

Please note: These commands apply only to **dynamic SPS tables**. They should not be used inside static SPS tables. The various types of tables used in Authentic View are described in the Using tables in Authentic View section of this documentation.

Creating and editing XML tables

You can insert your own tables should you want to present your data as a table. Such tables are inserted as XML tables. You can modify the structure of an XML table, and format the table. The icons for creating and editing XML tables are available in the toolbar, and are shown below. They are described in the section XML table editing icons.

The commands corresponding to these icons are **not available as menu items**. Note also that

for you to be able to use XML tables, this function must be enabled and suitably configured in the StyleVision Power Stylesheet.

A detailed description of the types of tables used in Authentic View and of how XML tables are to be created and edited is given in Using tables in Authentic View.

Text formatting icons
Text in Authentic View is formatted by applying to it an XML element or attribute that has the required formatting. If such formatting has been defined, the designer of the StyleVision Power Stylesheet can provide icons in the Authentic View toolbar to apply the formatting. To apply text formatting using a text formatting icon, highlight the text you want to format, and click the appropriate icon.

DB Row Navigation icons

The arrow icons are, from left to right, Go to First Record in the DB; Go to Previous Record; Open Go to Record # dialog; Go to Next Record; and Go to Last Record.

This icon opens the Edit Database Query dialog in which you can enter a query. Authentic View displays the queried record/s.

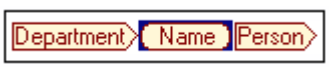
XML database editing
The **Select New Row with XML Data for Editing** command enables you to select a new row from the relevant table in an XML DB, such as IBM DB2. This row appears in Authentic View, can be edited there, and then saved back to the DB.

3.4.3 Authentic View Main Window

There are four viewing modes in Authentic View: Large Markup; Small Markup; Mixed Markup; and Hide All Markup. These modes enable you to view the document with varying levels of markup information. To switch between modes, use the commands in the **Authentic** menu or the icons in the toolbar (see the previous section, Authentic View toolbar icons).

Large markup
This shows the start and end tags of elements and attributes with the element/attribute names in the tags:

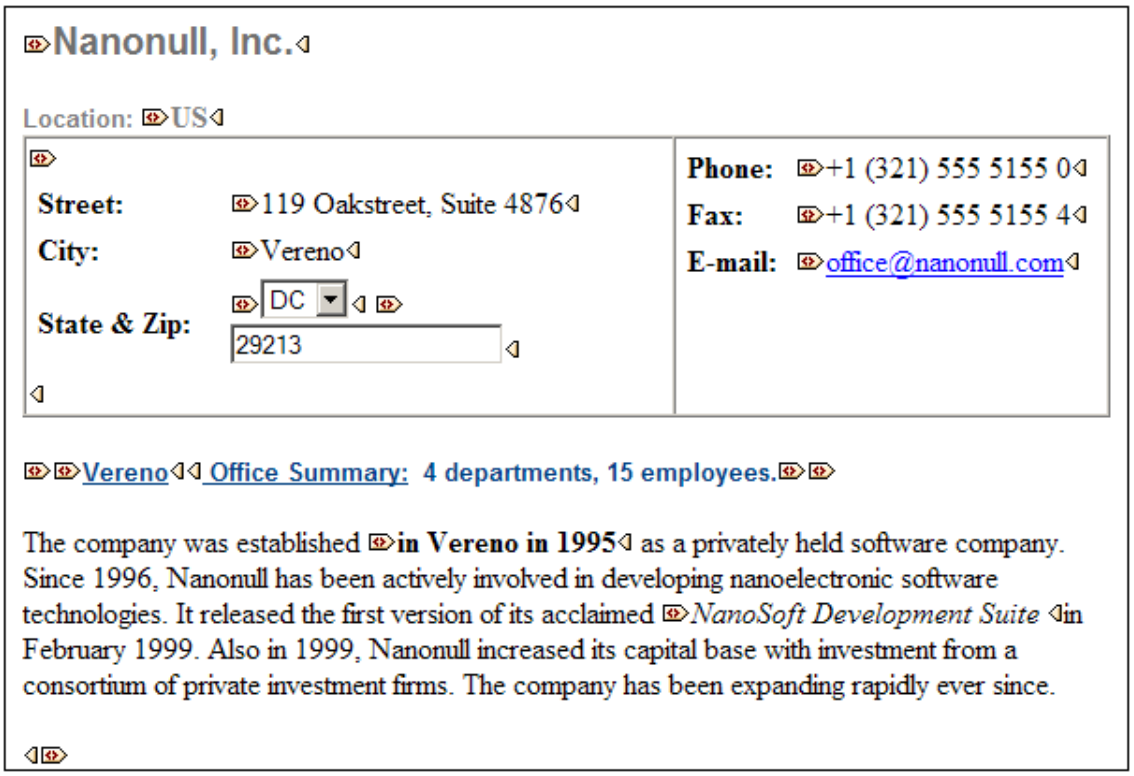
The element `Name` in the figure above is **expanded**, i.e. the start and end tags, as well as the content of the element, are shown. An element/attribute can be **contracted** by double-clicking either its start or end tag. To expand the contracted element/attribute, double-click the contracted tag.



In large markup, attributes are recognized by the equals-to symbol in the start and end tags of the attribute:

 © 2011 Altova GmbH

Small markup
This shows the start and end tags of elements/attributes without names:



Notice that start tags have a symbol inside it while end tags are empty. Also, element tags have an angular-brackets symbol while attribute tags have and equals sign as its symbol (*see screenshot below*).

To collapse or expand an element/attribute, double-click the appropriate tag. The example below shows a collapsed element (highlighted in blue). Notice the shape of the tag of the collapsed element and that of the start tag of the expanded element to its left.

Mixed markup
Mixed markup shows a customized level of markup. The person who has designed the StyleVision Power Stylesheet can specify either large markup, small markup, or no markup for individual elements/attributes in the document. The Authentic View user sees this customized markup in mixed markup viewing mode.

Hide all markup
All XML markup is hidden. Since the formatting seen in Authentic View is the formatting of the printed document, this viewing mode is a WYSIWYG view of the document.

Content display
In Authentic View, content is displayed in two ways:

- Plain text. You type in the text, and this text becomes the content of the element or the value of the attribute.

- Data-entry devices. The display contains either an input field (text box), a multiline input field, combo box, check box, or radio button. In the case of input fields and multiline input fields, the text you enter in the field becomes the XML content of the element or the value of the attribute.

In the case of the other data-entry devices, your selection produces a corresponding XML value, which is specified in the StyleVision Power Stylesheet. Thus, in a combo box, a selection of, say, "approved" (which would be available in the dropdown list of the combo box) could map to an XML value of "1", or to "approved", or anything else; while "not approved" could map to "0", or "not approved", or anything else.

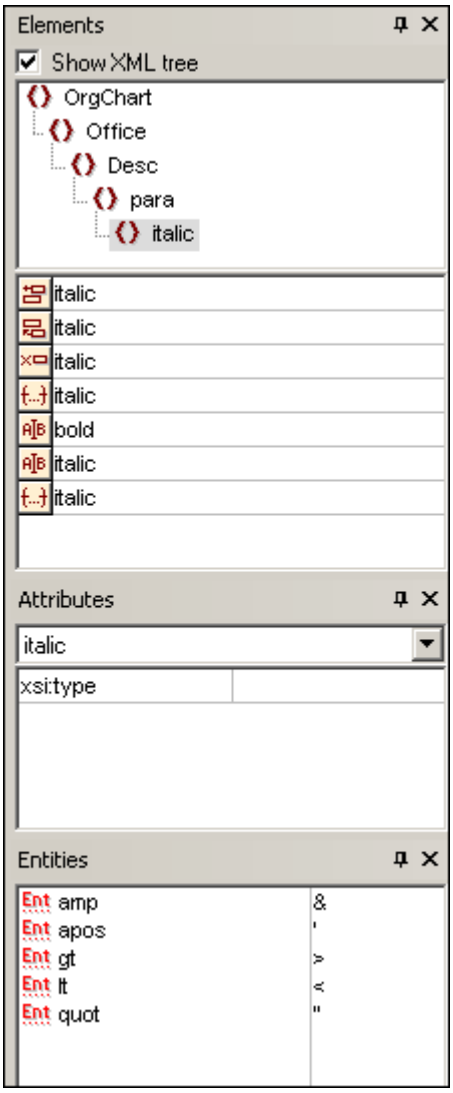
Optional nodes
When an element or attribute is **optional** (according to the referenced schema), a prompt of type `add [element/attribute]` is displayed:

Clicking the prompt adds the element, and places the cursor for data entry. If there are multiple optional nodes, the prompt `add...` is displayed. Clicking the prompt displays a menu of the optional nodes.

3.4.4 Authentic View Entry Helpers

There are three entry helpers in Authentic View: for Elements, Attributes, and Entities. They are displayed as windows down the right side of the Authentic View interface (*see screenshot below*).

 © 2011 Altova GmbH



The Elements and Attributes Entry Helpers are context-sensitive, i.e. what appears in the entry helper depends on where the cursor is in the document. The entities displayed in the Entities Entry Helper are not context-sensitive; all entities allowed for the document are displayed no matter where the cursor is.

Each of the entry helpers is described separately below.

Elements Entry Helper

The Elements Entry Helper consists of two parts:

- The upper part, containing an XML tree that can be toggled on and off using the **Show XML tree** check box. The XML tree shows the ancestors up to the document's root element for the current element. When you click on an element in the XML tree, elements corresponding to that element (as described in the next item in this list) appear in the lower part of the Elements Entry Helper.
- The lower part, containing a list of the nodes that can be inserted within, before, and after; removed; applied to or cleared from the selected element or text range in Authentic View. What you can do with an element listed in the Entry Helper is indicated by the icon to the left of the element name in the Entry Helper. The icons that occur in

the Elements Entry Helper are listed below, together with an explanation of what they mean.

To use node from the Entry Helper, click its icon.

Insert After Element
The element in the Entry Helper is inserted after the selected element. Note that it is appended at the correct hierarchical level. For example, if your cursor is inside a `//sect1/para` element, and you append a `sect1` element, then the new `sect1` element will be appended not as a following sibling of `//sect1/para` but as a following sibling of the `sect1` element that is the parent of that `para` element.

Insert Before Element
The element in the Entry Helper is inserted before the selected element. Note that, just as with the Append After Element command, the element is inserted at the correct hierarchical level.

Remove Element
Removes the element and its content.

Insert Element
An element from the Entry Helper can also be inserted within an element. When the cursor is placed within an element, then the allowed child elements of that element can be inserted. Note that allowed child elements can be part of an elements-only content model as well as a mixed content model (text plus child elements).

An allowed child element can be inserted either when a text range is selected or when the cursor is placed as an insertion point within the text.

- When a text range is selected and an element inserted, the text range becomes the content of the inserted element.
- When an element is inserted at an insertion point, the element is inserted at that point.

After an element has been inserted, it can be cleared by clicking either of the two Clear Element icons that appear (in the Elements Entry Helper) for these inline elements. Which of the two icons appears depends on whether you select a text range or place the cursor in the text as an insertion point (see below).

Apply Element
If you select an element in your document (by clicking either its start or end tag in the Show large markup view) and that element can be replaced by another element (for example, in a mixed content element such as `para`, an `italic` element can be replaced by the `bold` element), this icon indicates that the element in the Entry Helper can be applied to the selected (original) element. The **Apply Element** command can also be applied to a text range within an element of mixed content; the text range will be created as content of the applied element.

- If the applied element has a **child element with the same name** as a child of the original element and an instance of this child element exists in the original element, then the child element of the original is retained in the new

 © 2011 Altova GmbH

element's content.

- If the applied element has **no child element with the same name** as that of an instantiated child of the original element, then the instantiated child of the original element is appended as a sibling of any child element or elements that the new element may have.
- If the applied element has **a child element for which no equivalent exists** in the original element's content model, then this child element is not created directly but Authentic View offers you the option of inserting it.

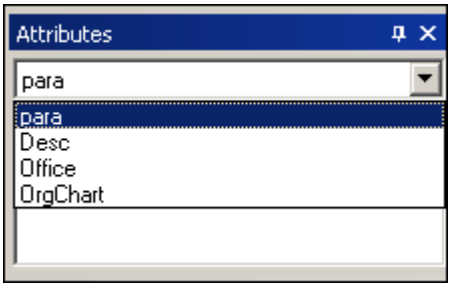
If a text range is selected rather than an element, applying an element to the selection will create the applied element at that location with the selected text range as its content. Applying an element when the cursor is an insertion point is not allowed.

Clear Element (when range selected)
This icon appears when text within an element of mixed content is selected. Clicking the icon clears the element from around the selected text range.

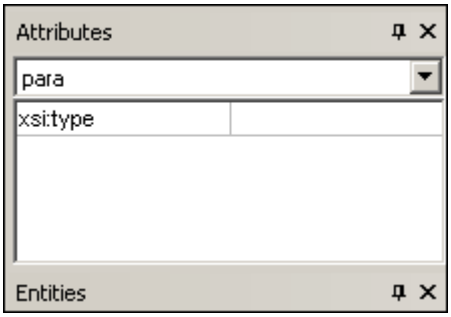
Clear Element (when insertion point selected)
This icon appears when the cursor is placed within an element that is a child of a mixed-content element. Clicking the icon clears the inline element.

Attributes Entry Helper
The Attributes Entry Helper consists of a drop-down combo box and a list of attributes. The element that you have selected (you can click the start or end tag, or place the cursor anywhere in the element content to select it) appears in the combo box.

The Attributes Entry Helper shown in the figures below has a `para` element in the combo box. Clicking the arrow in the combo box drops down a list of all the `para` element's **ancestors up to the document's root element**, which in this case is `OrgChart`.



Below the combo box, a list of valid attributes for that element is displayed, in this case for `para` . If an attribute is mandatory on a given element, then it appears in bold. (In the example below, there are no mandatory attributes except the built-in attribute `xsi:type`.)

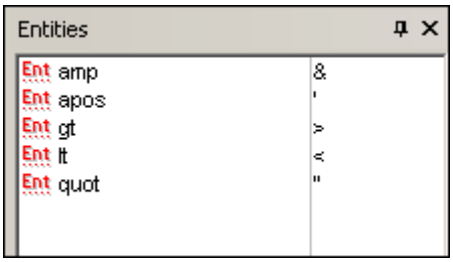


To enter a value for an attribute, click in the value field of the attribute and enter the value. This creates the attribute and its value in the XML document.

In the case of the `xsi:nil` attribute, which appears in the Attributes Entry Helper when a nillable element has been selected, the value of the `xsi:nil` attribute can only be entered by selecting one of the allowed values (`true` or `false`) from the dropdown list for the attribute's value.

The `xsi:type` attribute can be changed by clicking in the value field of the attribute and then selecting, from the dropdown list that appears, one of the listed values. The listed values are the available abstract types defined in the XML Schema on which the Authentic View document is based.

Entities Entry Helper
The Entities Entry Helper allows you to insert an entity in your document. Entities can be used to insert special characters or text fragments that occur often in a document (such as the name of a company). To insert an entity, place the cursor at the point in the text where you want to have the entity inserted, then double-click the entity in the Entities Entry Helper.



Note: An internal entity is one that has its value defined within the DTD. An external entity is one that has its value contained in an external source, e.g. another XML file. Both internal and external entities are listed in the Entities Entry Helper. When you insert an entity, whether internal or external, the entity—not its value—is inserted into the XML text. If the entity is an internal entity, Authentic View displays **the value of the entity**. If the entity is an external entity, Authentic View displays the entity—and not its value. This means, for example, that an XML file that is an external entity will be shown in the Authentic View display as an entity; its content does not replace the entity in the Authentic View display.

You can also **define your own entities** in Authentic View and these will also be displayed in the entry helper: see Define Entities in the Editing in Authentic View section.

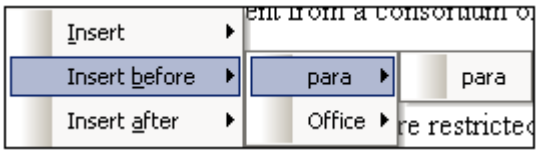
3.4.5 Authentic View Context Menus

Right-clicking on some selected document content or node pops up a context menu with commands relevant to the selection or cursor location.

Inserting elements
The figure below shows the **Insert** submenu, which is a list of all elements that can be inserted at that current cursor location. The **Insert Before** submenu lists all elements that can be inserted before the current element. The **Insert After** submenu lists all elements that can be inserted after the current element. In the figure below, the current element is the `para` element. The `bold` and `italic` elements can be inserted within the current `para` element.

 © 2011 Altova GmbH

As can be seen below, the `para` and `Office` elements can be inserted before the current `para` element.



The node insertion, replacement (**Apply**), and markup removal (**Clear**) commands that are available in the context menu are also available in the Authentic View entry helpers and are fully described in that section.

Insert entity
Positioning the cursor over the Insert Entity command rolls out a submenu containing a list of all declared entities. Clicking an entity inserts it a the selection. See Define Entities for a description of how to define entities for the document.

Insert CDATA Section
This command is enabled when the cursor is placed within text. Clicking it inserts a CDATA section at the cursor insertion point. The CDATA section is delimited by start and end tags; to see these tags you should switch on large or small markup. Within CDATA sections, XML markup and parsing is ignored. XML markup characters (the ampersand, apostrophe, greater than, less than, and quote characters) are not treated as markup, but as literals. So CDATA sections are useful for text such as program code listings, which have XML markup characters.

Remove node
Positioning the mouse cursor over the **Remove** command pops up a menu list consisting of the selected node and all its removable ancestors (those that would not invalidate the document) up to the document element. Click the element to be removed. This is a quick way to delete an element or any removable ancestor. Note that clicking an ancestor element will remove all its descendants, including the selected element.

Clear
The **Clear** command clears the element markup from around the selection. If the entire node is selected, then the element markup is cleared for the entire node. If a text segment is selected, then the element markup is cleared from around that text segment only.

Apply
The **Apply** command applies a selected element to your selection in the main Window. For more details, see Authentic View entry helpers.

Copy, Cut, Paste
These are the standard Windows commands. Note, however, that the **Paste** command pastes copied text either as XML or as Text, depending on what the designer of the stylesheet has

specified for the SPS as a whole. For information about how the **Copy as XML** and **Copy as Text** commands work, see the description of the **Paste As** command immediately below.

Paste As
The **Paste As** command offers the option of pasting as XML or as text an Authentic View XML fragment (which was copied to the clipboard). If the copied fragment is pasted as XML it is pasted together with its XML markup. If it is pasted as text, then only the text content of the copied fragment is pasted (not the XML markup, if any). The following situations are possible:

- An ***entire node together with its markup tags*** is highlighted in Authentic View and copied to the clipboard. (i) The node can be pasted as XML to any location where this node may validly be placed. It will not be pasted to an invalid location. (ii) If the node is pasted as text, then only the node's *text content* will be pasted (not the markup); the text content can be pasted to any location in the XML document where text may be pasted.
- A ***text fragment*** is highlighted in Authentic View and copied to the clipboard. (i) If this fragment is pasted as XML, then the XML markup tags of the text—even though these were not explicitly copied with the text fragment—will be pasted along with the text, but only if the XML node is valid at the location where the fragment is pasted. (ii) If the fragment is pasted as text, then it can be pasted to any location in the XML document where text may be pasted.

Note: Text will be copied to nodes where text is allowed, so it is up to you to ensure that the copied text does not invalidate the document. The copied text should therefore be:

(i) lexically valid in the new location (for example, non-numeric characters in a numeric node would be invalid), and
(ii) not otherwise invalidate the node (for example, four digits in a node that accepts only three-digit numbers would invalidate the node).

If the pasted text does in any way invalidate the document, this will be indicated by the text being displayed in red.

Delete
The **Delete** command removes the selected node and its contents. A node is considered to be selected for this purpose by placing the cursor within the the node or by clicking either the start or end tag of the node.

 © 2011 Altova GmbH

3.5 Browser View

Browser View is typically used to view:

- XML files that have an associated XSLT file. When you switch to Browser View, the XML file is transformed on the fly using the associated XSLT stylesheet and the result is displayed directly in Browser View.
- HTML files which are either created directly as HTML or created via an XSLT transformation of an XML file.

To view XML and HTML files in Browser View, click the **Browser** tab.

Note about Microsoft Internet Explorer and XSLT
Browser View requires Microsoft's Internet Explorer 5.0 or later. If you wish to use Browser View for viewing XML files transformed by an XSLT stylesheet, we strongly recommend Internet Explorer 6.0 or later, which uses MSXML 3.0, an XML parser that fully supports the XSLT 1.0 standard. You might also wish to install MSXML 4.0. Please see our Download Center for more details.(Note that support for XSLT in IE 5 is not 100% compatible with the official XSLT Recommendation. So if you encounter problems in Browser View with IE 5, you should upgrade to IE 6 or later.) You should also check the support for XSLT of your version of Internet Explorer.

Browser View features
The following features are available in Browser View. They can be accessed via the **Browser** menu, **File** menu, and **Edit** menu.

- ***Open in separate window*:** When Browser View is a separate window, it can be positioned side-by-side with an editing view of the same document. This command is in the **Browser** menu and is a toggle-command that can be used to return a separate Browser View window as a tabbed view. In the View tab of the Options dialog, you can set whether Browser View should, by default, be a separate window.
- ***Forward** and **Back*:** The common browser commands to navigate through pages that were loaded in Browser View. These commands are in the **Browser** menu.
- ***Font size:*** Can be adjusted via the **Browser** menu command.
- ***Stop, Refresh, Print:*** More standard browser commands, these can be found in the **Browser** and **File** menus.
- ***Find:*** Enables searches for text strings, this command is in the **Edit** menu.

4 XML

This section describes how to work with XML documents in XMLSpy. It covers the following aspects:

- How to create, open, and save XML documents. In this section, some important XMLSpy settings relating to file creation are also explained.
- XML documents can be edited in Text View, Grid View, and Authentic View. You can select the view that is most useful for you and switch among the views while editing. Each of the views offers different advantages.
- Entry helpers for XML documents have certain specific features, and these are described.
- How to process XML documents with XSLT and XQuery. Various XMLSpy features related to processing are explained.
- Miscellaneous other features for working with XML documents are described.

Altova website: XML Editor

© 2011 Altova GmbH

4.1 Creating, Opening, and Saving XML Documents

When creating, opening, or saving XML documents, the following issues are involved:

- In what view will the XML document open: Text View, Grid View, or Authentic View
- When a new XML document is created, whether a schema (XML Schema or DTD) will be automatically assigned, manually assigned, or not assigned
- If a schema is assigned to the XML document, whether the document will be validated automatically on opening and/or saving

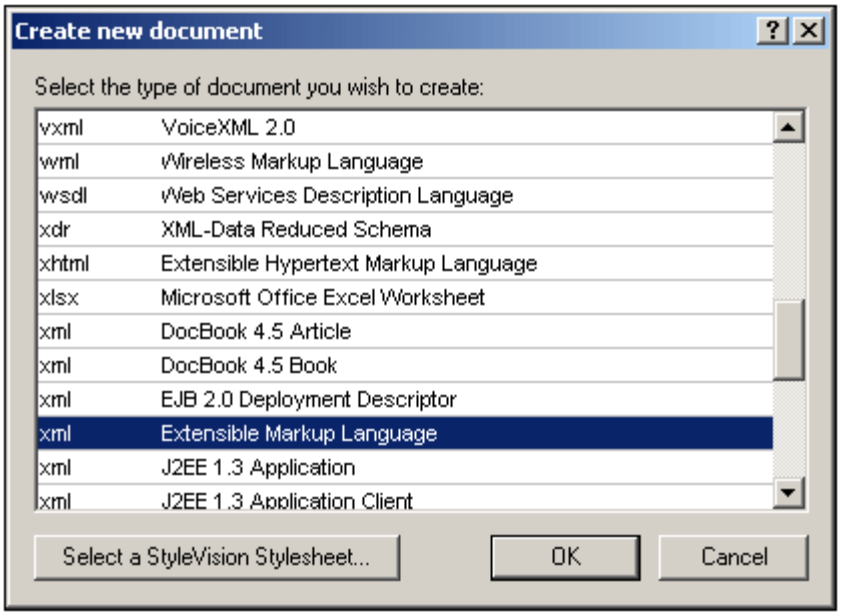
Default view
There are application-wide settings for specifying in what view XML documents (new and existing) should open. These settings are in the Options dialog (**Tools | Options**).

In the ***File Types*** tab of the Options dialog, select a file type of `.xml` and, in the Default View pane, check the required editing view (Text or Grid). Note that: (i) Schema View can be used only for XML Schema; and (ii) Browser View is a display view, not an editing view.

In the ***File Types*** tab, you can also set XMLSpy as the default editor for the selected file type.

An XML document can be edited in Authentic View if a StyleVision Power Stylesheet (SPS) has been assigned to it. When an XML file with an associated SPS is opened, you can specify that it opens directly in Authentic View. Do this by checking the *Always open in Authentic View* option in the ***View*** tab of the Options dialog. If this option is not checked, the file will open in the default view specified for `.xml` files in the ***File Types*** tab (*see above*).

Assigning schemas
When a new XML file is to be created, select the menu command **File | New**. This pops up the Create New Document dialog (*screenshot below*).



Notice that there are several options for the XML document type. The option marked *Extensible Markup Language* creates a generic XML document. Each of the other options is associated with a schema, for example the DocBook DTD. If you select one of these options, an XML document is created that has (i) the corresponding schema automatically assigned to it, and (ii)

a skeleton document structure that is valid according to the assigned schema. Note that you can create your own skeleton XML document. If you save it in the `Template` folder of the application folder, your skeleton document will be available for selection in the Create New Document dialog.

If you select the generic Extensible Markup Language document type, you will be prompted for a schema (DTD or XML Schema) to assign to the document. At this point, you can choose to browse for a schema or go ahead and create an XML document with no schema assigned to it.

You can, of course, assign a schema via the **DTD/Schema** menu at any subsequent time during editing.

Automatic validation

If an existing XML document has a schema assigned to it, then it can be automatically validated on opening and/or saving. The setting for this is in the ***File*** tab of the Options dialog (**Tools | Options**).

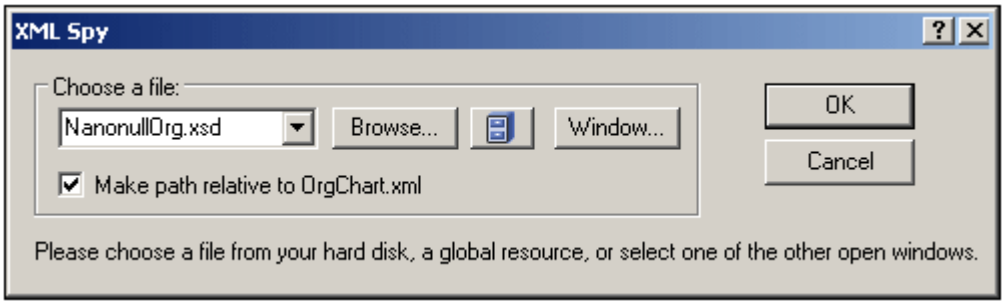
The automatic validation settings in the ***File*** tab can be combined with a setting in the File Types tab to disable automatic validation for specific file types. Using the settings in the two tabs together enables you to specify automatic validation for specific file types.

© 2011 Altova GmbH

4.2 Assigning Schemas and Validating

Altova website: XML Validator, XML Validation

A schema (DTD or XML Schema) can be assigned to an XML document when it is first created. A schema can also be assigned, or changed, at any subsequent time using the **Assign DTD** or **Assign Schema** commands in the **DTD/Schema** menu.



The path to the schema file that is inserted in the XML document can be made relative by checking the relative path check box in the dialog.

Global resources for schemas

A global resource is an alias for a file or folder. The target file or folder can be changed within the GUI by changing the active configuration of the global resource (via the menu command **Tools | Active Configuration**). Global resources therefore enable the assigned schema to be switched among multiple schemas, which can be useful for testing. How to use global resources is described in the section Altova Global Resources.

XML Schema plus DTD

One very useful DTD feature that XML Schema does not have is the use of entities. However, if you wish to use entities in your XML-Schema-validated XML document, you can add a `DOCTYPE` declaration to the XML document and include your entity declarations in it.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE OrgChart [
  <!ENTITY name-int "value">
  <!ENTITY name-ext SYSTEM "extfile.xml">
]>
<OrgChart xmlns="http://www.xs.com/org"
          xsi:schemaLocation="http://www.xs.com/org OrgChart.xsd">
  ...
</OrgChart>
```

After declaring the entities in the DTD, they can be used in the XML document. The document will be well-formed and valid. Note, however, that external parsed entities are not supported in Authentic View..

Going to schema definitions

With the XML document open, you can directly open the DTD or XML Schema on which it is based by clicking the Go to DTD or Go to Schema commands in the **DTD/Schema** menu. Additionally, you can place the cursor within a node in the XML document and go to the schema definition of that node via the **Go to Definition** command in the **DTD/Schema** menu.

Validating and checking well-formedness
To validate and/or check for well-formedness, use the **Validate XML (F8)** and **Check Well-Formedness (F7)** commands in the XML menu or the corresponding commands in the toolbar. Any error is reported in the Messages window.

© 2011 Altova GmbH

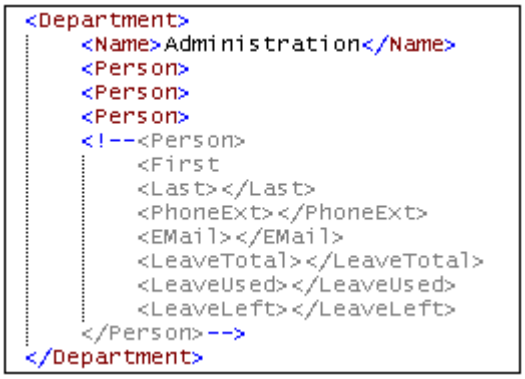
4.3 Editing XML in Text View

XMLSpy offers some specialized XML text editing features in addition to the generally available editing features in Text View, which are described in Text View in the Editing Views section.

- Commenting text in and out
- Inserting file paths
- Inserting XML fragments via XInclude
- Copying XPath and XPointer expressions to the clipboard

Commenting text in/out
Text in an XML document can be commented out using the XML start-comment and end-comment delimiters, respectively <! -- and -->. In XMLSpy, these comment delimiters can be easily inserted using the **Edit | Comment In/Out** menu command.

To comment out a block of text, select the text to be commented out and then select the command **Comment In/Out**, either from the **Edit** menu or the context menu that you get on right-clicking the selected text. The commented text will be grayed out (*see screenshot below*).



To uncomment a commented block of text, select the commented block **excluding** the comment delimiters, and select the command **Comment In/Out**, either from the **Edit** menu or the context menu that you get on right-clicking the selected text. The comment delimiters will be removed and the text will no longer be grayed out.

Note about empty lines
In XML documents, empty lines are discarded when you change views or save the document. If you wish to retain empty lines, enclose them in comment delimiters.

Inserting file paths
The **Edit | Insert File Path** command enables you to browse for the file in question and insert its file path at the selected location in the XML document being edited. This command enables you to quickly and accurately enter a file path. See the command description for more details.

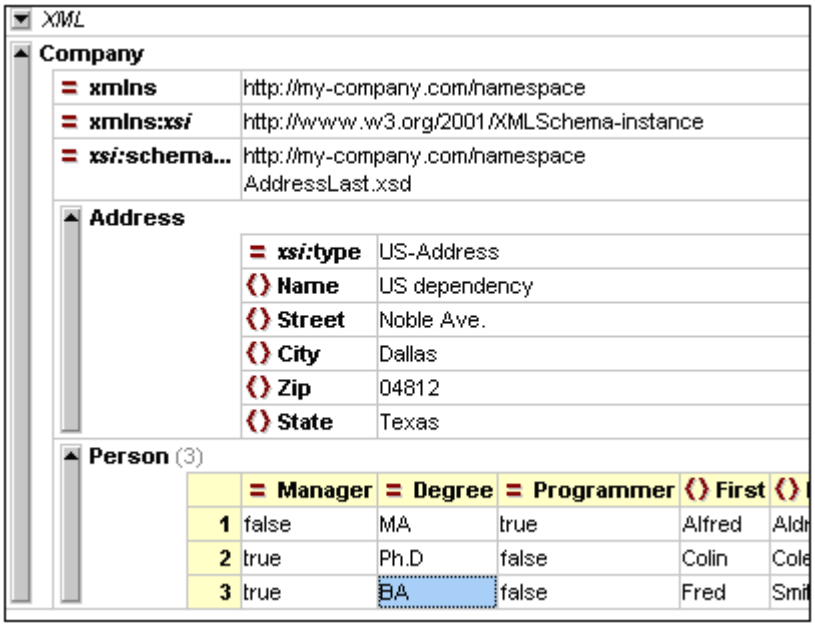
Inserting XML fragments via XInclude
The **Edit | Insert XInclude** enables you, via XInclude, to insert the contents of an entire XML document, or a fragment of one, in the XML document being edited. This command enables you to quickly and accurately enter a file path. See the command description for more details.

Copying XPath and XPointer expressions to the clipboard
The XPath and XPointer expressions of the selected node can be copied to the clipboard using the **Edit | Copy XPath** and **Edit | Copy XPointer** commands, respectively. This enables you to obtain the correct XPath and XPointer expressions for a given node. The copied expressions can then be inserted at the required location, for example, an XPath expression in an XSLT stylesheet and an XPointer expression in the `href` attribute of an `xinclude` element. For more detailed descriptions of the commands, see their descriptions in the User Reference section.

© 2011 Altova GmbH

4.4 Editing XML in Grid View

Grid View shows the hierarchical structure of **XML documents** through a set of nested containers that can be expanded and collapsed. This provides a clear picture of the document's structure. In Grid View, both structure and contents can be easily manipulated. In Standard Edition, Grid View is read-only; in Enterprise and Professional Editions, you can also edit your document in Grid View.

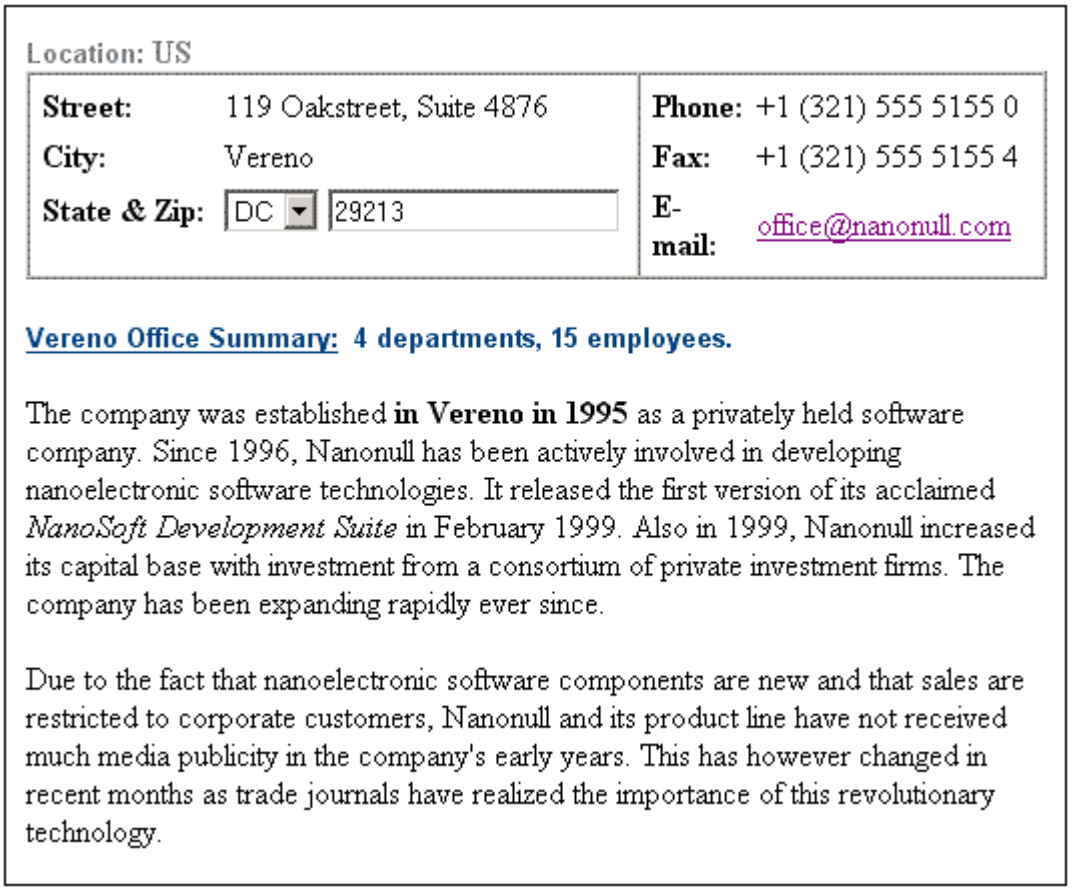


In the screenshot above, notice that the document is displayed as a hierarchy in a grid form. When a node can contains content, it is divided into two fields: for name and for content. Node names are displayed in bold face and node content in normal face.

More about Grid View
For a more detailed description of Grid View, see under Editing Views.

4.5 Editing XML in Authentic View

Authentic View enables a user to edit an XML document as if it were a text document (*screenshot below*). The XML markup and all other non-content text can be hidden from the person editing the document. This can be useful for people who are unfamiliar with XML, enabling them to a valid XML document even while concentrating entirely on the content of the document..



The Authentic View of a document is enabled when a StyleVision Power Stylesheet (SPS) is assigned to an XML document. An SPS is based on the same schema source as that on which the XML document is based, and it defines the structure of the XML document. The SPS also defines the layout and formatting of the document in Authentic View. For example, in the document shown in the screenshot above, the following Authentic formatting and editing features are used:

- Paragraph and other block formatting
- Table structures
- Text formatting, such as color and font face
- Combo boxes (see the State and Zip fields) enable the user to select from a group of valid choices, which can be taken from schema enumerations, as has been done in the case above
- Additional information can be calculated from the data in the document and be presented (in the example above, the office summary details have not been entered by the user but calculated from other data in the document)

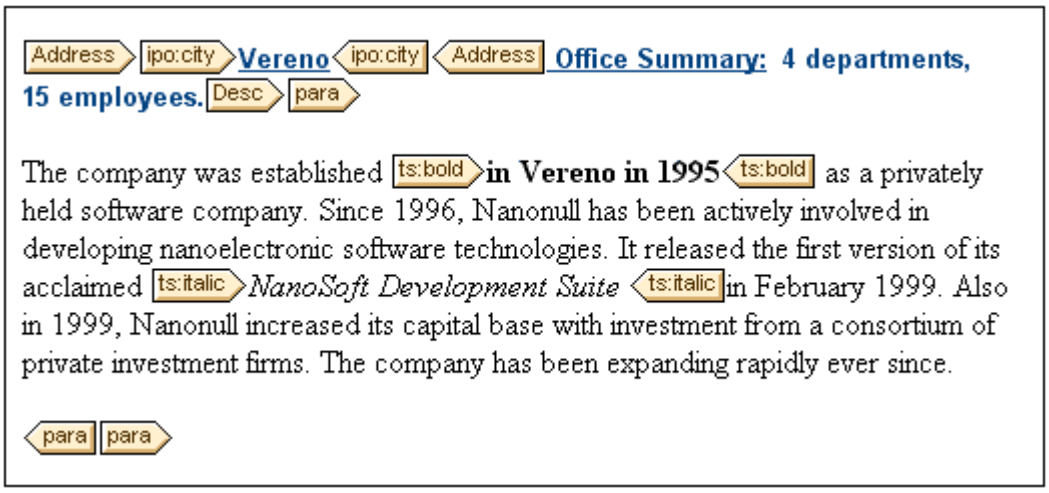
SPSs are created specifically for viewing and editing XML documents in Authentic View and for

 © 2011 Altova GmbH

generating standard output (such as HTML, PDF, RTF, and Word 2007 documents) from XML. SPSs are created with Altova StyleVision.

Editing document structure
Valid nodes can be added to the document at any time by selecting a location and then adding the required node via the entry helpers (Elements and Attributes) or context menu. The nodes available at any given location are restricted to the nodes that can be validly added as siblings or children at the selected location. For example, when the cursor is located within a paragraph, you can append another paragraph if this is allowed by the schema.

When editing the structure of an XML document in Authentic View, it could be useful to see the markup of the document. Markup can therefore be switched on as tags (*screenshot below*) using the **Authentic | Show Large Markup** command (or the corresponding toolbar icon).



Editing content
Content is created and edited by typing it into the nodes of the document. Entities and CDATA sections can be added via the context menu (entities also via the Entities entry helper).

More about editing in Authentic View
For more details of how to edit in Authentic View, see the Authentic View section.

4.6 Entry Helpers for XML Documents

For XML documents, there are three entry helpers: Elements entry helper; Attributes entry helper; and Entities entry helper. When an element is added via the Elements entry helper, it can be added together with mandatory child elements, mandatory attributes, all child elements, or no child element or attribute, according to the respective settings in the Editing tab of the Options dialog. When empty attributes are added, they are added with quotes.

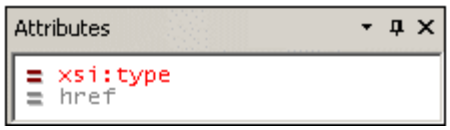
Note that in the different views, the entry helpers are designed differently, in accordance with the functionality of the respective view.

Elements entry helper
The following points should be noted:

- *Text View:* Elements are inserted at the cursor insertion point. Unused elements are displayed in red, used elements in gray. Mandatory elements are indicated with an exclamation mark "**!**" before the name of the element.
- *Grid View:* Elements can be appended after, inserted before, or added as a child of the selected element. There are therefore three tabs, each displaying the elements that may be added. Unused elements are displayed in black, used elements in gray.
- *Authentic View:* Elements can be inserted before, after, or within the selected element. Additionally, there is a document tree that shows the location of the currently selected element in the document's tree structure. For more details of how to edit in Authentic View, see the Authentic View section.

Attributes entry helper
The following points should be noted:

- *Text View:* When the cursor is placed inside the start tag of an element and after a space, the attributes declared for that element become visible. Unused attributes are displayed in red, used attributes in gray. Mandatory attributes are indicated with an exclamation mark "**!**" before the name of the attribute.



  To insert an attribute, double-click the required attribute. The attribute is inserted at the cursor point together with an equals-to sign and quotes to delimit the attribute value. The cursor is placed between the quotes, so you can start typing in the attribute value directly.
- *Grid View:* When an element is selected, the attributes that can be added as a child are listed in the Add Child tab of then entry helper. When an attribute is selected, the available attributes are listed in the Append (after) and Insert (before) tabs. Unused attributes are displayed in black, used attributes in gray.
- *Authentic View:* When an element is selected, the attributes declared for that element become visible. Enter he value of the attribute in the entry helper.

Entities entry helper
Any parsed or unparsed entity that is declared inline (within the XML document) or in an external DTD, is displayed in the Entities entry helper. In all three views (Text, Grid, and Authentic), an entity is inserted at the cursor insertion point by double-clicking it. In Grid View, entities are displayed in the Append, Insert, and Add Child tabs.

 © 2011 Altova GmbH

Note that if you add an internal entity, you will need to save and reopen your document before the entity appears in the Entities entry helper.

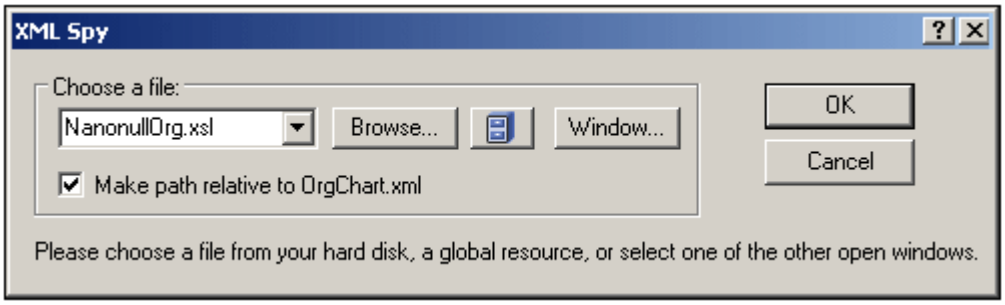
4.7 Processing with XSLT and XQuery

XML documents can be processed with XSLT or XQuery documents to produce output documents. XMLSpy has built-in XSLT 1.0, XSLT 2.0, and XQuery 1.0 processors. The following processing-related features are available in the GUI:

- Assigning XSLT stylesheets
- Go to XSLT
- XSLT parameters and XQuery variables
- XSLT transformations
- XQuery executions
- Automating XML tasks with AltovaXML

Assigning XSLT stylesheets
You can assign an XSLT stylesheet to an XML document via the **XSL/XQuery | Assign XSL** command (browse for the file in the dialog (*screenshot below*) that pops up). The assignment is entered in the XML document as a processing instruction (PI) having the standard XSLT target defined by the W3C: `xml-stylesheet`. This assignment is used when an XSLT transformation is invoked (**XSL/XQuery | XSL Transformation**).

Additionally, an XSLT-for-FO stylesheet can be assigned with the **XSL/XQuery | Assign XSL: FO** command (browse for the file in the dialog (*screenshot below*) that pops up). The assignment is entered in the XML document as a processing instruction (PI) having the Altova-defined target: `altova_xslfo`. This assignment is used when an XSLT-for-FO transformation is invoked (**XSL/XQuery | XS:FO Transformation**).



You can also select a global resource to specify the XSLT file. A global resource is an alias for a file or folder. The target file or folder can be changed within the GUI by changing the active configuration of the global resource (via the menu command **Tools | Active Configuration**). Global resources therefore enable the assigned XSLT file to be switched from one to another, which can be useful for testing. How to use global resources is described in the section Altova Global Resources.

If a previous assignment using either of these PI targets exists, then you are asked whether you wish to overwrite the existing assignment.

Go to XSLT
The **XSL/XQuery | Go to XSL** command opens the XSLT file that has been assigned to the XML document.

XSLT parameters and XQuery variables
XSLT parameters and XQuery variables can be defined, edited, and deleted in the dialog that

 © 2011 Altova GmbH

appears on clicking the command **XSL/XQuery | XSLT Parameters / XQuery Variables**. The parameter/variable values defined here are used for all XSLT transformations and XQuery executions in XMLSpy. However, these values will not be passed to external engines such as MSXML. For the details of how to use this feature, see the User Reference section.

XSLT transformations
Two types of XSLT transformation are available:

- Standard XSLT transformation (**XSL/XQuery | XSL Transformation**): The output of the transformation is displayed in a new window or, if specified in the stylesheet, is saved to a file location. The engine used for the transformation is specified in the XSL tab of the Options dialog (**Tools | Options**).
- XSL-for-FO transformation (**XSL/XQuery | XSL-FO Transformation**): The XML document is transformed to PDF in a two-step process. In the first step, the XML document is transformed to an FO document using the XSLT processor specified in the XSL tab of the Options dialog (**Tools | Options**); note that you can also select (at the bottom of the tab) the XSLT engine that comes with some FO processors such as FOP. In the second step, the FO document is processed by the FO processor specified in the XSL tab of the Options dialog (**Tools | Options**) to produce PDF output.

Note: An FO document (which is a particular type of XML document) can be transformed to PDF by clicking the XSL:FO transformation command. When the source document is an FO document, the second step of the two-step process for this command is executed directly.

XQuery executions
An XQuery document can be executed on the active XML document by clicking the command **XSL/XQuery | XQuery Execution**. You are prompted for the XQuery file, and the result document is displayed in a new window in the GUI.

Automating XML tasks with AltovaXML
AltovaXML is a free application which contains Altova's XML Validator, XSLT 1.0, XSLT 2.0, and XQuery 1.0 engines. It can be used from the command line, via a COM interface, in Java programs, and in .NET applications to validate XML documents, transform XML documents using XSLT 1.0 and 2.0 stylesheets, and execute XQuery documents.

XSLT transformation tasks can therefore be automated with the use of AltovaXML. For example, you can create a batch file that calls AltovaXML to transform a set of documents. See the Altova XML documentation for details.

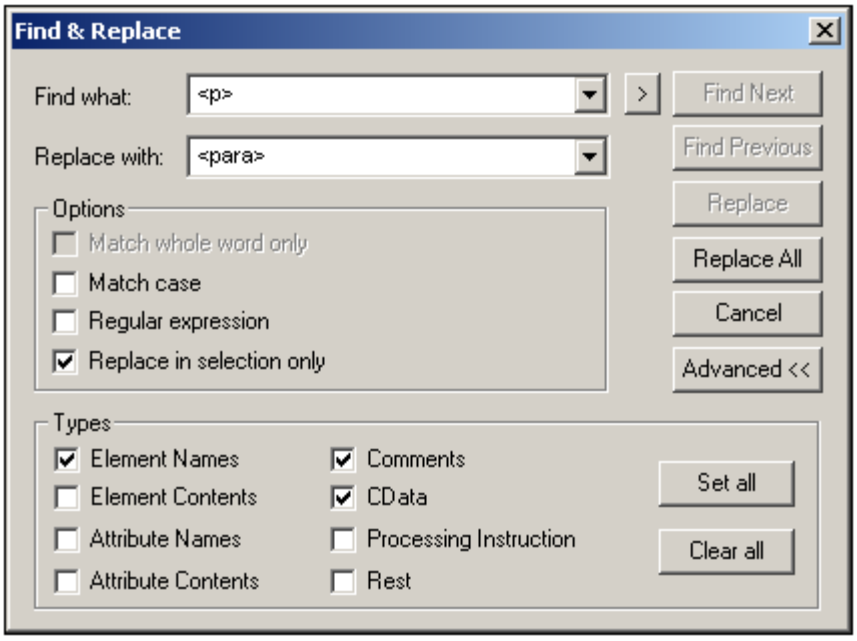
4.8 Additional Features

Additional features for working with XML files are listed below.

- Encoding
- Find and Replace

Encoding
The encoding of XML files (and other types of documents) can be set via the menu command **File | Encoding**. The default encoding of XML and non-XML files can be specified in the Options | Encoding tab.

Find and Replace
The Find and Replace features (accessed via the **Edit** menu) has powerful search capabilities. The search term can be defined additionally in terms of casing and whether whole words should be matched, and it can also be expressed as a regular expression. The search range can be restricted to a selection in the document and to particular node types (*see screenshot below*).



The screenshot above shows the Find & Replace dialog in Text View. The dialog and functionality varies according to the view that is active.

© 2011 Altova GmbH

5 DTDs and XML Schemas

Altova website: XML Schema Editor

This section provides an overview of how to work with DTDs and XML Schemas. In addition to the editing features, XMLSpy provides the following powerful DTD/Schema features:

Catalog mechanism
Support for the OASIS catalog mechanism enables the re-direction of URIs to local addresses, thus facilitating use across multiple workstations.

Generate Sample XML file
You can generate, via the **DTD/Schema | Generate Sample XML File** menu command, a skeleton XML document based on the active DTD or XML Schema file. This is very useful for quickly creating an XML file based on a schema.

Go to definition
When the cursor is located within a node in an XML document, clicking the **DTD/Schema | Go to Deinition** menu command opens the schema file and highlights the definition of the selected XML node.

5.1 DTDs

A DTD document can be edited in Text View. You can also view a DTD in Grid View, but not edit it.

Text View

In Text View, the document is displayed with syntax coloring and must be typed in. Given below is a sample of a DTD fragment:

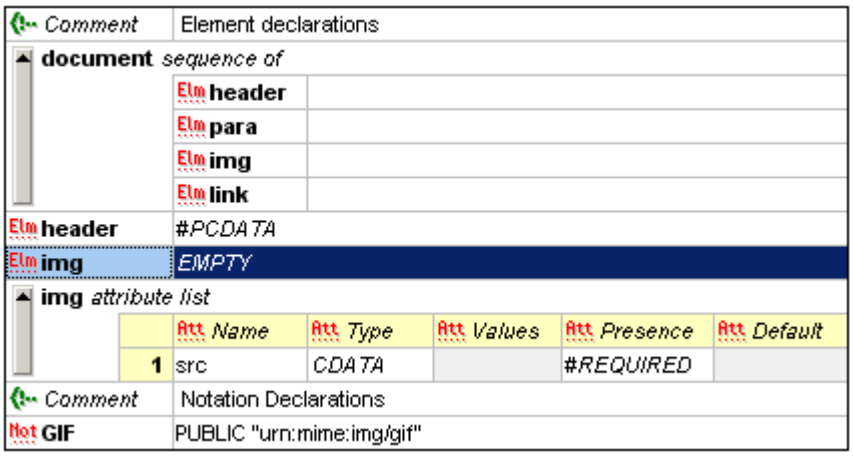
```
<!-- Element declarations -->
<!ELEMENT document (header, para, img, link)>
<!ELEMENT header (#PCDATA)>
<!ELEMENT img EMPTY>
  <!ATTLIST img
    src CDATA #REQUIRED
  >

<!-- Notation Declarations -->
<!NOTATION GIF PUBLIC "urn:mime:img/gif">
```

Indentation is indicated by indentation guides and is best obtained by using the tab key. The amount of tab indentation can be set in the Text View Settings dialog.

Grid View

In Grid View, the DTD document is displayed as a table. The screenshot below shows the Grid View display of the DTD listed above.



Comment	Element declarations				
document *sequence of*					
	Elm header				
	Elm para				
	Elm img				
	Elm link				
Elm header	#PCDATA				
Elm img	EMPTY				
img *attribute list*					
	Att Name	Att Type	Att Values	Att Presence	Att Default
1	src	CDATA		#REQUIRED	
Comment	Notation Declarations				
Not GIF	PUBLIC "urn:mime:img/gif"				

DTD features in XMLSpy

XMLSpy offers the following very useful features:

- *Generate sample XML file from DTD:* With the **DTD/Schema | Generate Sample XML File** command, an XML document can be generated that is based on the active DTD.

© 2011 Altova GmbH

5.2 XML Schemas

XML Schema documents can be edited in Text View, and can be viewed but not edited in Grid View and Schema View. XML Schema documents are typically saved with the extension `.xsd` or `.xs`.

Editing in Text View

In Text View an XML Schema is edited as an XML document; the editing features available for XML documents are also available for XML Schemas. As with all XML documents where the schema is identified and accessible, Text View entry helpers display the items available for addition at the cursor location point.

XML Schema features in XMLSpy

Additionally, XMLSpy offers the following very useful features:

- *Generate sample XML file from XML Schema:* With the **DTD/Schema | Generate Sample XML File** command, an XML document can be generated that is based on the active XML Schema.

5.3 Catalogs in XMLSpy

XMLSpy supports a subset of the OASIS XML catalogs mechanism. The catalog mechanism enables XMLSpy to retrieve commonly used schemas (as well as stylesheets and other files) from local user folders. This increases the overall processing speed, enables users to work offline (that is, not connected to a network), and improves the portability of documents (because URIs would then need to be changed only in the catalog files.)

The catalog mechanism in XMLSpy works as outlined below.

RootCatalog.xml

When XMLSpy starts, it loads a file called `RootCatalog.xml` (*structure shown in listing below*), which contains a list of catalog files that will be looked up. You can modify this file and enter as many catalog files to look up as you like, each in a `nextCatalog` element. Each of these catalog files is looked up and the URIs in them are resolved according to the mappings specified in them.

```
<?xml version="1.0" encoding="UTF-8"?>
<catalog xmlns="urn:oasis:names:tc:entity:xmlns:xml:catalog"
         xmlns:spy="http://www.altova.com/catalog ext"
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xsi:schemaLocation="urn:oasis:names:tc:entity:xmlns:xml:catalog
Catalog.xsd">
  <nextCatalog catalog="%PersonalFolder%/Altova/%AppAndVersionName%/
CustomCatalog.xml"/>
  <nextCatalog catalog="CoreCatalog.xml"/>
  <!-- Include all catalogs under common schemas folder on the first directory
level -->
  <nextCatalog spy:recurseFrom="%AltovaCommonFolder%/Schemas" catalog="
catalog.xml" spy:depth="1"/>
  <!-- Include all catalogs under common XBRL folder on the first directory
level -->
  <nextCatalog spy:recurseFrom="%AltovaCommonFolder%/XBRL" catalog="
catalog.xml" spy:depth="1"/>
</catalog>
```

In the listing above, notice that in the `Schemas` and `XBRL` folders of the folder identified by the variable `%AltovaCommonFolder%` are catalog files named `catalog.xml`. (The value of the `%AltovaCommonFolder%` variable is given in the table below.)

The catalog files in the Altova Common Folder map the pre-defined public and system identifiers of commonly used schemas (such as SVG) and XBRL taxonomies to URIs that point to locally saved copies of the respective schemas. These schemas are installed in the Altova Common Folder when XMLSpy is installed.You should take care not to duplicate mappings in these files, as this could lead to errors.

CoreCatalog.xml, CustomCatalog.xml, and Catalog.xml

In the `RootCatalog.xml` listing above, notice that `CoreCatalog.xml` and `CustomCatalog.xml` are listed for lookup:

- `CoreCatalog.xml` contains certain Altova-specific mappings for locating schemas in the Altova Common Folder.
- `CustomCatalog.xml` is a skeleton file in which you can create your own mappings. You can add mappings to `CustomCatalog.xml` for any schema you require but that is not addressed by the catalog files in the Altova Common Folder. Do this using the supported elements of the OASIS catalog mechanism (*see below*).
- There are a number of `Catalog.xml` files in the Altova Common Folder. Each is inside

 © 2011 Altova GmbH

the folder of a specific schema or XBRL taxonomy in the Altova Common Folder, and each maps public and/or system identifiers to URIs that point to locally saved copies of the respective schemas.

Location of catalog files and schemas

The files `RootCatalog.xml` and `CoreCatalog.xml` are installed in the XMLSpy application folder. The file `CustomCatalog.xml` is located in your `MyDocuments\Altova\XMLSpy` folder. The `catalog.xml` files are each in a specific schema folder, these schema folders being inside the folders: `%AltovaCommonFolder%\Schemas` and `%AltovaCommonFolder%\XBRL`.

Shell environment variables and Altova variables

Shell environment variables can be used in the `nextCatalog` element to specify the path to various system locations (*see RootCatalog.xml listing above*). The following shell environment variables are supported:

`%AltovaCommonFolder%`	C:\Program Files\Altova\Common2011
`%DesktopFolder%`	Full path to the Desktop folder for the current user.
`%ProgramMenuFolder%`	Full path to the Program Menu folder for the current user.
`%StartMenuFolder%`	Full path to Start Menu folder for the current user.
`%StartUpFolder%`	Full path to Start Up folder for the current user.
`%TemplateFolder%`	Full path to the Template folder for the current user.
`%AdminToolsFolder%`	Full path to the file system directory that stores administrative tools for the current user.
`%AppDataFolder%`	Full path to the Application Data folder for the current user.
`%CommonAppDataFolder%`	Full path to the file directory containing application data for all users.
`%FavoritesFolder%`	Full path of the Favorites folder for the current user.
`%PersonalFolder%`	Full path to the Personal folder for the current user.
`%SendToFolder%`	Full path to the SendTo folder for the current user.
`%FontsFolder%`	Full path to the System Fonts folder.
`%ProgramFilesFolder%`	Full path to the Program Files folder for the current user.
`%CommonFilesFolder%`	Full path to the Common Files folder for the current user.
`%WindowsFolder%`	Full path to the Windows folder for the current user.
`%SystemFolder%`	Full path to the System folder for the current user.
`%CommonAppDataFolder%`	Full path to the file directory containing application data for all users.

%LocalAppDataFolder%	Full path to the file system directory that serves as the data repository for local (nonroaming) applications.
%MyPicturesFolder%	Full path to the MyPictures folder.

How catalogs work

Catalogs are commonly used to redirect a call to a DTD to a local URI. This is achieved by mapping, in the catalog file, public or system identifiers to the required local URI. So when the DOCTYPE declaration in an XML file is read, the public or system identifier locates the required local resource via the catalog file mapping.

For popular schemas, the PUBLIC identifier is usually pre-defined, thus requiring only that the URI in the catalog file point to the correct local copy. When the XML document is parsed, the PUBLIC identifier in it is read. If this identifier is found in a catalog file, the corresponding URL in the catalog file will be looked up and the schema will be read from this location. So, for example, if the following SVG file is opened in XMLSpy:

```
<?xml version="1.0" standalone="no"?>
<!DOCTYPE svg PUBLIC "-//W3C//DTD SVG 1.1//EN"
"http://www.w3.org/Graphics/SVG/1.1/DTD/svg11.dtd">

<svg width="20" height="20" xml:space="preserve">
  <g style="fill:red; stroke:#000000">
     <rect x="0" y="0" width="15" height="15"/>
     <rect x="5" y="5" width="15" height="15"/>
  </g>
</svg>
```

This document is read and the catalog is searched for the PUBLIC identifier. Let's say the catalog file contains the following entry:

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

In this case, there is a match for the PUBLIC identifier, so the lookup for the SVG DTD is redirected to the URI schemas/svg/svg11.dtd (this path is relative to the catalog file), and this local file will be used as the DTD. If there is no mapping for the Public ID in the catalog, then the URL in the XML document will be used (in the example above: http://www.w3.org/Graphics/SVG/1.1/DTD/svg11.dtd).

The catalog subset supported by XMLSpy

When creating entries in CustomCatalog.xml (or any other catalog file that is to be read by XMLSpy), use only the following elements of the OASIS catalog specification. Each of the elements below is listed with an explanation of their attribute values. For a more detailed explanation, see the XML Catalogs specification. Note that each element can take the xml:base attribute, which is used to specify the base URI of that element.

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

 © 2011 Altova GmbH

```
rewritePrefix="Replacement string to locate resource locally"/>
```

In cases where there is no public identifier, as with most stylesheets, the system identifier can be directly mapped to a URL via the `system` element. Also, a URI can be mapped to another URI using the `uri` element. The `rewriteURI` and `rewriteSystem` elements enable the rewriting of the starting part of a URI or system identifier, respectively. This allows the start of a filepath to be replaced and consequently enables the targeting of another directory. For more information on these elements, see the XML Catalogs specification.

File extensions and intelligent editing according to a schema

Via catalog files you can also specify that documents with a particular file extension should have XMLSpy's intelligent editing features applied in conformance with the rules in a schema you specify. For example, if you create a custom file extension `.myhtml` for (HTML) files that are to be valid according to the HTML DTD, then you can enable intelligent editing for files with these extensions by adding the following element of text to `CustomCatalog.xml` as a child of the `<catalog>` element.

```
<spy:fileExtHelper ext="myhtml" uri="schemas/xhtml/xhtml1-transitional.dtd"/>
```

This would enable intelligent editing (auto-completion, entry helpers, etc) of `.myhtml` files in XMLSpy according to the XHTML 1.0 Transitional DTD. Refer to the `catalog.xml` file in the `%AltovaCommonFolder%\Schemas\xhtml` folder, which contains similar entries.

XML Schema and catalogs

XML Schema information is built into XMLSpy and the validity of XML Schema documents is checked against this internal information. In an XML Schema document, therefore, no references should be made to any schema for XML Schema.

The `catalog.xml` file in the `%AltovaCommonFolder%\Schemas\schema` folder contains references to DTDs that implement older XML Schema specifications. You should not validate your XML Schema documents against either of these schemas. The referenced files are included solely to provide XMLSpy with entry helper info for editing purposes should you wish to create documents according to these older recommendations.

More information

For more information on catalogs, see the XML Catalogs specification.

6 XSLT and XQuery

XMLSpy provides editing support for XSLT and XQuery documents, has built-in engines for transforming with XSLT and executing XQuery documents. This section provides an overview of the XSLT and XQuery functionality available in XMLSpy. It is organized into the following sections:

- XSLT
- XQuery

Additional and more detailed information about commands is in the descriptions of the relevant menu commands (in the User Reference section). For more information on editing support, also see the Editing Views section and the XML section.

© 2011 Altova GmbH

6.1 XSLT

Altova website: XSLT Editor

This section on XSLT is organized into the following sections:

- Editing XSLT documents: describes the editing support for XSLT documents in XMLSpy
- XSLT Processing: shows the various ways in which XSLT transformations can be carried out in the XMLSpy GUI using engines of your choice. This section also explains important XSLT settings in XMLSpy.

Additional XSLT features
Additional and more detailed information about the various features described in this section is in the descriptions of the relevant menu commands (in the User Reference section).

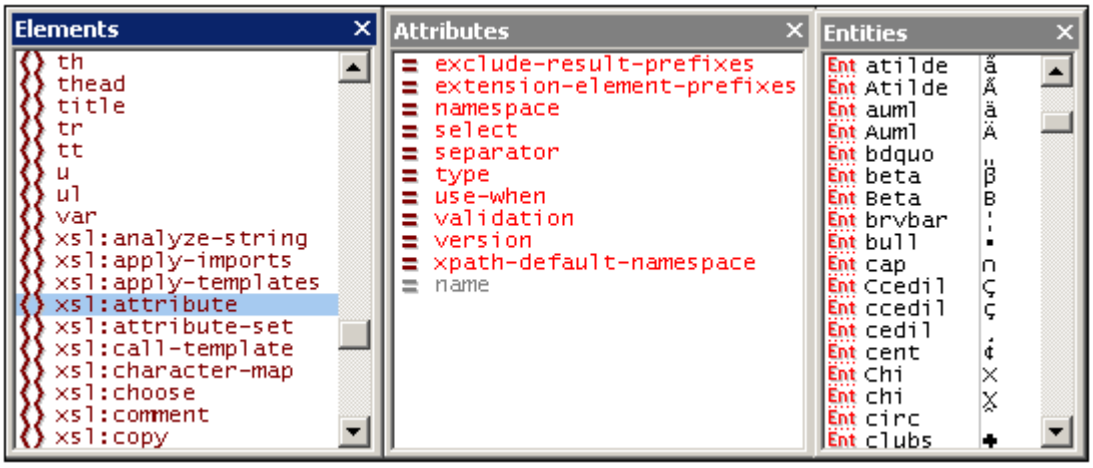
Altova XSLT Engines
For details about how the Altova XSLT 1.0 and 2.0 Engines are implemented, see Engine Information in the Appendices.

AltovaXML for command line and batch processing
The XMLSpy GUI enables batch processing via the projects functionality. However, if you are looking for more flexibility, you should try Altova's free AltovaXML product, which contains the Altova XSLT 1.0 and 2.0 Engines that are built into XMLSpy. AltovaXML is ideal if you wish to perform XSLT transformations from the command line, or batch processing.

6.1.1 XSLT Documents

XSLT 1.0 and 2.0 documents can be edited in Text View, and are edited like any other XML document in Text View. In Standard Edition, XSLT documents can also be viewed, but not edited, in Grid View.

Entry helpers
Entry helpers are available for elements, attributes, and entities. Information about the items displayed in the entry helpers is built into XMLSpy, and is not dependent on references contained in the XSLT document.



The following points should be noted:

1. If a new XSLT document is created via the Create a New Document dialog (**File | New**), then the appropriate XSLT elements and attributes (XSLT 1.0 or XSLT 2.0, depending upon which document type was created) are loaded into the entry helpers. Additionally, HTML elements and attributes are loaded, as well as the HTML 4.0 entity sets, Latin-1, special characters, and symbols.
2. If an XML document is created via the Create a New Document dialog (**File | New**) and given XSLT content, no entry helper items are available except for XML character entities.
3. If an XSLT document is opened that was created as an XSLT document via the Create a New Document dialog (**File | New**), then the entity helpers will be as in Point 1 above.
4. If an XSLT document is opened that was **not** created as an XSLT document via the Create a New Document dialog (**File | New**), then the entity helpers will be as in Point 1 above. Additionally, XSL-FO elements and attributes will be listed in the Text View entry helpers.
5. The prefixes of elements in the Elements entry helper are as follows and are invariable: `xsl:` prefix for XSLT elements; no prefix for HTML elements; `fo:` prefix for XSL-FO elements. Consequently, in order to use the entry helpers, the namespace declarations in the XSLT document must define prefixes that match the built-in prefixes shown in the entry helpers.

Auto-completion

In Text View, auto-completion is available in a pop-up as you type, with the first item in the pop-up list being highlighted that matches the typed text. When an element is being typed, a list of elements pops up with the first nearest match in alphabetical order being highlighted. Similarly, when an attribute is being typed in, a list of applicable attributes pops up. The items in the list are determined according to the rules described in the previous section on entry helpers.

XPath intelligent editing

At locations in the XSLT document where XPath expressions can be entered (for example, the value of a `select` attribute), intelligent XPath editing is available. XPath functions and XPath axes become available in popup as you type.

Validating XSLT documents

The XSLT document can be validated against the XSLT schema built into XMLSpy (click **XML | Validate (F8)**). The correct built-in schema is automatically selected according to whether the XSLT document is XSLT 1.0 or XSLT 2.0 (specified in the `version` attribute of the `xsl:stylesheet` element).

 © 2011 Altova GmbH

6.1.2 XSLT Processing

In the XMLSpy GUI, two types of XSLT transformation are available:

- The **XSL/XQuery | XSL Transformation (F10)** command is used for straightforward XML transformations with an XSLT stylesheet to result formats specified and described in the stylesheets.
- The **XSL/XQuery | XSL-FO Transformation** command is used: (i) for transformations of XML to FO to PDF in two steps, and (ii) for one-step transformations of FO to PDF.

Specifying the XSLT processor for the transformation
The XSLT engine that will be used for transformations is specified in the XSL tab of the Options dialog

The available options are explained in the User Reference section. The engine specified in the XSL tab will be used for all XSLT transformations. Note that for the XSL:FO transformation, an additional XSLT engine option is available: the XSLT engine that is packaged with some FO processors. To select this option, select the corresponding radio button at the bottom of the XSL tab (*see screenshot above*).

Specifying the FO processor
The FO processor that will be used for transformations of FO to PDF is specified in the text box at the bottom of the XSL tab of the Options dialog (*screenshot above*).

XSLT 1.0 and 2.0 and Altova's XSLT engines
The XSLT version of a stylesheet is specified in the `version` attribute of the `xsl:stylesheet` (or `xsl:transform`) element. XMLSpy contains the built-in Altova XSLT 1.0 and Altova XSLT 2.0 engines, and the appropriate engine is selected according to the value of the version attribute (`1.0` or `2.0`).

XSLT Transformation
The **XSLT Transformation (F8)** command can be used in the following scenarios:

- To transform an XML document that is active in the GUI and has an XSLT document assigned to it. If no XSLT document is assigned, you are prompted to make an assignment when you click the **XSLT Transformation (F8)** command.
- To transform an XSLT document that is active in the GUI. On clicking the **XSLT Transformation (F8)** command, you are prompted for the XML file you wish to process with the active XSLT stylesheet.
- To transform project folders and files. Right-click the project folder or file and select the command.

XSL:FO Transformation
The **XSL:FO Transformation** command can be used in the following scenarios:

- To transform an XML document that is active in the GUI and has an XSLT document assigned to it. The XML document will first be transformed to FO using the specified XSLT engine. The FO document will then be processed with the specified FO processor to produce the PDF output. If no XSLT document is assigned, you are prompted to make an assignment when you click the **XSL:FO Transformation** command.

- To transform an FO document to PDF using the specified FO processor.
- To transform an XSLT document that is active in the GUI. On clicking the **XSL:FO Transformation** command, you are prompted for the XML file you wish to process with the active XSLT stylesheet.
- To transform project folders and files. Right-click the project folder or file and select the command.

For a description of the options in the XSL:FO output dialog, see the User Reference section.

Parameters for XSLT
If you are using the Altova XSLT engines, XSLT parameters can be stored in a convenient GUI dialog. All the stored parameters are passed to the XSLT document each time you transform. For more information, see the description of the XSLT Parameters / XQuery Variable command.

Batch processing with AltovaXML
AltovaXML is a free standalone application that contains Altova's XML validator, XSLT engines, and XQuery engine. It can be used from the command line, via a COM interface, in Java programs, and in .NET applications to validate XML documents, transform XML documents using XSLT 1.0 and 2.0 stylesheets, and execute XQuery documents.

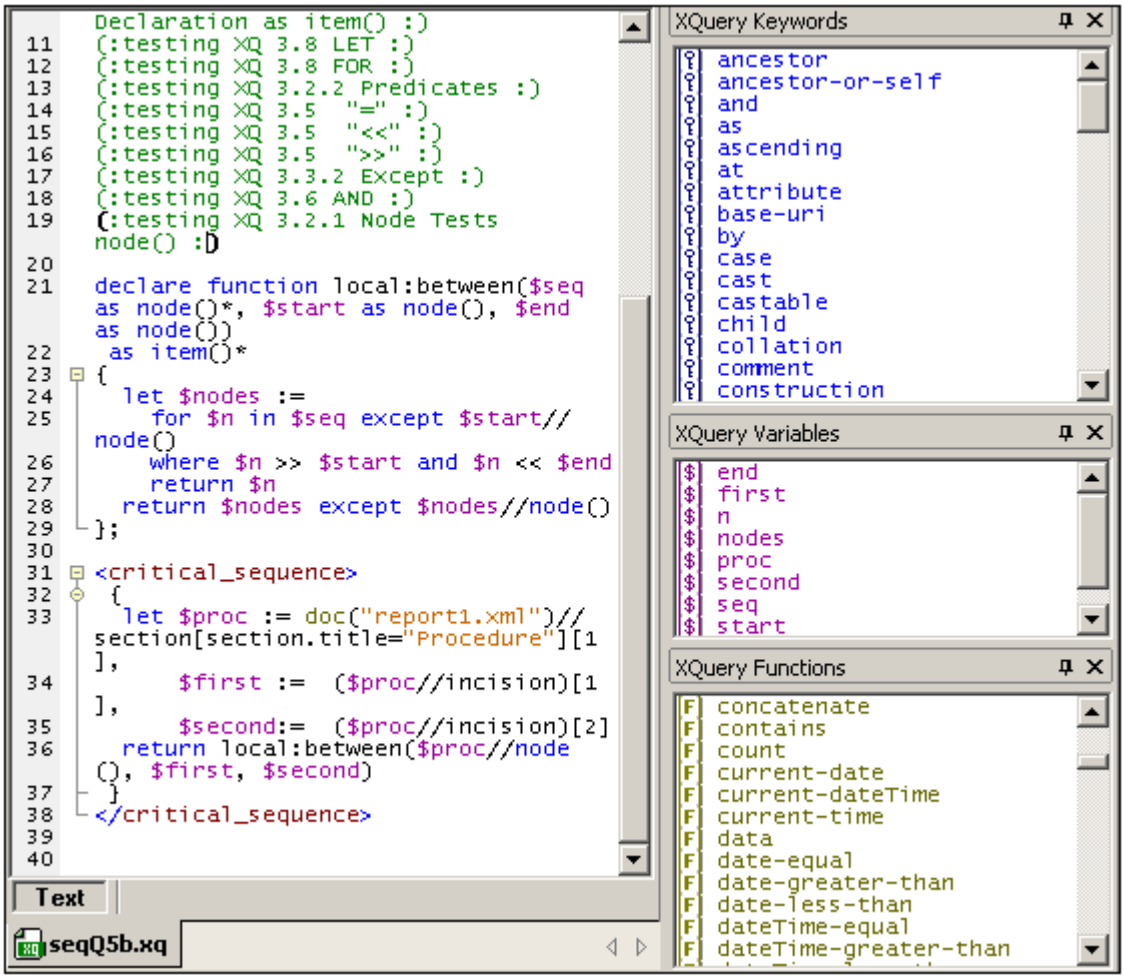
XSLT transformation tasks can therefore be automated with the use of Altova XML. For example, you can create a batch file that calls AltovaXML to transform a set of documents. See the Altova XML documentation for details.

© 2011 Altova GmbH

6.2 XQuery

Altova website: XQuery Editor

XQuery documents can be edited in Text View. The Entry Helpers, syntax coloring, and intelligent editing are different than for XML documents (*see screenshot; line numbering and folding margins in Enterprise and Professional Editions only*). We call this mode of Text View its XQuery Mode. In addition, you can validate your XQuery document in Text View and execute the code in an XQuery document (with an optional XML file if required) using the built-in Altova XQuery Engine.



Please note: XQuery files can be edited only in Text View. No other views of XQuery files are available.

Altova XQuery Engine

For details about how the Altova XQuery Engine is implemented and will process XQuery files, see XQuery Engine Implementation.

AltovaXML for command line and batch processing

The XMLSpy GUI enables batch processing via the projects functionality. However, if you are

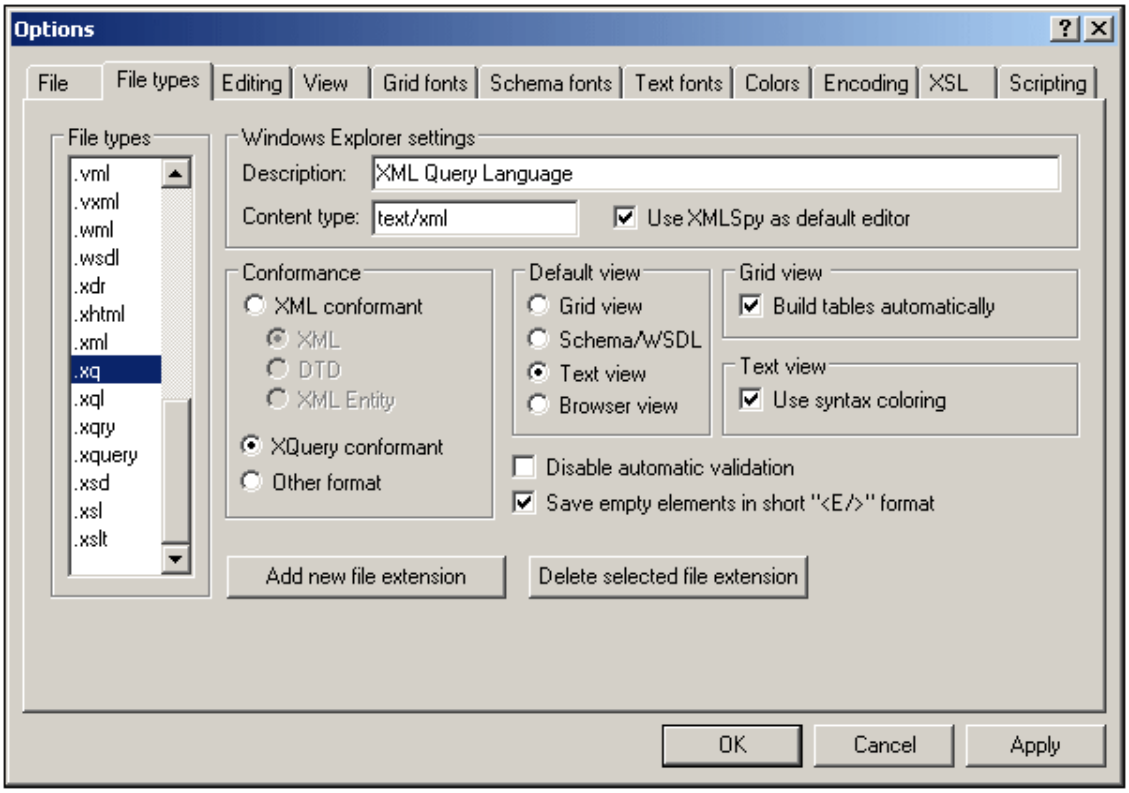
looking for more flexibility, you should try Altova's free AltovaXML product, which contains the Altova XQuery Engine that is built into XMLSpy. AltovaXML is ideal if you wish to perform XQuery executions from the command line, or batch processing.

6.2.1 XQuery Documents

An XQuery document is opened automatically in XQuery Mode of Text View if it is XQuery conformant. Files that have the file extension `.xq`, `.xql`, and `.xquery` are pre-defined in XMLSpy as being XQuery conformant.

Setting additional file extensions to be XQuery conformant
To set additional file extensions to be XQuery conformant:

1. Select **Tools | Options**. The Options dialog appears (*see screenshot*).
2. Select the **File types** tab.
3. Click Add new file extension to add the new file extension to the list of file types.
4. Under **Conformance**, select **XQuery conformant**.



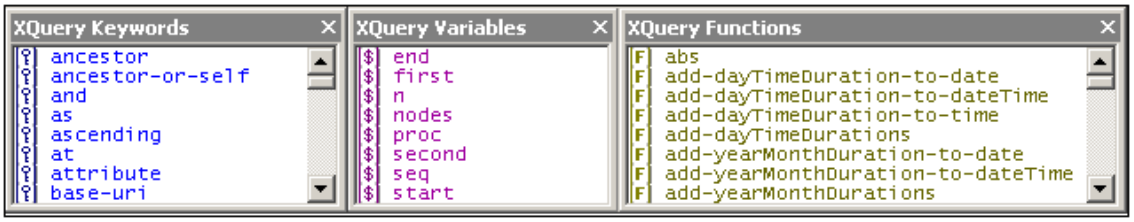
You should also make the following **Windows Explorer settings** in this dialog:

- **Description:** XML Query Language
- **Content** type: text/xml
- If you wish to use XMLSpy as the default editor for XQuery files, activate the **Use XMLSpy as default editor** check box.

 © 2011 Altova GmbH

6.2.2 XQuery Entry Helpers

There are three Entry Helpers in the XQuery Mode of Text View: XQuery Keywords (blue), XQuery Variables (purple), and XQuery Functions (olive).



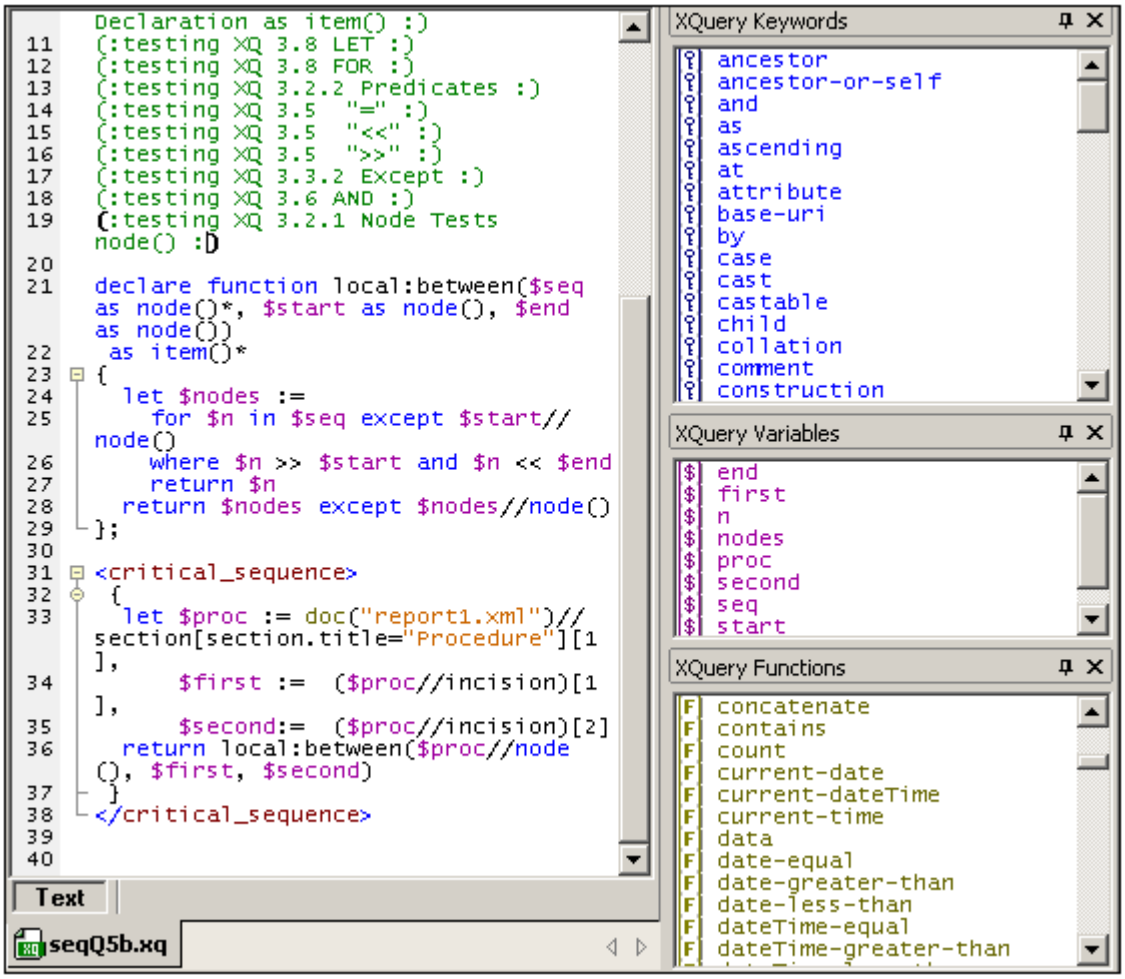
Note the following points:

- The color of items in the three Entry Helpers are different and correspond to the syntax coloring used in the text. These colors cannot be changed.
- The listed keywords and functions are those supported by the Altova XQuery Engine.
- The variables are defined in the XQuery document itself. When a `$` and a character are entered in Text View, the character is entered in the Variables Entry Helper (unless a variable consisting of exactly that character exists). As soon as a variable name that is being entered matches a variable name that already exists, the newly entered variable name disappears from the Entry Helper.
- To navigate in any Entry Helper, click an item in the Entry Helper, and then use either the scrollbar, mouse wheel, or page-down and page-up to move up and down the list.

To insert any of the items listed in the Entry Helpers into the document, place the cursor at the required insertion point and double-click the item. In XQuery, some character strings represent both a keyword and a function (`empty`, `unordered`, and `except`). These strings are always entered as keywords (in blue)—even if you select the function of that name in the Functions Entry Helper. When a function appears in blue, it can be distinguished by the parentheses that follow the function name.

6.2.3 XQuery Syntax Coloring

An XQuery document can consist of XQuery code as well as XML code. The default syntax coloring for the XQuery code is described in this section. The syntax coloring for XML code in an XQuery document is the same as that used for regular XML documents. All syntax coloring (for both XQuery code and XML code) is set in the Text Fonts tab of the Options dialog (**Tools | Options**). Note that XQuery code can be contained in XML elements by enclosing the XQuery code in curly braces **{ }** (*see screenshot for example*).



In XQuery code in the XQuery Mode of Text View, the following default syntax coloring is used:

- `(: Comments, including 'smiley' delimiters, are in green :)`
- XQuery Keywords are in blue: `keyword`
- XQuery Variables, including the dollar sign, are in purple: `$start`
- XQuery Functions, but **not** their parentheses, are in olive: `function()`
- Strings are in orange: `"Procedure"`
- All other text, such as path expressions, is black (*shown underlined below*). So:

```
for $s in doc("report1.xml")//section[section.title =
"Procedure"]
return ($s//incision)[2]/instrument
```

You can change these default colors and other font properties in the Text Fonts tab of the Options dialog (**Tools | Options**).

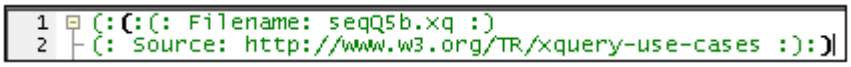
Please note: In the above screenshot, one pair of colored parentheses for a comment is displayed black and bold. This is because of the bracket-matching feature (see XQuery Intelligent Editing).

 © 2011 Altova GmbH

6.2.4 XQuery Intelligent Editing

The XQuery Mode of Text View provides the following intelligent editing features.

Bracket-matching
The bracket-matching feature highlights the opening and closing brackets of a pair of brackets, enabling you to clearly see the contents of a pair of brackets. This is particularly useful when brackets are nested, as in XQuery comments (*see screenshot below*).

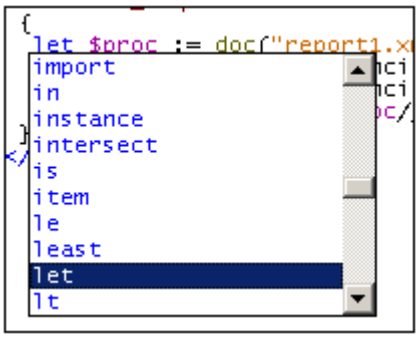


Bracket-matching is activated when the cursor is placed either immediately before or immediately after a bracket (either opening or closing). That bracket is highlighted (bold black) together with its corresponding bracket. Notice the cursor position in the screenshot above.

Bracket-matching is enabled for round parentheses **( )** , square brackets **[ ]** , and curly braces **{ }** . The exception is angular brackets **<>**, which are used for XML tags.

Please note: When you place the cursor just inside a start or end bracket, both brackets are highlighted. Pressing **Ctrl+E** moves the cursor to the other member of the pair. Pressing **Ctrl+E** repeatedly enables you to switch between the start and end brackets. This is another aid to quickly navigating your document.

Keywords
XQuery keywords are instructions used in query expressions, and they are displayed in blue. You select a keyword by placing the cursor inside a keyword, or immediately before or after it. With a keyword selected, pressing **Ctrl+Space** causes a complete list of keywords to be displayed in a pop-up menu. You can scroll through the list and double-click a keyword you wish to have replace the selected keyword.

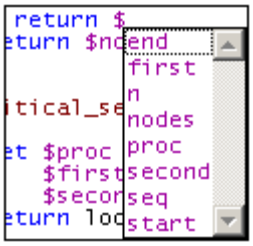


In the screenshot above, the cursor was placed in the `let` keyword. Double-clicking a keyword from the list causes it to replace the `let` keyword.

Variables
Names of variables are prefixed with the **$** sign, and they are displayed in purple. This mechanism of the intelligent editing feature is similar to that for keywords. There are two ways to access the pop-up list of all variables in a document:

- After typing a **$** character, press **Ctrl+Space**
- Select a variable and press **Ctrl+Space**. (A variable is selected when you place the cursor immediately after the **$** character, or within the name of a variable, or

immediately after the name of a variable.)

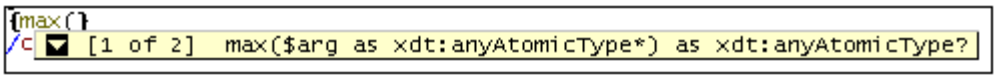


To insert a variable after the **$** character (when typing), or to replace a selected variable, double-click the variable you want in the pop-up menu.

Functions
Just as with keywords and variables, a pop-up menu of built-in functions is displayed when you select a function (displayed in olive) and press **Ctrl+Space**. (A function is selected when you place the cursor within a function name, or immediately before or after a function name. The cursor must not be placed between the parentheses that follow the function's name.) Double-clicking a function name in the pop-up menu replaces the selected function name with the function from the pop-up menu.

To display a tip containing the signature of a function (*screenshot below*), place the cursor immediately after the opening parenthesis and press **Ctrl+Space**.

Please note: The signature can be displayed only for standard XQuery functions.



The downward-pointing arrowhead indicates that there is more than one function with the same name but with different arguments or return types. Click on the signature to display the signature of the next function (if available); click repeatedly to cycle through all the functions with that name. Alternatively, you can use the **Ctrl+Shift+Up** or **Ctrl+Shift+Down** key-combinations to move through a sequence.

6.2.5 XQuery Validation and Execution

Validating XQuery documents
To validate an XQuery document:

1. Make the XQuery document the active document.
2. Select **XML | Validate**, or press the **F8** key, or click the toolbar icon.

The document will be validated for correct XQuery syntax.

Executing XQuery documents
XQuery documents are executed within XMLSpy using the built-in XQuery 1.0 engine. The output is displayed in a window in XMLSpy.

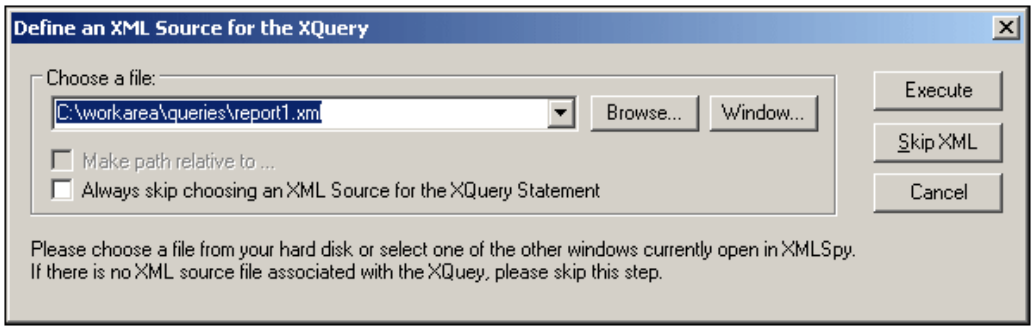
Typically, an XQuery document is not associated with any single XML document. This is because XQuery expressions can select any number of XML documents with the `doc()` function. In XMLSpy, however, before executing individual XQuery documents you can select a source XML document for the execution. In such cases, the document node of the selected

 © 2011 Altova GmbH

XML source is the starting context item available at the root level of the XQuery document. Paths that begin with a leading slash are resolved with this document node as its context item.

To execute an XQuery document:

1. Make the XQuery document the active document.
2. Select **XSL/XQuery | XQuery Execution** or click the toolbar icon. This opens the Define an XML Source for the XQuery dialog.



3. Do one of the following:
   - To select an XML file, use either the **Browse** button or the **Window** button (which lists files that are open in XMLSpy and that are in XMLSpy projects). Select an XML source if you wish to assign its document node as the context item for the root level of the XQuery document. Click **Execute**.
   - To skip this dialog click **Skip XML**.

The result document is generated as a temporary file that can be saved to any location with the desired file format and extension.

XQuery Variables

If you are using the Altova XQuery engine, XQuery variables can be stored in a convenient GUI dialog. All the stored variables are passed to the XQuery document each time you execute an XQuery document via XMLSpy. For more information, see the description of the XSLT Parameters / XQuery Variable command.

Altova XQuery Engine

For details about how the Altova XQuery Engine is implemented and will process XQuery files, see XQuery Engine Implementation.

7 Authentic

Authentic View (*screenshot below*) is a graphical representation of your XML document. It enables XML documents to be displayed without markup and with appropriate formatting and data-entry features such as input fields, combo boxes, and radio buttons. Data that the user enters in Authentic View is entered into the XML file.

To be able to view and edit an XML document in Authentic View, the XML document must be associated with a **StyleVision Power Stylesheet (SPS),** which is created in Altova's StyleVision product. An SPS (. `sps` file) is, in essence, an XSLT stylesheet. It specifies an output presentation for an XML file that can include data-entry mechanisms. Authentic View users can, therefore, write data back to the XML file or DB. An SPS is based on a schema and is specific to it. If you wish to use an SPS to edit an XML file in Authentic View, you must use one that is based on the same schema as that on which the XML file is based.

Using Authentic View

- If an XML file is open, you can switch to Authentic View by clicking the **Authentic** button at the bottom of the Main Window. If an SPS is not already assigned to the XML file, you will be prompted to assign one to it. You must use an SPS that is based on the same schema as the XML file.
- A new XML file is created and displayed in Authentic View by selecting the **File | New** command and then clicking the "Select a StyleVision Stylesheet" button. This new file is a template file associated with the SPS you open. It can have a variable amount of starting data already present in it. This starting data is contained in an XML file (a Template XML File) that may optionally be associated with the SPS. After the Authentic View of an XML file is displayed, you can enter data in it and save the file.
- You can also open an SPS via the **Authentic | New Document** command. If a

 © 2011 Altova GmbH

Template XML File has been assigned to the SPS, then the data in the Template XML File is used as the starting data of the XML document template created in Authentic View.

In this section

This section contains an Authentic View tutorial, which shows you how to use Authentic View. It is followed by the section, Editing in Authentic View, which explains individual editing features in detail.

More information about Authentic View

For more information about Authentic View information, see (i) the section Editing Views | Authentic View in this documentation, which describes the Authentic View editing window, and (ii) the Authentic menu section of the User Reference part of this documentation.

7.1 Authentic View Tutorial

In Authentic View, you can edit XML documents in a graphical WYSIWYG interface (*screenshot below*), just like in word-processor applications such as Microsoft Word. In fact, all you need to do is enter data. You do not have to concern yourself with the formatting of the document, since the formatting is already defined in the stylesheet that controls the Authentic View of the XML document. The stylesheet (StyleVision Power Stylesheet, shortened to SPS in this tutorial) is created by a stylesheet designer using Altova's StyleVision product.

Editing an XML document in Authentic View involves two user actions: (i) editing the structure of the document (for example, adding or deleting document parts, such as paragraphs and headlines); and (ii) entering data (the content of document parts).

This tutorial takes you through the following steps:

- Opening an XML document in Authentic View. The key requirement for Authentic View editing is that the XML document be associated with an SPS file.
- A look at the Authentic View interface and a broad description of the central editing mechanisms.
- Editing document structure by inserting and deleting nodes.
- Entering data in the XML document.
- Entering (i) attribute values via the Attributes entry helper, and (ii) entity values.
- Printing the document.

Remember that this tutorial is intended to get you started, and has intentionally been kept simple. You will find additional reference material and feature descriptions in the Authentic View interface section.

 © 2011 Altova GmbH

Tutorial requirements
All **the files** you need for the tutorial are in the `C:\Documents and Settings\<username>\My Documents\Altova\<ProductName>\Examples` folder of your Altova application folder. These files are:

- `NanonullOrg.xml` (the XML document you will open)
- `NanonullOrg.sps` (the StyleVision Power Stylesheet to which the XML document is linked)
- `NanonullOrg.xsd` (the XML Schema on which the XML document and StyleVision Power Stylesheet are based, and to which they are linked)
- `nanonull.gif` and `Altova_right_300.gif` (two image files used in the tutorial)

Please note: At some points in the tutorial, we ask you to look at the XML text of the XML document (as opposed to the Authentic View of the document). If the Altova product edition you are using does not include a Text View (as with Authentic Desktop and Authentic Browser), then use a plain **text editor** like Wordpad or Notepad to view the text of the XML document.

Caution: We recommend that you use a copy of `NanonullOrg.xml` for the tutorial, so that you can always retrieve the original should the need arise.

7.1.1 Opening an XML Document in Authentic View

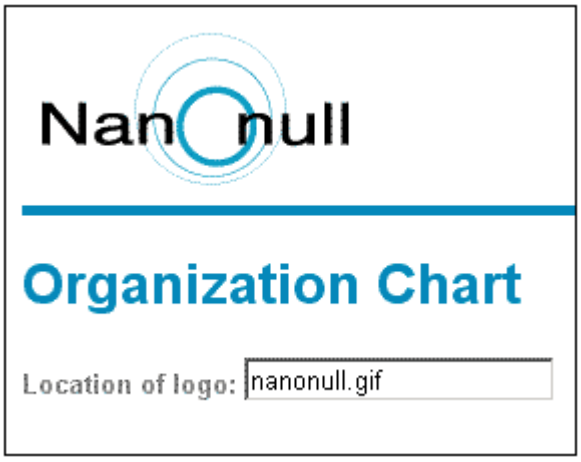
In Authentic View, you can edit an existing XML document or create and edit a new XML document. In this tutorial, you will open an existing XML document in Authentic View (described in this section) and learn how you can edit it (subsequent sections). Additionally, in this section is a description of how a new XML document can be created for editing in Authentic View.

Opening an existing XML document
The file you will open is `NanonullOrg.xml`. It is in the `Examples` folder of your Altova application. You can open `NanonullOrg.xml` in one of two ways:

- Click **File | Open** in your Altova product, then browse for `NanonullOrg.xml` in the dialog that appears, and click **Open**.
- Use Windows Explorer to locate the file, right-click, and select your Altova product as the application with which to open the file.

The file `NanonullOrg.xml` opens directly in Authentic View (*screenshot below).* This is because

1. The file already has a StyleVision Power Stylesheet (SPS) assigned to it, and
2. In the Options dialog (**Tools | Options**), in the View tab, the option to open XML files in Authentic View if an SPS file is assigned has been checked. (Otherwise the file would open in Text View.)



Remember: It is the SPS that defines and controls how an XML document is displayed in Authentic View. Without an SPS, there can be no Authentic View of the document.

Creating a new XML document based on an SPS
You can also create a new XML document that is based on an SPS. You can do this in two ways: via the **File | New** menu command and via the **Authentic | New Document** menu command. In both cases an SPS is selected.

Via File | New

1. Select **File | New**, and, in the Create a New Document dialog, select XML as the new file type to create.
2. Click **Select a STYLEVISION Stylesheet**, and browse for the desired SPS.

Via Authentic | New Document

1. Select **Authentic | New Document.**
2. In the Create a New Document dialog, browse for the desired SPS.

If a Template XML File has been assigned to the SPS, then the data in the Template XML File is used as the starting data of the XML document template created in Authentic View.

7.1.2 The Authentic View Interface

The Authentic View editing interface consists of a main window in which you enter and edit the document data, and three entry helpers. Editing a document is simple. If you wish to see the markup of the document, switch on the markup tags. Then start typing in the content of your document. To modify the document structure, you can use either the context menu or the Elements entry helper.

Displaying XML node tags (document markup)
An XML document is essentially a hierarchy of nodes. For example:

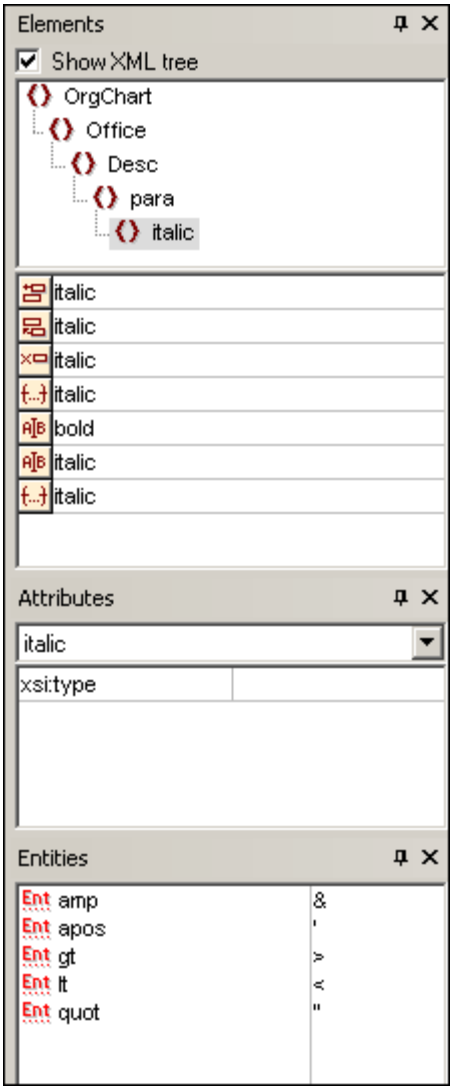
```
<DocumentRoot>
   <Person id="ABC001">
      <Name>Alpha Beta</Name>
      <Address>Some Address</Address>
      <Tel>1234567</Tel>
   </Person>
</DocumentRoot>
```

 © 2011 Altova GmbH

By default, the node tags are not displayed in Authentic View. You can switch on the node tags by selecting the menu item **Authentic | Show Large Markup** (or the toolbar icon). Large markup tags contain the names of the respective nodes. Alternatively, you can select small markup (no node names in tags) and mixed markup (a mixture of large, small, and no markup tags, which is defined by the designer of the stylesheet; the default mixed markup for the document is no markup).

You can view the text of the XML document in the Text View of your Altova product or in a text editor.

Entry helpers
There are three entry helpers in the interface (*screenshot below*), located by default along the right edge of the application window. These are the Elements, Attributes, and Entity entry helpers.



Elements entry helper: The Elements entry helper displays elements that can be inserted and removed with reference to the current location of the cursor or selection in the Main Window.

Note that the entry helper is context-sensitive; its content changes according to the location of the cursor or selection. The content of the entry helper can be changed in one other way: when another node is selected in the XML tree of the Elements entry helper, the elements relevant to that node are displayed in the entry helper. The Elements entry helper can be expanded to show the XML tree by checking the Show XML Tree check box at the top of the entry helper (*see screenshot above*). The XML tree shows the hierarchy of nodes from the top-level element node all the way down to the node selected in the Main Window.

Attributes entry helper: The Attributes entry helper displays the attributes of the element selected in the Main Window, and the values of these attributes. Attribute values can be entered or edited in the Attributes entry helper. Element nodes from the top-level element down to the selected element are available for selection in the combo box of the Atributes entry helper. Selecting an element from the dropdown list of the combo box causes that element's attributes to be displayed in the entry helper, where they can then be edited.

Entities entry helper: The Entities entry helper is not context-sensitive, and displays all the entities declared for the document. Double-clicking an entity inserts it at the cursor location. How to add entities for a document is described in the section Authentic View Interface.

Context menu
Right-clicking at a location in the Authentic View document pops up a context menu relevant to that (node) location. The context menu provides commands that enable you to:

- Insert nodes at that location or before or after the selected node. Submenus display lists of nodes that are allowed at the respective insert locations.
- Remove the selected node (if this allowed by the schema) or any removable ancestor element. The nodes that may be removed (according to the schema) are listed in a submenu.
- Insert entities and CDATA sections. The entities declared for the document are listed in a submenu. CDATA sections can only be inserted withn text.
- Cut, copy, paste (including pasting as XML or text), and delete document content.

Note: For more details about the interface, see Authentic View Interface

7.1.3 Node Operations

There are two major types of nodes you will encounter in an Authentic View XML document: **element nodes** and **attribute nodes**. These nodes are marked up with tags, which you can switch on. There are also other nodes in the document, such as text nodes (which are not marked up) and CDATA section nodes (which are marked up, in order to delimit them from surrounding text).

The node operations described in this section refer only to element nodes and attribute nodes. When trying out the operations described in this section, it is best to have large markup switched on.

Note: It is important to remember that **only same- or higher-level elements** can be inserted before or after the selected element. Same-level elements are **siblings**. Siblings of a paragraph element would be other paragraph elements, but could also be lists, a table, an image, etc. Siblings could occur before or after an element. Higher-level elements are **ancestor** elements and siblings of ancestors. For a paragraph element, ancestor elements could be a section, chapter, article, etc. A paragraph in a valid XML file would already have ancestors. Therefore, adding a higher-level element in Authentic View, creates the new element as a sibling of the relevant ancestor. For example, if a section

element is inserted after a paragraph, it is created as a sibling of the section that contains the current paragraph element.

Carrying out node operations

Node operations can be carried out by selecting a command in the context menu or by clicking the node operation entry in the Elements entry helper. In some cases, an element or attribute can be added by clicking the Add Node link in the Authentic View of the document. In the special cases of elements defined as paragraphs or list items, pressing the Enter key when within such an element creates a new sibling element of that kind. This section also describes how nodes can be created and deleted by using the Apply Element, Remove Node, and Clear Element mechanisms.

Inserting elements

Elements can be inserted at the following locations:

- The cursor location within an element node. The elements available for insertion at that location are listed in a submenu of the context menu's **Insert** command. In the Elements entry helper, elements that can be inserted at a location are indicated with the icon. In the `NanonullOrg.xml` document, place the cursor inside the `para` element, and create `bold` and `italic` elements using both the context menu and Elements entry helper.
- Before or after the selected element or any of its ancestors, if allowed by the schema. Select the required element from the submenu/s that roll out. In the Elements entry helper, elements that can be inserted before or after the selected element are indicated with the and icons, respectively. Note that in the Elements entry helper, you can insert elements before/after the selected element only; you cannot insert before/after an ancestor element. Try out this command, by first placing the cursor inside the `para` element and then inside the table listing the employees.

Add Node link

If an element or attribute is included in the document design, and is not present in the XML document, an `Add Node` link is displayed at the location in the document where that node is specified. To see this link, in the line with the text, *Location of logo*, select the `@href` node within the `CompanyLogo` element and delete it (by pressing the **Delete** key). The `add @href` link appears within the `CompanyLogo` element that was edited (*screenshot below*). Clicking the link adds the `@href` node to the XML document. The text box within the `@href` tags appears because the design specifies that the `@href` node be added like this. You still have to enter the value (or content) of the `@href` node. Enter the text `nanonull.gif`.

If the content model of an element is ambiguous, for example, if it specifies that a sequence of child elements may appear in any order, then the `add...` link appears. Note that no node name

is specified. Clicking the link will pop up a list of elements that may validly be inserted.

Note: The Add Node link appears directly in the document template; there is no corresponding entry in the context menu or Elements entry helper.

Creating new elements with the Enter key
In cases where an element has been formatted as a paragraph or list item (by the stylesheet designer), pressing the Enter key when inside such a node causes a new node of that kind to be inserted after the current node. You can try this mechanism in the `NanonullOrg.xml` document by going to the end of a `para` node (just before its end tag) and pressing **Enter**.

Applying elements
In elements of mixed content (those which contain both text and child elements), some text content can be selected and an allowed child element be applied to it. The selected text becomes the content of the applied element. To apply elements, in the context menu, select Apply and then select from among the applicable elements. (If no elements can be applied to the selected text, then the Apply command does not appear in the context menu.) In the Elements entry helper, elements that can be applied for a selection are indicated with the icon. In the `NanonullOrg.xml` document, select text inside the mixed content `para` element and experiment with applying the `bold` and `italic` elements.

The stylesheet designer might also have created a toolbar icon to apply an element. In the `NanonullOrg.xml` document, the `bold` and `italic` elements can be applied by clicking the bold and italic icons in the application's Authentic toolbar.

Removing nodes
A node can be removed if its removal does not render the document invalid. Removing a node causes a node and all its contents to be deleted. A node can be removed using the **Remove** command in the context menu. When the Remove command is highlighted, a submenu pops up which contains all nodes that may be removed, starting from the selected node and going up to the document's top-level node. To select a node for removal, the cursor can be placed within the node, or the node (or part of it) can be highlighted. In the Elements entry helper, nodes that can be removed are indicated with the icon. A removable node can also be removed by selecting it and pressing the **Delete** key. In the `NanonullOrg.xml` document, experiment with removing a few nodes using the mechanisms described. You can undo your changes with **Ctrl +Z**.

Clearing elements
Element nodes that are children of elements with mixed content (both text and element children) can be cleared. The entire element can be cleared when the node is selected or when the cursor is placed inside the node as an insertion point. A text fragment within the element can be cleared of the element markup by highlighting the text fragment. With the selection made, select **Clear** in the context menu and then the element to clear. In the Elements entry helper, elements that can be cleared for a particular selection are indicated with the icon (insertion point selection) and icon (range selection). In the `NanonullOrg.xml` document, try the clearing mechanism with the `bold` and `italic` child elements of `para` (which has mixed content).

Tables and table structure
There are two types of Authentic View table:

 © 2011 Altova GmbH

- *SPS tables (static and dynamic)*. The broad structure of SPS table is determined by the stylesheet designer. Within this broad structure, the only structural changes you are allowed are content-driven. For example, you could add new rows to a dynamic SPS table.
- *XML tables*, in which you decide to present the contents of a particular node (say, one for person-specific details) as a table. If the stylesheet designer has enabled the creation of this node as an XML table, then you can determine the structure of the table and edit its contents. XML tables are discussed in detail in the Using tables in Authentic View section.

7.1.4 Entering Data in Authentic View

Data is entered into the XML document directly in the main window of Authentic View. Additionally for attributes, data (the value of the attribute) can be entered in the Attributes entry helper. Data is entered (i) directly as text, or (ii) by selecting an option in a data-entry device, which is then mapped to a predefined text entry.

Adding text content

You can enter element content and attribute values directly as text in the main window of Authentic View. To insert content, place the cursor at the location where you want to insert the text, and type. You can also copy text from the clipboard into the document. Content can also be edited using standard editing mechanisms, such as the **Caps** and **Delete** keys. For example, you can highlight the text to be edited and type in the replacement text with the **Caps** key on.

For example, to change the name of the company, in the `Name` field of `Office`, place the cursor after Nanonull, and type in `USA` to change the name from Nanonull, Inc. to Nanonull USA, Inc.

If text is editable, you will be able to place your cursor in it and highlight it, otherwise you will not be able to. Try changing any of the **field names** (not the field values), such as "Street", "City", or "State/Zip," in the address block. You are not able to place the cursor in this text because such text is not XML content; it is derived from the StyleVision Power Stylesheet.

Inserting special characters and entities

When entering data, the following type of content is handled in a special way:

- *Special characters that are used for XML markup* (ampersand, apostrophe, greater than, less than, and quotes). These characters are available as built-in entities and can be entered in the document by double-clicking the respective entity in the Entities entry helper. If these characters occur frequently (for example, in program code listings), then they can be entered within CDATA sections. To insert a CDATA section, right-click at the location where you wish to enter the CDATA section, and select **Insert CDATA Section** from the context menu. The XML processor ignores all markup characters within CDATA sections. This also means that if you want a special character inside a CDATA section, you should enter that character and not its entity reference.
- *Special characters that cannot be entered via the keyboard* should be entered by copying them from the character map of your system to the required location in the document.

- *A frequently used text string* can be defined as an entity, which appears in the Entities entry helper. The entity is inserted at the required locations by placing the cursor at each required location and double-clicking the entity in the entry helper. This is useful for maintenance because the value of the text string is held in one location; if the value needs to be changed, then all that needs to be done is to change the entity definition.

Note: When markup is hidden in Authentic View, an empty element can easily be overlooked. To make sure that you are not overlooking an empty element, switch large or small markup on.

Try using each type of text content described above.

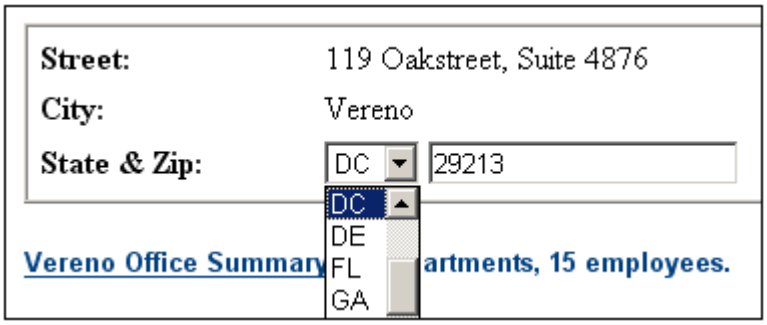
Adding content via a data-entry device

In the content editing you have learned above, content is added by directly typing in text as content. There is one other way that **element content** (or attribute values) can be entered in Authentic View: via data-entry devices.

Given below is a list of data-entry devices in Authentic View, together with an explanation of how data is entered in the XML file for each device.

Data-Entry Device	Data in XML File
Input Field (Text Box)	Text entered by user
Multiline Input Field	Text entered by user
Combo box	User selection mapped to value
Check box	User selection mapped to value
Radio button	User selection mapped to value
Button	User selection mapped to value

In the static table containing the address fields (*shown below*), there are two data-entry devices: an input field for the `Zip` field and a combo-box for the State field. The values that you enter in the text fields are entered directly as the XML content of the respective elements. For other data-entry devices, your selection is mapped to a value.



For the Authentic View shown above, here is the corresponding XML text:

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

 © 2011 Altova GmbH

Notice that the combo-box selection `DC` is mapped to a value of `DC`. The value of the `Zip` field is entered directly as content of the `ipo:zip` element.

7.1.5 Entering Attribute Values

An attribute is a property of an element, and an element can have any number of attributes. Attributes have values. You may sometimes be required to enter XML data as an attribute value. In Authentic View, you enter attribute values in two ways:

- As content in the main window if the attribute has been created to accept its value in this way
- In the Attributes entry helper

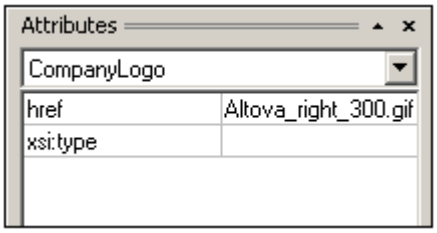
Attribute values in the main window

Attribute values can be entered as normal text or as text in an input field, or as a user selection that will be mapped to an XML value. They are entered in the same way that element content is entered: see Entering Data in Authentic View. In such cases, the distinction between element content and attribute value is made by the StyleVision Power Stylesheet and the data is handled appropriately.

Attribute values in the Attributes Entry Helper

If you wish to enter or change an attribute value, you can also do this in the Attributes Entry Helper. First, the attribute node is selected in Authentic View, then the value of the attribute is entered or edited in the Attributes entry helper. In the `NanonullOrg.xml` document, the location of the logo is stored as the value of the `href` attribute of the `CompanyLogo` element. To change the logo to be used:

1. Select the `CompanyLogo` element by clicking a CompanyLogo tag. The attributes of the `CompanyLogo` element are displayed in the Attributes Entry Helper.
2. In the Attributes Entry Helper, change the value of the `href` attribute from `nanonull.gif` to `Altova_right_300.gif` (an image in the `Examples` folder).



This causes the Nanonull logo to be replaced by the Altova logo.

Note: Entities cannot be entered in the Attributes entry helper.

7.1.6 Adding Entities

An entity in Authentic View is typically XML data (but not necessarily), such as a single character; a text string; and even a fragment of an XML document. An entity can also be a binary file, such as an image file. All the entities available for a particular document are displayed in the Entities Entry Helper (*screenshot below*). To insert an entity, place the cursor at the location in the document where you want to insert it, and then double-click the entity in the Entities entry helper. Note that you cannot enter entities in the Attributes entry helper.

The ampersand character (&) has special significance in XML (as have the apostrophe, less than and greater than symbols, and the double quote). To insert these characters, entities are used so that they are not confused with XML-significant characters. These characters are available as entities in Authentic View.

In `NanonullOrg.xml`, change the title of Joe Martin (in Marketing) to Marketing Manager Europe & Asia. Do this as follows:

1. Place the cursor where the ampersand is to be inserted.
2. Double-click the entity listed as "amp". This inserts an ampersand (*screenshot below*).

Note: The Entities Entry Helper is not context-sensitive. All available entities are displayed no matter where the cursor is positioned. This does not mean that an entity can be inserted at all locations in the document. If you are not sure, then validate the document after inserting the entity: **XML | Validate XML** (**F8**).

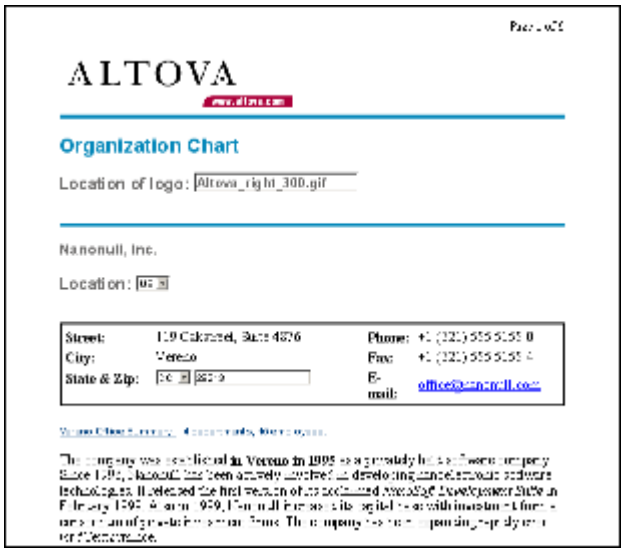
Defining your own entities
As a document editor, you can define your own document entities. How to do this is described in the section Defining Entities in Authentic View.

7.1.7 Printing the Document

A printout from Authentic View of an XML document preserves the formatting seen in Authentic View.

To print `NanonullOrg.xml`, do the following:

1. Switch to Hide Markup mode if you are not already in it. You must do this if you do not want markup to be printed.
2. Select **File | Print Preview** to see a preview of all pages. Shown below is part of a print preview page, reduced by 50%.



Notice that the formatting of the page is the same as that in Authentic View.

3. To print the file, click **File | Print**.

Note that you can also print a version of the document that displays markup. To do this, switch Authentic View to Show small markup mode or Show large markup mode, and then print.

7.2 Editing in Authentic View

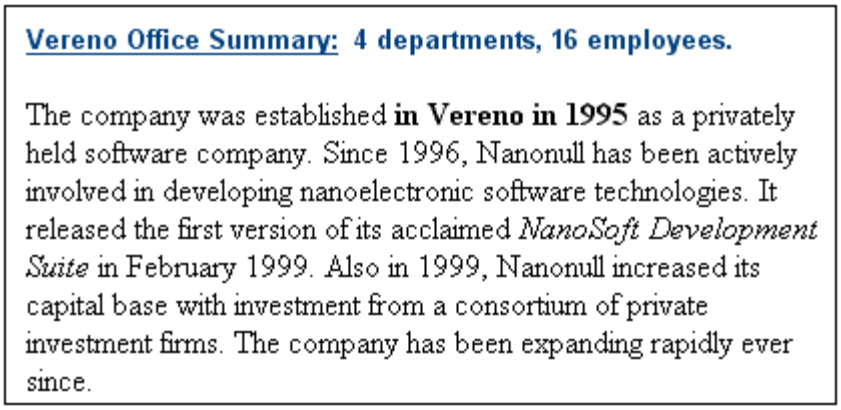
This section describes important features of Authentic View in detail. Features have been included in this section either because they are frequently used or because the mechanisms or concepts involved require explanation.

The section explains the following:

- There are three distinct types of tables used in Authentic View. The section Using tables in Authentic View explains the three types of tables (static SPS, dynamic SPS, and XML), and when and how to use them. It starts with the broad, conceptual picture and moves to the details of usage.
- The Date Picker is a graphical calendar that enters dates in the correct XML format when you click a date. See Using the Date Picker.
- An entity is shorthand for a special character or text string. You can define your own entities, which allows you to insert these special characters or text strings by inserting the corresponding entities. See Defining Entities for details.
- What image formats can be displayed in Authentic View.

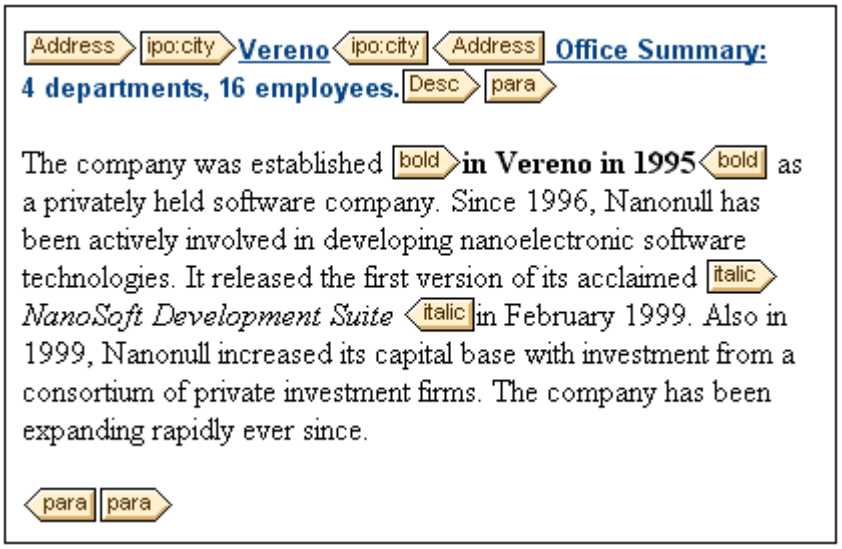
7.2.1 Basic Editing

When you edit in Authentic View, you are editing an XML document. Authentic View, however, can hide the structural XML markup of the document, thus displaying only the content of the document (*first screenshot below*). You are therefore not exposed to the technicalities of XML, and can edit the document as you would a normal text document. If you wish, you could switch on the markup at any time while editing (*second screenshot below*).



An editable Authentic View document with no XML markup.

 © 2011 Altova GmbH

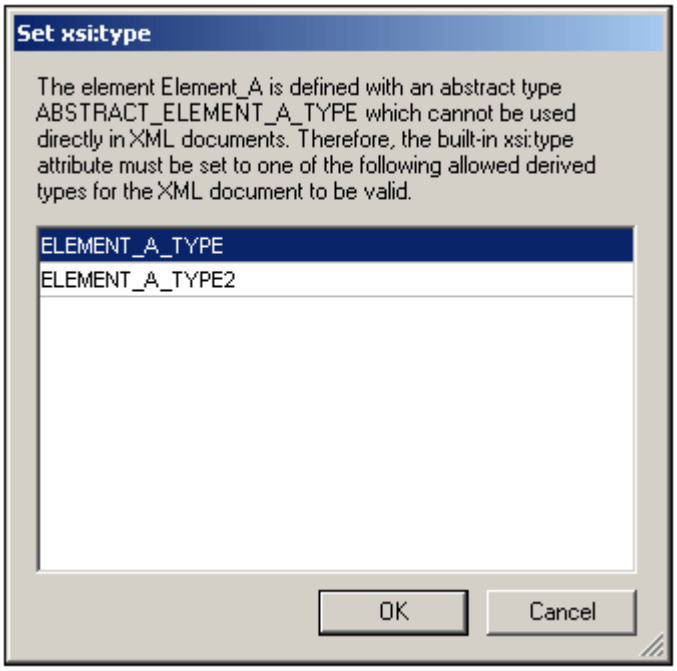


An editable Authentic View document with XML markup tags.

Inserting nodes

Very often you will need to add a new node to the Authentic XML document. For example, a new `Person` element might need to be added to an address book type of document. In such cases the XML Schema would allow the addition of the new element. All you need to do is right-click the node in the Authentic View document before which or after which you wish to add the new node. In the context menu that appears, select **Insert Before** or **Insert After** as required. The nodes available for insertion at that point in the document are listed in a submenu. Click the required node to insert it. The node will be inserted. All mandatory descendant nodes are also inserted. If a descendant node is optional, a clickable link, `Add NodeName`, appears to enable you to add the optional node if you wish to.

If the node being added is an element with an abstract type, then a dialog (*something like in the screenshot below*) appears containing a list of derived types that are available in the XML Schema.



Selecting one of the available derived types and clicking **OK** does the following:

- Sets the selected derived type as the value of the `xsi:type` attribute of the element
- Inserts the element together with the descendant nodes defined in the content model of the selected derived type.

The selected derived type can be changed subsequently by changing the value of the element's `xsi:type` attribute in the Attributes Entry Helper. When the element's type is changed in this way, all nodes of the previous type's content model are removed and nodes of the new type's content model are inserted.

Text editing

An Authentic View document will essentially consist of text and images. To edit the text in the document, place the cursor at the location where you wish to insert text, and type. You can copy, move, and delete text using familiar keystrokes (such as the **Delete** key) and drag-and-drop mechanisms. One exception is the **Enter** key. Since the Authentic View document is pre-formatted, you do not—and cannot—add extra lines or space between items. The **Enter** key in Authentic View therefore serves to append another instance of the element currently being edited, and should be used exclusively for this purpose.

Copy as XML or as text

Text can be copied and pasted as XML or as text.

- If text is pasted as XML, then the XML markup is pasted together with the text content of nodes. The XML markup is pasted even if only part of a node's contents has been copied. For the markup to be pasted it must be allowed, according to the schema, at the location where it is pasted.
- If text is pasted as text, XML markup is not pasted.

To paste as XML or text, first copy the text (**Ctrl+C**), right-click at the location where the text is to be pasted, and select the context menu command **Paste As | XML** or **Paste As | Text**. If the shortcut **Ctrl+V** is used, the text will be pasted in the default Paste Mode of the SPS. The default Paste Mode will have been specified by the designer of the SPS. For more details, see

 © 2011 Altova GmbH

the section Context Menus.

Alternatively, highlighted text can be dragged to the location where it is to be pasted. When the text is dropped, a pop-up appears asking whether the text is to be pasted as text or XML. Select the desired option.

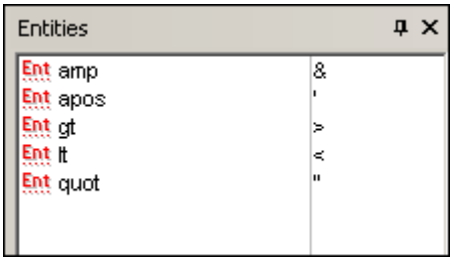
Text formatting

A fundamental principle of XML document systems is that content be kept separate from presentation. The XML document contains the content, while the stylesheet contains the presentation (formatting). In Authentic View, the XML document is presented via the stylesheet. This means that all the formatting you see in Authentic View is produced by the stylesheet. If you see bold text, that bold formatting has been provided by the stylesheet. If you see a list or a table, that list format or table format has been provided by the stylesheet. The XML document, which you edit in Authentic View contains only the content; it contains no formatting whatsoever. The formatting is contained in the stylesheet. What this means for you, the Authentic View user, is that you do not have to—nor can you—format any of the text you edit. You are editing content. The formatting that is automatically applied to the content you edit is linked to the semantic and/or structural value of the data you are editing. For example, an email address (which could be considered a semantic unit) will be formatted automatically in a certain way because of it is an email. In the same way, a headline must occur at a particular location in the document (both a structural and semantic unit) and will be formatted automatically in the way the stylesheet designer has specified that headlines be formatted. You cannot change the formatting of either email address or headline. All that you do is edit the content of the email address or headline.

In some cases, content might need to be specially presented; for example, a text string that must be presented in boldface. In all such cases, the presentation must be tied in with a structural element of the document. For example, a text string that must be presented in boldface, will be structurally separated from surrounding content by markup that the stylesheet designer will format in boldface. If you, as the Authentic View user, need to use such a text string, you would need to enclose the text string within the appropriate element markup. For information about how to do this, see the Insert Element command in the Elements Entry Helper section of the documentation.

Inserting entities

In XML documents, some characters are reserved for markup and cannot be used in normal text. These are the ampersand (&), apostrophe ('), less than (<), greater than (>), and quote (") characters. If you wish to use these characters in your data, you must insert them as entity references, via the Entities Entry Helper (*screenshot below*).



XML also offers the opportunity to create custom entities. These could be: (i) special characters that are not available on your keyboard, (ii) text strings that you wish to re-use in your document content, (iii) XML data fragments, or (iv) other resources, such as images. You can define your own entities within the Authentic View application. Once defined, these entities appear in the Entities Entry Helper and can then be inserted as in the document.

Inserting CDATA sections
CDATA sections are sections of text in an XML document that the XML parser does not process as XML data. They can be used to escape large sections of text if replacing special characters by entity references is undesirable; this could be the case, for example, with program code or an XML fragment that is to be reproduced with its markup tags. CDATA sections can occur within element content and are delimited by `<![CDATA[` and `]]>` at the start and end, respectively. Consequently the text string `]]>` should not occur within a CDATA section as it would prematurely signify the end of the section. In this case, the greater than character should be escaped by its entity reference (`>`). To insert a CDATA section within an element, place the cursor at the desired location, right-click, and select **Insert CDATA Section** from the context menu. To see the CDATA section tags in Authentic View, switch on the markup display. Alternatively, you could highlight the text that is to be enclosed in a CDATA section, and then select the **Insert CDATA section** command.

Editing and following links
A hyperlink consists of two parts: the link text and the target of the link. You can edit the link text by clicking in the text and editing. But you cannot edit the target of the link. (The target of the link is set by the designer of the stylesheet (either by typing in a static target address or by deriving the target address from data contained in the XML document).) From Authentic View, you can go to the target of the link by pressing **Ctrl** and clicking the link text. (Remember: merely clicking the link will set you up for editing the link text.)

7.2.2 Tables in Authentic View

The three table types fall into two categories: SPS tables (static and dynamic) and CALS/HTML Tables.

SPS tables are of two types: static and dynamic. SPS tables are designed by the designer of the StyleVision Power Stylesheet to which your XML document is linked. You yourself cannot insert an SPS table into the XML document, but you can enter data into SPS table fields and add and delete the rows of dynamic SPS tables. The section on SPS tables below explains the features of these tables.

CALS/HTML tables are inserted by you, the user of Authentic View. Their purpose is to enable you to insert tables at any allowed location in the document hierarchy should you wish to do so. The editing features of CALS/HTML Tables and the CALS/HTML Table editing icons are described below.

SPS Tables

Two types of SPS tables are used in Authentic View: static tables and dynamic tables.

Static tables are fixed in their structure and in the content-type of cells. You, as the user of Authentic View, can enter data into the table cells but you cannot change the structure of these tables (i.e. add rows or columns, etc) or change the content-type of a cell. You enter data either by typing in text, or by selecting from options presented in the form of check-box or radio button alternatives or as a list in a combo-box. After you enter data, you can edit it.

 © 2011 Altova GmbH

Nanonull, Inc.

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

Please note: The icons or commands for editing dynamic tables **must not** be used to edit static tables.

Dynamic tables have rows that represent a repeating data structure, i.e. each row has an identical data structure (not the case with static tables). Therefore, you can perform row operations: append row, insert row, move row up, move row down, and delete row. These commands are available under the **Authentic** menu and as icons in the toolbar (shown below).

To use these commands, place the cursor anywhere in the appropriate row, and then select the required command.

Administration

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

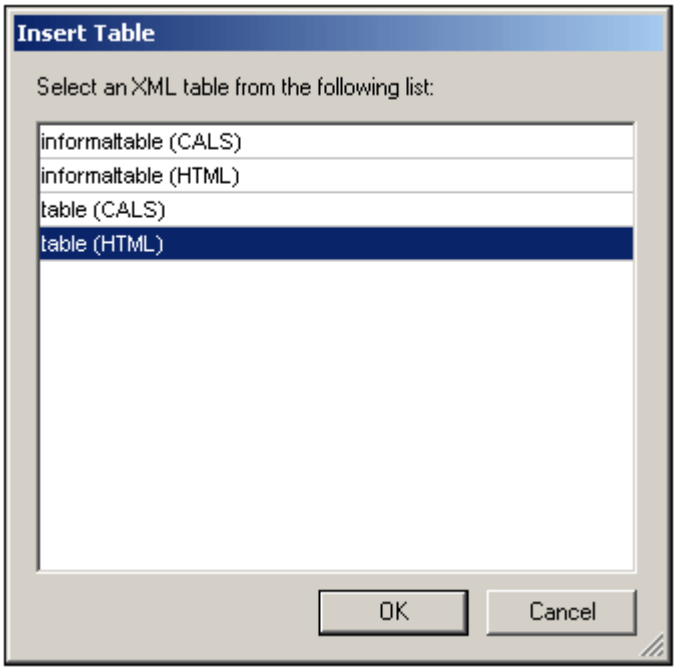
To move among cells in the table, use the Up, Down, Left, and Right arrow keys. To move forward from one cell to the next, use the **Tab** key. Pressing the **Tab** key in the last cell of a row creates a new row.

CALS/HTML Tables

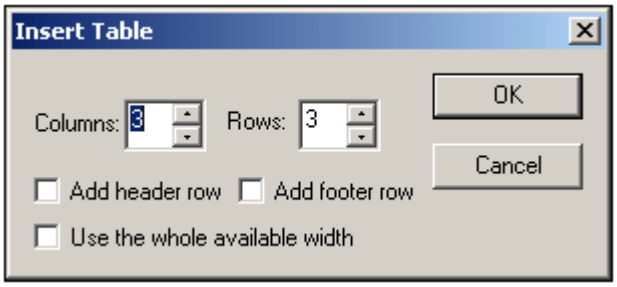
CALS/HTML tables can be inserted by you, the user of Authentic View, for certain XML data structures that have been specified to show a table format. There are three steps involved when working with CALS/HTML tables: inserting the table; formatting it; and entering data. The commands for working with CALS/HTML tables are available as icons in the toolbar (see CALS/HTML table editing icons).

Inserting tables
To insert a CALS/HTML table do the following:

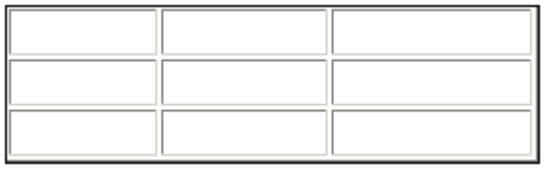
1. Place your cursor where you wish to insert the table, and click the icon. (Note that where you can insert tables is determined by the schema.) The Insert Table dialog (*screenshot below*) appears. This dialog lists all the XML element data-structures for which a table structure has been defined. For example, in the screenshot below, the `informaltable` element and `table` element have each been defined as both a CALS table as well as an HTML table.



2. Select the entry containing the element and table model you wish to insert, and click **OK** .
3. In the next dialog (*screenshot below*), select the number of columns and rows, and specify whether a header and/or footer is to be added to the table and whether the table is to extend over the entire available width. Click **OK** when done.



For the specifications given in the dialog box shown above, the following table is created.



 © 2011 Altova GmbH

By using the **Table** menu commands, you can add and delete columns, and create row and column joins and splits. But to start with, you must create the broad structure.

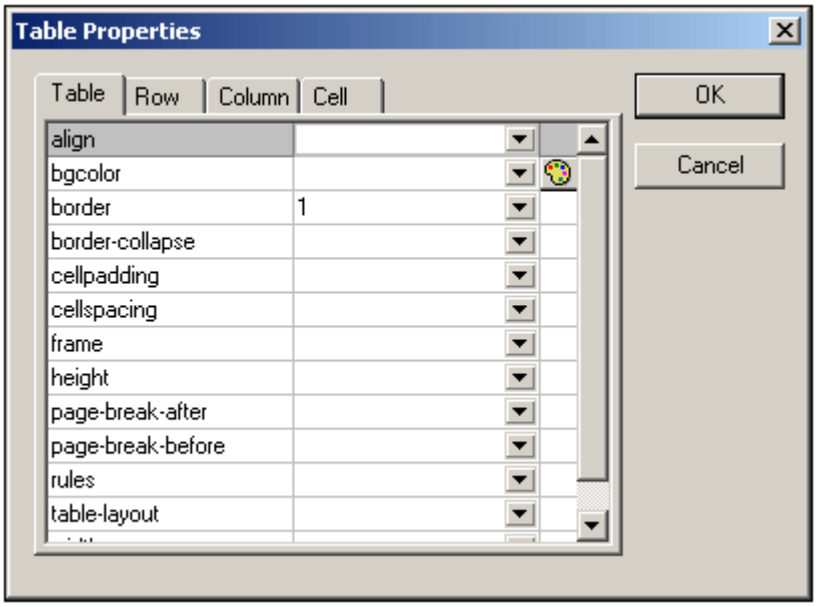
Formatting tables and entering data

The table formatting will already have been assigned in the document design. However, you might, under certain circumstances, be able to modify the table formatting. These circumstances are as follows:

- The elements corresponding to the various table structure elements must have the relevant CALS or HTML table properties defined as attributes (in the underlying XML Schema). Only those attributes that are defined will be available for formatting. If, in the design, values have been set for these attributes, then you can override these values in Authentic View.
- In the design. no `style` attribute containing CSS styles must have been set. If a style attribute containing CSS styles has been specified for an element, the `style` attribute has precedence over any other formatting attribute set on that element. As a result, any formatting specified in Authentic View will be overridden.

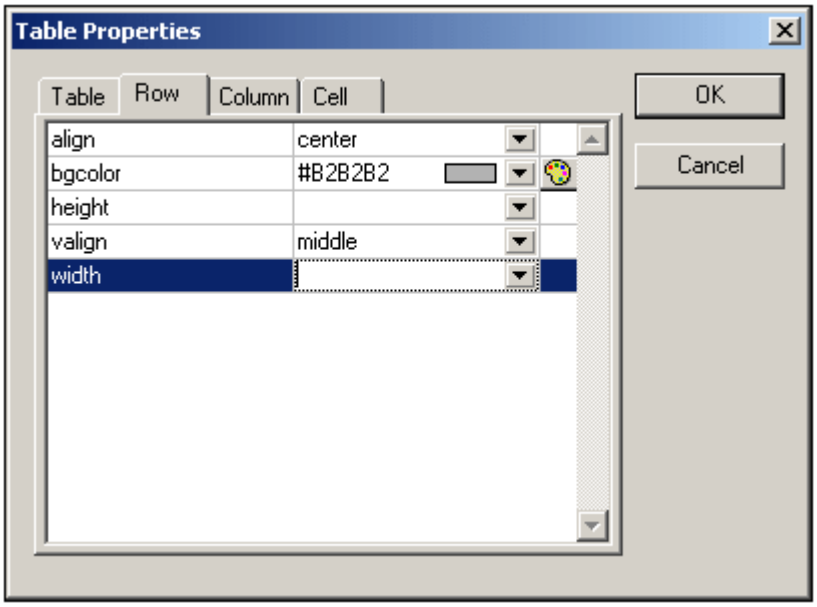
To format a table, row, column, or cell, do the following:

1. Place the cursor anywhere in the table and click the (Table Properties) icon. This opens the Table Properties dialog (*see screenshot*), where you specify formatting for the table, or for a row, column, or cell.



2. Set the cellspacing and cellpadding properties to "0". Your table will now look like this:

3. Place the cursor in the first row to format it, and click the (Table Properties) icon. Click the **Row** tab.



Since the first row will be the header row, set a background color to differentiate this row from the other rows. Note the Row properties that have been set in the figure above. Then enter the column header text. Your table will now look like this:

Name	Telephone	Email

Notice that the alignment is centered as specified.

4. Now, say you want to divide the "Telephone" column into the sub-columns "Office" and "Home", in which case you would need to split the horizontal width of the Telephone column into two columns. First, however, we will split the vertical extent of the header cell to make a sub-header row. Place the cursor in the "Telephone" cell, and click the (Split vertically) icon. Your table will look like this:

Name	Telephone	Email

5. Now place the cursor in the cell below the cell containing "Telephone", and click the (Split horizontally) icon. Then type in the column headers "Office" and "Home". Your table will now look like this:

Name	Telephone		Email
	Office	Home	

Now you will have to split the horizontal width of each cell in the "Telephone" column.

 © 2011 Altova GmbH

You can also add and delete columns and rows, and vertically align cell content, using the table-editing icons. The CALS/HTML table editing icons are described in the section titled, CALS/HTML Table Editing Icons.

Moving among cells in the table
To move among cells in the CALS/HTML table, use the Up, Down, Right, and Left arrow keys.

Entering data in a cell
To enter data in a cell, place the cursor in the cell, and type in the data.

Formatting text
Text in a CALS/HTML table, as with other text in the XML document, must be formatted using XML elements or attributes. To add an element, highlight the text and double-click the required element in the Elements Entry Helper. To specify an attribute value, place the cursor within the text fragment and enter the required attribute value in the Attributes Entry Helper. After formatting the header text bold, your table will look like this.

Name	Telephone		Email
	Office	Home	

The text above was formatted by highlighting the text, and double-clicking the element `strong`, for which a global template exists that specifies bold as the font-weight. The text formatting becomes immediately visible.

Please note: For text formatting to be displayed in Authentic View, a global template with the required text formatting must have been created in StyleVision for the element in question.

CALS/HTML Table Editing Icons

The commands required to edit CALS/HTML tables are available as icons in the toolbar, and are listed below. Note that no corresponding menu commands exist for these icons.

For a full description of when and how CALS/HTML Tables are to be used, see CALS/HTML Tables.

Insert table

The "Insert Table" command inserts a **CALS/HTML table** at the current cursor position.

Delete table

The "Delete table" command deletes the currently active table.

Append row

The "Append row" command appends a row to the end of the currently active table.

Append column

The "Append column" command appends a column to the end of the currently active table.

Insert row

The "Insert row" command inserts a row above the current cursor position in the currently active table.

Insert column

The "Insert column" command inserts a column to the left of the current cursor position in the currently active table.

Join cell left

The "Join cell left" command joins the current cell (current cursor position) with the cell to the left. The tags of both cells remain in the new cell, the column headers remain unchanged and are concatenated.

Join cell right

The "Join cell right" command joins the current cell (current cursor position) with the cell to the right. The contents of both cells are concatenated in the new cell.

Join cell below

The "Join cell below" command joins the current cell (current cursor position) with the cell below. The contents of both cells are concatenated in the new cell.

Join cell above

The "Join cell above" command joins the current cell (current cursor position) with the cell above. The contents of both cells are concatenated in the new cell.

Split cell horizontally

The "Split cell Horizontally" command creates a new cell to the right of the currently active cell. The size of both cells, is now the same as the original cell.

Split cell vertically

The "Split cell Vertically" command creates a new cell below the currently active cell.

Align top

This command aligns the cell contents to the top of the cell.

Center vertically

This command centers the cell contents.

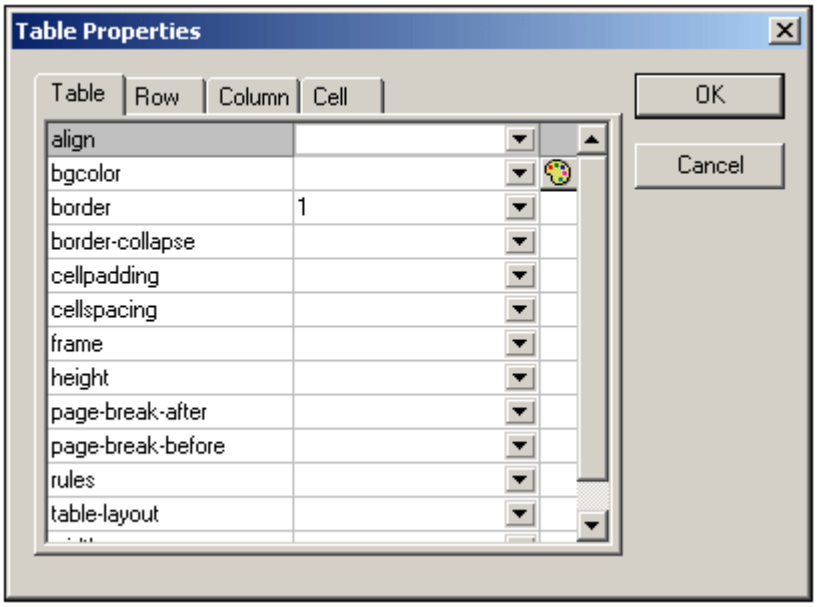
 © 2011 Altova GmbH

Align bottom

This command aligns the cell contents to the bottom of the cell.

Table properties

The "Table properties" command opens the Table Properties dialog box. This icon is only made active for HTML tables, it cannot be clicked for CALS tables.



7.2.3 Editing a DB

In Authentic View, you can edit database (DB) tables and save data back to a DB. This section contains a full description of interface features available to you when editing a DB table. The following general points need to be noted:

- The number of records in a DB table that are displayed in Authentic View may have been deliberately restricted by the designer of the StyleVision Power Stylesheet in order to make the design more compact. In such cases, only that limited number of records is initially loaded into Authentic View. Using the DB table row navigation icons (*see* Navigating a DB Table), you can load and display the other records in the DB table.
- You can query the DB to display certain records.
- You can add, modify, and delete DB records, and save your changes back to the DB. See Modifying a DB Table.

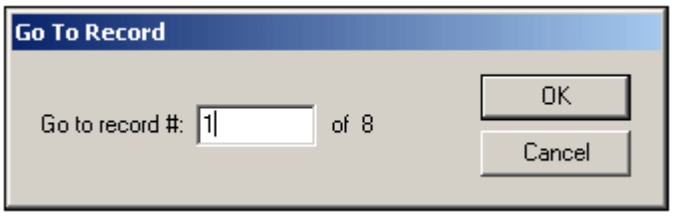
To open a DB-based StyleVision Power Stylesheet in Authentic View:

- Click **Authentic** | **Edit Database Data**, and browse for the required StyleVision Power Stylesheet.

Navigating a DB Table

The commands to navigate DB table rows are available as buttons in the Authentic View document. Typically, one navigation panel with either four or five buttons accompanies each DB table.

The arrow icons are, from left to right, Go to First Record in the DB Table; Go to Previous Record; Open the Go to Record dialog (*see screenshot*); Go to Next Record; and Go to Last Record.



To navigate a DB table, click the required button.

XML Databases
In the case of XML DBs, such as IBM DB2, one cell (or row) contains a single XML document, and therefore a single row is loaded into Authentic View at a time. To load an XML document that is in another row, use the menu command.

DB Queries

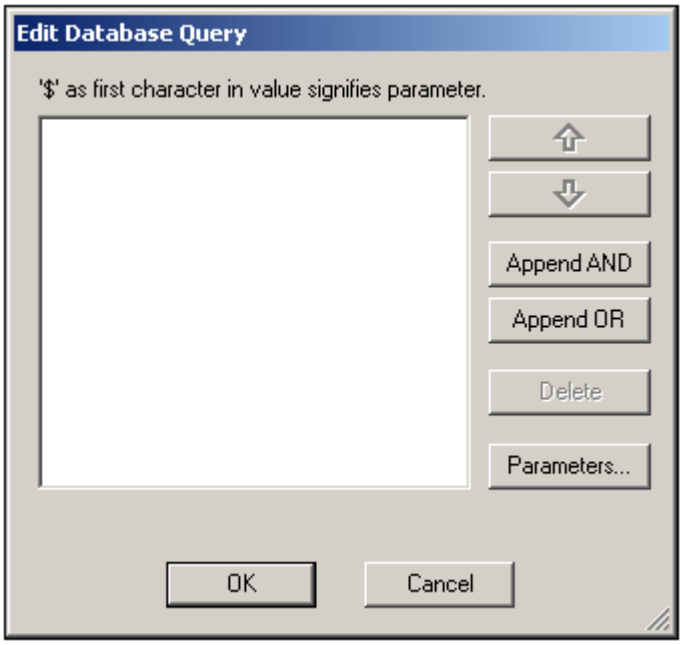
A DB query enables you to query the records of a table displayed in Authentic View. A query is made for an individual table, and only one query can be made for each table. You can make a query at any time while editing. If you have unsaved changes in your Authentic View document at the time you submit the query, you will be prompted about whether you wish to save **all** changes made in the document or discard **all** changes. Note that even changes made in other tables will be saved/discarded. After you submit the query, the table is reloaded using the query conditions.

Please note: If you get a message saying that too many tables are open, then you can reduce the number of tables that are open by using a query to filter out some tables.

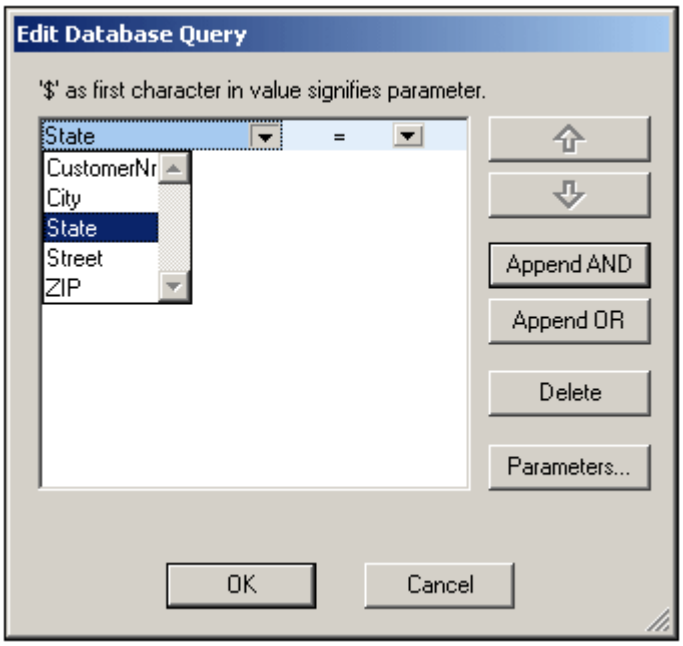
To create and submit a query:

1. Click the Query button for the required table in order to open the Edit Database Query dialog (*see screenshot*). This button typically appears at the top of each DB table or below it. If a Query button is not present for any table, the designer of the StyleVision Power Stylesheet has not enabled the DB Query feature for that table.

 © 2011 Altova GmbH



2. Click the **Append AND** or **Append OR** button. This appends an empty criterion for the query (shown below).



4. Enter the expression for the criterion. An expression consists of: (i) a field name (available from the associated combo-box); (ii) an operator (available from the associated combo-box); and (iii) a value (to be entered directly). For details of how to construct expressions see the Expressions in criteria section.
5. If you wish to add another criterion, click the **Append AND** or **Append OR** button according to which logical operator (AND or OR) you wish to use to join the two criteria. Then add the new criterion. For details about the logical operators, see the section Re-ordering criteria in DB Queries.

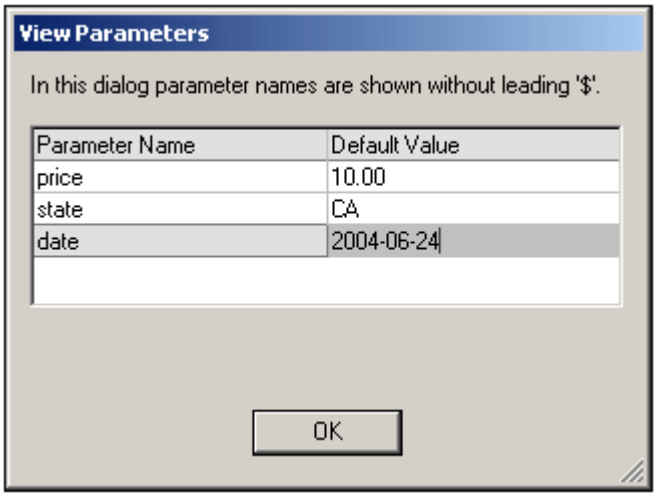
Expressions in criteria
Expressions in DB Query criteria consist of a field name, an operator, and a value. The **available field names** are the child elements of the selected top-level data table; the names of these fields are listed in a combo-box (*see screenshot above*). The **operators** you can use are listed below:

=	Equal to
<>	Not equal to
<	Less than
<=	Less than or equal to
>	Greater than
>=	Greater than or equal to
LIKE	Phonetically alike
NOT LIKE	Phonetically not alike
IS NULL	Is empty
NOT NULL	Is not empty

If IS NULL or NOT NULL is selected, the Value field is disabled. **Values** must be entered without quotes (or any other delimiter). Values must also have the same formatting as that of the corresponding DB field; otherwise the expression will evaluate to `FALSE`. For example, if a criterion for a field of the `date` datatype in an MS Access DB has an expression `StartDate=25/05/2004`, the expression will evaluate to `FALSE` because the `date` datatype in an MS Access DB has a format of `YYYY-MM-DD`.

Using parameters with DB Queries
You can enter the name of a **parameter** as the value of an expression when creating queries. Parameters are variables that can be used instead of literal values in queries. When you enter it in an expression, its value is used in the expression. Parameters that are available have been defined by the SPS designer in the SPS and can be viewed in the View Parameters dialog (*see screenshot below*). Parameters have been assigned a default value in the SPS, which can be overridden by passing a value to the parameter via the command line (if and when the output document is compiled via the command line).

To view the parameters defined for the SPS, click the **Parameters** button in the Edit Database Query dialog. This opens the **View Parameters** dialog (*see screenshot*).

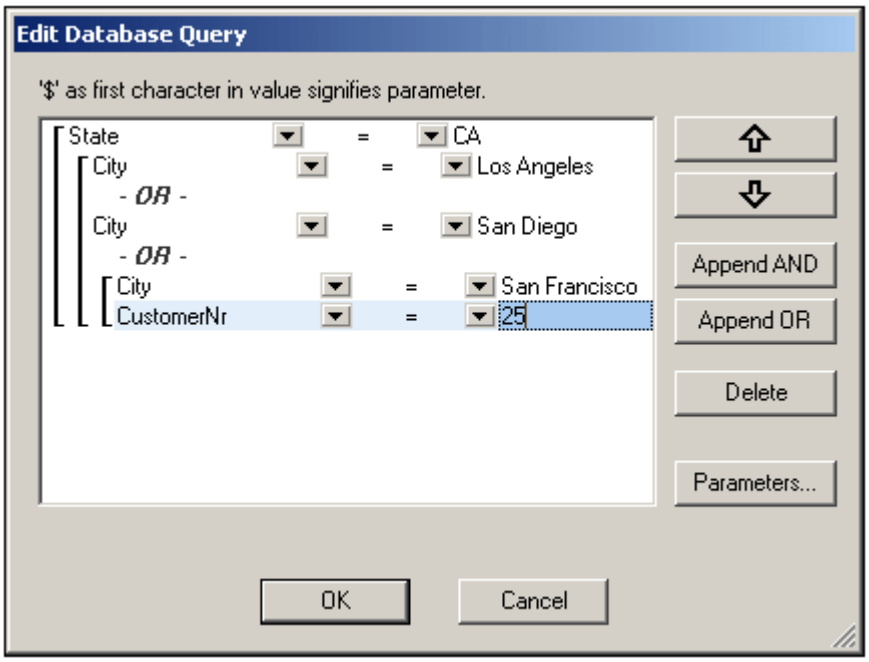


The View Parameters dialog contains **all** the parameters that have been defined for the stylesheet in the SPS and parameters must be edited in th stylesheet design.

 © 2011 Altova GmbH

Re-ordering criteria in DB Queries

The logical structure of the DB Query and the relationship between any two criteria or sets of criteria is indicated graphically. Each level of the logical structure is indicated by a square bracket. Two adjacent criteria or sets of criteria indicate the AND operator, whereas if two criteria are separated by the word `OR` then the OR operator is indicated. The criteria are also appropriately indented to provide a clear overview of the logical structure of the DB Query.



The DB Query shown in the screenshot above may be represented in text as:

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

You can re-order the DB Query by moving a criterion or set of criteria up or down relative to the other criteria in the DB Query. To move a criterion or set of criteria, do the following:

1. Select the criterion by clicking on it, or select an entire level by clicking on the bracket that represents that level.
2. Click the Up or Down arrow button in the dialog.

The following points should be noted:

- If the adjacent criterion in the direction of movement is at the same level, the two criteria exchange places.
- A set of criteria (i.e. criterion within a bracket) changes position within the same level; it does not change levels.
- An individual criterion changes position within the same level. If the adjacent criterion is further outward/inward (i.e. not on the same level), then the selected criterion will move outward/inward, **one level at a time**.

To delete a criterion in a DB Query, select the criterion and click **Delete**.

Modifying a DB Query

To modify a DB Query:

1. Click the Query button . The Edit Database Query dialog box opens. You can now edit the expressions in any of the listed criteria, add new criteria, re-order criteria, or delete criteria in the DB Query.
2. Click **OK**. The data from the DB is automatically re-loaded into Authentic View so as to reflect the modifications to the DB Query.

Modifying a DB Table

Adding a record

To add a record to a DB table:

1. Place the cursor in the DB table row and click the icon (to append a row) or the icon (to insert a row). This creates a new record in the temporary XML file.
2. Click the **File | Save** command to add the new record in the DB. In Authentic View a row for the new record is appended to the DB table display. The `AltovaRowStatus` for this record is set to `A` (for Added).

When you enter data for the new record it is entered in bold and is underlined. This enables you to differentiate added records from existing records—if existing records have not been formatted with these text formatting properties. Datatype errors are flagged by being displayed in red.

The new record is added to the DB when you click **File | Save**. After a new record is saved to the DB, its `AltovaRowStatus` field is initialized (indicated with `---`) and the record is displayed in Authentic View as a regular record.

Modifying a record

To modify a record, place the cursor at the required point in the DB table and edit the record as required. If the number of displayed records is limited, you may need to navigate to the required record (see Navigating a DB Table).

When you modify a record, entries in all fields of the record are underlined and the `AltovaRowStatus` of all primary instances of this record is set to `U` (for Updated). All secondary instances of this record have their `AltovaRowStatus` set to `u` (lowercase). Primary and secondary instances of a record are defined by the structure of the DB—and correspondingly of the XML Schema generated from it. For example, if an Address table is included in a Customer table, then the Address table can occur in the Design Document in two types of instantiations: as the Address table itself and within instantiations of the Customer table. Whichever of these two types is modified is the type that has been primarily modified. Other types—there may be more than one other type—are secondary types. Datatype errors are flagged by being displayed in red.

The modifications are saved to the DB by clicking **File | Save**. After a modified record is saved to the DB, its `AltovaRowStatus` field is initialized (indicated with `---`) and the record is displayed in Authentic View as a regular record.

Please note:

- If even a single field of a record is modified in Authentic View, the entire record is updated when the data is saved to the DB.
- The date value `0001-01-01` is defined as a `NULL` value for some DBs, and could result in an error message.

 © 2011 Altova GmbH

Deleting a record

To delete a record:

1. Place the cursor in the row representing the record to be deleted and click the icon. The record to be deleted is marked with a strikethrough. The `AltovaRowStatus` is set as follows: primary instances of the record are set to `D`; secondary instances to `d`; and records indirectly deleted to `X`. Indirectly deleted records are fields in the deleted record that are held in a separate table. For example, an Address table might be included in a Customer table. If a Customer record were to be deleted, then its corresponding Address record would be indirectly deleted. If an Address record in the Customer table were deleted, then the Address record in the Customer table would be primarily deleted, but the same record would be secondarily deleted in an independent Address table if this were instantiated.
2. Click **File | Save** to save the modifications to the DB.

Please note: Saving data to the DB resets the Undo command, so you cannot undo actions that were carried out prior to the save.

7.2.4 Working with Dates

There are two ways in which dates can be edited in Authentic View:

- Dates are entered or modified using the Date Picker.
- Dates are entered or modified by typing in the value.

The method the Authentic View user will use is defined in the SPS. Both methods are described in the two sub-sections of this section.

Note on date formats

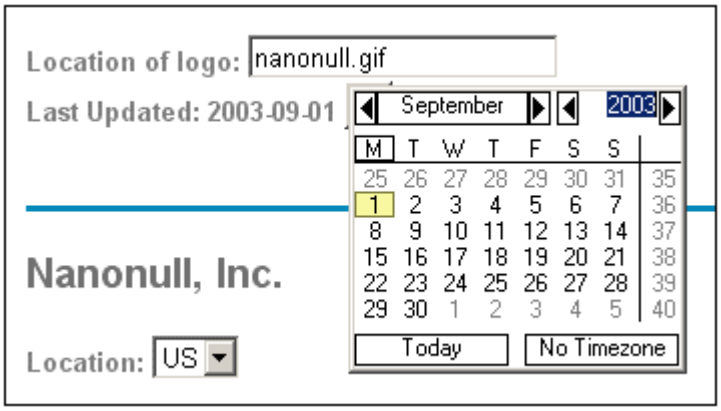
In the XML document, dates can be stored in one of several date datatypes. Each of these datatypes requires that the date be stored in a particular lexical format in order for the XML document to be valid. For example, the `xs:date` datatype requires a lexical format of `YYYY-MM-DD`. If the date in an `xs:date` node is entered in anything other than this format, then the XML document will be invalid.

In order to ensure that the date is entered in the correct format, the SPS designer can include the graphical Date Picker in the design. This would ensure that the date selected in the Date Picker is entered in the correct lexical format. If there is no Date Picker, the Authentic View should take care to enter the date in the correct lexical format. Validating the XML document could provide useful tips about the required lexical format.

Date Picker

The Date Picker is a graphical calendar used to enter dates in a standard format into the XML document. Having a standard format is important for the processing of data in the document. The Date Picker icon appears near the date field it modifies (*see screenshot*).

To display the Date Picker (*see screenshot*), click the Date Picker icon.

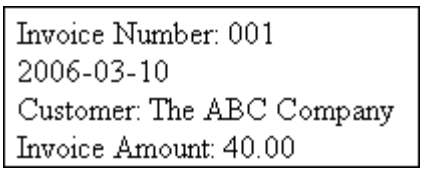


To select a date, click on the desired date, month, or year. The date is entered in the XML document, and the date in the display is modified accordingly. You can also enter a time zone if this is required.

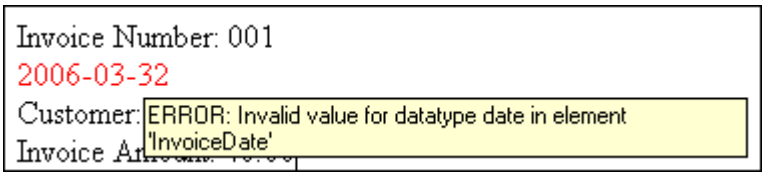
Text Entry

For date fields that do not have a Date Picker (*see screenshot*), you can edit the date directly by typing in the new value.

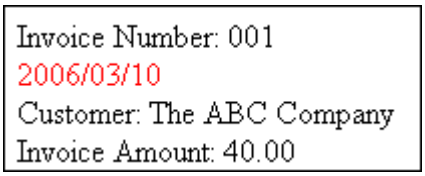
Please note: When editing a date, you must not change its format.



If you edit a date and change it such that it is out of the valid range for dates, the date turns red to alert you to the error. If you place the mouse cursor over the invalid date, an error message appears (*see screenshot*).



If you try to change the format of the date, the date turns red to alert you to the error (*see screenshot*).



7.2.5 Defining Entities

You can define entities for use in Authentic View, whether your document is based on a DTD or an XML Schema. Once defined, these entities are displayed in the Entities Entry Helper and in the **Insert Entity** submenu of the context menu. When you double-click on an entity in the Entities Entry Helper, that entity is inserted at the cursor insertion point.

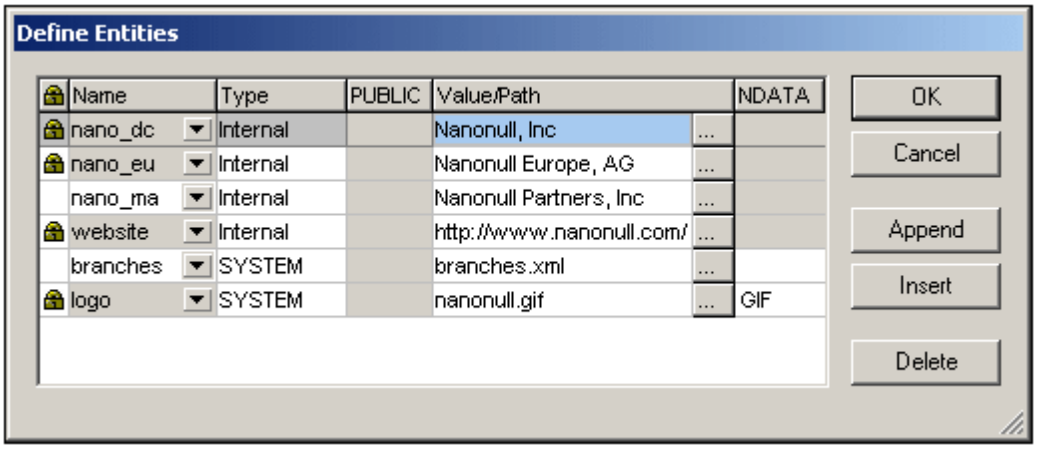
 © 2011 Altova GmbH

An entity is useful if you will be using a text string, XML fragment, or some other external resource in multiple locations in your document. You define the entity, which is basically a short name that stands in for the required data, in the Define Entities dialog. After defining an entity you can use it at multiple locations in your document. This helps you save time and greatly enhances maintenance.

There are two broad types of entities you can use in your document: a **parsed entity**, which is XML data (either a text string or a fragment of an XML document), or an **unparsed entity**, which is non-XML data such as a binary file (usually a graphic, sound, or multimedia object). Each entity has a name and a value. In the case of parsed entities the entity is a placeholder for the XML data. The value of the entity is either the XML data itself or a URI that points to a `.xml` file that contains the XML data. In the case of unparsed entities, the value of the entity is a URI that points to the non-XML data file.

To define an entity:

1. Click **Authentic | Define XML Entities...**. This opens the Define Entities dialog (*screenshot below*).



2. Enter the name of your entity in the Name field. This is the name that will appear in the Entities Entry Helper.
3. Enter the type of entity from the drop-down list in the Type field. The following types are possible: An **Internal** entity is one for which the text to be used is stored in the XML document itself. Selecting **PUBLIC** or **SYSTEM** specifies that the resource is located outside the XML file, and will be located with the use of a public identifier or a system identifier, respectively. A system identifier is a URI that gives the location of the resource. A public identifier is a location-independent identifier, which enables some processors to identify the resource. If you specify both a public and system identifier, the public identifier resolves to the system identifier, and the system identifier is used.
4. If you have selected PUBLIC as the Type, enter the public identifier of your resource in the PUBLIC field. If you have selected Internal or SYSTEM as your Type, the PUBLIC field is disabled.
5. In the Value/Path field, you can enter any one of the following:

   - If the entity type is Internal, enter the text string you want as the value of your entity. Do not enter quotes to delimit the entry. Any quotes that you enter will be treated as part of the text string.
   - If the entity type is SYSTEM, enter the URI of the resource or select a resource on your local network by using the Browse button. If the resource contains parsed data, it must be an XML file (i.e., it must have a `.xml` extension). Alternatively, the resource can be a binary file, such as a GIF file.
   - If the entity type is PUBLIC, you must additionally enter a system identifier in this field.

6. The NDATA entry tells the processor that this entity is not to be parsed but to be sent to the appropriate processor. The NDATA field must therefore contain some value to indicate that the entity is an unparsed entity.

Dialog features
You can do the following in the Define Entities dialog:

- Append entities
- Insert entities
- Delete entities
- Sort entities by the alphabetical value of any column by clicking the column header; clicking once sorts in ascending order, twice in descending order.
- Resize the dialog box and the width of columns.
- Locking. Once an entity is used in the XML document, it is locked and cannot be edited in the Define Entities dialog. Locked entities are indicated by a lock symbol in the first column. Locking an entity ensures that the XML document valid with respect to entities. (The document would be invalid if an entity is referenced but not defined.)
- Duplicate entities are flagged.

Limitations of entities

- An entity contained within another entity is not resolved, either in the dialog, Authentic View, or XSLT output, and the ampersand character of such an entity is displayed in its escaped form, i.e. `&` .
- External unparsed entities that are not image files are not resolved in Authentic View. If an image in the design is defined to read an external unparsed entity and has its URI set to be an entity name (for example: `'logo'`), then this entity name can be defined in the Define Entities dialog (*see screenshot above*) as an external unparsed entity with a value that resolves to the URI of the image file (as has been done for the `logo` entity in the screenshot above).

7.2.6 Images in Authentic View

Authentic View allows you to specify images that will be used in the final output document (HTML, RTF, PDF and Word 2007). You should note that some image formats might not be supported in some formats or by some applications. For example, the SVG format is supported in PDF, but not in RTF and would require a browser add-on for it to be viewed in HTML. So, when selecting an image format, be sure to select a format that is supported in the output formats of your document. Most image formats are supported across all the output formats (*see list below*).

Authentic View is based on Internet Explorer, and is able to display most of the image formats that your version of Internet Explorer can display. The following commonly used image formats are supported:

- GIF
- JPG
- PNG
- BMP
- WMF (Microsoft Windows Metafile)
- EMF (Enhanced Metafile)
- SVG (for PDF output only)

Relative paths
Relative paths are resolved relative to the SPS file.

 © 2011 Altova GmbH

7.2.7 Keystrokes in Authentic View

Enter key
In Authentic View the **Enter** key is used to append additional elements when it is in certain cursor locations. For example, if the chapter of a book may (according to the schema) contain several paragraphs, then pressing **Enter** inside the text of the paragraph causes a new paragraph to be appended immediately after the current paragraph. If a chapter can contain one title and several paragraphs, pressing **Enter** inside the chapter but outside any paragraph element (including within the title element) causes a new chapter to be appended after the current chapter (assuming that multiple chapters are allowed by the schema).

Please note: The **Enter** key does **not** insert a new line. This is the case even when the cursor is inside a text node, such as paragraph.

Using the keyboard
The keyboard can be used in the standard way, for typing and navigating. Note the following special points:

- The **Tab** key moves the cursor forward, stopping before and after nodes, and highlighting node contents; it steps over static content.
- The `add...` and `add Node` hyperlinks are considered node contents and are highlighted when tabbed. They can be activated by pressing either the spacebar or the **Enter** key.

7.3 Authentic Scripting

The **Authentic Scripting** feature provides more flexibility and interactivity to SPS designs. These designs can be created or edited in StyleVision Enterprise and Professional editions, and can be viewed in the Authentic View of the Enterprise and Professional editions of Altova products.

A complete listing of support for this feature in Altova products is given in the table below. Note, however, that in the trusted version of Authentic Browser plug-in, internal scripting is turned off because of security concerns.

Altova Product	Authentic Scripts Creation	Authentic Scripts Enabled
StyleVision Enterprise	Yes	Yes
StyleVision Professional	Yes	Yes
StyleVision Standard *	No	No
XMLSpy Enterprise	No	Yes
XMLSpy Professional	No	Yes
XMLSpy Standard	No	No
AuthenticDesktop Community	No	No
AuthenticDesktop Enterprise	No	Yes
Authentic Browser Plug-in Community **	No	No
Authentic Browser Plug-in Enterprise Trusted ***	No	Yes
Authentic Browser Plug-in Enterprise Untrusted	No	Yes

* *No AuthenticView*
** *Both Trusted and Untrusted versions*
*** *Scripted designs displayed. No internal macro execution or event handling. External events fired.*

Authentic Scripts behave in the same way in all Altova products, so no product-specific code or settings are required.

How Authentic Scripting works
The designer of the SPS design can use Authentic Scripting in two ways to make Authentic documents interactive:

- By assigning scripts for user-defined actions (macros) to design elements, toolbar buttons, and context menu items.
- By adding to the design event handlers that react to Authentic View events.

All the scripting that is required for making Authentic documents interactive is done within the StyleVision GUI (Enterprise and Professional editions). Forms, macros and event handlers are created within the Scripting Editor interface of StyleVision and these scripts are saved with the SPS. Then, in the Design View of StyleVision, the saved scripts are assigned to design elements, toolbar buttons, and context menus. When an XML document based on the SPS is opened in an Altova product that supports Authentic Scripting (*see table above*), the document will have the additional flexibility and interactivity that has been created for it.

 © 2011 Altova GmbH

Documentation for Authentic Scripting
The documentation for Authentic Scripting is available in the documentation of StyleVision. It can be viewed online via the Product Documentation page of the Altova website.

8 HTML, CSS, JSON

XMLSpy provides intelligent editing features for HTML and CSS documents. (JSON intelligent editing features are available in the Enterprise and Professional editions of XMLSpy; they are not available in the Standard Edition.) Both types of documents can be edited in Text View. Additionally, the active HTML document can be previewed in Browser View.

The intelligent editing features of each type of document is described separately in the sub-sections of this section: HTML, CSS.

Note: JSON support is available in the Enterprise and Professional editions only. It is not available in Standard Edition.

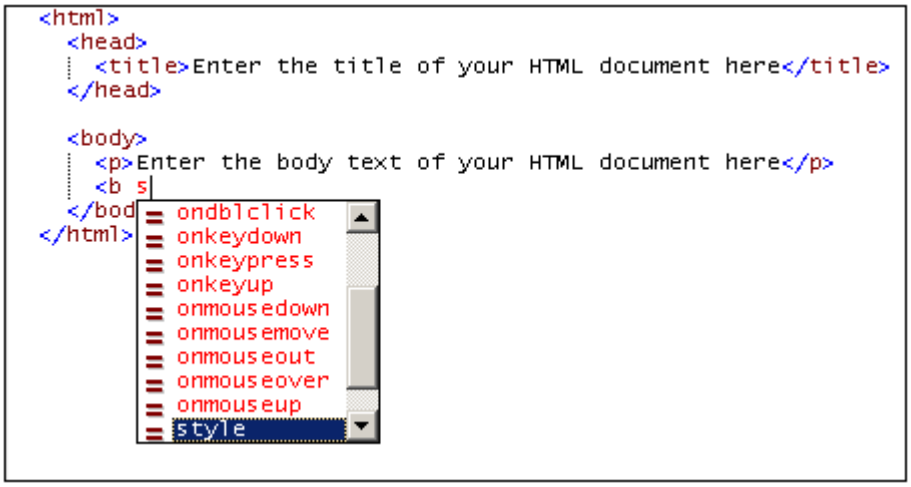
© 2011 Altova GmbH

8.1 HTML

HTML documents can be edited in Text View, and the edited page can then be viewed immediately in Browser View. Text View provides a number of useful HTML editing features. These are described in detail in Text View, but the main features, as well as HTML-specific options, are listed below.

Entry helpers
Elements, Attributes and Entities entry helpers are available when an HTML document is active. The entry helpers are context-sensitive; the items displayed in the entry helpers are those available at the current cursor location. Use the HTML entry helpers as described in Text View.

Auto-completion
As you type markup text into your HTML document, XMLSpy provides Auto-completion help. A pop-up containing a list of all nodes available at the cursor insertion point is displayed. As you type, the selection jumps to the first closest match in the list (*see screenshot below*). Click the selected item to insert it at the cursor insertion point.

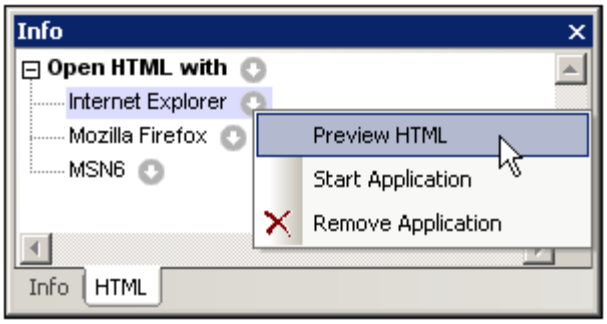


Auto-completion for elements appears when the left bracket of node tags is entered. When the start tag of an element node is entered in the document, the end tag is automatically inserted as well. This ensures well-formedness.

Auto-completion for attributes appears when a space is entered after the element name in a start tag. When you click an attribute name in the Auto-completion pop-up, the attribute is entered with quotes characters and the cursor positioned between the quotes.

The Entities entry helper contains character entities from the HTML 4.0 entity sets, Latin-1, special characters, and symbols.

HTML Info window
The HTML Info window (*screenshot below*) lists applications that can be used to quickly access the active HTML file. For example, if an HTML file is active in XMLSpy, double-clicking the Mozilla Firefox item in the HTML Info Window starts an instance of Mozilla Firefox and loads the active HTML document in it.



Note the following usage points:

- The icon to the right of the *Open HTML With* item enables applications to be added to the *Open HTML With* list. All the browsers installed on the system, or any other application (such as a text editor), can be added via the menu commands accessed via the *Open HTML With* icon. The associated applications would typically be browser or editor applications.
- After an application has been added to the *Open HTML With* list (except when added with the **Add Installed Browsers** command), its name in the *Open HTML With* list can be changed by selecting it, pressing **F2**, and editing the name.
- The icons to the right of each application listed in the *Open HTML With* list each opens a menu containing commands to: (i) open the application; (ii) open the application and load the linked HTML file; (iii) remove the application from the list. Double-clicking an application name opens the linked HTML file in that application.
- Applications added to or removed from the *Open HTML With* list are also added to or removed from the CSS Info window.

Assigning a DTD

For XHTML documents, a DTD or XML Schema can be assigned via the **DTD/Schema** menu, which enables you to browse for the required DTD or XML Schema file. An XHTML document can be edited exactly like an XML document.

Browser View commands

Browser View commands are available in the **Browser** menu.

 © 2011 Altova GmbH

8.2 CSS

CSS documents can be edited using Text View's intelligent editing features. These features, as they apply to the editing of CSS documents, are listed below.

Syntax coloring: A CSS rule consists of a selector, one or more properties, and the values of those properties. These three components may be further sub-divided into more specific categories; for example, a selector may be a class, pseudo-class, ID, element, or attribute. Additionally, a CSS document can contain other items than rules: for example, comments. In Text View, each such category of items can be displayed in a different color (*screenshot below*) according to settings you make in the Options dialog (*see below*).

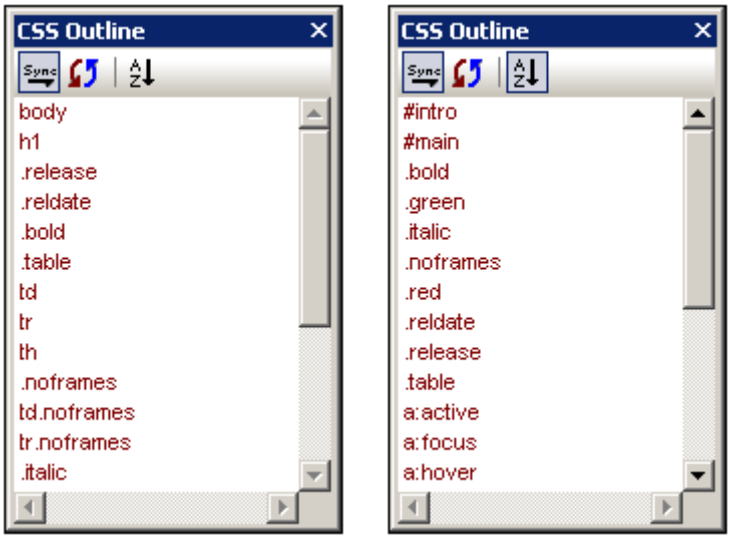
```
.header
{
    font-family: "Arial", sans-serif;
    font-weight: bold;
    color: red;
}
```

You can set the colors of the various CSS components in the Text Fonts tab of the Options dialog. In the combo box at top left, select CSS, and then select the required color (in the Styles pane) for each CSS item.

Folding margins: Each rule can be collapsed and expanded by clicking, respectively, the minus and plus icons to the left of the rule (*see screenshot above*). Note also that the pair of curly braces that delimit a rule (*screenshot above*) turns bold when the cursor is placed either before or after one of the curly braces. This indicates clearly where the definition of a particular rule starts and ends.

CSS outline: The CSS Outline entry helper (*screenshots below*) provides an outline of the document in terms of its selectors. Clicking a selector in the CSS Outline highlights it in the document. In the screenshot at left below, the selectors are unsorted and are listed in the order in which they appear in the document. In the screenshot at right, the Alphabetical Sorting feature has been toggled on (using the toolbar icon), and the selectors are sorted alphabetically.

You should note the following points: (i) For evaluating the alphabetical order of selectors, all parts of the selector are considered, including the period, hash, and colon characters; (ii) If the CSS document contains several selectors grouped together to define a single rule (e.g. `h4, h5, h6 {...}`), then each selector in the group is listed separately.



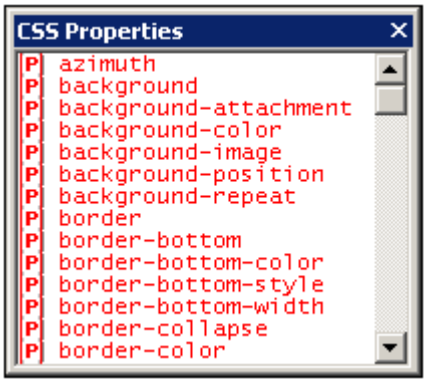
The icons in the toolbar of the CSS Outline entry helper, from left to right, do the following:

Toggles automatic synchronization (with the document) on and off. When auto-synchronization is switched on, selectors are entered in the entry helper even as you type them into the document.

Synchronizes the entry helper with the current state of the document.

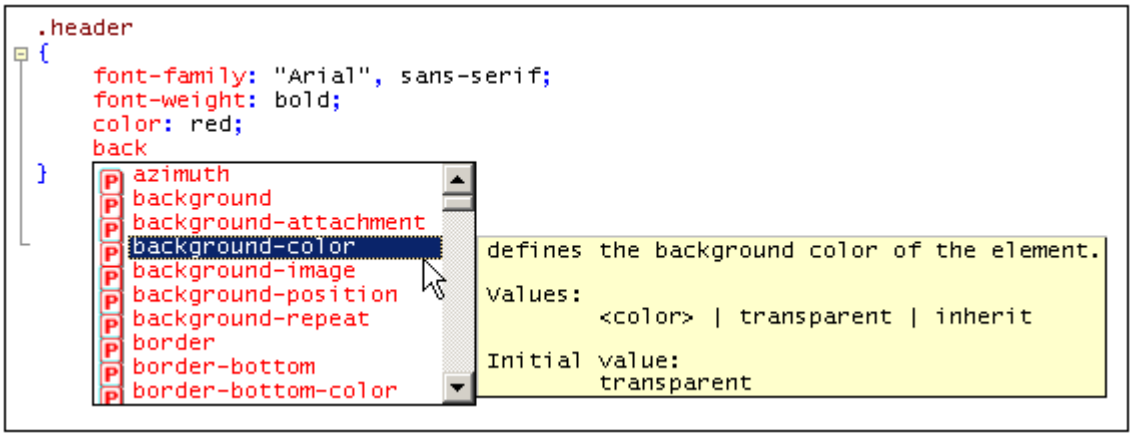
Toggles alphabetical sorting on and off. When off, the selectors are listed in the order in which they appear in the document. When sorted alphabetically, ID selectors appear first because they are prefaced by a hash (e.g. `#intro`).

Properties entry helper: The Properties entry helper (*screenshot below*) provides a list of all CSS properties, arranged alphabetically. A property can be inserted at the cursor insertion point by double-clicking the property.



Auto-completion of properties and tooltips for properties: As you start to type the name of a property, XMLSpy prompts you with a list of properties that begin with the letters you have typed (*screenshot below*). Alternatively, you can place the cursor anywhere inside a property name and then press **Ctrl+Space** to pop up the list of CSS properties.

 © 2011 Altova GmbH

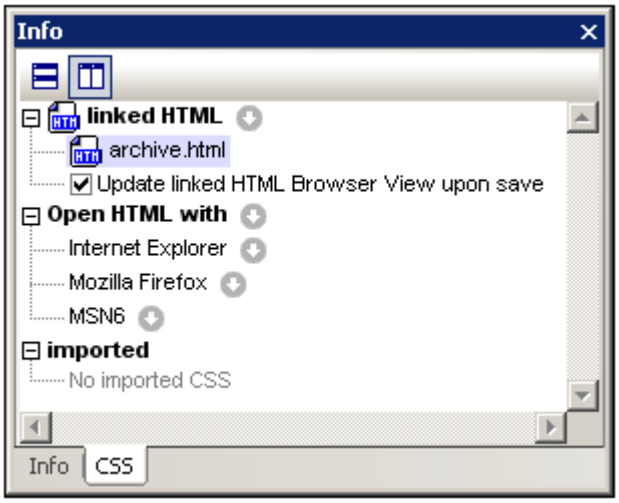


You can view a tooltip containing the definition of a property and its possible values by scrolling down the list or navigating the list with the Up and Down keys of your keyboard. The tooltip for the highlighted property is displayed. To insert a property, either press **Enter** when it is selected, or click it.

CSS Info window

When a CSS file is active, the CSS Info window (*screenshot below*) is enabled. The CSS Info window provides the following functionality:

- It enables the CSS file to be linked to an HTML file. This functionality enables you to modify the CSS document and view the effect of changes immediately. Additionally, the linked HTML file can be opened in multiple browsers via the CSS Info window, thus enabling changes in the CSS document to be viewed in multiple browsers.
- The CSS Info window lists the imported CSS stylesheets, thus giving you an overview of the import structure of the active CSS stylesheet.



Note the following usage points:

- Only one HTML file can be linked to the active CSS document. Do this by clicking the icon to the right of the *Linked HTML* item, then selecting the command **Set Link to HTML** and browsing for the required HTML file. The linked HTML file will be listed under the *Linked HTML* item in the Info window (*see screenshot below*). Creating this link does not modify the CSS document or the HTML document in any way. The link serves to set up an HTML file to which the active CSS document can be applied for

testing.

- Double-clicking the Linked HTML file listing opens the HTML file in XMLSpy.
- The toolbar icons enable you to horizontally and vertically tile the CSS document and the HTML file.
- When changes to the CSS document are saved, the HTML file that is open in XMLSpy can be automatically updated. To enable these automatic updates, check the *Update Linked HTML Browser* check box. Note that these updates will only occur if the HTML file contains a reference to the CSS document being edited.
- To change the linked HTML file, select another HTML file via the **Set Link to HTML** command.
- To remove the link to the HTML file, click the icon to the right of the *Linked HTML* item and select the command **Remove Link**.
- The icon to the right of the *Open HTML With* item enables applications to be added to the *Open HTML With* list. All the browsers installed on the system, or any other application (such as a text editor), can be added via the menu commands accessed via the *Open HTML With* icon. The associated applications would typically be browser or editor applications.
- After an application has been added to the *Open HTML With* list (except when added with the **Add Installed Browsers** command), its name in the *Open HTML With* list can be changed by selecting it, pressing **F2**, and editing the name.
- The icons to the right of each application listed in the *Open HTML With* list each opens a menu containing commands to: (i) open the application; (ii) open the application and load the linked HTML file; (iii) remove the application from the list. Double-clicking an application name opens the linked HTML file in that application.
- Applications added to or removed from the *Open HTML With* list are also added to or removed from the HTML Info window.
- The *Imported* item displays a list of the CSS files imported by the active CSS document.

 © 2011 Altova GmbH

9 Altova Global Resources

Altova Global Resources is a collection of aliases for file, and folder resources. Each alias can have multiple configurations, and each configuration maps to a single resource

Therefore, when a global resource is used as an input, the global resource can be switched among its configurations. This is done easily via controls in the GUI. For example, if an XSLT stylesheet for transforming an XML document is assigned via a global resource, then we can set up multiple configurations for the global resource, each of which points to a different XSLT file. After setting up the global resource in this way, switching the configuration would switch the XSLT file used for the transformation.

A global resource can not only be used to switch resources within an Altova application, but also to generate and use resources from other Altova applications. So, files can be generated on-the-fly in one Altova application for use in another Altova application. All of this tremendously eases and speeds up development and testing. For example, an XSLT stylesheet in XMLSpy can be used to transform an XML file generated on-the-fly by an Altova MapForce mapping.

Using Altova Global Resources involves two processes:

- Defining Global Resources: Resources are defined and the definitions are stored in an XML file. These resources can be shared across multiple Altova applications.
- Using Global Resources: Within an Altova application, files can be located via a global resource instead of via a file path. The advantage is that the resource being used can be instantly changed by changing the active configuration in XMLSpy.

Global resources in other Altova products
Currently, global resources can be defined and used in the following individual Altova products: XMLSpy, StyleVision, MapForce, and DatabaseSpy.

9.1 Defining Global Resources

Altova Global Resources are defined in the Manage Global Resources dialog, which can be accessed in two ways:

- Click Tools in the menu bar to pop up the **Tools** menu (*screenshot below*), and select the command **Global Resources**. This pops up the Global Resources dialog.

- Click the menu command **Tools | Customize** to display the Customize dialog. Then select the **Toolbars** tab and check the Global Resources option. This switches on the display of the Global Resources toolbar (*screenshot below*).

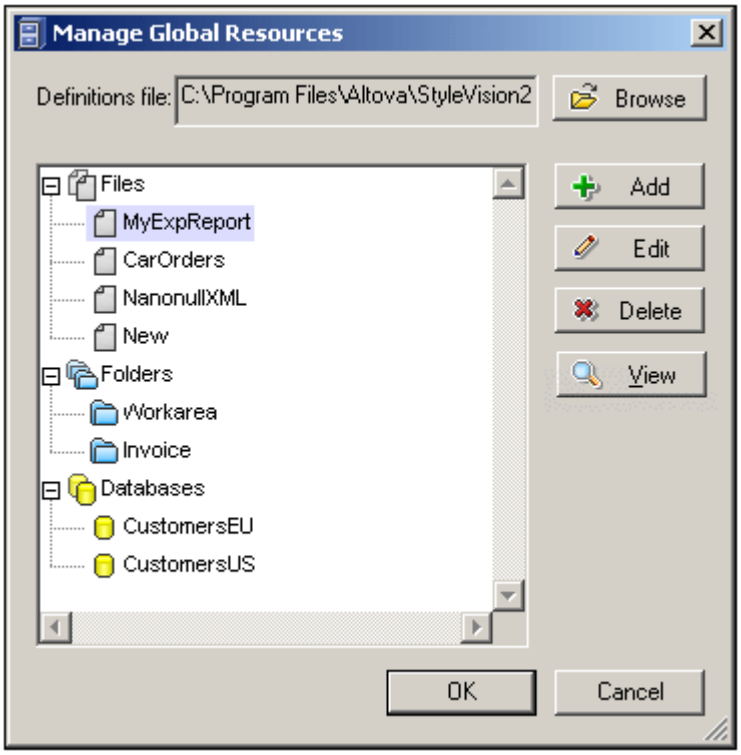
  Once the toolbar is displayed, click the Manage Global Resources icon. This pops up the Global Resources dialog.

The Global Resources XML File
Information about global resources that you define is stored in an XML file. By default, this XML file is called `GlobalResources.xml`, and it is stored in the `\Altova\` sub-folder of the (My) documents folder. This file is set as the default Global Resources XML File for all Altova applications. As a result, a global resource defined in any application will be available to all Altova applications—assuming that all applications use this file.

You can also re-name the file and save it to any location, if you wish. Consequently, you may have multiple Global Resources XML files. However, only one of these Global Resources XML File can be active at any time, and only the definitions contained in this file will be available to the application.

To select a Global Resources XML file to be the active file, in the Manage Global Resources dialog (*screenshot below*), browse for it in the Definitions File entry and select it.

 © 2011 Altova GmbH

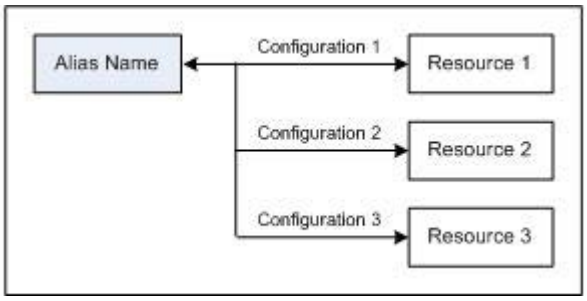


Managing global resources: adding, editing, deleting
In the Manage Global Resources dialog (*screenshot above*), you can add a global resource to the selected Global Resources XML File, or edit or delete a selected global resource. The Global Resources XML File organizes the aliases you add into a list of several sections: files, and folders (*see screenshot above*).

To add a global resource, click the **Add** button and define the global resource in the **Global Resource** dialog that pops up (*see description below*). After you define a global resource and save it, the global resource (or alias) is added to the library of global definitions in the selected Global Resources XML File. To edit a global resource, select it and click **Edit**. This pops up the **Global Resource** dialog, in which you can make the necessary changes (see the descriptions of files, folders in the sub-sections of this section). To delete a global resource, select it and click **Delete**.

After you finish adding, editing, or deleting, make sure to click **OK** in the **Manage Global Resources** dialog to save your modifications to the Global Resources XML File.

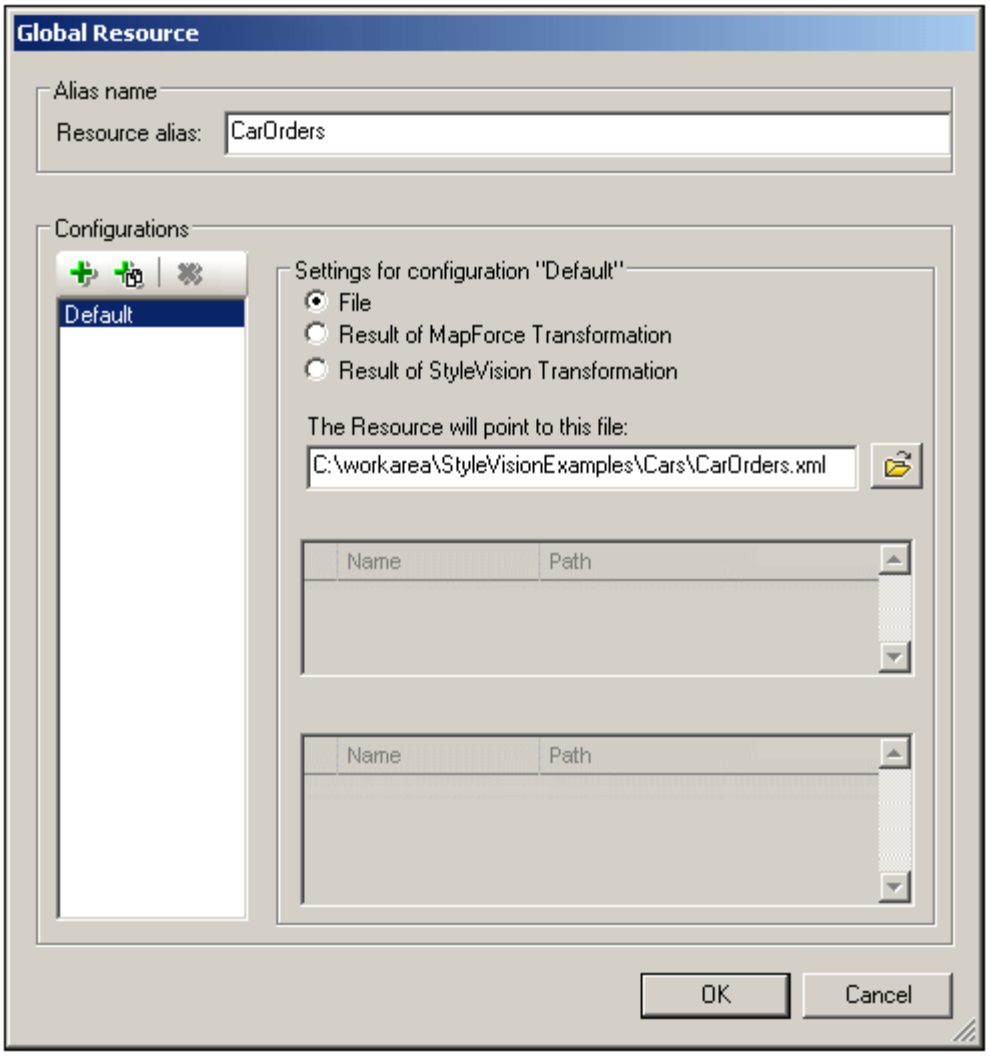
Adding a global resource
Creating a global resource involves mapping one alias name to one or more resources). Each mapping is called a configuration. A single alias name can therefore be associated with several resources via different configurations (*screenshot below*).



In the **Manage Global Resources** dialog (*screenshot above*), when you click the **Add** button, you can select whether you wish to add a file-type, or folder-type resource. How to add and edit each type of resource is described in the sub-sections of this section.

9.1.1 Files

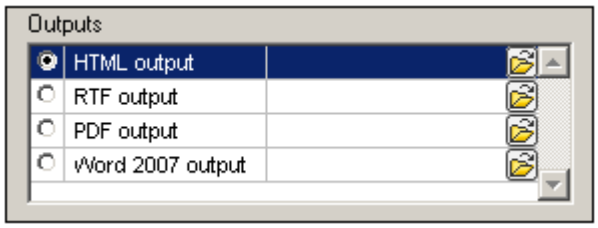
In the Global Resource dialog for Files (*screenshot below*), you can add a file resource as follows:



1. Enter an alias name.

 © 2011 Altova GmbH

2. The Configurations pane will have a configuration named Default (*screenshot above*). This Default configuration cannot be deleted nor have its name changed. You can enter as many additional configurations for the selected alias as you like. Add a configuration by clicking the **Add Configuration** icon and, in the **Add Configuration** dialog which pops up, enter the configuration name. Click **OK**. The new configuration will be listed in the Configurations pane. Repeat for as many configurations as required for this particular alias (global resource). You can also copy a configuration (using the Add Configuration as Copy icon) and then modify it.
3. Select one of the configurations in the Configurations pane and then define the resource to which this configuration will map. In the Settings for Configuration X pane, you can select whether the resource is a file, or the result of either an Altova MapForce or Altova StyleVision transformation. After selecting the resource type by clicking its radio button, browse for the file, MapForce file, or StyleVision file. Where multiple inputs or outputs for the transformation are possible, a selection of the options will be presented. For example, if the Result of StyleVision Transformation was selected as the resource type, the output options are displayed according to the what edition of StyleVision is installed (*the screenshot below shows the outputs for Enterprise Edition*).

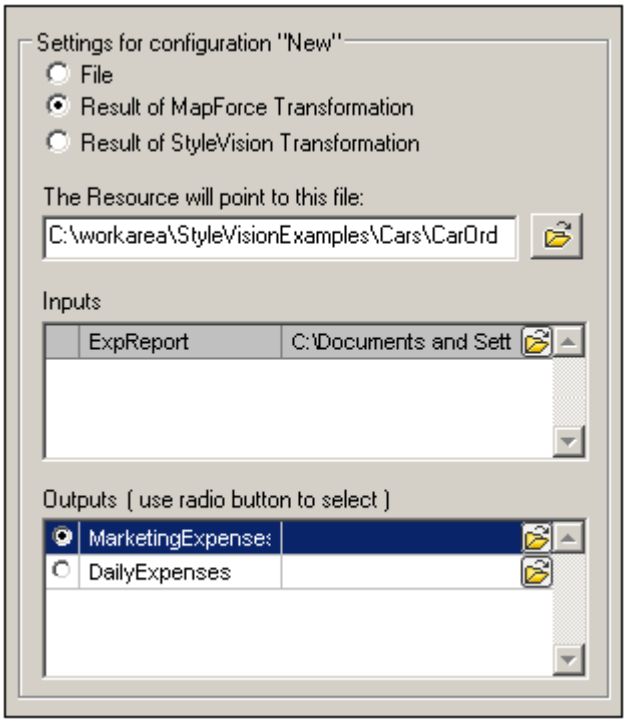


   Select the radio button of the desired option (in the screenshot above, 'HTML output' is selected). The result of a transformation can itself be saved as a global resource or as a file path (click the icon and select, respectively, Global Resource or Browse). If neither of these two saving options is selected, the transformation result will be loaded as a temporary file when the global resource is invoked.
4. Specify a resource for each configuration (that is, repeat Step 3 above for the various configurations you have created).
5. Click **OK** in the Global Resource dialog to save the alias and all its configurations as a global resource. The global resource will be listed under Files in the Manage Global Resources dialog.

Selecting Result of MapForce transformations as a global resource

Altova MapForce maps one or more (already existing) schemas to one or more (new) schemas designed by the MapForce user. XML files corresponding to the input schemas are used as data sources, and an output XML file based on the user-designed schema can be generated by MapForce. This generated output file (Result of MapForce Transformation) is the file that will be used as a global resource.

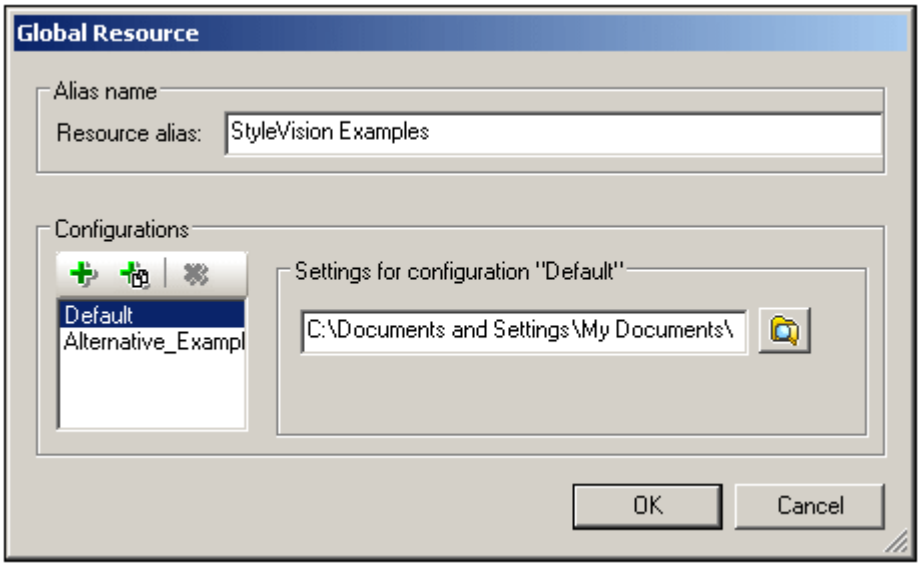
In a MapForce transformation that has multiple output schemas, you can select which one of the output schemas should be used for the global resource by clicking its radio button (*screenshot below*). The XML file that is generated for this schema can be saved as a global resource or as a file path (click the icon and select, respectively, Global Resource or Browse). If neither of these options is selected, a temporary XML file is created when the global resource is used.



Note that each Input can also be saved as a global resource or as a file path (click the icon and select, respectively, Global Resource or Browse).

9.1.2 Folders

In the Global Resource dialog for Folders (*screenshot below*), you can add a folder resource as follows:



Enter an alias name.

1. The Configurations pane will have a configuration named Default (*screenshot above*). This Default configuration cannot be deleted nor have its name changed. You can enter as many additional configurations for the selected alias as you like. Add a configuration by clicking the **Add Configuration** icon and, in the **Add Configuration** dialog

 © 2011 Altova GmbH

which pops up, enter the configuration name. Click **OK**. The new configuration will be listed in the Configurations pane. Repeat for as many configurations as required for this particular alias (global resource).

2. Select one of the configurations in the Configurations pane and browse for the folder you wish to create as a global resource.
3. Specify a folder resource for each configuration (that is, repeat Step 3 above for the various configurations you have created).
4. Click **OK** in the Global Resource dialog to save the alias and all its configurations as a global resource. The global resource will be listed under Folders in the Manage Global Resources dialog.

9.1.3 Copying Configurations

The Manage Global resources dialog allows you to duplicate existing configurations for all types of resources. To do so, select a configuration and click the **Copy Configuration** icon . Then select or enter a configuration name and click **OK**. This creates a copy of the selected configuration which you can now change as required.

9.2 Using Global Resources

There are several types of global resources (file-type, folder-type).Particular scenarios in XMLSpy allow the use of particular types of global resources. For example, you can use file-type or folder-type global resources for a Working XML File or a CSS file. The various scenarios in which you can use global resources in XMLSpy are listed in this section: Files and Folders.

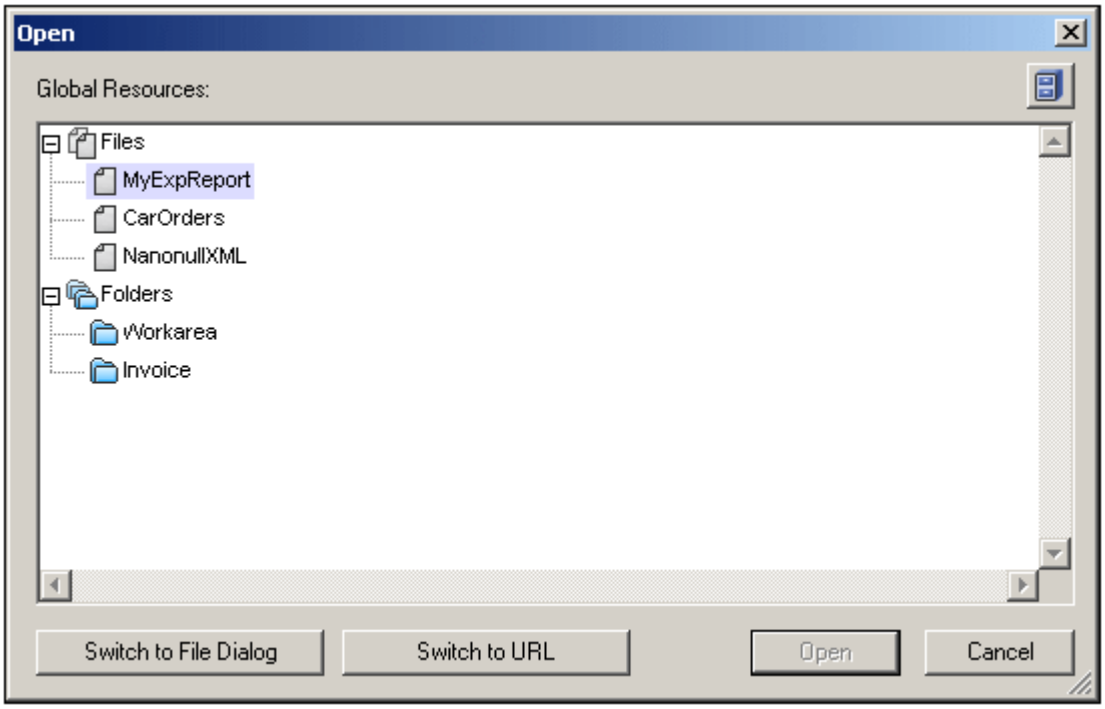
Selections that determine which resource is used
There are two application-wide selections that determine what global resources can be used and which global resources are actually used at any given time:

- *The active Global Resources XML File* is selected in the Global Resource dialog. The global-resource definitions that are present in the active Global Resources XML File are available to all files that are open in the application. Only the definitions in the active Global Resources XML File are available. The active Global Resources XML File can be changed at any time, and the global-resource definitions in the new active file will immediately replace those of the previously active file. The active Global Resources XML File therefore determines: (i) what global resources can be assigned, and (ii) what global resources are available for look-up (for example, if a global resource in one Global Resource XML File is assigned but there is no global resource of that name in the currently active Global Resources XML File, then the assigned global resource (alias) cannot be looked up).
- *The active configuration* is selected via the menu item **Tools | Active Configuration** or via the Global Resources toolbar. Clicking this command (or drop-down list in the toolbar) pops up a list of configurations across all aliases. Selecting a configuration makes that configuration active application-wide. This means that wherever a global resource (or alias) is used, the resource corresponding to the active configuration of each used alias will be loaded. The active configuration is applied to all used aliases. If an alias does not have a configuration with the name of the active configuration, then the default configuration of that alias will be used. The active configuration is not relevant when assigning resources; it is significant only when the resources are actually used.

9.2.1 Assigning Files and Folders

In this section, we describe how file-type and folder-type global resources are assigned. File-type and folder-type global resources are assigned differently. In any one of the usage scenarios below, clicking the **Switch to Global Resources** button pops up the Open Global Resource dialog (*screenshot below*).

 © 2011 Altova GmbH



Selecting a *file-type global resource* assigns the file. Selecting a *folder-type global resource* causes an Open dialog to open, in which you can brows for the required file. The path to the selected file is entered relative to the folder resource. So if a folder-type global resource were to have two configurations, each pointing to different folders, files having the same name but in different folders could be targeted via the two configurations. This could be useful for testing purposes.

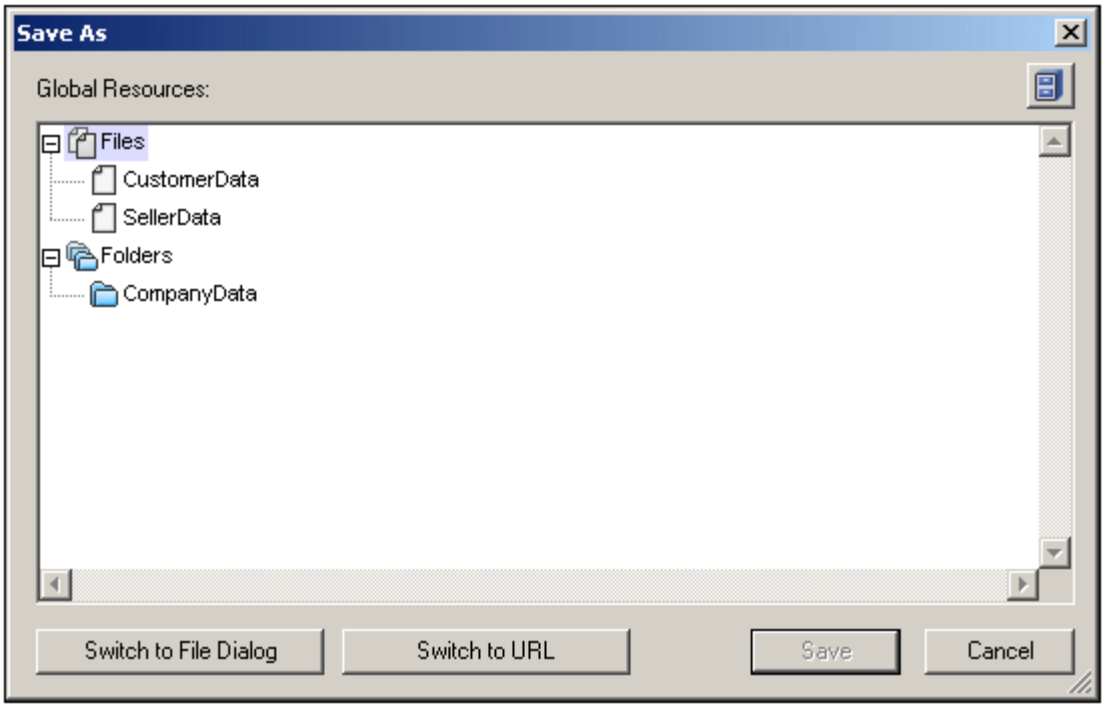
In the Open Global Resource dialog, you can switch to the file dialog or the URL dialog by clicking the respective button at the bottom of the dialog. The **Manage Global Resources** icon in the top right-hand corner pops up the **Manage Global Resources** dialog.

Usage scenarios
File-type and folder-type global resources can be used in the following scenarios:

- Opening global resources
- Saving as global resource
- Assigning files for XSLT transformations
- XSLT transformation and XQuery executions
- Assigning an SPS

Opening global resources
A global resource can be opened in XMLSpy with the **File | Open (Switch to Global Resource)** command and can be edited. In the case of a file-type global resource, the file is opened directly. In the case of a folder-type global resource, an Open dialog pops up with the associated folder selected. You can then browse for the required file in descendant folders. One advantage of addressing files for editing via global resources is that related files can be saved under different configurations of a single global resource and accessed merely by changing configurations. Any editing changes would have to be saved before changing the configuration.

Saving as global resource
A newly created file can be saved as a global resource. Also, an already existing file can be opened and then saved as a global resource. When you click the **File | Save** or **File | Save As** commands, the Save dialog appears. Click the **Switch to Global Resource** button to access the available global resources (*screenshot below*), which are the aliases defined in the current Global Resources XML File.



Select an alias and the click Save. If the alias is a file alias, the file will be saved directly. If the alias is a folder alias, a dialog will appear that prompts for the name of the file under which the file is to be saved. In either case the file will be saved to the location that was defined for the currently active configuration.

Note: Each configuration points to a specific file location, which is specified in the definition of that configuration. If the file you are saving as a global resource does not have the same filetype extension as the file at the current file location of the configuration, then there might be editing and validation errors when this global resource is opened in XMLSpy. This is because XMLSpy will open the file assuming the filetype specified in the definition of the configuration.

Assigning files for XSLT transformations
XSLT files can be assigned to XML documents and XML files to XSLT documents via global resources.. When the commands for assigning XSLT files (**XSL/XQuery | Assign XSL** and **XSL/XQuery | Assign XSL:FO**) and XML files (**XSL/XQuery | Assign Sample XML**) are clicked the assignment dialog pops up. Clicking the **Browse** button pops up the Open dialog, in which you can switch to the Open Global Resource dialog and select the required global resource. A major advantage of using a global resource to specify files for XSLT transformations is that the XSLT file (or XML file) can be changed for a transformation merely by changing the active configuration in XMLSpy; no new file assignments have to be made each time a transformation is required with a different file. When an XSLT transformation is started, it will use the file/s associated with the active configuration.

 © 2011 Altova GmbH

XSLT transformations and XQuery executions
Clicking the command **XSL/XQuery | XSL Transformation**, **XSL/XQuery | XSL:FO Transformation**, or **XSL/XQuery | XQuery Execution** pops up a dialog in which you can browse for the required XSLT, XQuery, or XML file. Click the **Browse** button and then the **Switch to Global Resource** button to pop up the Open Global Resource dialog (*screenshot at top of section*). The file that is associated with the currently active configuration of the selected global resource is used for the transformation.

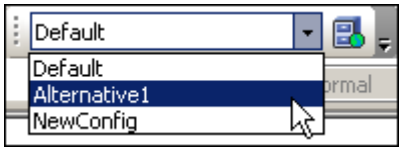
Assigning an SPS
When assigning a StyleVision stylesheet to an XML file (**Authentic | Assign StyleVision Stylesheet**), you can select a global resource to locate the stylesheet. Click the **Browse** button and then the **Switch to Global Resource** button to pop up the Open Global Resource dialog (*screenshot at top of section*). With a global resource selected as the assignment, the Authentic View of the XML document can be changed merely by changing the active configuration in XMLSpy.

9.2.2 Changing Configurations

One global resource configuration can be active at any time, and it is active application-wide. This means that the active configuration is active for all aliases in all currently open files. If an alias does not have a configuration with the name of the active configuration, then the default configuration of that alias will be used.

As an example of how to change configurations, consider the case in which an XSLT file has been assigned to an XML document via a global resource with multiple configurations. The XSLT file can be switched merely by changing the configuration of the global resource. This can be done in two ways:

- When you hover over the menu command **Tools | Active Configuration**, a submenu with a list of all configurations in the Global Resources XML File pops out. Select the required configuration.
- In the combo box of the Global Resources toolbar (*screenshot below*), select the required configuration. (The Global Resources toolbar can be toggled on and off with the menu command **Tools | Customize | Toolbars | Global Resources**.)

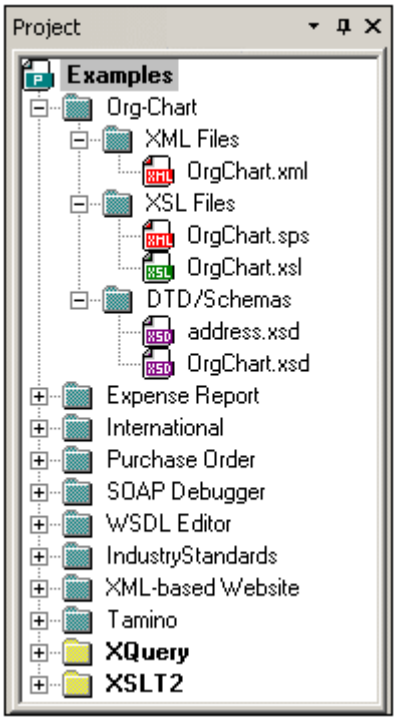


The XSLT file will be changed immediately.

In this way, by changing the active configuration, you can change source files that are assigned via a global resource.

10 Projects

A project is a collection of files that are related to each other in some way you determine. For example, in the screenshot below, a project named `Examples` collects the files for various examples in separate example folders, each of which can be organized further into sub-folders. Within the `Examples` project, for instance, the `OrgChart` example folder is organized further into sub-folders for XML, XSL, and Schema files.



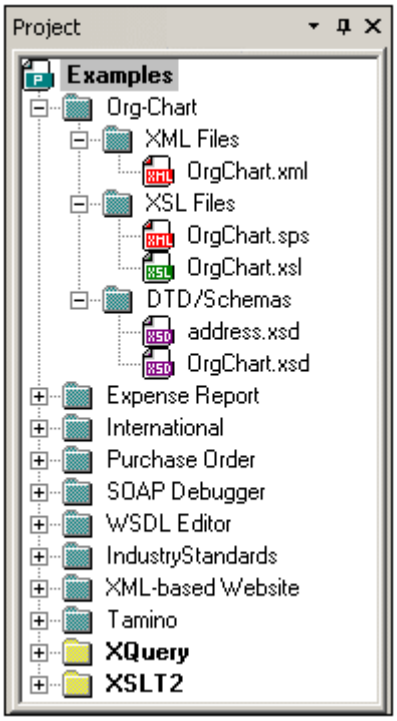
Projects thus enable you to gather together files that are used together and to access them quicker. Additionally, you can define schemas and XSLT files for individual folders, thus enabling the batch processing of files in a folder.

This section describes how to create and edit projects and how to use projects.

© 2011 Altova GmbH

10.1 Creating and Editing Projects

Projects are managed via the Project Window (*screenshot below*) and the Project menu. One project can be open at a time in the application. The open project is displayed in the Project Window.



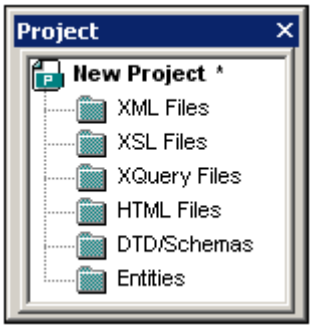
Creating new projects, opening existing projects
A new project is created with the menu command **Project | New Project**. An existing project is opened with the menu command **Project | Open Project**. The newly opened project (whether new or existing) replaces the previously opened project in the Project Window. If the previously opened project contains unsaved changes (indicated by an asterisk next to the folder name; *see screenshot below*), you are asked whether you wish to save these changes.

Naming and saving projects
A new project is named when you save it. A project is saved with the **Project | Save Project** command and has the . `spp` file extension. After a project has been modified, the project must be saved for the modifications to be stored. Note that a project (indicated by the top-level folder in the Project Window) can only be re-named by changing its name in Windows File Explorer; the name cannot be changed in the GUI. (The names of sub-folders, however, can be changed in the GUI.)

Project structure
A project has a tree structure of folders and files. Folders and files can be created at any level and to an unlimited depth. Do this by selecting a folder in the Project Window and then using the commands in the **Project** menu or context menu to add folders, files, or resources. Folders, files, and resources that have been added to a project can be deleted or dragged to other locations in the project tree.

When a new project is created, the default project structure organizes the project by file type

(XML, XSL, etc) (*see screenshot below*).



File-type extensions are associated with a folder via the property definitions for that folder. When a file is added to a folder, it is automatically added to the appropriate child folder according to the file-type extension. For each folder, you can define what file-type extensions are to be associated with it.

What can be added to a project

Folder, files, and other resources can be added either to the top-level project folder or to a folder at any level in the project. There are three types of folders: (i) project folders; (ii) external folders; (iii) external web folders.

To add an object, select the relevant folder and then the required command from the **Project** menu or context menu of the selected folder. The following objects are available for addition to a project folder

- *Project folders* (green) are folders that you add to the project in order to structure the project's contents. You can define what file extensions are to be associated with a project folder (in the properties of that folder). When files are added to a folder, they are automatically added to the first child folder that has that file's extension associated with it. Consequently, when multiple files are added to a folder, they will be distributed by file extension among the child folders that have the corresponding file-extension associations.
- *External folders* (yellow) are folders in a file system. When an external folder is added to a folder, the external folder and all its files, sub-folders, and sub-folder files are included in the project. Defining file extensions on an external folder serves to filter the files available in the project.
- *External web folders* are like external folders, except that they are located on a web server and require user authentication to access. Defining file extensions on an external web folder serves to filter the files available in the project.
- *Files* can be added to a folder by selecting the folder and then using one of the three Add-File commands: (i) **Add Files**, to select the file/s via an Open dialog; (ii) **Add Active File**, to add the file that is active in the Main Window; (iii) **Add Active and Related Files**, additionally adds files related to an active XML file, for example, an XML Schema or DTD. Note that files associated by means of a processing instruction (for example, XSLT files), are not considered to be related files.
- *Global Resources* are aliases for file, folder, and database resources. How they are defined and used is described in the section on Global Resources.
- *URLs* identify a resource object via a URL.

Project properties

The properties of a folder are stored in the Properties dialog of that folder. It is accessed by first selecting the folder and then the **Properties** command in the **Project** menu or context menu (obtained by right-clicking the folder). Note that properties can be defined not only for the top-

 © 2011 Altova GmbH

level project folder, but also for folders at various levels of the project hierarchy. The following properties of a folder can be defined and edited in the Properties dialog:

- *Folder name*: cannot be edited for the top-level project folder (for which, instead of a name, a filepath is displayed).
- *File extensions*: cannot be edited for the top-level project.
- *Validation*: specifies the DTD or XML Schema file that should be used to validate XML files in a folder.
- *Transformations*: specifies (i) the XSLT files to be used for transforming XML files in the folder, and (ii) the XML files to be transformed with XSLT files in the folder.
- *Destination files*: for the output of transformations, specifies the file extension and the folder where the files are to be saved.
- *SPS files for Authentic View*: specifies the SPS files to be used so that XML files in a folder can be viewed and edited in Authentic View.

Source control in projects
Source control systems that are compatible with Microsoft Visual Source-Safe are supported in projects. How to use this feature is described in the User Reference section of the manual.

Saving projects
Any changes you make to a project, such as adding or deleting a file, or modifying a project property, must be saved with the **Save Project** command.

Refreshing projects
If a change is made to an external folder, this change will not be reflected in the Project Window till the project is refreshed.

10.2 Using Projects

Projects are very useful for organizing your workspace, applying settings to multiple files, and for setting up and executing batch commands. Using projects can therefore greatly help speed up and ease your work.

Benefits of using projects
The following list lists the benefits of using projects.

- Files and folders can be grouped into folders by file extension or any other desired criterion.
- Schemas and XSLT files can be assigned to a folder. This can be useful if you wish to quickly validate or transform a single XML file using different schema or XSLT files. Add the XML file to different folders and define different schemas and XSLT files for the different folders.
- Batch processing can be applied to individual folders. The commands available for batch processing are listed below.
- Output folders can be specified for transformations.

Organizing resources for quick access
Folder and file resources can be organized into a tree structure, giving you a clear overview of the various folders and files in your project, and enabling you to quickly access any and all files in a project. Simply double-click a file in the Project window to open it. You can quickly add files and folders to a project as required and delete unwanted files and folders. When you wish to work with another project, close the project currently open in the Project Window and open the required project.

Batch processing
The commands for batch processing of files in a folder, whether the top-level project folder or a folder at any other level, are **available in the context menu of that folder** (obtained by right-clicking the folder). The steps for batch processing are as follows:

1. Define the files to be used for validation or transformation in the Properties dialog of that folder.
2. Specify the folder in which the output of transformations should be saved. If no output folder is specified for a folder, the output folder of the next ancestor folder in the project tree is used.
3. Use the commands in the context menu for batch execution. If you use the corresponding commands in the XML, DTD/Schema, or XSL/XQuery menus, the command will be executed only on the document active in the Main Window, not on any project folder in the Project Window.

The following commands in the context menu of a project folder (top-level or other) are available for batch processing:

- *Well-formed check*: If any error is detected during the batch execution, it is reported in the Messages Window.
- *Validation*: If any error is detected during the batch execution, it is reported in the Messages Window.
- *Transformations*: Transformation outputs are saved to the folder specified as the output folder in the Properties dialog of that folder. If no folder is specified, the output folder of the next ancestor project folder is used. If no ancestor project folder has an output folder defined, a document window is opened and the results of each transformation is displayed successively in this document window. An XSL-FO transformation transforms an XML document or FO document to PDF.

 © 2011 Altova GmbH

Note: To execute batch commands use the context menu of the relevant folder in the Project Window. Do not use the commands in the XML, DTD/Schema, or XSL/XQuery menus. These commands will be executed on the document active in the Main Window.

11 Source Control

Project files can be placed under source control. A variety of source control systems are supported and Altova has tested support with a large number of drivers and source control systems. The tested systems are listed in the section, Supported Source Control Systems.

Since XMLSpy implements the Microsoft Source Code Control Interface (MSSCCI) v1.1 – v1.3, multiple source control systems, including Microsoft SourceSafe and other compatible repositories, are supported.

Note: The **64-bit** version of XMLSpy automatically supports any of the supported 32-bit source control programs mentioned in this documentation. When using a 64-bit version of XMLSpy with a 32-bit source control program, the "Perform background status updates every... ms" option (Tools | Options) is automatically **grayed-out** and cannot be selected!

Overview of the Source Control feature
The mechanism for placing files in a XMLSpy project under source control is as follows:

1. In XMLSpy, a project folder containing the files to be placed under source control is created. Typically, the project folder will correspond to a local folder in which the project files are located. The path to the local folder is referred to as the local path.
2. In the source control system's database (also referred to as source control or repository), a folder is created that will contain the files to be placed under source control.
3. Project files are added to source control using the command **Project | Source Control | Add to Source Control**.
4. Source control actions, such as checking in to, checking out from , and removing files from source control, can be carried out by using the commands in the **Project | Source Control** submenu. The commands in this submenu are listed and described in the Project menu section of the User Reference.

Note: If you wish to change the current source control provider, this can be done in any of two ways: (i) via the Source Control options (**Tools | Options | Source Control**), or (ii) in the Change Source Control dialog (**Project | Source Control | Change Source Control**).

Note: Note that a Source Control project is not the same as an XMLSpy project. Source Control projects are directory dependent, while XMLSpy projects are logical constructions without direct directory dependence.

Registry entry and plug-ins
Microsoft has defined a Registry entry, where all Source Control-compatible programs can register themselves. This is the entry for XMLSpy:

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

Note that Source Control (SC) plug-ins are not automatically installed by all SC products. Please read the documentation supplied with your specific SC software for more information about plug-ins.

In this section
This section as organized as follows:

 © 2011 Altova GmbH

- Supported Source Control Systems lists the Source Control Systems (SCSs) that are supported by XMLSpy. Systems are listed alphabetically by server, and for each server the supported clients are also listed.
- Installing Source Control Systems contains notes about how to install the various SCSs.
- SCSs and Altova DiffDog Differencing describes how various Source Control Systems can be set up to carry out differencing with Altova DiffDog.

Menu commands
The menu commands for using Source Control Systems are in the submenu **Project | Source Control** and are described in the User Reference section.

Resource / Speed issues
Very large source control databases might be introducing a speed/resource penalty when automatically performing background status updates.

You might be able to speed up your system by disabling (or increasing the interval of) the "Perform background status updates every xxx seconds" field in the Source Control tab accessed through **Tools | Options**.

11.1 Supported Source Control Systems

The list below shows the Source Control Servers (SCSs) supported by XMLSpy, together with their respective Source Control Client/s (SCCs). The list is organized alphabetically by SCS. Please read the notes following the list for information about support levels.

AccuRev

Version: AccuRev 4.7.0 Windows

Clients:
- AccuBridge for Microsoft SCC 2008.2

Bazaar

Version: Bazaar 1.9 Windows

Clients:
- Aigenta Unified SCC 1.0.6

Borland StarTeam 2008

Version: StarTeam 2008 Release 2

Clients:
- Borland StarTeam Cross-Platform Client 2008 R2

Codice Software Plastic SCM

Version: Codice Software Plastic SCM Professional 2.7.127.10 (Server)

Clients:
- Codice Software Plastic SCM Professional 2.7.127.10 (SCC Plugin)

Collabnet Subversion 1.5

Version: 1.5.4

Clients:
- Aigenta Unified SCC 1.0.6
- PushOK SVN SCC 1.5.1.1
- PushOK SVN SCC x64 version 1.6.3.1
- TamTam SVN SCC 1.2.24

ComponentSoftware CS-RCS (PRO)

Version: ComponentSoftware CS-RCS (PRO) 5.1

Clients:
- ComponentSoftware CS-RCS (PRO) 5.1

Dynamsoft SourceAnywhere for VSS

 © 2011 Altova GmbH

Version: Dynamsoft SourceAnywhere for VSS 5.3.2 Standard/Professional Server

Clients:
- Dynamsoft SourceAnywhere for VSS 5.3.2 Client

Dynamsoft SourceAnywhere Hosted

Version: Server hosted in a Bell Data Center

Clients:
- Dynamsoft SourceAnywhere Hosted Client (22252)

Dynamsoft SourceAnywhere Standalone

Version: SourceAnywhere Standalone 2.2 Server

Clients:
- Dynamsoft SourceAnywhere Standalone 2.2 Client

IBM Rational ClearCase 7

Version: 7.0.1 (LT)

Clients:
- IBM Rational ClearCase 7.0.1 (LT)

March-Hare CVSNT 2.5

Version: 2.5.03.2382

Clients:
- Aigenta Unified SCC 1.0.6

March-Hare CVS Suite 2008

Version: Server 2008 [3321]

Clients:
- Jalindi Igloo 1.0.3
- March-Hare CVS Suite Client 2008 (3321)
- PushOK CVS SCC NT 2.1.2.5
- PushOK CVS SCC x64 version 2.2.0.4
- TamTam CVS SCC 1.2.40

Mercurial

Version: Mercurial 1.0.2 for Windows

Clients:
- Sergey Antonov HgSCC 1.0.1

Microsoft SourceSafe 2005

Version: 2005 with CTP

Clients:
- Microsoft SourceSafe 2005 with CTP

Microsoft Visual Studio Team System 2008 Team Foundation Server

Version: 2008

Clients:
- Microsoft Team Foundation Server 2008 MSSCCI Provider

Perforce 2008

Version: P4S 2008.1

Clients:
- Perforce P4V 2008.1

PureCM

Version: PureCM Server 2008/3a

Clients:
- PureCM Client 2008/3a

QSC Team Coherence Version Manager

Version: QSC Team Coherence Server 7.2.1.35

Clients:
- QSC Team Coherence Client 7.2.1.35

Qumasoft QVCS Enterprise

Version: QVCS Enterprise 2.1.18

Clients:
- Qumasoft QVCS Enterprise 2.1.18

Qumasoft QVCS Pro

Version: 3.10.18

Clients:
- Qumasoft QVCS Pro 3.10.18

Reliable Software Code Co-Op

Version: Code Co-Op 5.1a

Clients:
- Reliable Software Code Co-Op 5.1a

© 2011 Altova GmbH

Seapine Surround SCM

Version: Surround SCM Client/Server for Windows 2009.0.0

Clients:
- Seapine Surround SCM Client 2009.0.0

Serena Dimensions

Version: Dimensions Express/CM 10.1.3 for Win32 Server

Clients:
- Serena Dimensions 10.1.3 for Win32 Client

Softimage Alienbrain

Version: Alienbrain Server 8.1.0.7300

Clients:
- Softimage Alienbrain Essentials/Advanced Client 8.1.0.7300

SourceGear Fortress

Version: 1.1.4 Server

Clients:
- SourceGear Fortress 1.1.4 Client

SourceGear SourceOffsite

Version: SourceOffsite Server 4.2.0

Clients:
- SourceGear SourceOffsite Client 4.2.0 (Windows)

SourceGear Vault

Version: 4.1.4 Server

Clients:
- SourceGear Vault 4.1.4 Client

VisualSVN Server 1.6

Version: 1.6.2

Clients:
- Aigenta Unified SCC 1.0.6
- PushOK SVN SCC 1.5.1.1
- PushOK SVN SCC x64 version 1.6.3.1
- TamTam SVN SCC 1.2.24

Note the following:

- Altova has implemented the Microsoft Source Code Control Interface (MSSCCI) v1.1 – v1.3 in XMLSpy, and has tested support for the drivers and revision control systems listed above. It is expected that XMLSpy will continue to support these products if, and when, they are updated.
- Source Control plugins not listed in the table above, but that adhere to the MSCCI 1.1-1.3 specification, should also work together with XMLSpy.

© 2011 Altova GmbH

11.2 Installing Source Control Systems

This section gives information on how to install and set up the various supported Source Control Systems.

AccuBridge for Microsoft SCC 2008.2
http://www.accurev.com/

1. Install AccuRev client software, run the installer and specify the server you want to connect to (hostname and port) then create a workspace.
2. Install the AccuBridge SCC provider. Extract the ZIP archive into the <AccuRev installation dir>\bin directory.

   Register the AccuRev.dll and SccAcc.dll as follows:
3. Open a command prompt window (if you work with Vista, start Windows Explorer, go to C:\Windows\System32, right click and run cmd.exe “As administrator”).
4. Go to the <installation AccuRev dir>\bin directory.
5. Enter the following command at the command prompt:
   - Regsvr32 AccuRev.dll
   - Regsvr32 SccAcc.dll
6. Run the SwitchScc.exe program and set AccuRev as the provider.
7. Perform a Windows log off and log in again.

Aigenta Unified SCC 1.0.6
http://aigenta.com/products/UnifiedScc.aspx

Requirements: source control client. Aigenta Unified SCC works with:

- subversion command line client 1.5.4 at http://subversion.tigris.org
- CVSNT 2.5 (client) at http://www.cvsnt.org
- Bazaar 1.9 Windows (and related pre-requisites)at http://bazaar-vcs.org/ (http://bazaar-vcs.org/WindowsInstall)

A standard installation will work correctly with Altova products.

Borland StarTeam Cross-Platform Client 2008 R2
http://www.borland.com/us/products/starteam

To install the Borland StarTeam Microsoft SCC integration run the setup program and choose to install the SCC API Integration. Altova products can now connect to the repository by specifying the server address, the end point, user and password to log on, your project and working path.

Codice Software Plastic SCM Professional 2.7.127.10 (SCC Plugin)
http://www.codicesoftware.com/xpproducts.aspx

A standard installation will work correctly with Altova products. It is sufficient to install the “client” and the “Visual Studio SCC plug-in” components.

ComponentSoftware CS-RCS (PRO) 5.1
http://www.componentsoftware.com/Products/RCS

1. To install ComponentSoftware CS-RCS (PRO) start the setup and choose the option “Workstation Setup”.
2. Specify your repository tree root and when the installation is finished, restart your machine as requested.
3. Use the “ComponentSoftware RCS Properties” to choose, or create, a project and to specify a work folder.

Dynamsoft SourceAnywhere for VSS 5.3.2 Client
http://www.dynamsoft.com/Products/SAW_Overview.aspx

A standard installation will work correctly with Altova products. To integrate with Altova products you do not need to install the plug-in for Adobe DreamWeaver CS3. After the installation, establish a server connection and set a working folder.

Dynamsoft SourceAnywhere Hosted Client (22252)
http://www.dynamsoft.com/Products/SourceAnywhere-Hosting-Version-Control-Source-Control.aspx

A standard installation will work correctly with Altova products. To integrate with the Altova products you do not need to install the plug-in for Adobe DreamWeaver CS3.

Dynamsoft SourceAnywhere Standalone 2.2 Client
http://www.dynamsoft.com/Products/SourceAnywhere-SourceSafe-VSS.aspx

A standard installation will work correctly with Altova products. To integrate with the Altova products you do not need to install the plug-in for Adobe DreamWeaver CS3. After the installation, establish a server connection and set a working folder.

IBM Rational ClearCase 7.0.1 (LT)
http://www-01.ibm.com/software/awdtools/clearcase/

To install IBM Rational ClearCase LT run the setup.

- You will be asked to update the version of the InstallShield scripting engine if it is older than version 10.5, choose “Update it if necessary”. The update runs prior to the installation starting.
- Choose the default option “Enterprise deployment, create a network release area and customize it using Siteprep”.

To integrate with Altova products, it is sufficient to install only the client. Check only the client check box.

- Provide a server name and the license server element(s) following the examples provided by the installer (port@server_name).
- Provide a configuration description name by editing a name you like and insert the path to a Release area. This path must specify a shared folder.
- You can create a new folder on your machine, share it, and use it as a Release Area. (In Vista, you must set the Network discovery to "on" in Network and Sharing Center to set this path.) The Release Area is now created, some files are copied into it and a shortcut is created with the name **sitedefs.lnk**.
- When all files are copied, continue by clicking the shortcut from Windows Explorer. A

new setup will start to install the client.
- When setup starts, choose the option “Install IBM Rational ClearCase LT”.
- Keep clicking “Next”, accept the “Software License Agreement” and start the installation.

In **Vista**, the second setup could generate the internal error: 2739. In this case, start Windows Explorer and go to C:\Windows\System32.
- Right click and run “cmd.exe” “As Administrator”. A command window pops up.
- Type “regsvr32 jscript.dll”.
- Launch the setup again.

To work with files stored in ClearCase, you should create a view that points to your ClearCase project.

Jalindi Igloo 1.0.3
http://www.jalindi.com/igloo/

To use Jalindi Igloo with Altova products it is sufficient to run the setup to install Jalindi Igloo. Note that if you uninstall Jalindi Igloo, all other installed SCC Provider Windows registry keys (if any) are deleted as well and are not longer available.

When working with Altova products, setting the “Auto Commit” Mode is recommended.
- Auto Commit Mode is found in the advanced Source Control options.
- After defining a workspace, you can start to work.

March-Hare CVS Suite Client 2008 (3321)
http://www.march-hare.com/cvsnt/en.asp

A “typical” installation will work correctly with Altova products.

Mercurial
see under Sergey Antonov HgScc 1.0.1

Microsoft SourceSafe 2005 with CTP
http://msdn.microsoft.com/en-us/vstudio/aa718670.aspx

A standard installation of Microsoft Source Safe 2005 will work correctly with Altova products.

Microsoft Team Foundation Server 2008 MSSCCI Provider
http://www.microsoft.com/downloads

Requirements: Visual Studio 2008 Team Explorer, or Visual Studio 2008 with Team Explorer 2008. A standard installation will work correctly with Altova products.

Perforce P4V 2008.1

http://www.perforce.com/

The Perforce Visual Client (P4V) offers a choice:

- To install all client features (default behavior)
- To install only the “SCC Plug-in (P4SCC)” feature.

The default installation will work correctly with all Altova products.

If the “SCC Plug-in (P4SCC)” feature is chosen:

- Two SCC functions “Show differences” and “Source control manager” will not work.
- The “Show differences” functionality and the possibility to launch the source control manager will not work, because they rely on the non-installed features” Visual Merge Tool” and “Visual Client (P4V)” respectively.
- The differencing functionality will need 3rd party software, while the launch of the source control manager will only be possible after the explicit installation of the “Visual Client (P4V)".
- After starting your Perforce Visual Client installation, specify your own client configuration settings (server name, Text Editing Application, User Name).
- When the installation is finished, do not forget to create a new workspace or to select an existing one.

PureCM Client 2008/3a

http://www.purecm.com/

A standard installation will work correctly with Altova products. After the installation, start the PureCM client to register a server.

PushOK CVS SCC NT 2.1.2.5

http://www.pushok.com/soft_cvs.php

A standard installation is sufficient for using PushOK CVS SCC NT.

- After installation is complete, make sure your copy of the CVS proxy plug-in is correctly registered.
- After defining a workspace, you can start to work.

PushOK CVS SCC x64 version 2.2.0.4

http://www.pushok.com/soft_cvs.php

A standard installation is sufficient for using PushOK CVS SCC.

- After installation is complete, make sure your copy of the CVS proxy plug-in is correctly registered.
- After defining a workspace, you can start to work.

PushOK SVN SCC 1.5.1.1

http://www.pushok.com/soft_svn.php

A standard installation of PushOK SVN SCC is sufficient for use with Altova products. When installing under Vista, it is possible that the COM library **svncom.dll** cannot be registered. In this case, finish the installation, and then register the library manually by following these steps:

1. Start a command window using the option "Run as administrator".
2. Enter: cd “C:\Program Files\PushOK Software\SVNSCC\svn”

© 2011 Altova GmbH

3. Type the command > regsvr32 svncom.dll.

PushOK SVN SCC x64 version 1.6.3.1
http://www.pushok.com/soft_svn.php

A standard installation of PushOK SVN SCC is sufficient for use with Altova products. When installing under Vista, it is possible that the COM library **svncom.dll** cannot be registered. In this case, finish the installation, and then register the library manually by following these steps:

1. Start a command window using the option "Run as administrator".
2. Enter: cd "C:\Program Files\PushOK Software\SVNSCC\svn"
3. Type the command > regsvr32 svncom.dll.

QSC Team Coherence Client 7.2.1.35
http://www.teamcoherence.com

A standard installation will work correctly with Altova products.

- If the server is installed on the client machine, a default connection is created after the client installation.
- If the server resides on a different machine, you need to change the "HOSTNAME" property in the Connection Properties dialog of Team Coherence client, to point to the relevant machine.

Qumasoft QVCS Enterprise 2.1.18
http://www.qumasoft.com/

Requirements: J2SE 1.5 or later http://java.sun.com/j2se/downloads.html

To install Qumasoft QVCS-Enterprise client, run the installer. If your operating system is **Vista**, you must modify the installation directory from the default value "C:\Program Files\QVCS-Enterprise Client" to "C:\QVCS-Enterprise Client". This must be done as Vista does not let applications write to the C:\Program Files area. Edit the "**setEnv.cmd**" file that resides in the installation directory so that the JAVA_HOME environment variable points to the location of your JVM.

If you work with **Vista** you might have problem when saving the file.

1. If this is the case, start Windows Explorer and go to C:\Windows\System32.
2. Right click and run "cmd.exe" "As Administrator".
3. A command window pops up.
4. Type "cd <installation folder of the QVCS –Enterprise client>"
5. Type "Notepad setEnv.cmd" and then edit the file and save it.
6. From the installation directory of the Qumasoft QVCS-Enterprise client run the batch file "gui.bat".
7. Add a server from the "Server menu" specifying the requested name, IP address and ports, log in and define a local workspace.

Qumasoft QVCS Pro 3.10.18
http://www.qumasoft.com/

To install Qumasoft QVCS-Pro run the installer.

- If your operating system is **Vista**, you must modify the installation directory from the default value "C:\Program Files\QVCSBin" to "C:\QVCSBin". This must be done as Vista does not let applications write to the C:\Program Files area.
- After installation is finished, launch the QVCS 3.10 client, create a new user and enable **Ide integration** by selecting the submenu "Ide Integration" in the Admin menu and adding QVCS as a Version Control Tool.
- Create a project and set a workspace.

Reliable Software Code Co-Op 5.1a
http://www.relisoft.com/co_op/index.htm

A standard installation will work correctly with Altova products.

Seapine Surround SCM Client 2009.0.0
http://www.seapine.com/surroundscm.html

A standard installation will work correctly with Altova products. After installation, a server connection must be established.

Serena Dimensions 10.1.3 for Win32 Client
http://www.serena.com/Products/dimensions/

Supported Versions: Dimensions Express/CM 10.1.3 for Win32 Client

- Perform a "Typical" installation of the Serena Dimension client.
- Specify the WEB client hostname and port number as requested.

Sergey Antonov HgSCC 1.0.1 for Mercurial SCS
http://www.newsupaplex.pp.ru/hgscc_news_eng.html

A standard installation will work correctly with Altova products.

Softimage Alienbrain Essentials/Advanced Client 8.1.0.7300
http://www.alienbrain.com/

- Perform a "typical" installation of the Alienbrain Client Software, then install the Alienbrain Microsoft Visual Studio Integration. To work with Altova products you do not need to install Microsoft Visual Studio.
- The first time you try to open a project from VCS, or to add a project to VCS you will be asked to enter some user settings, e.g. to specify your server and choose the project database you want to connect to.

SourceGear Fortress 1.1.4 Client
http://www.sourcegear.com/fortress

A standard installation of SourceGear Fortress client will work with Altova products.

© 2011 Altova GmbH

SourceGear SourceOffsite Client 4.2.0 (Windows)
http://www.sourcegear.com/sos/

A standard installation of SourceOffsite client will work with Altova products.

SourceGear Vault 4.1.4 Client
http://www.sourcegear.com/vault

A standard installation of SourceGear Vault client will work correctly with Altova products.

TamTam CVS SCC 1.2.40
http://www.daveswebsite.com/software/tamtam

Requirements: CVSNT 2.5 (client) at http://www.cvsnt.org. A standard installation will work correctly with Altova products.

- To connect to the CVS repository you need to install CVSNT.
- In the Altova product open the "Source control" Advanced options and enter the path to the **cvs.exe** executable.

TamTam SVN SCC 1.2.24
http://www.daveswebsite.com/software/tamtamsvn

Requirements: subversion command line client 1.5.4 at http://subversion.tigris.org . A standard installation will work correctly with Altova products.

- To connect to the SVN repository you need to install the subversion command line client and specify the path to the executable **svn.exe** in the Altova product Source control options.
- After starting XMLSpy, you must register the SCC provider.

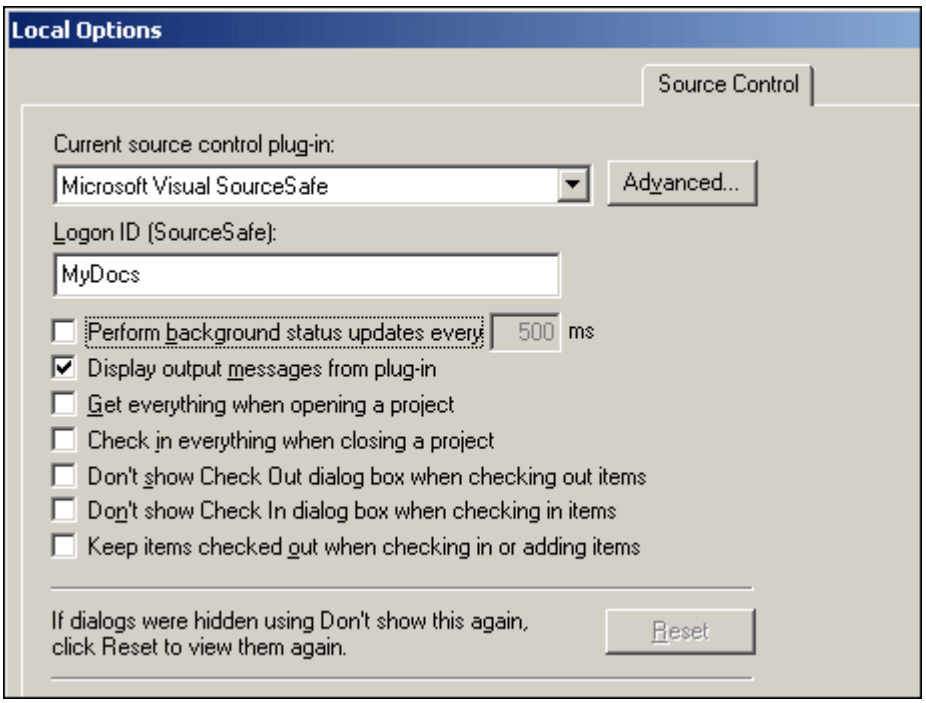
On a **Vista** machine the SCC registration could fail.

- If this is the case, use Windows explorer and browse to the directory that contains the Altova application executable.
- Right click and run the Altova executable "As Administrator".
  The SCC registration will now be successful.

11.3 SCSs and Altova DiffDog Differencing

You can configure certain Source Control Systems so that they use Altova DiffDog as their differencing tools. The systems that support this feature are listed below, together with the setup steps for each. In XMLSpy, you access the setup process via the Source Control tab of the Options dialog (**Tools | Options**, *screenshot below*).

When using a 64-bit version of XMLSpy with a 32-bit source control program, the "Perform background status updates every... ms" option (Tools | Options) is automatically **grayed-out** and cannot be selected!



The "Perform background status updates every xx ms" check box is unchecked per default, which means that status updates are not performed at all. Activate the check box and enter a value in the field, if you want to perform status updates every xx ms. For 64-bit versions using 32-bit source control plugins, this option has no effect.

In the Source Control tab, select the required Source Control System and then click the **Advanced** button. The dialog box that opens will be different for each Source Control System. For the setup process, note the following:

- If you have performed a standard installation of Altova DiffDog, the file path to the Altova DiffDog executable is:
  `c:\program files\altova\diffdog2011\DiffDog.exe`
  If Altova DiffDog is installed elsewhere on your system, insert the appropriate value when the filepath is required.

Aigenta Unified SCC 1.0.6
http://aigenta.com/products/UnifiedScc.aspx

The following steps will integrate Altova DiffDog into Aigenta Unified SCC:

1. Click the **Advanced** button of the Source Control tab.
2. Select the *Comparison and merging tab* and specify the full DiffDog filepath as comparison tool.

 © 2011 Altova GmbH

Borland StarTeam Cross-Platform Client 2008 R2
http://www.borland.com/us/products/starteam

The following steps integrate Altova DiffDog into Borland Star Team:
1. Use the StarTeam client personal options (**Tools | Personal options | File | Alternate applications**)
2. Compare utility: Enter the DiffDog full path.
3. Compare utility options: `$file1 $file2.`

ComponentSoftware CS-RCS (PRO) 5.1
http://www.componentsoftware.com/Products/RCS

The following steps will integrate Altova DiffDog into ComponentSoftware CS-RCS (Pro):
1. Go to the ComponentSoftware CS-RCS Properties.
2. In the *File Types* tab, choose a file extension and edit it.
3. Enter/select the value *Custom Tool* for the "Difference Analysis Tool", and browse to insert the DiffDog full path.

Dynamsoft SourceAnywhere for VSS 5.3.2 Client
http://www.dynamsoft.com/Products/SAW_Overview.aspx

The following steps will integrate Altova DiffDog into Dynamsoft SourceAnywhere for VSS:
1. Go to the Dynamic SourceAnywhere For VSS client Options.
2. Specify the DiffDog full path as External application for diff/merge, with the arguments: `%FIRST_FILE%" "%SECOND_FILE%.`

Warning: Do not perform these settings from the Altova product options, as there is no possibility of inserting the external application parameters.

Dynamsoft SourceAnywhere Hosted Client (22252)
http://www.dynamsoft.com/Products/SourceAnywhere-Hosting-Version-Control-Source-Control.aspx

Dynamsoft SourceAnywhere Standalone 2.2 Client
http://www.dynamsoft.com/Products/SourceAnywhere-SourceSafe-VSS.aspx

The following steps will integrate Altova DiffDog into Dynamsoft SourceAnywhere Hosted and Dynamsoft SourceAnywhere Standalone:
1. Click the **Advanced** button of the Source Control tab.
2. Specify the DiffDog full path as External program application for diff/merge with arguments `%FIRST_FILE%" "%SECOND_FILE%.`

Jalindi Igloo 1.0.3
http://www.jalindi.com/igloo/

The following steps will integrate Altova DiffDog into Jalindi Igloo:
1. Start the **Show differences** command in XMLSpy.

2. Open the **Show Differences or Merge Files** panel.
3. Set the *External Diff Command* by entering the DiffDog full file path as the External Diff EXE path.

Warning: When using the default diff editor CvsConflictEditor, you might have problems comparing files with excessively long lines. We recommended that you "pretty print" all files (particularly `.ump` files) before storing them in the repository. This limits the line length, thus avoiding problems with the CVSConflictEditor.

March-Hare CVS Suite Client 2008 (3321)
http://www.march-hare.com/cvsnt/en.asp

The following steps will integrate Altova DiffDog into Marc-Hare CVS Suite 2008:
1. Go to the TortoiseCVS Preferences and choose the Tools tab.
2. Specify the DiffDog full path as Diff application, and the parameters `%1 %2` as two-way differencing parameters.

Mercurial
see under Sergey Antonov HgScc 1.0.1

Microsoft SourceSafe 2005 with CTP
http://msdn.microsoft.com/en-us/vstudio/aa718670.aspx

The following steps will integrate Altova DiffDog into Microsoft SourceSafe 2005:
1. Click the **Advanced** button of the Source Control tab.
2. Click the Custom Editors tab and enter `C:\Program Files\Altova\DiffDog2011\DiffDogexe %1 %2` in the Command Line field.
3. In the Operation combo box, select *File Difference*.

Microsoft Team Foundation Server 2008 MSSCCI Provider
http://www.microsoft.com/downloads

Requirements: Visual Studio 2008 Team Explorer *or* Visual Studio 2008 *with* Team Explorer 2008. The following steps will integrate Altova DiffDog into Microsoft Visual Studio Team System 2008 Team Foundation Server MSSCCI Provider:
1. In the manager (Visual Studio 2008 Team Explorer or Visual Studio 2008) options, configure Altova DiffDog as new user tool
2. Choose Visual Studio Team Foundation Server source as the plug-in.
3. Configure a new user tool specifying: (i) the extensions of the files you wish to compare with DiffDog; and (ii) the DiffDog full file path.

Perforce P4V 2008.1
http://www.perforce.com/

The following steps will integrate Altova DiffDog into Perforce 2008:
1. Click the **Advanced** button of the Source Control tab.
2. Choose the tab Diff in the Preferences panel.
3. Check as default differencing application the field “Other application” and enter the DiffDog full file path.

 © 2011 Altova GmbH

PushOK CVS SCC NT 2.1.2.5,
PushOK SVN SCC 1.5.1.1
PushOK CVS SCC x64 version 2.2.0.4
PushOK SVN SCC x64 version 1.6.3.1
http://www.pushok.com/soft_cvs.php

The following steps will integrate Altova DiffDog into PushOK CVS NT and PushOK SVN SCC:

1. Click the **Advanced** button of the Source Control tab.
2. Choose the CVS Executables tab.
3. Select the value *External merge/compare tool* into the Diff/Merge field.
4. Insert the DiffDog full file path.
5. Edit the value `%first %second` into the "2 way diff cmd" field.

Warning: When using the default differencing editor CvsConflictEditor, you might have problems comparing files with excessively long lines. We recommended that you "pretty print" all files (particularly `.ump` files) before storing them in the repository. This limits the line length, thus avoiding problems with the CVSConflictEditor.

QSC Team Coherence Client 7.2.1.35
http://www.teamcoherence.com

The following steps will integrate Altova DiffDog into Team Coherence Version Manager:

1. Go to Team Coherence client Options “Difference Viewer”.
2. Specify as the Default Difference Viewer application, the DiffDog full file path.
3. Specify as parameters: `"$LF $RF"`.

Warning: It is possible that the new settings will only be applied after a Windows log off.

Qumasoft QVCS Enterprise 2.1.18
http://www.qumasoft.com/

The following steps will integrate Altova DiffDog into Qumasoft QVCS-Enterprise:

1. Add the Qumasoft QVCS-Enterprise installation directory to the Path environment variable.
2. Use the QVCS Enterprise User Preferences.
3. In Utilities, enable the checkbox, Use External Visual Compare Tool.
4. Specify as Visual Compare Command Line:
   `<DiffDog full path> “file1Name file2Name”`

Qumasoft QVCS Pro 3.10.18
http://www.qumasoft.com/

The following steps will integrate Altova DiffDog into Qumasoft QVCS-Pro:

1. Use the QVCS 3.10 client preferences.
2. In Utilities, specify the DiffDog full path as the visual compare utility with parameters “%s %s “.

Seapine Surround SCM Client 2009.0.0
http://www.seapine.com/surroundscm.html

The following steps will integrate Altova DiffDog into Seapine Surround SCM:
1. Go to the Surround SCM client user options (Diff/Merge) section.
2. Edit the Diff/Merge settings to compare with a selected application.
3. Enter the DiffDog full path with the parameters “%1” “%2”.
4. Restart the Surround SCM client and the Altova products.

Sergey Antonov HgSCC 1.0.1
http://www.newsupaplex.pp.ru/hgscc_news_eng.html

The following steps will integrate Altova DiffDog into Mercurial:
1. Click the **Advanced** button of the Source Control tab.
2. Select differencing tool “custom”, and specify the DiffDog full path.

SourceGear Fortress 1.1.4 Client
http://www.sourcegear.com/fortress

SourceGear Vault 4.1.4 Client
http://www.sourcegear.com/vault

The following steps will integrate Altova DiffDog into SourceGear Fortress and SourceGear Vault:
1. Click the **Advanced** button of the Source Control tab.
2. Set the Diff/Merge Vault options by specifying as the differencing program the DiffDog full path and using the Arguments:
   /ro1 /ro2 /title1:"%LEFT_LABEL%" /title2:"%RIGHT_LABEL%" "%LEFT_PATH%" "%RIGHT_PATH%"

SourceGear SourceOffsite Client 4.2.0 (Windows)
http://www.sourcegear.com/sos/

The following steps will integrate DiffDog into SourceGear SourceOffsite:
1. Click the **Advanced** button of the Source Control tab.
2. Specify as “External Programs”, “Application for comparing files” the DiffDog full path.

TamTam CVS SCC 1.2.40,
TamTam SVN SCC 1.2.24
http://www.daveswebsite.com/software/tamtamGo

The following steps will integrate Altova DiffDog into TamTam CVS SCC and TamTam SVN SCC:
1. Click the **Advanced** button of the Source Control tab.
2. Specify the DiffDog full file path as the external tool for Diff/Merge and Conflict.

Warning: The default differencing editor CvsConflictEditor, has problems comparing files with excessively long lines. We recommended that you "pretty print" all files (particularly `.ump` files) before storing them in the repository. This limits the line length, avoiding problems with the CVSConflictEditor.

 © 2011 Altova GmbH

12 User Reference

The **User Reference** section contains a complete description of all XMLSpy menu commands and explains their use in general. We have tried to be comprehensive. If, however, you have questions which are not covered in the User Reference or other parts of this documentation, please look up the FAQs and Discussion Forums on the Altova website. If you cannot find a suitable answer at these locations, please do not hesitate to contact the Altova Support Center.

Standard Windows commands, such as (**Open**, **Save**, **Cut**, **Copy** and **Paste**) are in the File and Edit menus. These menus additionally contain XML- and Internet-related commands.

© 2011 Altova GmbH

12.1 File Menu

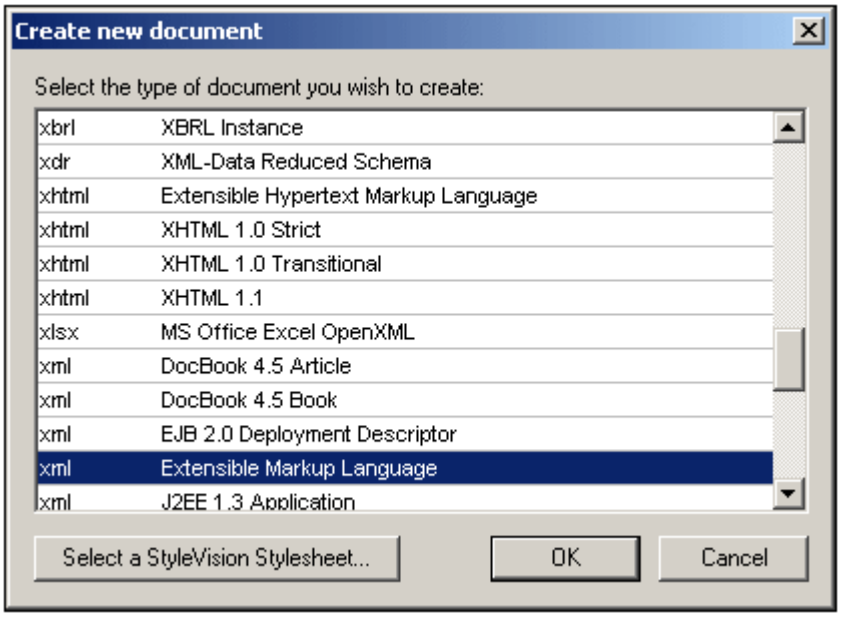
The **File** menu contains commands relevant to manipulating files, in the order common to most Windows software products.

In addition to the standard New, Open, Save, Print, Print Setup, and Exit commands, XMLSpy offers a range of XML- and application-specific commands.

12.1.1 New

Ctrl+N

The **New** command is used to create a new document. Clicking **New** opens the Create New Document dialog, in which you can select the type of document you wish to create. If the document type you wish to create is not listed, select XML and change the file extension when you save the file. Note that you can add new file types to the list in this dialog using the Tools | Options | File types tab.

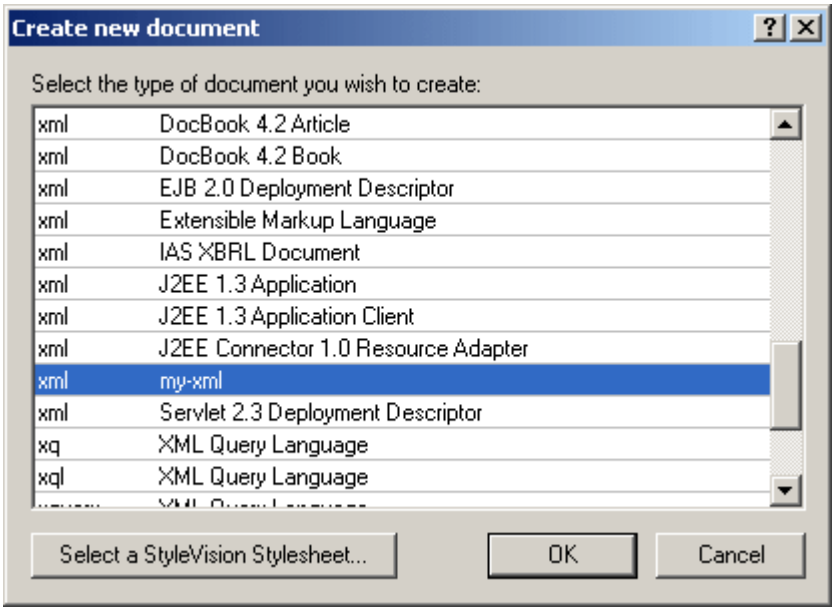


Creating templates for new documents
You can create multiple templates for various file types. These templates can then be opened directly from the Create New Document dialog and edited. To create your own template so that it appears in the list of documents in the Create New Document dialog, you first create the template document and then save it to the folder that contains all the templates.

Do the following:

1. Open the **XMLSpy\Template** folder using Windows Explorer or your preferred navigation tool, and select a rudimentary template file from among the files named **new.xxx** (where .xxx is a file extension, such as .xml and .xslt).
2. Open the file in XMLSpy, and modify the file as required. This file will be the template file.
3. When you are done, select **File | Save as...** to save the file back to the \Template

folder with a suitable name, say `my-xml.xml`. You now have a template called `my-xml`, which will appear in the list of files in the Create New Document dialog.

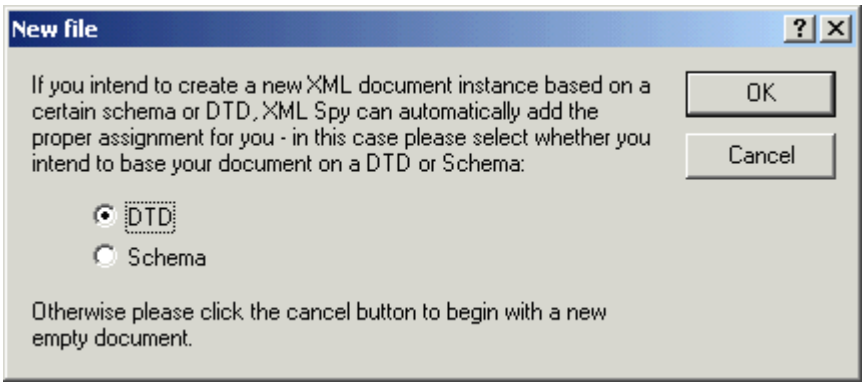


4. To open the template, select **File | New**, and then the template (`my-xml`, in this case).

Please note: To delete a template, delete the template file from the template folder.

Assigning a DTD/XML Schema to a new XML document
When you create a new document of a certain type that is based on a standard schema (DTD or XML Schema), the document is automatically opened with the correct DTD or XML Schema association. For example, an XHTML file will be opened with the DTD `http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd` associated with it. And an XML Schema (`.xsd`) file is associated with the `http://www.w3.org/2001/XMLSchema` schema document.

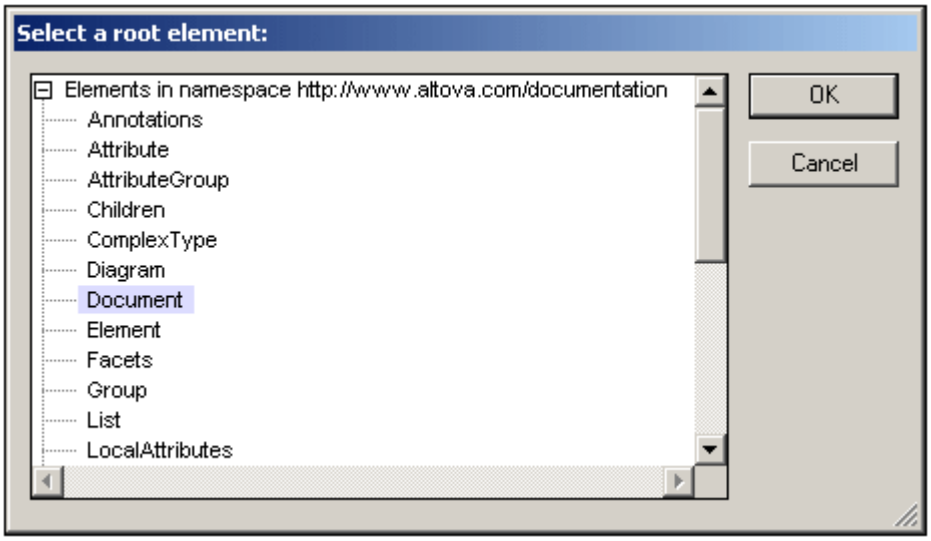
If you are creating a new file for which the schema is not known (for example, an XML file), then you are prompted to associate a schema (DTD or XML Schema) with the document that is to be created.



If you choose to associate a DTD or XML Schema with your document, clicking **OK** in the New File dialog enables you to browse for the schema. Clicking **Cancel** in this dialog will create a new file that is not associated with any schema.

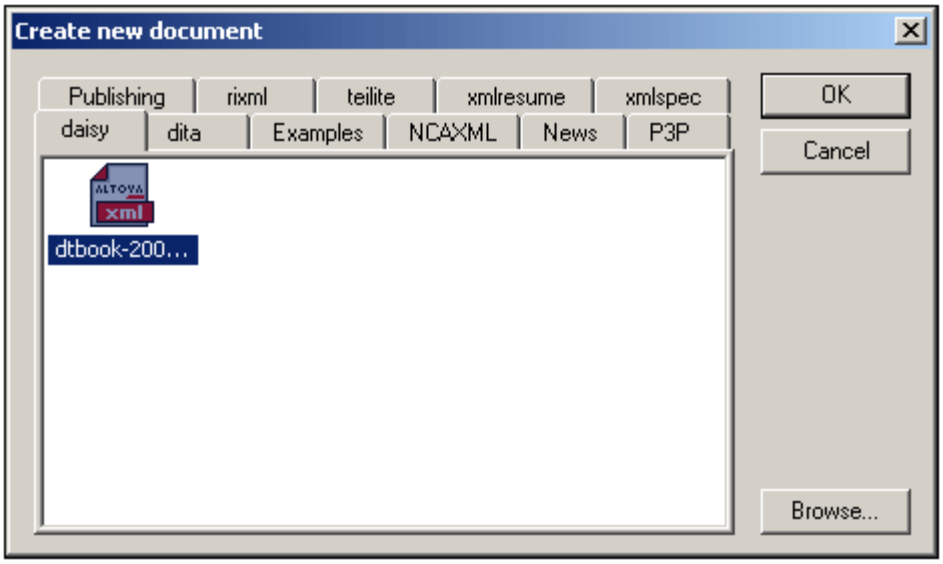
 © 2011 Altova GmbH

Specifying the document element of a new XML document
If you select an XML Schema, there can be more than one global element in it, all of which are potential document (or root) elements. You can select which of these is to be the root element of the XML document in the Select a Root Element dialog, which pops up if you select Schema in the New File dialog and if the XML Schema has more than one global element.



The new XML document is created with this element as its document element.

Assigning a StyleVision Power Stylesheet when creating a new document
When a new XML document is created, you can associate a StyleVision Power Stylesheet (`.sps` file) to view the document in Authentic View. In the Create New Document dialog (*see screenshot above*), when you click the **Select StyleVision Stylesheet**, the Create New Document dialog (*shown below*) appears.



You can browse for the required StyleVision Power Stylesheet in the folder tabs displayed in the New dialog. Alternatively, you can click the **Browse** button to navigate for and select the StyleVision Power Stylesheet. The tabs that appear in the New dialog correspond to folders in

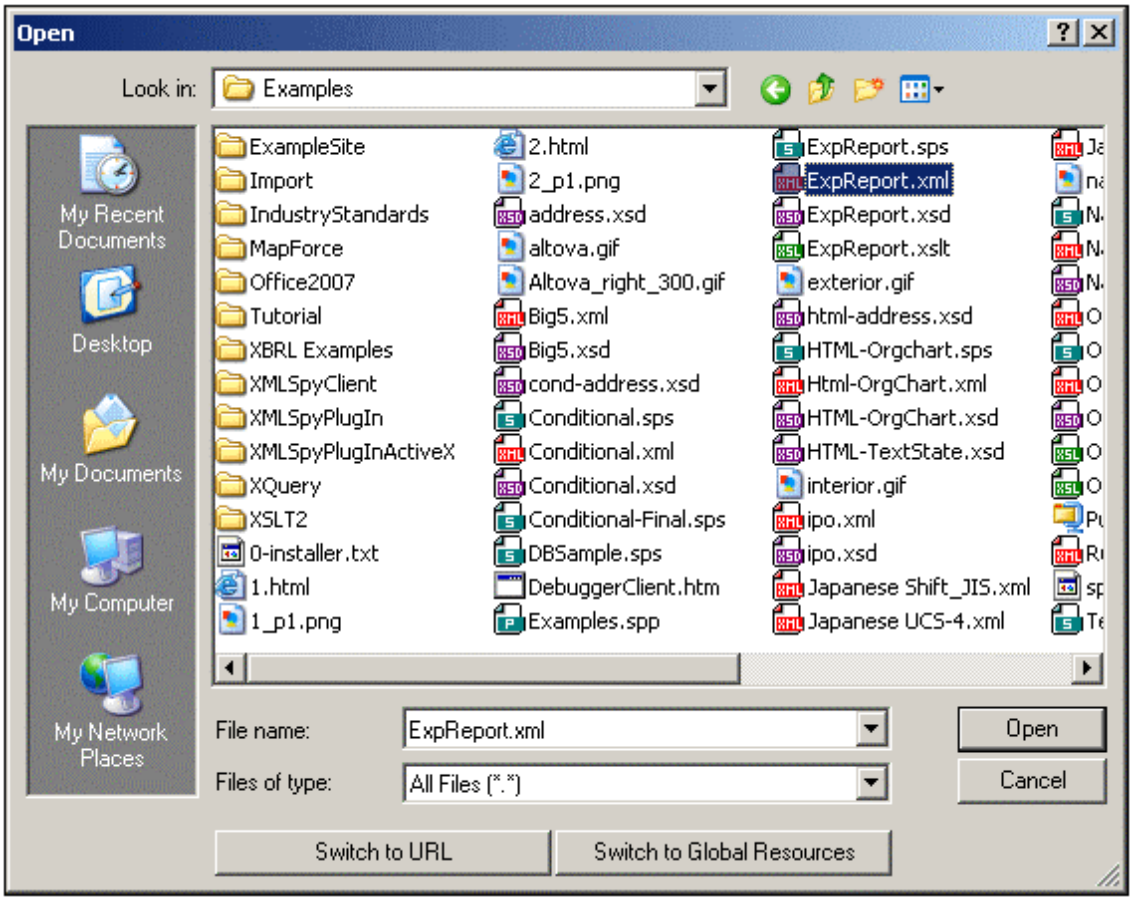
the `sps/Template` folder of your application folder.

12.1.2 Open

Ctrl+O

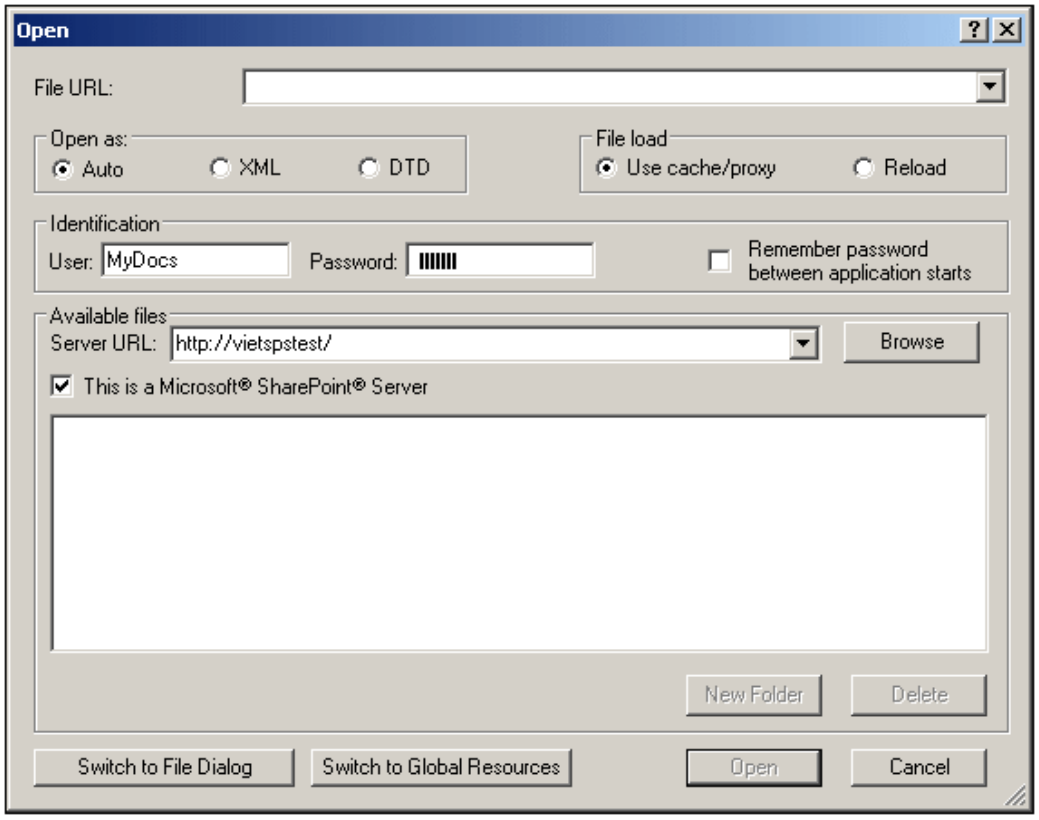
The **Open** command pops up the familiar Windows Open dialog, and allows you to open any XML-related document or text document. In the Open dialog, you can select more than one file to open. Use the Files of Type combo box to restrict the kind of files displayed in the dialog box. (The list of available file types can be configured in the File Types tab of the Options dialog (**Tools | Options**).) When an XML file is opened, it is checked for well-formedness. If the file is not well-formed, you will get a file-not-well-formed error. Fix the error and select the menu command **XML | Check Well-Formedness (F7)** to recheck. If you have opted for automatic validation upon opening and the file is invalid, you will get an error message. Fix the error and select the menu command **XML | Validate XML (F8)** to revalidate.

Selecting files via URLs and Global Resources
In several File Open and File Save dialogs, you can choose to select the required file or save a file via a URL or a global resource (*see screenshot below*). Select the **Switch to URL** or **Switch to Global Resource** to go to one of these selection processes.

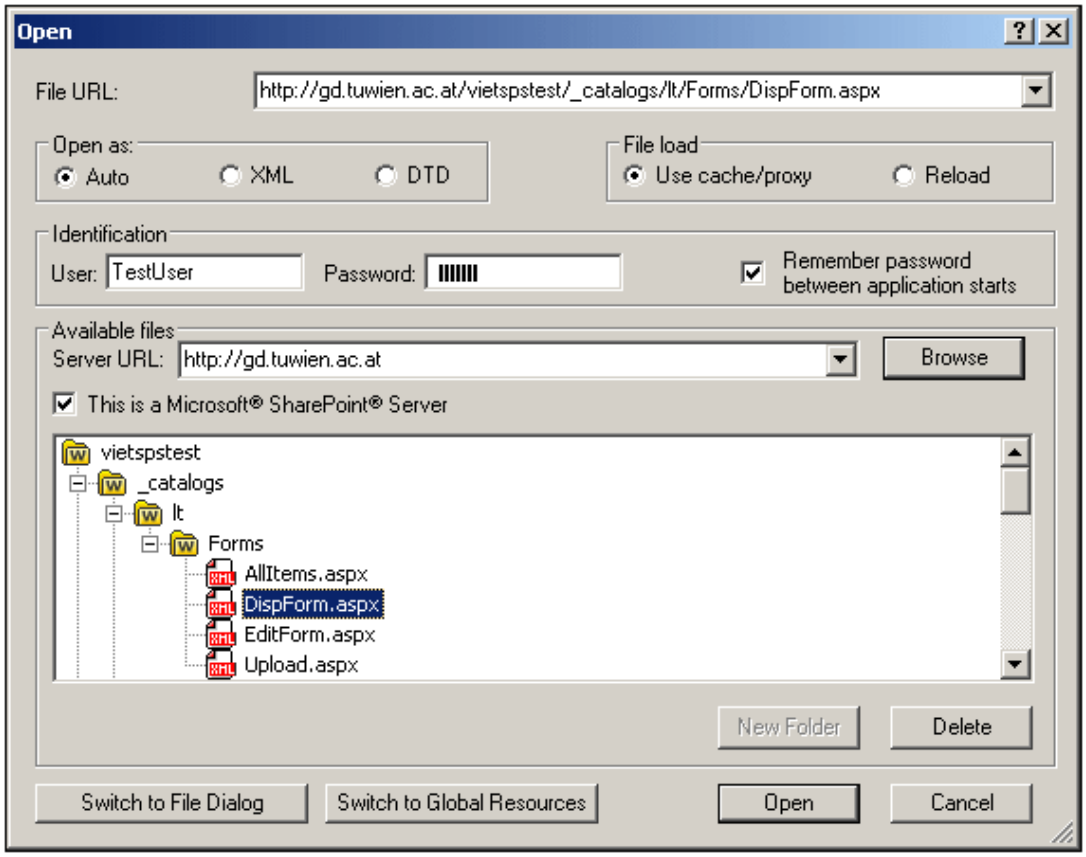


Selecting files via URLs
To select a file via a URL, do the following:

1. Click the **Switch to URL** command. This switches to the URL mode of the Open dialog (*screenshot below*).



2. Enter the URL you want to access in the *Server URL* field (*screenshot above*). If the server is a Microsoft® SharePoint® Server, check the *Microsoft® SharePoint® Server* check box. See the Microsoft® SharePoint® Server Notes below for further information about working with files on this type of server.
3. If the server is password protected, enter your User-ID and password in the *User* and *Password* fields.
4. Click **Browse** to view and navigate the directory structure of the server.
5. In the folder tree, browse for the file you want to load and click it.



The file URL appears in the File URL field (*screenshot above*). The **Open** button only becomes active at this point.

6. Click the **Open** button to load the file. The file you open appears in the main window.

Note: The Browse function is only available on servers which support WebDAV and on Microsoft SharePoint Servers. The supported protocols are FTP, HTTP, and HTTPS.

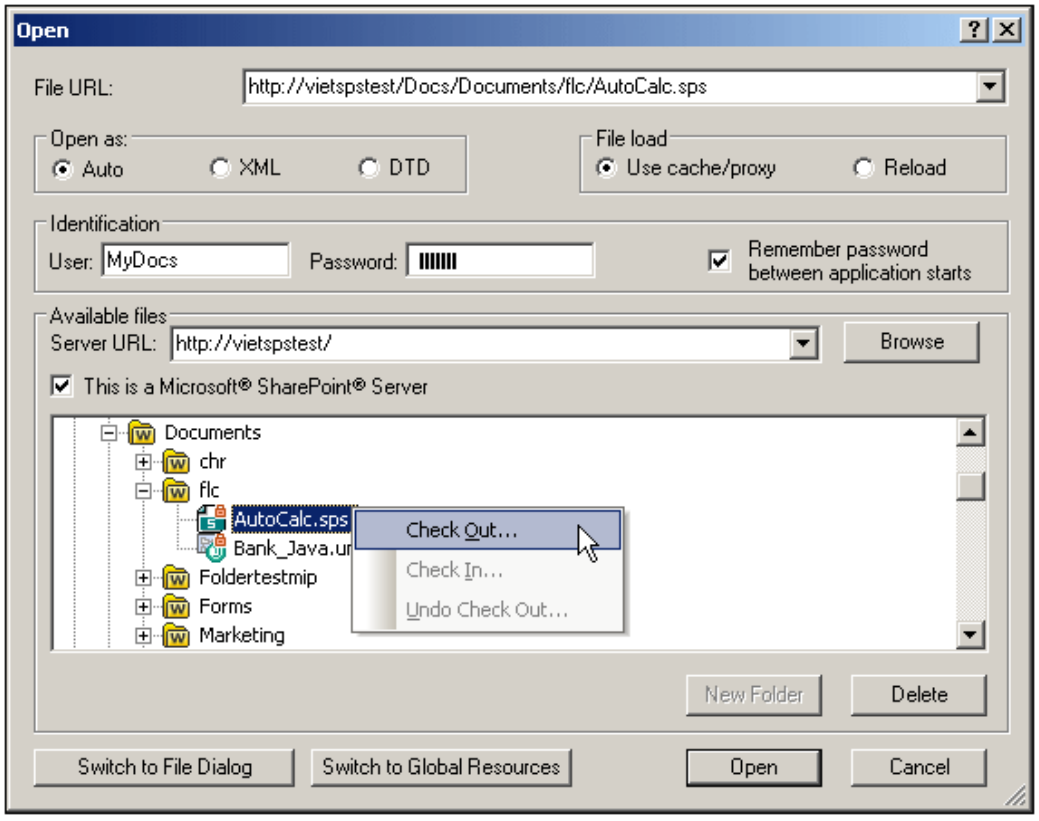
Note: To give you more control over the loading process, you can choose to load the file through the local cache or a proxy server (which considerably speeds up the process if the file has been loaded before). Alternatively, you may want to reload the file if you are working, say, with an electronic publishing or database system; select the **Reload** option in this case

Microsoft® SharePoint® Server Notes

Note the following points about files on Microsoft® SharePoint® Servers:

- In the directory structure that appears in the Available Files pane (*screenshot below*), file icons have symbols that indicate the check-in/check-out status of files.

© 2011 Altova GmbH



Right-clicking a file pops up a context menu containing commands available for that file (*screenshot above*).

- The various file icons are shown below:

	Checked in. Available for check-out.
	Checked out by another user. Not available for check-out.
	Checked out locally. Can be edited and checked-in.

- After you check out a file, you can edit it in your Altova application and save it using **File | Save (Ctrl+S)**.
- You can check-in the edited file via the context menu in the Open URL dialog (*see screenshot above*), or via the context menu that pops up when you click the file tab in the Main Window of your application (*screenshot below*).

- When a file is checked out by another user, it is not available for check out.
- When a file is checked out locally by you, you can undo the check-out with the Undo Check-Out command in the context menu. This has the effect of returning the file unchanged to the server.
- If you check out a file in one Altova application, you cannot check it out in another Altova application. The file is considered to be already checked out to you. The available commands at this point in any Altova application supporting Microsoft® SharePoint® Server will be: **Check In** and **Undo Check Out**.

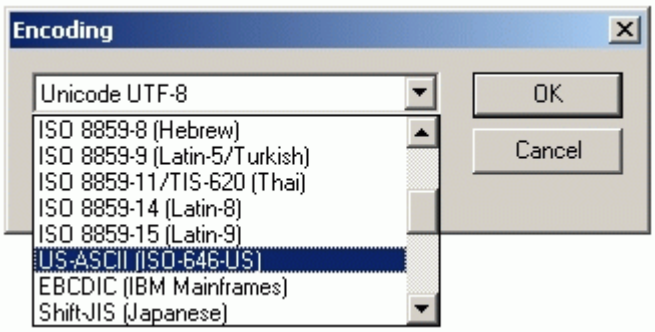
Opening and saving files via Global Resources
To open or save a file via a global resources, click **Switch to Global Resource**. This pops up a dialog in which you can select the global resource. These dialogs are described in the section, Using Global Resources. For a general description of Global Resources, see the Global Resources section in this documentation.

12.1.3 Reload

The **Reload** command allows you to reload open documents. This is useful if an open document has been modified outside XMLSpy. If a modification occurs, XMLSpy asks whether you wish to reload the file. If you reload, then any changes you may have made to the file since the last save will be lost. This option can be changed in the Options dialog (Tools | Options).

12.1.4 Encoding

The **Encoding** command lets you view the current encoding of the active document (XML or non-XML) and to select a different encoding with which the active document will be saved the next time.



In XML documents, if you select a different encoding than the one in use before, the encoding specification in the XML declaration will be adjusted accordingly. For two-byte and four-byte character encodings (UTF-16, UCS-2, and UCS-4) you can also specify the byte-order to be used for the file. Another way to change the encoding of an XML document is to directly edit the encoding attribute of the document's XML declaration.

Default encodings for existing and new XML and non-XML documents can be set in the Encoding tab of the Options dialog.

Note: When saving a document, XMLSpy automatically checks the encoding specification and opens a dialog box if it cannot recognize the encoding entered by the user. Also, if your document contains characters that cannot be represented in the selected encoding, you will get a warning message when you save your file.

12.1.5 Close, Close All, Close All But Active

The **Close** command closes the active document window. If the file was modified (indicated by an asterisk * after the file name in the title bar), you will be asked if you wish to save the file first.

 © 2011 Altova GmbH

The **Close All** command closes all open document windows. If any document has been modified (indicated by an asterisk * after the file name in the title bar), you will be asked if you wish to save the file first.

The **Close All But Active** command closes all open document windows except the active document window. If any document has been modified (indicated by an asterisk * after the file name in the title bar), you will be asked if you wish to save the file first.

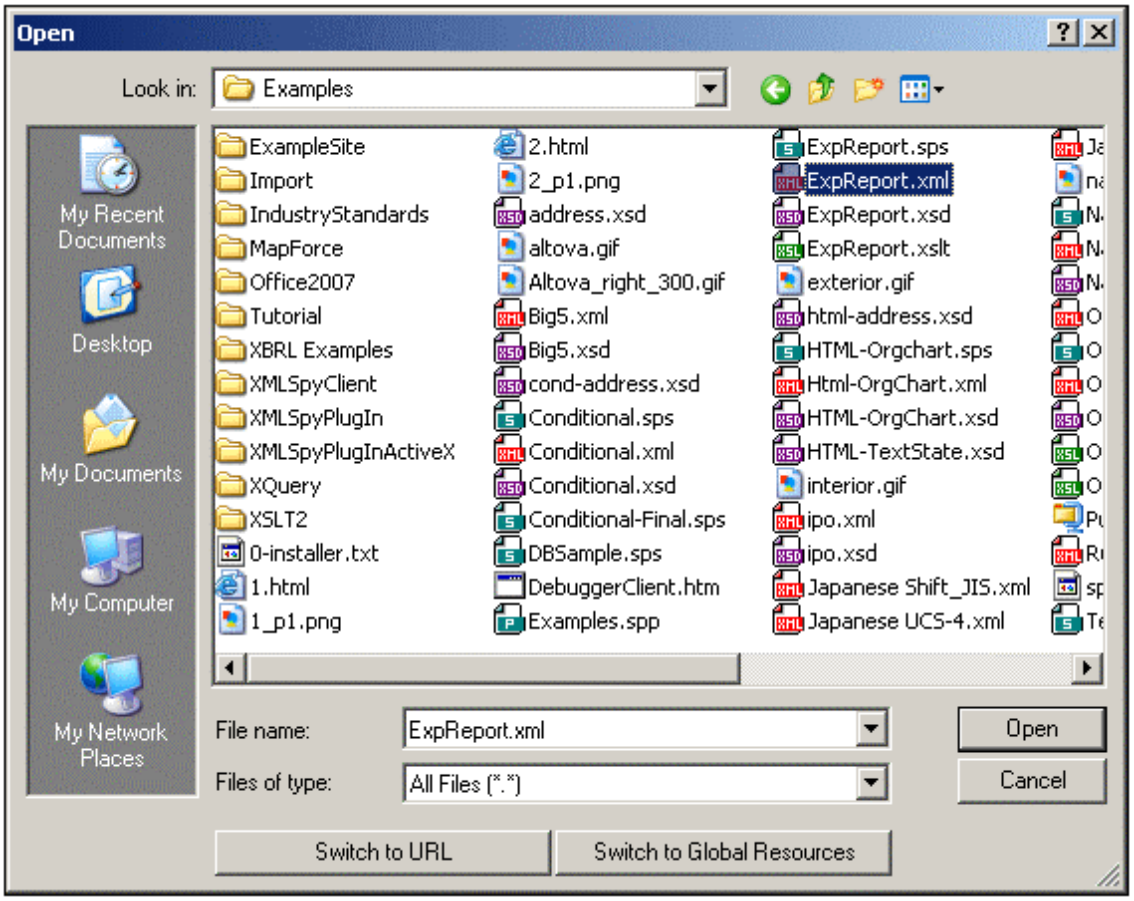
12.1.6 Save, Save As, Save All

The **Save (Ctrl+S)** command saves the contents of the active document to the file from which it has been opened. When saving a document, the file is automatically checked for well-formedness. The file will also be validated automatically if this option has been set in the File tab of the Options dialog (**Tools | Options**). The XML declaration is also checked for the encoding specification, and this encoding is applied to the document when the file is saved.

The **Save As** command pops up the familiar Windows Save As dialog box, in which you enter the name and location of the file you wish to save the active file as. The same checks and validations occur as for the **Save** command.

The **Save All** command saves all modifications that have been made to any open documents. The command is useful if you edit multiple documents simultaneously. If a document has not been saved before (for example, after being newly created), the Save As dialog box is presented for that document.

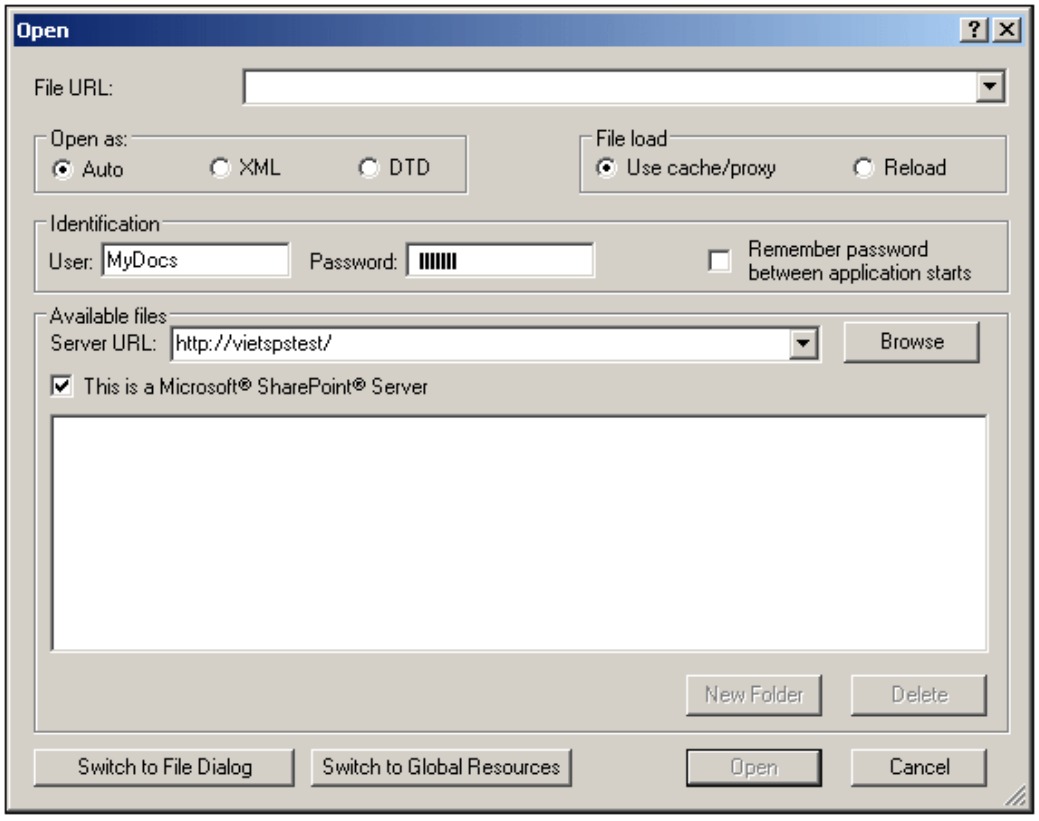
Selecting files via URLs and Global Resources
In several File Open and File Save dialogs, you can choose to select the required file or save a file via a URL or a global resource (*see screenshot below*). Select the **Switch to URL** or **Switch to Global Resource** to go to one of these selection processes.



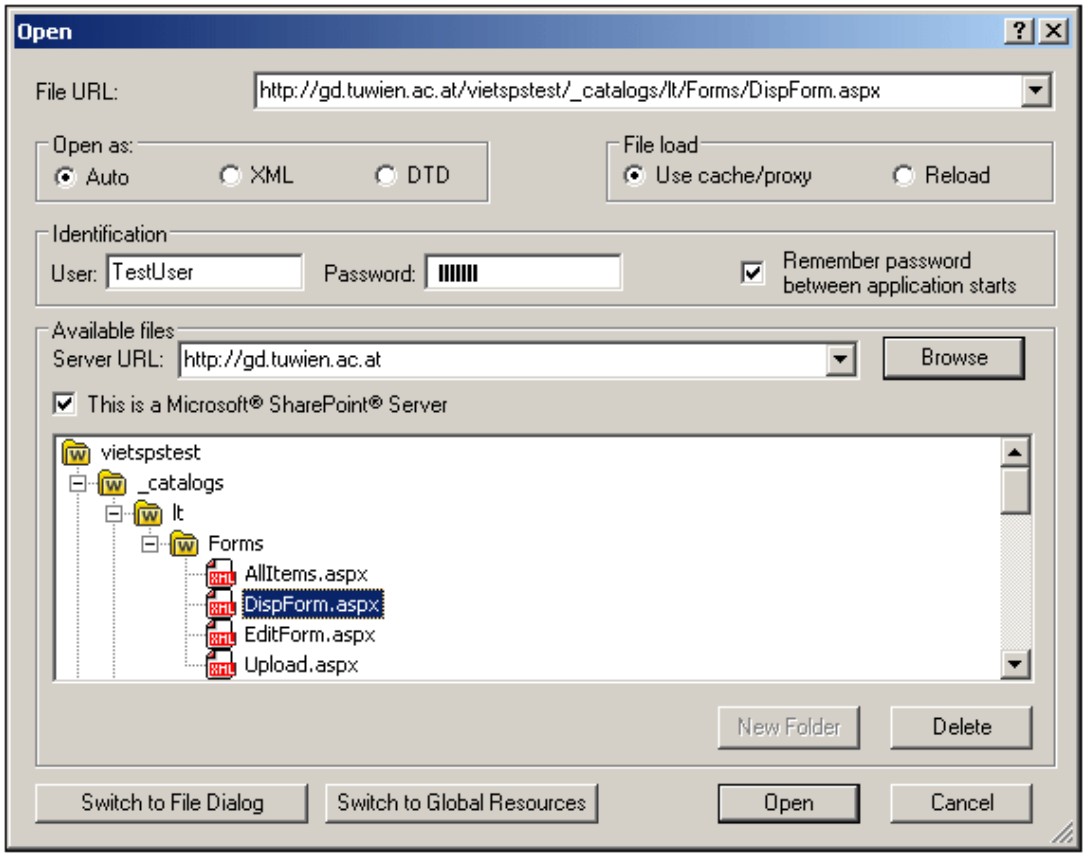
Selecting files via URLs
To select a file via a URL, do the following:

1. Click the **Switch to URL** command. This switches to the URL mode of the Open dialog (*screenshot below*).

© 2011 Altova GmbH



2. Enter the URL you want to access in the *Server URL* field (*screenshot above*). If the server is a Microsoft® SharePoint® Server, check the *Microsoft® SharePoint® Server* check box. See the Microsoft® SharePoint® Server Notes below for further information about working with files on this type of server.
3. If the server is password protected, enter your User-ID and password in the *User* and *Password* fields.
4. Click **Browse** to view and navigate the directory structure of the server.
5. In the folder tree, browse for the file you want to load and click it.



The file URL appears in the File URL field (*screenshot above*). The **Open** button only becomes active at this point.

6. Click the **Open** button to load the file. The file you open appears in the main window.

Note: The Browse function is only available on servers which support WebDAV and on Microsoft SharePoint Servers. The supported protocols are FTP, HTTP, and HTTPS.

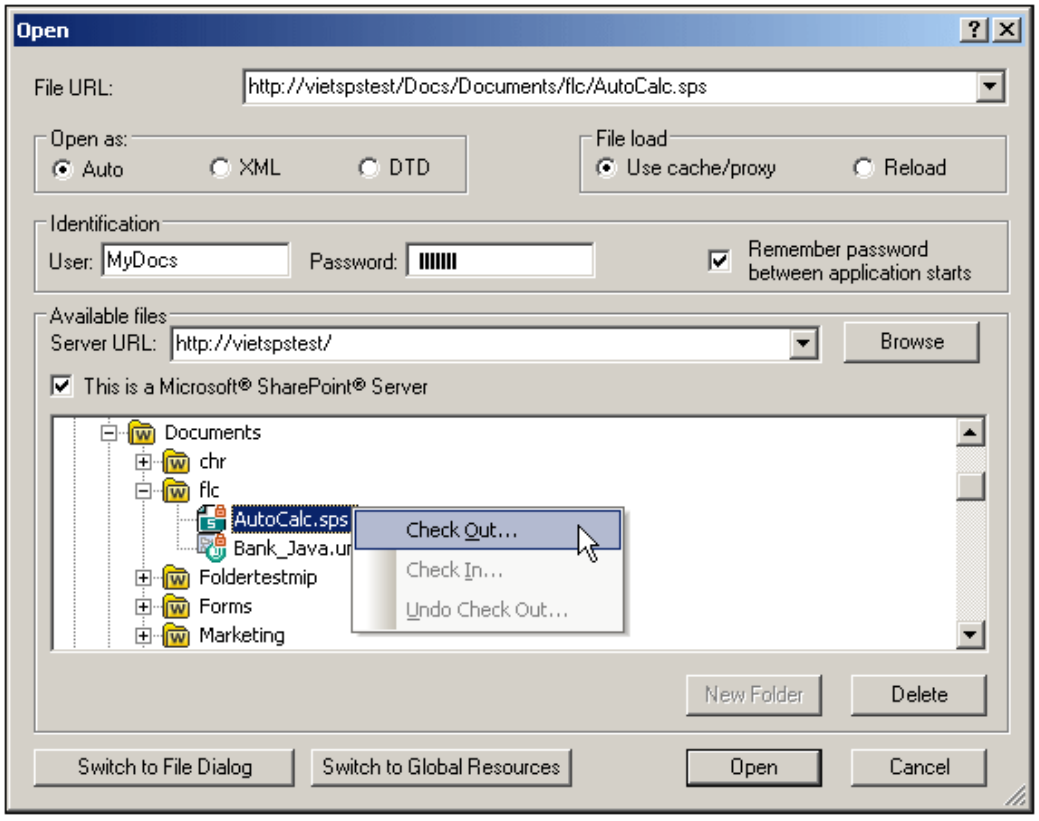
Note: To give you more control over the loading process, you can choose to load the file through the local cache or a proxy server (which considerably speeds up the process if the file has been loaded before). Alternatively, you may want to reload the file if you are working, say, with an electronic publishing or database system; select the **Reload** option in this case

Microsoft® SharePoint® Server Notes

Note the following points about files on Microsoft® SharePoint® Servers:

- In the directory structure that appears in the Available Files pane (*screenshot below*), file icons have symbols that indicate the check-in/check-out status of files.

© 2011 Altova GmbH



Right-clicking a file pops up a context menu containing commands available for that file (*screenshot above*).

- The various file icons are shown below:

	Checked in. Available for check-out.
	Checked out by another user. Not available for check-out.
	Checked out locally. Can be edited and checked-in.

- After you check out a file, you can edit it in your Altova application and save it using **File | Save (Ctrl+S)**.
- You can check-in the edited file via the context menu in the Open URL dialog (*see screenshot above*), or via the context menu that pops up when you click the file tab in the Main Window of your application (*screenshot below*).

- When a file is checked out by another user, it is not available for check out.
- When a file is checked out locally by you, you can undo the check-out with the Undo Check-Out command in the context menu. This has the effect of returning the file unchanged to the server.
- If you check out a file in one Altova application, you cannot check it out in another Altova application. The file is considered to be already checked out to you. The available commands at this point in any Altova application supporting Microsoft® SharePoint® Server will be: **Check In** and **Undo Check Out**.

Opening and saving files via Global Resources
To open or save a file via a global resources, click **Switch to Global Resource**. This pops up a dialog in which you can select the global resource. These dialogs are described in the section, Using Global Resources. For a general description of Global Resources, see the Global Resources section in this documentation.

12.1.7 Send by Mail

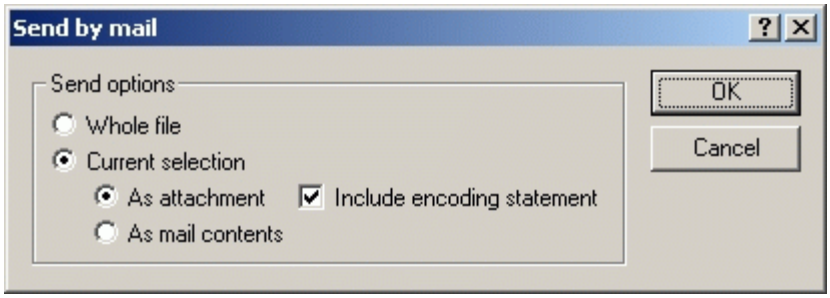
The **Send by Mail...** command lets you send XML document/s or selections from an XML document by e-mail. You can do any of the following:

- Send an XML document as an attachment or as the content of an e-mail.
- Send a selection in an XML document as an attachment or as the content of an e-mail.
- Send a group of files (selected in the Project Window) as an attachment to an e-mail.
- Send a URL (selected in the Project Window) as an attachment or as a link.

Please note: To use this function you must have a MAPI compliant e-mail system.

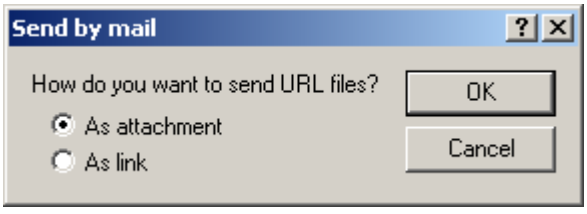
Sending documents and document fragments
To send an XML document:

1. Make that document the active document in the Main Window. If you wish to send a selection or a group of files from a project, make the selection in the document or select the required files in the Project.
2. Click **Send by Mail...**. The following dialog opens:



3. Make the required choices and click **OK**. The selected documents/contents/URLs are attached to the e-mail or content is inserted into the e-mail.

Sending URLs by mail
To send one or more URLs, select the URLs in the Project Window, and click **Send by Mail...**. The following dialog opens:



Select how the URL is to be sent and click **OK**.

 © 2011 Altova GmbH

12.1.8 Print

Ctrl+P

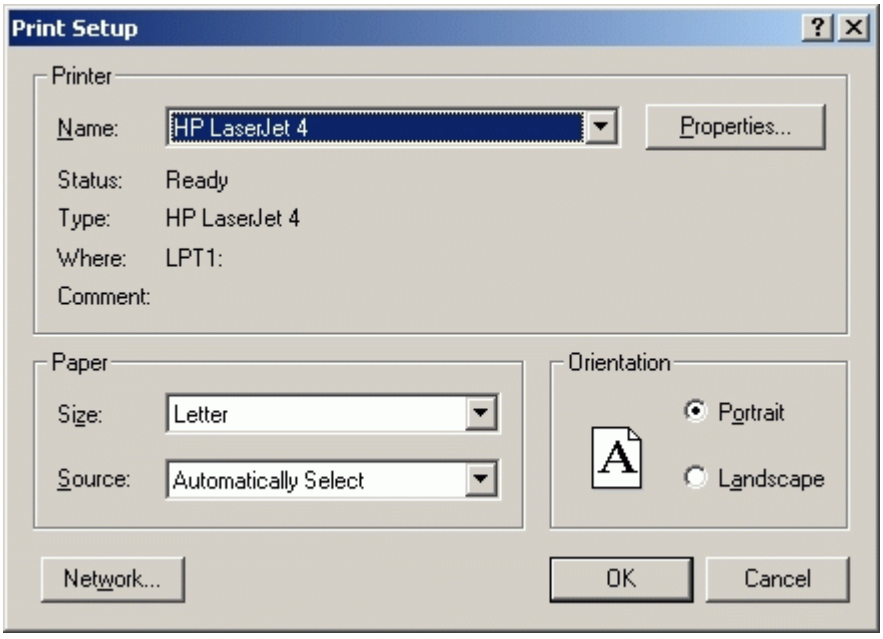
The **Print** command opens the Print dialog box, in which you can select printer options. The currently active document, as seen in the current view, can then be printed.

12.1.9 Print Preview, Print Setup

The **Print Preview** command opens the Print dialog box. Click the **Preview** button to display a print preview of the currently active document. In Print Preview mode, the Print Preview toolbar at top left of the preview window provides print- and preview-related options.

The preview can be magnified or miniaturized using the the **Zoom In** and **Zoom Out** buttons. When the page magnification is such that an entire page length fits in a preview window, then the **One Page / Two Page** button toggles the preview to one or two pages at a time. The **Next Page** and **Previous Page** buttons can be used to navigate among the pages. The toolbar also contains buttons to print all pages and to close the preview window.

The **Print Setup** command, displays the printer-specific Print Setup dialog box, in which you specify such printer settings as paper format and page orientation. These settings are applied to all subsequent print jobs.



The screenshot above shows the Print Setup dialog in which an HP LaserJet 4 printer attached to a parallel port (LPT1) is selected.

Note: To enable background colors and images in Print Preview, do the following: (i) In the **Tools** menu of Internet Explorer, click **Internet Options**, and then click the Advanced tab; (ii) In the Settings box, under Printing, select the *Print background colors and images* check box, and (iii) Then click **OK**.

12.1.10 Recent Files, Exit

The **File** menu displays a list of the nine most recently used files, with the most recently opened file shown at the top of the list. You can open any of these files by clicking its name. To open a file in the list using the keyboard, press **ALT+F** to open the **File** menu, and then press the number of the file you want to open.

The **Exit** command is used to quit XMLSpy. If you have any open files with unsaved changes, you are prompted to save these changes. XMLSpy also saves modifications to program settings and information about the most recently used files.

© 2011 Altova GmbH

12.2 Edit Menu

The **Edit** menu contains commands for editing documents in XMLSpy.

12.2.1 Undo, Redo

The **Undo** (**Ctrl+Z**) command contains support for unlimited levels of Undo. Every action can be undone and it is possible to undo one command after another. The Undo history is retained after using the Save command, enabling you go back to the state the document was in before you saved your changes.

The **Redo** (**Ctrl+Y**) command allows you to redo previously undone commands, thereby giving you a complete history of work completed. You can step back and forward through this history using the Undo and Redo commands.

12.2.2 Cut, Copy, Paste, Delete

The **Cut** (**Shift+Del** or **Ctrl+X**) command copies the selected text or items to the clipboard and deletes them from their present location.

The **Copy** (**Ctrl+C**) command copies the selected text or items to the clipboard. This can be used to duplicate data within XMLSpy or to move data to another application.

The **Paste** (**Ctrl+V**) command inserts the contents of the clipboard at the current cursor position.

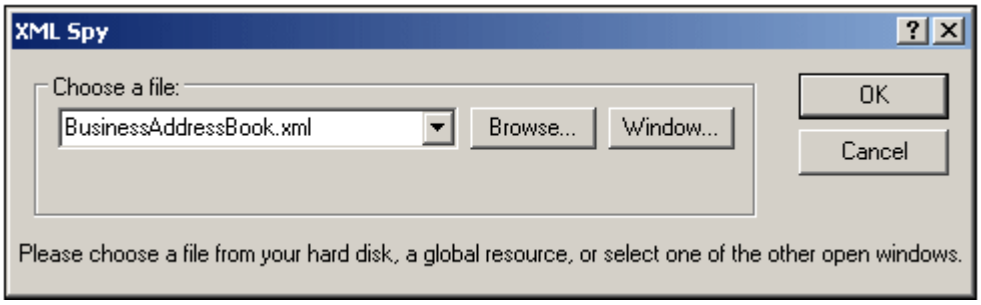
The **Delete** (**Del**) command deletes the currently selected text or items without placing them in the clipboard.

12.2.3 Insert

Mousing over or selecting the **Insert** command rolls out a submenu with three commands, which are described below:

- Insert File Path
- Insert XInclude

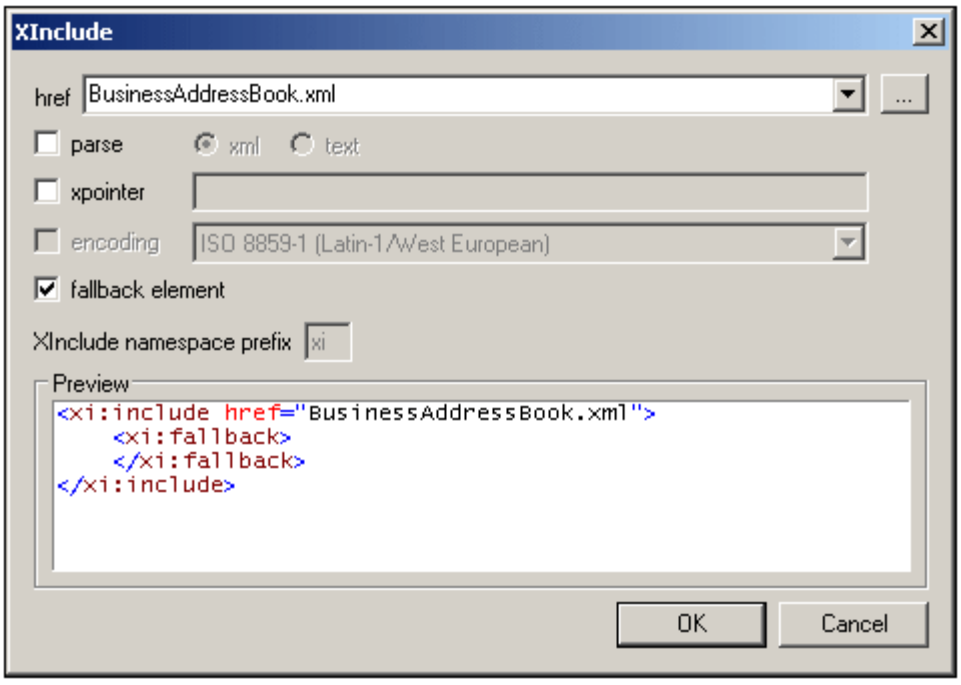
Insert File Path
The **File Path** command is enabled in the Text View and Grid View of documents of any file type. Using it, you can insert the path to a file at the cursor selection point. Clicking the command pops up a dialog (*screenshot below*) in which you select the required file.



The required file can be selected in one of the following ways: (i) by browsing for the file, URL, or global resource (use the **Browse** button); (ii) by selecting the window in which the file is open (the **Window** button). When done, click **OK**. The path to the selected file will be inserted in the active document at the cursor selection point.

Insert XInclude

The **XInclude** command is available in Text View and Grid View, and enables you to insert a new XInclude element at the cursor selection point in Text View, or before the selected item in both Text View and Grid View. If in Grid View the current selection is an attribute, the XInclude element is inserted after the attribute and before the first child element of the attribute's parent element. Selecting this command pops up the XInclude dialog (*screenshot below*).



The XML file to be included is entered in the `href` text box (alternatively, you can browse for the file by clicking the **Browse (...)** button to the right of the text box). The filename will be entered in the XML document as the value of the `href` attribute. The `parse`, `xpointer`, and `encoding` attributes of the XInclude element (`xi:include`), and the `fallback` child element of `xi:include` can also be inserted via the dialog. Do this by first checking the appropriate check box and then selecting/entering the required values. In the case of the `fallback` element, checking its check box only inserts the empty element. The content of the `fallback` element must be added subsequently in one of the editing views.

 © 2011 Altova GmbH

The parse attribute determines whether the included document is to be parsed as XML or text. (XML is the default value and therefore need not be specified.) The xpointer attribute identifies a specific fragment of the document located with the href attribute; it is this fragment that will be included. The encoding attribute specifies the encoding of the included document so that XMLSpy can transcode this document (or the part of it to be included) into the encoding of the including document. The contents of the fallback child element replace the xi:include element if the document to be included cannot be located.

Here is an example of an XML document that uses XInclude to include two XML documents:

```
<?xml version="1.0" encoding="UTF-16"?>
<AddressBook xsi:schemaLocation="http://www.altova.com/sv/myaddresses
AddressBook.xsd"
       xmlns="http://www.altova.com/stylevision/tutorials/myaddresses"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xmlns:xi="http://www.w3.org/2001/XInclude">
       <xi:include href="BusinessAddressBook.xml"/>
       <xi:include href="PersonalAddressBook.xml"/>
</AddressBook>
```

When this XML document is parsed, it will replace the two XInclude elements with the files specified in the respective href attributes.

xml:base

When the XML validator of XMLSpy reads an XML document and encounters the include element in the XInclude namespace (hereafter xi:include), it replaces this element (xi:include) with the XML document named in the href attribute of the xi:include element. The document element (root element) of the included XML document (or the element identified by an XPointer) will be included with an attribute of xml:base in order to preserve the base URIs of the included element. If the resulting XML document (containing the included XML document/s or tree fragment/s) must be valid according to a schema, then the document element of the included document (or the top-level element of the tree fragment) must be created with a content model that allows an attribute of xml:base. If, according to the schema, the xml:base attribute is not allowed on this element, then the resulting document will be invalid. How to define an xml:base attribute in an element's content model using XMLSpy's Schema View is described in the xml: Prefixed Attributes section of the Schema View section of the documentation.

XPointers

XMLSpy supports XPointers in XInclude. The relevant W3C recommendations are the XPointer Framework and XPointer element() Scheme recommendations. The use of an XPointer in an XInclude element enables a specific part of the XML document to be included, instead of the entire XML document. XPointers are used within an XInclude element as follows:

```
       <xi:include href="PersonalAddressBook.xml" xpointer="element(usa)"/>
       <xi:include href="BusinessAddressBook.xml" xpointer="element(/1/1)"/>
       <xi:include href="BobsAddressBook.xml" xpointer="element(usa/3/1)"/>
       <xi:include href="PatsAddressBook.xml" xpointer="
element(usa)element(/1/1)"/>
```

In the element() scheme of XPointer, an NCName or a child sequence directed by integers may be used.

- In the first xi:include element listed above, the xpointer attribute uses the element scheme with an NCName of usa. According to the XPointer Framework, this NCName identifies the element that has an ID of usa.
- In the second xi:include listed above, the xpointer attribute with a value of element

`(/1/1)` identifies, in the first step, the first child element of the document root (which, if the document is well-formed, will be its document (or root) element). In the second step, the first child element of the element located in the previous step is located; in our example, this would be the first child element of the document element.

- The `xpointer` attribute of the third `xi:include` listed above uses a combination of NCName and child sequence. This XPointer locates the first child element of the third child element of the element having an ID of `usa`.
- If you are not sure whether your first XPointer will work, you can back it up with a second one as shown in the fourth `xi:include` listed above: `xpointer="element(usa)element(/1/1)"`. Here, if there is no element with an ID of `usa`, the back-up XPointer specifies that the first child element of the document element is to be selected. Additional backups are also allowed. Individual XPointers may not be separated, or they may be separated by whitespace: for example, `xpointer="element(usa)element(addresses/1) element(/1/1)"`.

Note: The namespace binding context is not used in the element() scheme because the element() scheme does not support qualified names.

12.2.4 Copy XPath

The **Copy XPath** command is available in Text View and Grid View, and creates an XPath expression that selects the currently selected node/s and copies the expression to the clipboard. This enables you to paste the expression into a document (for example, in an XSLT document). All expressions start from the document root.

The XPath expression is resolved differently in Grid View and Text View. In Grid View, if a single element is highlighted, the XPath expression will select not that specific element but all elements of that name at that hierarchical level of the document. In Text View that specific element is selected. For example, if an element called `LastName` of the third `Person` element of the second `Company` element is selected, the XPath expressions would be as follows:

- Grid View: `/Companies/Company/Person/LastName`
- Text View: `/Companies/Company[2]/Person[3]/LastName`

Note: In Grid View the **Copy XPath** command can also be accessed via the context menu.

12.2.5 Copy XPointer

The **Copy XPointer** command is available in Text View and Grid View. It creates an element() scheme XPointer for the currently selected node/s and copies it to the clipboard. This enables you to paste the XPointer into a document (for example, in the `xpointer` attribute of an XInclude element in an XML document).

The element() scheme of XPointer returns results in the form `element(/1/3)`, which selects the third child of the document element (or root element). You should note the following points:

- Attributes cannot be represented using the element() scheme. If an attribute is selected, the following happens: In Grid View, the **Copy XPointer** command is disabled; in Text View, the XPointer of the element "parent" of that attribute is generated.
- Multiple elements cannot be selected. If selected in Grid View, the **Copy XPointer** command is disabled. In Text View, the XPointer of the parent element of the selection is generated.

Note: In Grid View the **Copy XPointer** command can also be accessed via the context menu.

 © 2011 Altova GmbH

12.2.6 Pretty-Print XML Text

The **Pretty-Print XML Text** command reformats your XML document in Text View. Two formatting options are available, depending upon whether the *Use Indentation* check box in the View tab of the Options dialog (**Tools | Options**) is checked or not:

- *Use Indentation* checked: The document is reformatted to give a structured display, indenting each deeper level in the hierarchy by an additional amount of the specified indentation space. This enables a clearer view of the document structure.
- *Use Indentation* unchecked: The document is reformatted so that each new line is left-aligned.

To set up a structured, indented view of the XML document, do the following:

1. In the View tab of the Options dialog (**Tools | Options**), check the *Use Indentation* check box.
2. In the Text View Settings dialog (**View | Text View Settings**), set the tab size you want for the indentation of the pretty-printed text.
3. In the File tab of the Options dialog (**Tools | Options**), enter the elements for which no output formatting (indentation) is wanted.
4. Click the **Pretty-Print XML Text** command (this command).

To reformat the document so that all lines are left-aligned, uncheck the *Use Indentation* check box.

Note the following points:

1. The XML document must be well-formed for this command to work.
2. Pretty-printing adds spaces or tabs to the document when the document is saved.

12.2.7 Select All

Ctrl+A

The **Select All** command selects the contents of the entire document.

12.2.8 Find, Find Next

The **Find** command (**Ctrl+F**) pops up the Find dialog, in which you can specify the string you want to find and other options for the search. To find text, enter the text in the Find What text box or use the combo box to select from one of the last 10 search criteria, and then specify the options for the search.

The **Find Next** command (**F3**) repeats the last Find command to search for the next occurrence of the requested text.

12.2.9 Replace

Ctrl+H

The **Replace** command enables you to find and replace one text string with another text string. It features the same options as the Find... command. You can replace each item individually, or you can use the **Replace All** button to perform a global search-and-replace operation.

12.2.10 Bookmark Commands

Insert/Remove Bookmark

The **Insert/Remove Bookmark** command (**Ctrl+F2**) inserts a bookmark at the current cursor position, or removes the bookmark if the cursor is in a line that has been bookmarked previously. This command is only available in Text View.

Bookmarked lines are displayed in one of the following ways:

- If the bookmarks margin has been enabled, then a solid blue ellipse appears to the left of the text in the bookmark margin.
- If the bookmarks margin has not been enabled, then the complete line containing the cursor is highlighted.

The **F2** key cycles through all the bookmarks in the document.

Remove All Bookmarks

The **Remove All Bookmarks** command (**Ctrl+Shift+F2**) removes all the currently defined bookmarks. This command is only available in Text View. Note that the **Undo** command does not undo the effects of this command.

Goto Next Bookmark

The **Goto Next Bookmark** command (**F2**) places the text cursor at the beginning of the next bookmarked line. This command is only available in the Text View.

Goto Previous Bookmark

The **Goto Previous Bookmark** command (**Shift+F2**) places the text cursor at the beginning of the previous bookmarked line. This command is only available in the Text View.

12.2.11 Comment In/Out

The **Comment In/Out** command is available in Text View and is used to comment and uncomment XML text fragments. Text in an XML document can be commented out using the XML start-comment and end-comment delimiters, respectively <!-- and -->. In XMLSpy, these comment delimiters can be easily inserted using the **Comment In/Out** menu command.

To comment out a block of text, select the text to be commented out and then select the command **Comment In/Out**, either from the **Edit** menu or the context menu that you get on right-clicking the selected text. The commented text will be grayed out (*see screenshot below*).

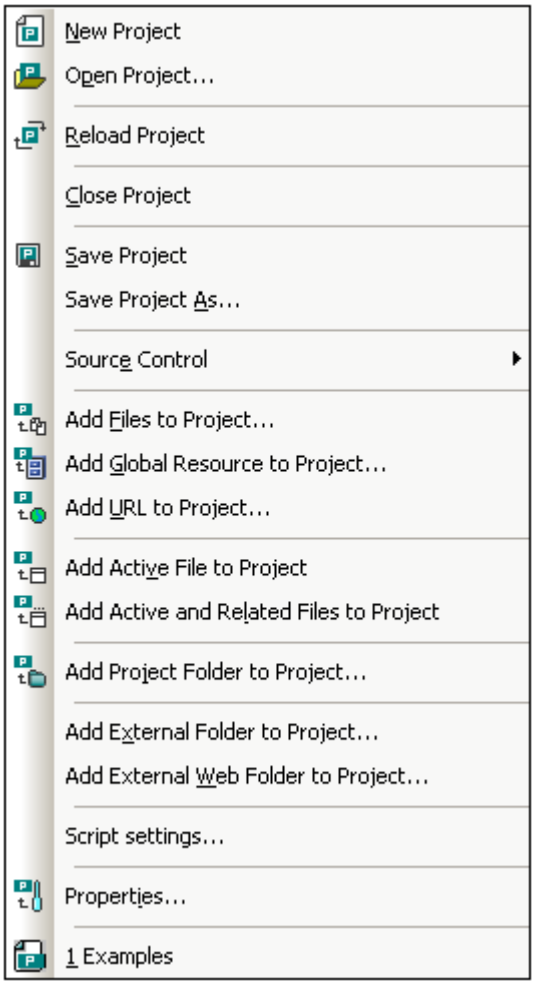
 © 2011 Altova GmbH

```
<Department>
    <Name>Administration</Name>
    <Person>
    <Person>
    <Person>
    <!--<Person>
        <First
        <Last></Last>
        <PhoneExt></PhoneExt>
        <EMail></EMail>
        <LeaveTotal></LeaveTotal>
        <LeaveUsed></LeaveUsed>
        <LeaveLeft></LeaveLeft>
    </Person>-->
</Department>
```

To uncomment a commented block of text, select the commented block **excluding** the comment delimiters, and select the command **Comment In/Out**, either from the **Edit** menu or the context menu that you get on right-clicking the selected text. The comment delimiters will be removed and the text will no longer be grayed out.

12.3 Project Menu

uses the familiar tree view to manage multiple files or URLs in XML projects. Files and URLs can be grouped into folders by common extension or any arbitrary criteria, allowing for easy structuring and batch manipulation.



Please note: Most project-related commands are also available in the context menu, which appears when you right-click any item in the project window.

Absolute and relative paths
Each project is saved as a project file, and has the **.spp** extension. These files are actually XML documents that you can edit like any regular XML File. In the project file, absolute paths are used for files/folders on the same level or higher, and relative paths for files/folders in the current folder or in sub-folders. For example, if your directory structure looks like this:

```
| -Folder1
|      |
|      | -Folder2
|             |
|             | -Folder3
|                    |
|                    | -Folder4
```

 © 2011 Altova GmbH

If your .spp file is located in Folder3, then references to files in Folder1 and Folder2 will look something like this:

```
c:\Folder1\NameOfFile.ext
c:\Folder1\Folder2\NameOfFile.ext
```

References to files in Folder3 and Folder4 will look something like this:

```
.\NameOfFile.ext
.\Folder4\NameOfFile.ext
```

If you wish to ensure that all paths will be relative, save the .spp files in the root directory of your working disk.

Drag-and-drop
In the Project window, a folder can be dragged to another folder or to another location within the same folder. A file can be dragged to another folder, but cannot be moved within the same folder (within which files are arranged alphabetically). Additionally, files and folders can be dragged from Windows File Explorer to the Project window.

Global resources in the context menu
When you right-click a folder in the Project window, in the context menu that appears, you can select the **Add Global Resource** menu item to add a global resource. The menu command itself pops up the Choose Global Resource dialog, which lists all the file-type and folder-type global resources in the currently active Global Resources XML File. Select the required global resource, and it will be added to the selected project folder.

Projects and source control providers
If you intend to add an project to a source control repository, please ensure that the project files position in the hierarchical file system structure is one which enables you to add files only from below it (taking the root directory to be the top of the directory tree).

In other words, the directory where the **project file** is located, essentially represents the **root directory** of the project within the source control repository. Files added from above it (the project root directory) will be added to the project, but their location in the repository may be an unexpected one—if they are allowed to be placed there at all.

For example, given the directory structure show above, if a project file is saved in Folder3 and placed under source control:

- Files added to Folder1 may not be placed under source control,
- Files added to Folder2 are added to the root directory of the repository, instead of to the project folder, but are still under source control,
- Files located in Folder3 and Folder4 work as expected, and are placed under source control.

12.3.1 New Project

The **New Project** command creates a **new** project in XMLSpy. If you are currently working with another project, a prompt appears asking if you want to close all documents belonging to the current project.

12.3.2 Open Project

The **Open Project...** command opens an existing project in XMLSpy. If you are currently working with another project, the previous project is closed first.

12.3.3 Reload Project

The **Reload Project** command reloads the current project from disk. If you are working in a multi-user environment, it can sometimes become necessary to reload the project from disk, because other users might have made changes to the project.

Please note: Project files (. `spp` files) are actually XML documents that you can edit like any regular XML File.

12.3.4 Close Project

The **Close Project** command **closes** the active project. If the project has been modified, you will be asked whether you want to save the project first. When a project is modified in any way, an asterisk is added to the project name in the Project Window.

12.3.5 Save Project, Save Project As

The **Save Project** command **saves** the current project. You can also save a project by making the project window active and clicking the icon.

The **Save Project As** command **saves** the current project with a new name that you can enter when prompted for one.

12.3.6 Source Control

XMLSpy supports Microsoft SourceSafe and other compatible repositories. The Source Control Systems supported by XMLSpy are listed in the section Supported Source Control Systems. How to install theses systems is described in the section, Installing Source Control Systems. This section describes the commands in the **Project | Source Control** submenu, which are used to work with the Source Control System from within XMLSpy.

Overview of the Source Control feature
The mechanism for placing files in a XMLSpy project under source control is as follows:

1. In XMLSpy, an application project folder containing the files to be placed under source control is created. Typically, the application project folder will correspond to a local

 © 2011 Altova GmbH

folder in which the project files are located. The path to the local folder is referred to as the local path.
2. In the source control system's database (also referred to as source control or repository), a folder is created that will contain the files to be placed under source control.
3. Application project files are added to source control using the command **Project | Source Control | Add to Source Control**.
4. Source control actions, such as checking in to, checking out from, and removing files from source control, can be carried out by using the commands in the **Project | Source Control** submenu. The commands in this submenu are listed in the sub-sections of this section.

Note: If you wish to change the current source control provider, this can be done in any of two ways: (i) via the Source Control options (**Tools | Options | Source Control**), or (ii) in the Change Source Control dialog (**Project | Source Control | Change Source Control**).

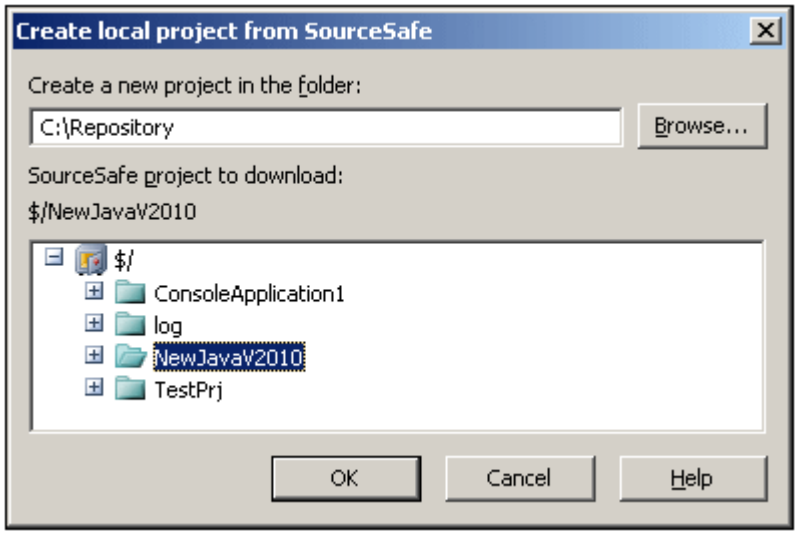
Note: Note that a Source Control project is not the same as an XMLSpy project. Source Control projects are directory-dependent, while XMLSpy projects are logical constructions without direct directory dependence.

For additional information, see the section, Source Control.

Open from Source Control

The **Open from Source Control** command creates an existing source control project locally. This command is enabled only if an XMLSpy project has previously been added to source control using the menu option Add to Source Control.

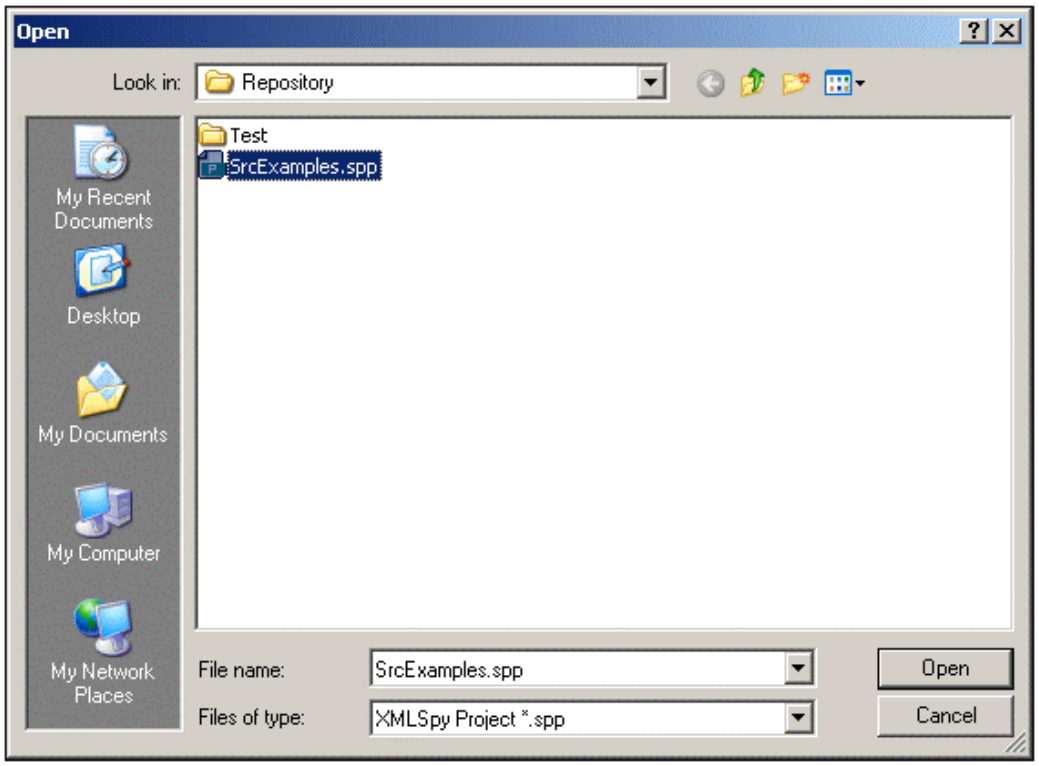
1. Select **Project | Source Control | Open from Source Control**. Enter your login details in the Login dialog that appears.
2. The Create Local Project from SourceSafe dialog box appears (*screenshot below*). Select the folder to contain the local project. This folder becomes the Working Folder or Checkout Folder (in the screenshot below: `C:\Repository`).



3. Select the source control project you want to download. In the screenshot above, the source control project folder, `NewJavaV2010`, has been selected. Files in the source control project folder will be downloaded to the local folder. If the local folder you have

specified does not exist, a dialog box opens prompting you to create it. Click **Yes** to confirm the new folder.

4. The Open dialog now pops up (*screenshot below*).



5. Click the application project file you wish to create and click **Open**. The selected application project will open in XMLSpy, and the file is placed under source control.

Source control symbols
Folders and files display certain symbols, the meanings of which are given below.

	Checked in. Available for check-out.
	Checked out by another user. Not available for check-out.
	Checked out locally. Can be edited and checked-in.

Enable Source Control

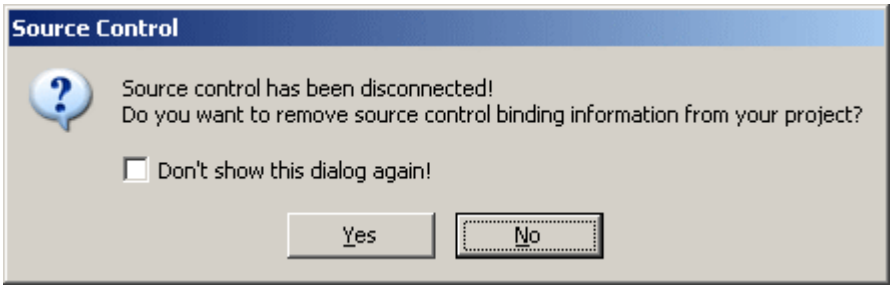
The **Enable Source Control** command allows you to enable or disable source control for a XMLSpy project. Selecting this option on any file or folder, enables/disables source control for the whole project.

Enabling Source Control for a project
To enable source control for a project, do the following:

1. Click on any file or folder in the Project window.
2. Select the menu option **Project | Source Control | Enable Source Code Control**. The previous check in/out status of the various files are retrieved and displayed in the Project window.

 © 2011 Altova GmbH

Disabling Source Control for a project
To disable source control for a project, select the menu option **Project | Source Control | Enable Source Control**. You are now prompted if you want to remove the binding information from the project (*see screenshot below*).



To provisionally disable source control for the project select **No**. To permanently disable source control for the project select **Yes**.

Get Latest Version

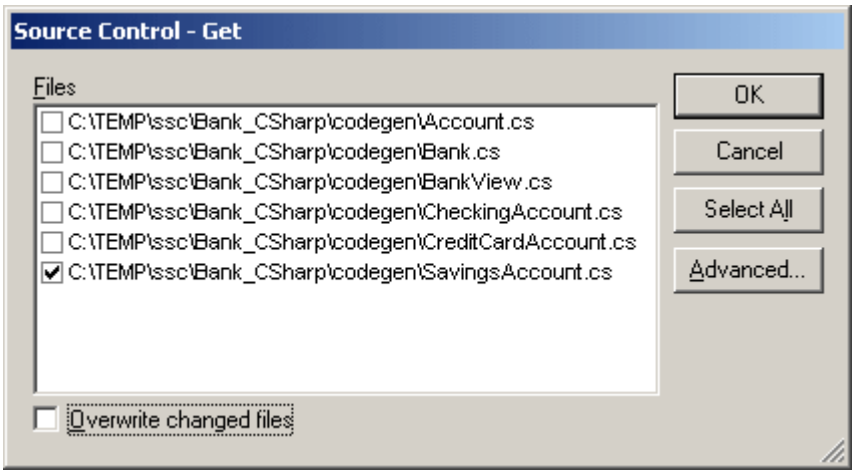
The **Get Latest Version** command retrieves and places the latest source control version of the selected file(s) in the working directory. The files are retrieved as read-only and are not checked out. No further options are available when using this command.

To get the latest version of a file, do the following:

1. Select the file(s) you want to get the latest version of in the Model Tree.
2. Select **Project** | **Source Control** | **Get Latest Version**.

Get

The **Get** command retrieves read-only copies of the selected files and places them in the working folder. By default, the files are not checked-out for editing. To get a file, select it in the Project window (multiple files can be selected) and then select the **Get** command. This pops up the Get dialog (*screenshot below*). Check the files you want to get.



Overwrite changed files check box
Overwrites those files that have been changed locally with those from the source control database.

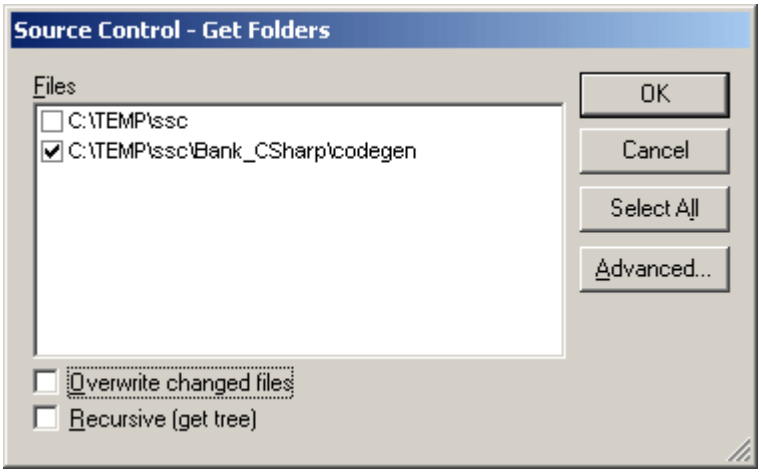
Select All
Selects all the files in the list box.

Advanced
Allows you to define the *Replace writable* and *Set timestamp* options in the respective combo boxes.

The *Make writable* check box removes the read-only attribute of the retrieved files.

Get Folders

The **Get Folders** command retrieves read-only copies of files in the selected folders and places them in the working folder. By default, the files are not checked-out for editing. To get a folder, s elect it in the Project window and then select the **Get Folders** command. This pops up the Get Folders dialog (*screenshot below*). Check the foldersyou want to get.



Overwrite changed files check box
Overwrites those files that have been changed locally with those from the source control database.

Recursive (get tree) check box
Retrieves all files of the folder tree below the selected folder.

 © 2011 Altova GmbH

Select All
Selects all the files in the list box.

Advanced
Allows you to define the *Replace writable* and *Set timestamp* options in the respective combo boxes.

The *Make writable* check box removes the read-only attribute of the retrieved files.

Check Out

The **Check Out** command checks out the latest version of the selected files and places writable copies in the working directory. The files are flagged as checked out for all other users. To check out files, do the following:

1. Select the file or folder you want to check out in the Model Tree.
2. Select **Project | Source Control | Check Out**.
3. In the Check Out dialog that pops up, select the files to check out, then click **OK**.

Note the following points:

- You can change the number of files to check out by activating the individual check boxes in the Files list box.
- The *Checkout local version* option checks out only the local versions of files, not those from the source control database.
- The following items can be checked out: (i) single files (in the Project window, click the required file to select it; use **Ctrl+Click** to select multiple files, use **Ctrl+Click**); (ii) folders (in the Project window, click the required folder to select it; use **Ctrl+Click** to select multiple folders).
- Checked out files and folders are indicated with a red check mark.

Advanced
Allows you to define the *Replace writable* and *Set timestamp* options in the respective combo boxes.

The *Make writable* check box removes the read-only attribute of the retrieved files.

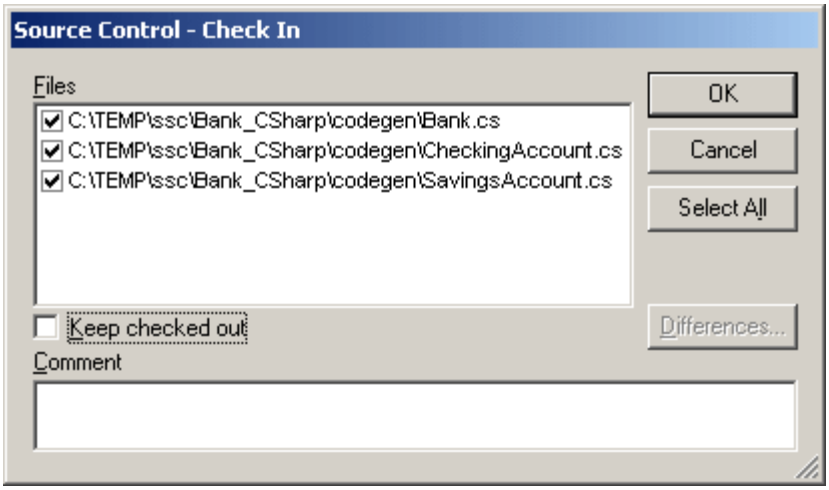
Source control symbols
Folders and files display certain symbols, the meanings of which are given below.

	Checked in. Available for check-out.
	Checked out by another user. Not available for check-out.
	Checked out locally. Can be edited and checked-in.

Check In

The Check In command checks in previously checked out files (that is, your locally updated files) and places them in the source control database. To check in files, do the following:

1. Select the files in the Model Tree.
2. Select **Project | Source Control | Check In**.
3. In the Check In dialog that pops up, select the files to check in, then click **OK**.



Note the following points:

- As a shortcut for the Check In command, right-click a checked out item in the project window and select **Check In** from the context menu.
- The following items can be checked in: (i) single files (in the Project window, click the required file to select it; use **Ctrl+Click** to select multiple files, use **Ctrl+Click**); (ii) folders (in the Project window, click the required folder to select it; use **Ctrl+Click** to select multiple folders).
- The lock symbol denotes that the file/folder is under source control, but is currently not checked out.
- The **Differences** button is enabled when a line in the Files pane is selected. Clicking it enables you to see differences between two file versions.

Source control symbols
Folders and files display certain symbols, the meanings of which are given below.

	Checked in. Available for check-out.

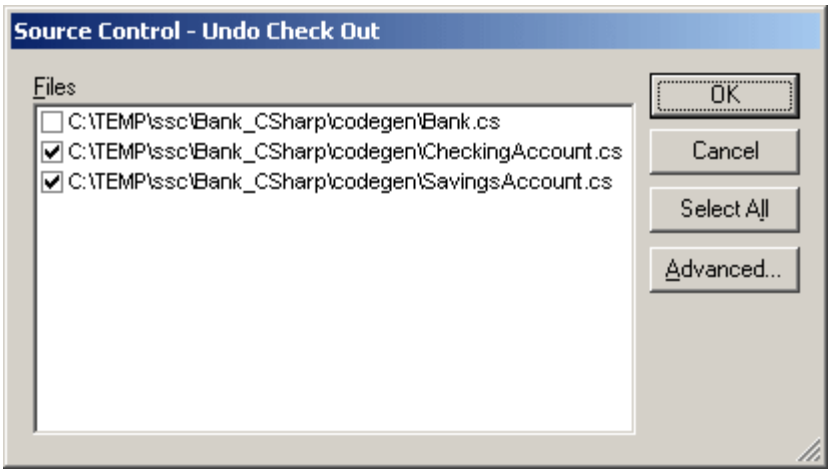
 © 2011 Altova GmbH

	Checked out by another user. Not available for check-out.
	Checked out locally. Can be edited and checked-in.

Undo Check Out

The **Undo Check Out** command rejects changes made to previously checked out files (that is, your locally updated files) and retains the old files from the source control database. To undo a check out, do the following:

1. Select the files in the Project window.
2. Select **Project | Source Control | Undo Check Out**.
3. In the Undo Check Out dialog that pops up, select the files for which check out should be undone, then click **OK**.



Check out of the following items can be undone: (i) single files (in the Project window, click the required file to select it; use **Ctrl+Click** to select multiple files, use **Ctrl+Click**); (ii) folders (in the Project window, click the required folder to select it; use **Ctrl+Click** to select multiple folders).

Advanced

Allows you to define the *Replace writable* and *Set timestamp* options in the respective combo boxes.

The *Make writable* check box removes the read-only attribute of the retrieved files.

Source control symbols

Folders and files display certain symbols, the meanings of which are given below.

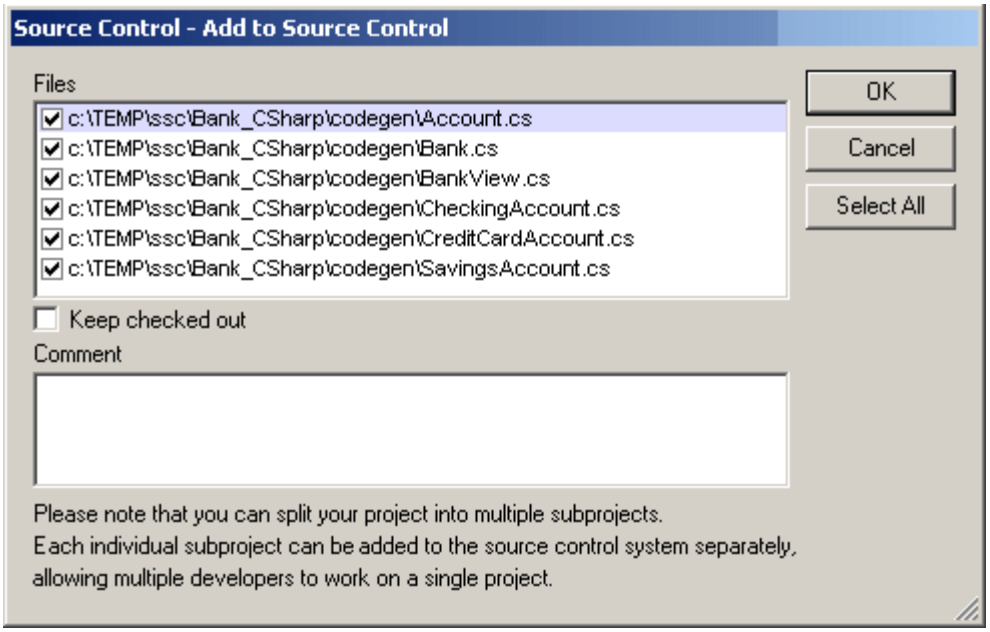
	Checked in. Available for check-out.

	Checked out by another user. Not available for check-out.
	Checked out locally. Can be edited and checked-in.

Add to Source Control

The **Add to Source Control** command adds the selected files in a project folder to the source control database and places them under source control. To add files to source control, do the following:

1. In the Project window, click a file and select the menu option **Project | Source Control | Add to Source Control**. If you select a folder, the files in that folder will be added to source control.
2. In the Add to Source Control dialog that pops up, ensure that the files to be added are checked. If the file to be added should be kept checked out, check the *Keep checked out* check box. Then click **OK**.



The lock symbol now appears next to each of the files placed under source control. If the files are checked out they will be shown with a red check symbol.

Source control symbols
Folders and files display certain symbols, the meanings of which are given below.

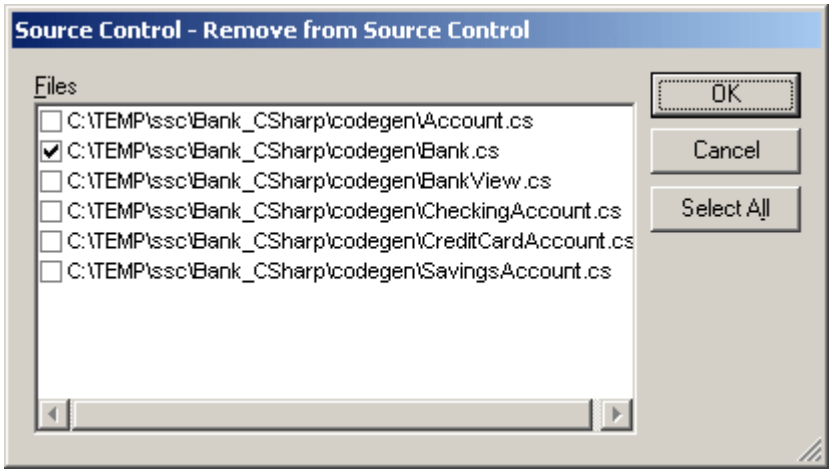
	Checked in. Available for check-out.
	Checked out by another user. Not available for check-out.
	Checked out locally. Can be edited and checked-in.

© 2011 Altova GmbH

Remove from Source Control

The **Remove from Source Control** command removes previously added files from the source control database. These files will be visible in the Project window but cannot be checked in or out. Use the **Add to Source Control** command to place them back under source control. To remove files from the source control provider, do the following:

1. In the Project window, select the files you wish to remove.
2. Select **Project | Source Control | Remove from Source Control**.
3. In the Remove from Source Control dialog that pops up, select the files to remove, then click **OK**.

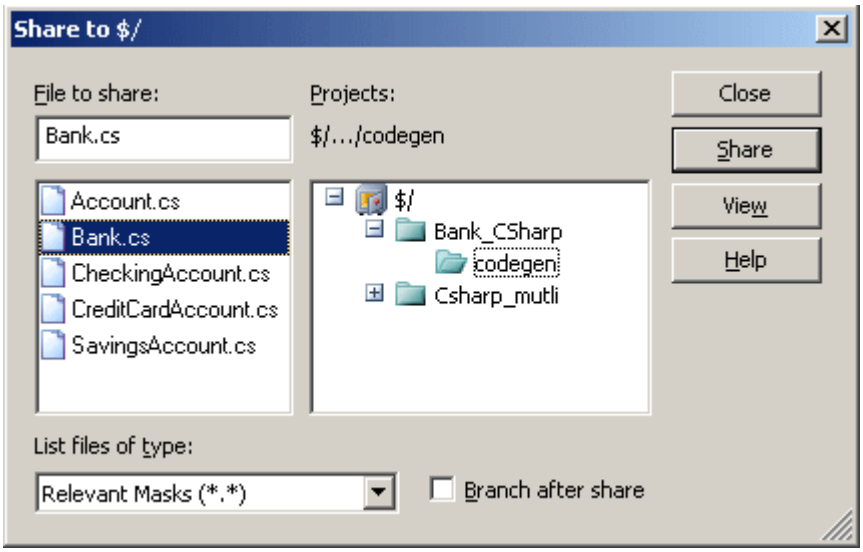


The following items can be removed from source control: (i) single files (in the Project window, click the required file to select it; use **Ctrl+Click** to select multiple files, use **Ctrl+Click**); (ii) folders (in the Project window, click the required folder to select it; use **Ctrl+Click** to select multiple folders).

Share from Source Control

To use the **Share from Source Control** command you must have the Check in/out rights to the project you are sharing from. To share a file from source control, do the following:

1. In the Project window, select the file you want to share and select **Project | Source Control | Share from Source Control**.
2. In the *Projects* list, select the project folder that contains the file you want to share.



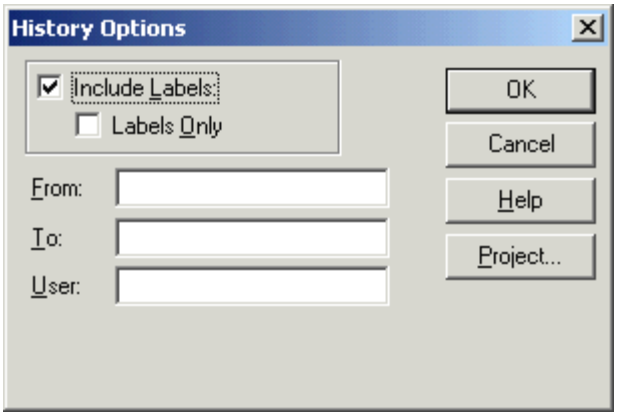
3. In the *Files to share* list box, select the file you want to share and click the **Share** button. The file will be removed from the *Files to share* list.
4. Click the **Close** button to continue.

Branch after share
The *Branch After Share* option shares the file and creates a new branch to create a separate version.

Show History

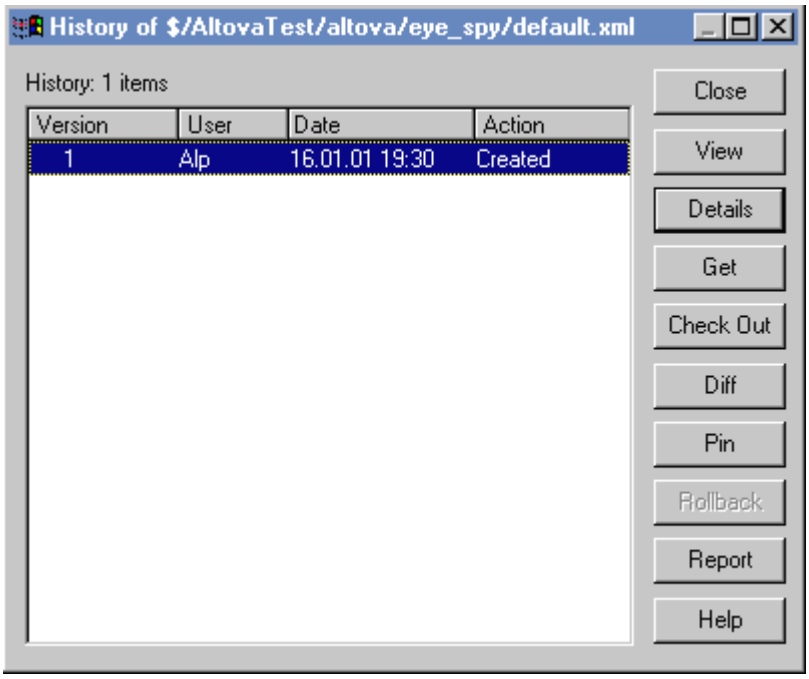
The **Show History** command displays the history of a file under source control and allows you to view detailed history info of previous versions of a file, view differences and retrieve previous versions of the file. To show the history of a file, do the following:

1. Click on the file in the Project window.
2. Select the menu option **Project | Source control | Show History**. A dialog box prompting for more information may appear (this example uses Visual Source-Safe).



3. Select the appropriate entries and confirm with **OK**.

 © 2011 Altova GmbH



This dialog box provides various way of comparing and getting specific versions of the file in question. Double-clicking an entry in the list opens the History Details dialog box for that file. The buttons in the dialog provide the following functionality:

- ***Close:*** closes this dialog box.
- ***View:*** opens a further dialog box in which you can select the type of viewer with which you wish to see the file.
- ***Details:*** opens a dialog box in which you can see the properties of the currently active file.
- ***Get:*** allows you to retrieve one of the previous versions of the file in the version list and place it in the working directory.
- ***Check Out:*** allows you to check out a previous version of the file.
- ***Diff:*** opens the Difference options dialog box, which allows you to define the difference options when viewing the differences between two file versions. Use **CTRL+Click** to mark two file versions in this window, then click Diff to view the differences between them.
- ***Pin:*** pins or unpins a version of the file, allowing you to define the specific file version to use when differencing two files.
- ***Rollback:*** rolls back to the selected version of the file.
- ***Report:*** Generates a history report which you can send to the printer, file, or clipboard.
- ***Help:*** opens the online help of the source control provider plugin.

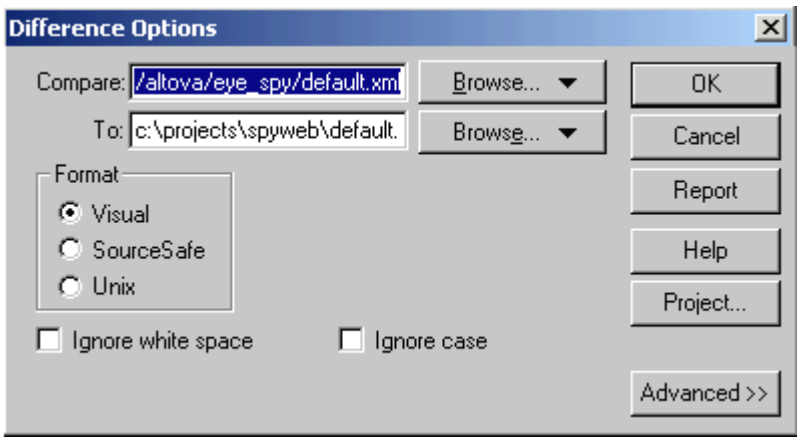
Show Differences

The **Show Differences** command displays the differences between the file currently in the source control repository and the **checked in/out** file of the same name in the working directory. If you have "pinned" one of the files in the history dialog box, then the pinned file will be used in the "Compare" text box. Any two files can be selected using the Browse buttons.

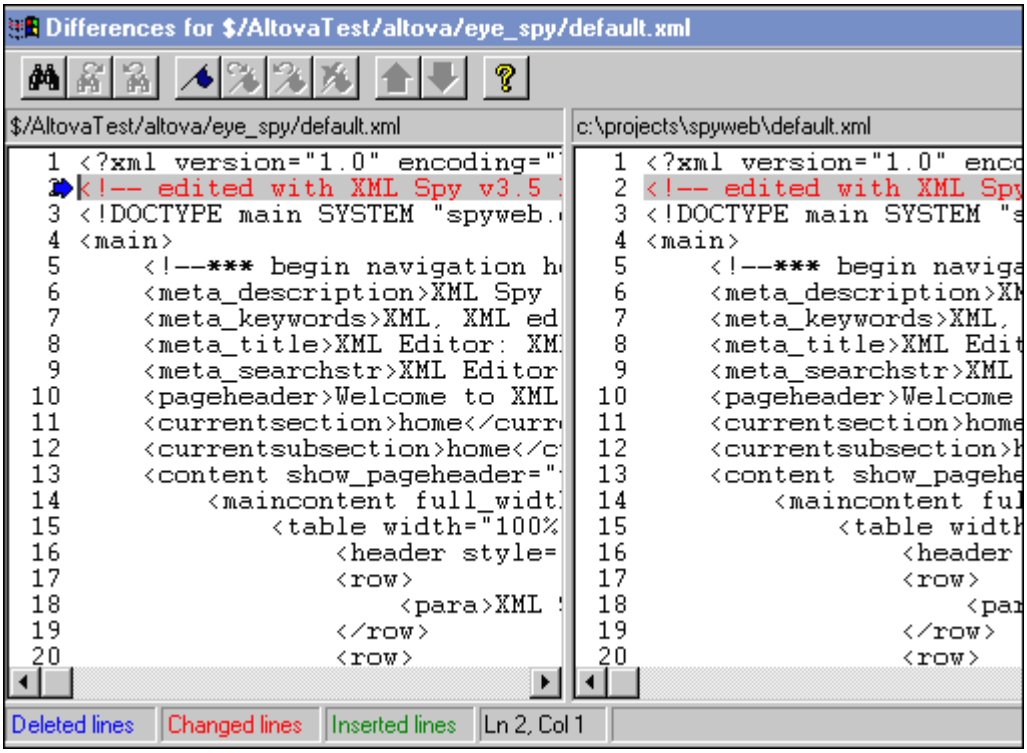
To show the differences between two files, do the following:

1. Check out a file from your project. Click on the file in the project window.

2. Select the menu option **Project | Source control | Show Differences**. A dialog box prompting for more information may appear at this time.



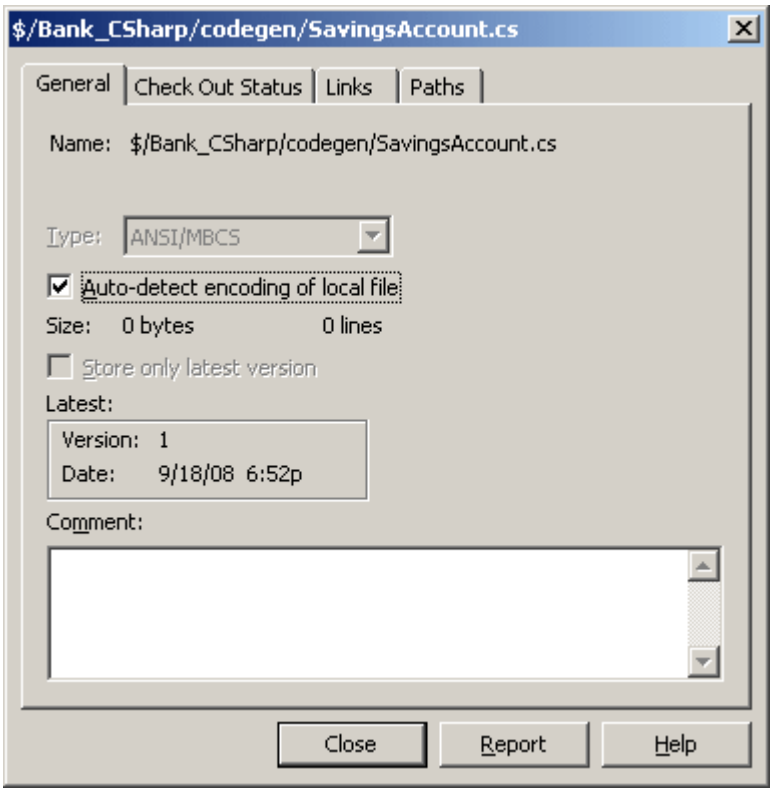
3. Select the appropriate entries and confirm with **OK**.



The differences between the two files are highlighted in both windows (this example uses MS Source-Safe).

Show Properties

The **Show Properties** command displays the properties of the currently selected file (*screenshot below*). The properties displayed depends on the source control provider you use.



Note that this command can only be used on single files.

Refresh Status

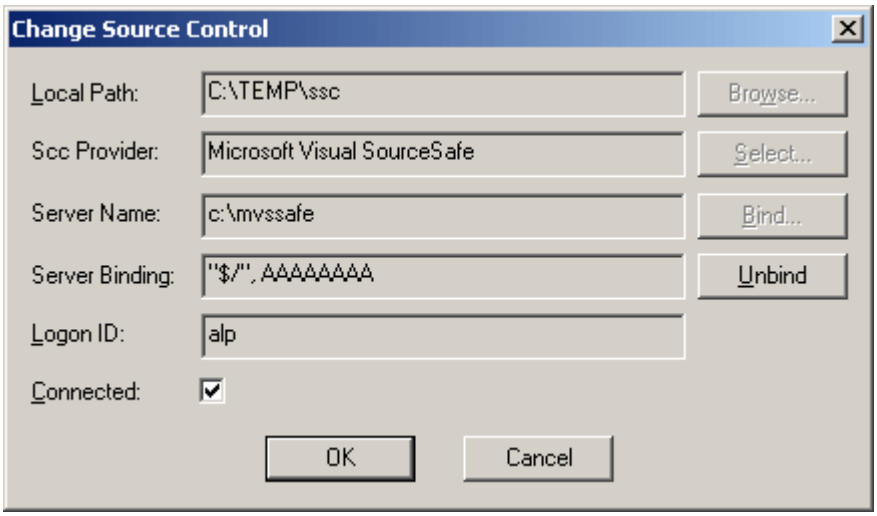
The **Refresh Status** command refreshes the status of all project files independent of their current status.

Source Control Manager

The **Source Control Manager** command starts your source control software with its native user interface.

Change Source Control

The **Change Source Control** command pops up the Change Source Control dialog box (*screenshot below*), which enables you to change the source control provider that you are using. Click the **Unbind** button first, then click the **Select** button to select the new source control provider.



After you have selected the source control provider, you will need to create a binding between the local folder and the folder on the source control server. The local folder is that entered in the Local Path text box. To select a folder on the source control server, click the **Bind** button. This enables you to log on to the server and then navigate to the required folder on the source control server. Click **OK** when done.

12.3.7 Add Files to Project

The **Project** | **Add Files to Project** command adds files to the current project. Use this command to add files to any folder in your project. You can either select a single file or any group of files (using **Ctrl+ click**) in the Open dialog box. If you are adding files to the project, they will be distributed among the respective folders based on the File Type Extensions defined in the Project Properties dialog box.

12.3.8 Add Global Resource to Project

The **Project** | **Add Global Resource to Project** command pops up the Choose Global Resource dialog, in which you can select a global resource of file or folder type to add to the project. If a file-type global resource is selected, then the file is added to the appropriate folder based on the File Type Extensions defined in the Project Properties dialog box. If a folder-type global resource is selected, that folder will be opened in a file-open dialog and you will be prompted to select a file; the selected file is added to the appropriate folder based on the File Type Extensions defined in the Project Properties dialog box. For a description of global resources, see the Global Resources section in this documentation.

12.3.9 Add URL to Project

The **Project** | **Add URL to Project** command adds a URL to the current project. URLs in a project cause the target object of the URL to be included in the project. Whenever a batch operation is performed on a URL or on a folder that contains a URL object, XMLSpy retrieves

the document from the URL, and performs the requested operation.

12.3.10 Add Active File to Project

The **Project** | **Add Active File to Project** command adds the active file to the current project. If you have just opened a file from your hard disk or through an URL, you can add the file to the current project using this command.

12.3.11 Add Active And Related Files to Project

The **Project** | **Add Active and Related Files to Project** command adds the currently active XML document and all related files to the project. When working on an XML document that is based on a DTD or Schema, this command adds not only the XML document but also all related files (for example, the DTD and all external parsed entities to which the DTD refers) to the current project.

Please note: Files referenced by processing instructions (such as XSLT files) are not considered to be related files.

12.3.12 Add Project Folder to Project

The **Project** | **Add Project Folder to Project** command adds a new folder to the current project. Use this command to add a new folder to the current project or a sub-folder to a project folder. You can also access this command from the context-menu when you right-click on a folder in the project window.

Note: A project folder can be dragged and dropped into another project folder or to any other location in the project. Also, a folder can be dragged from Windows (File) Explorer and dropped into any project folder.

Note: Project folders are green, while external folders are yellow.

12.3.13 Add External Folder to Project

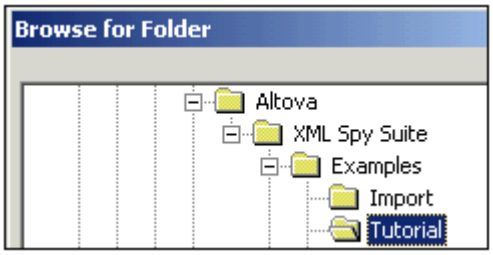
The **Project** | **Add External Folder to Project** command adds a new external folder to the current project. Use this command to add a local or network folder to the current project. You can also access this command from the context-menu when you right-click a folder in the project window.

Note: External folders are yellow, while project folders are green.

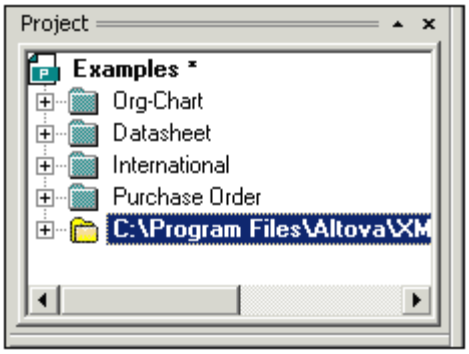
Note: Files contained in external folders cannot be placed under source control.

Adding external folders to projects
To add an external folder to the project:

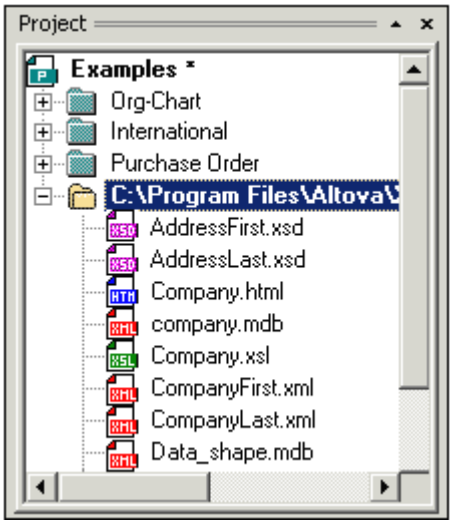
1. Select the menu option **Project | Add External Folder to Project**.
2. Select the folder you want to include from the Browse for Folder dialog box, and click **OK** to confirm.



The selected folder now appears in the project window.



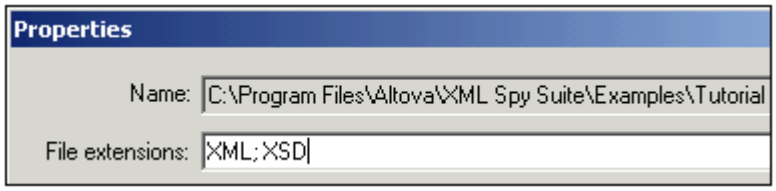
3. Click the plus icon to view the folder contents.



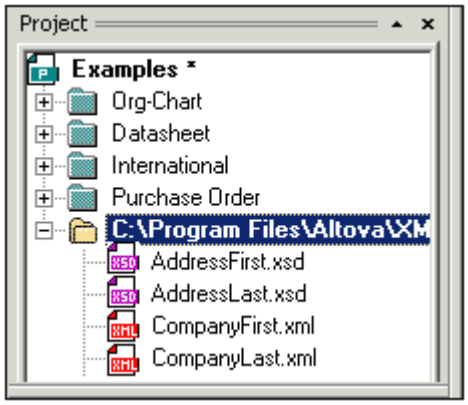
Filtering contents of folders
To filter the contents of the folder:

1. Right-click the local folder, and select the popup menu option **Properties**. This opens the Properties dialog box.

 © 2011 Altova GmbH



2. Click in the **File extensions** field and enter the file extensions of the file types you want to see. You can separate each file type with a **semicolon** to define multiple types (XML and Schema XSDs in this example).
3. Click **OK** to confirm.



The Project window now only shows the XML and XSD files of the tutorial folder.

Validating external folders
To validate and check an external folder for well-formedness:

1. Select the file types you want to see or check from the external folder,
2. Click the folder and click the **Check well-formedness** or **Validate** icon (hotkeys **F7** or **F8**). All the files visible under the folder are checked. If a file is malformed or invalid, then this file is opened in the Main Window, allowing you to edit it.
3. Correct the error and run the validation process once more to recheck.

Updating a project folder
You might add or delete files in the local or network directory at any time. To update the folder view, right-click the external folder, and select the popup menu option **Refresh external folder**.

Deleting external folders and files in them
Select an external folder and press the **Delete** key to delete the folder from the Project window. Alternatively, right-click the external folder and select the **Delete** command. Each of these actions only deletes the external folder from the Project window. The external folder is not deleted from the hard disk or network.

To delete a file in an external folder, you have to delete it physically from the hard disk or network. To see the change in the project, refresh the external folder contents (right-click the external folder and select **Refresh**).

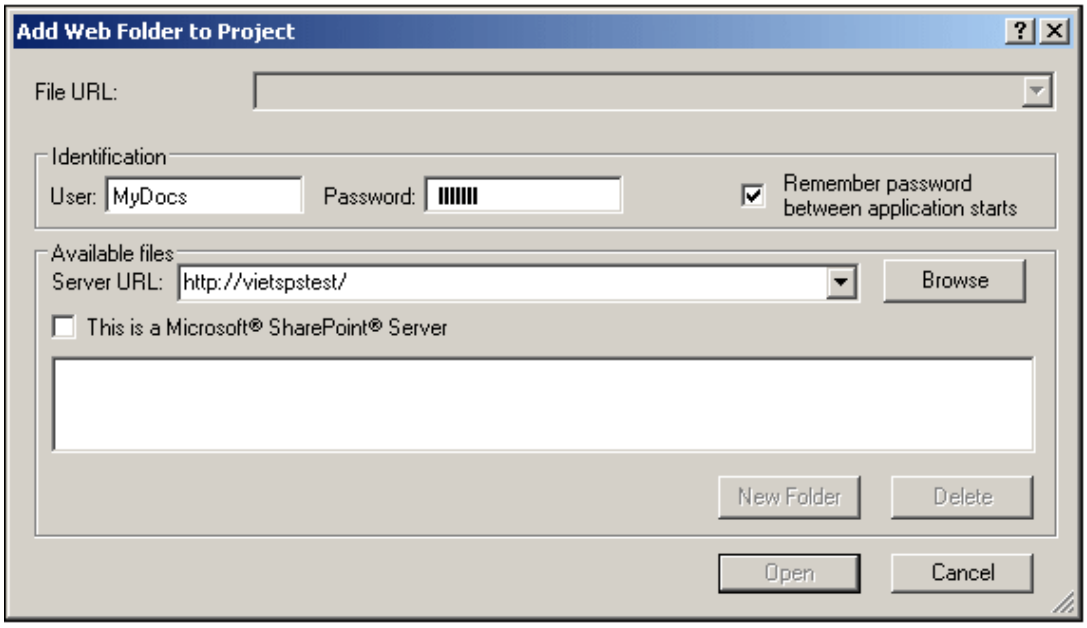
Note: An external folder can be dragged and dropped into a project folder or to any other location in the project (but not into another external folder). Also, an external folder can be dragged from Windows (File) Explorer and dropped into any location in the project window except into another external folder.

12.3.14 Add External Web Folder to Project

This command adds a new external web folder to the current project. You can also access this command from the context-menu when you right-click a folder in the project window. Note that files contained in external folders cannot be placed under source control.

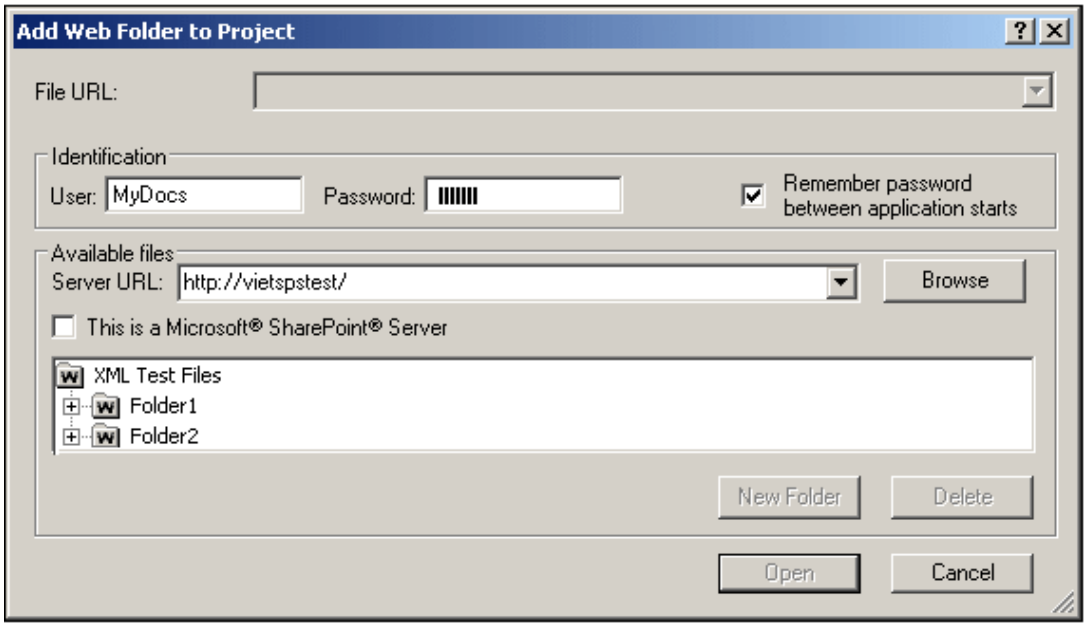
Adding an external web folder to the project
To add an external web folder to the project, do the following:

1. Select the menu option **Project | Add External Web Folder to Project**. This opens the Add Web Folder to Project dialog box (*screenshot below*).

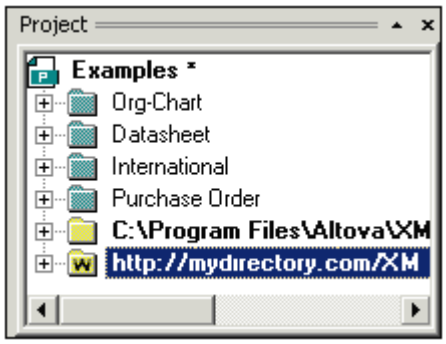


2. Click in the Server URL field and enter the URL of the server URL. If the server is a Microsoft® SharePoint® Server, check this option. See the *Folders on a Microsoft® SharePoint® Server* section below for further information about working with files on this type of server.
3. If the server is password-protected, enter your User ID and password in the *User* and *Password* fields.
4. Click **Browse** to connect to the server and view the available folders.

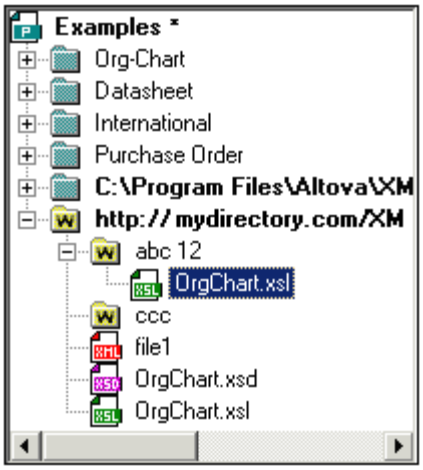
 © 2011 Altova GmbH



5. Click the folder you want to add to the project view. The **Open** button only becomes active once you do this. The URL of the folder now appears in the File URL field.
6. Click **Open** to add the folder to the project.



7. Click the plus icon to view the folder contents.



Filtering folder contents
To filter the contents of a folder, right-click the folder and select **Properties** from the context menu. In the Properties dialog that pops up, click in the *File Extensions* field and enter the file extensions of the file types you want to see (for example, XML and XSD files). Separate each file type with a semicolon (for example: `xml; xsd; sps`). The Project window will now show that folder only with files having the specified extension.

Validating and checking a folder for well-formedness
To check the files in a folder for well-formedness or to validate them, select the folder and then click the **Check well-formedness** or **Validate** icon (hotkeys **F7** or **F8**, respectively). All the files that are visible in the folder are checked. If a file is malformed or invalid, then this file is opened in the main window, allowing you to edit it. Correct the error and restart the process to recheck the rest of the folder. Note that you can select discontinuous files in the folder by holding **Ctrl** and clicking the files singly. Only these files are then checked when you press **F7** or **F8**.

Updating the contents of the project folder
Files may be added or deleted from the web folder at any time. To update the folder view, right-click the external folder and select the context menu option **Refresh**.

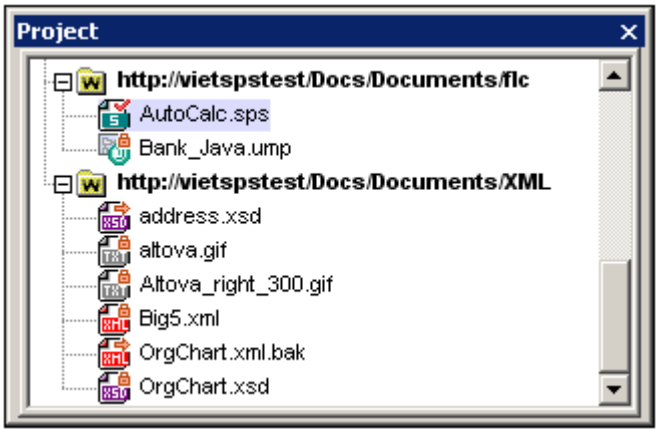
Deleting folders and files
Since it is the Web folder that has been added to the project, it is only the Web folder (and not files within it) that can be deleted from the project. You can delete a Web folder from a project, by either (i) right-clicking the folder and selecting **Delete**, or (ii) selecting the folder and pressing the **Delete** key. This only deletes the folder from the Project view; it does not delete anything on the web server.

Note: Right-clicking a single file and pressing the **Delete** key does not delete a file from the Project window. You have to delete it physically on the server and then refresh the contents of the external folder.

Folders on a Microsoft® SharePoint® Server
When a folder on a Microsoft® SharePoint® Server has been added to a project, files in the folder can be checked out and checked in via commands in the context menu of the file listing in the Project window (*see screenshot below*). To access these commands, right-click the file you wish to work with and select the command you want (**Check Out**, **Check In**, **Undo Check Out**).

The User ID and password can be saved in the properties of individual folders in the project, thereby enabling you to skip the verification process each time the server is accessed.

 © 2011 Altova GmbH



In the Project window (*screenshot below*), file icons have symbols that indicate the check-in/ check-out status of files. The various file icons are shown below:

	Checked in. Available for check-out.
	Checked out by another user. Not available for check-out.
	Checked out locally. Can be edited and checked-in.

The following points should be noted:

- After you check out a file, you can edit it in your Altova application and save it using **File | Save (Ctrl+S)**.
- You can check-in the edited file via the context menu in the Project window (*see screenshot above*), or via the context menu that pops up when you right-click the file tab in the Main Window of your application (*screenshot below*).

- When a file is checked out by another user, it is not available for check out.
- When a file is checked out locally by you, you can undo the check-out with the Undo Check-Out command in the context menu. This has the effect of returning the file unchanged to the server.
- If you check out a file in one Altova application, you cannot check it out in another Altova application. The file is considered to be already checked out to you. The available commands at this point in any Altova application supporting Microsoft® SharePoint® Server will be: **Check In** and **Undo Check Out**.

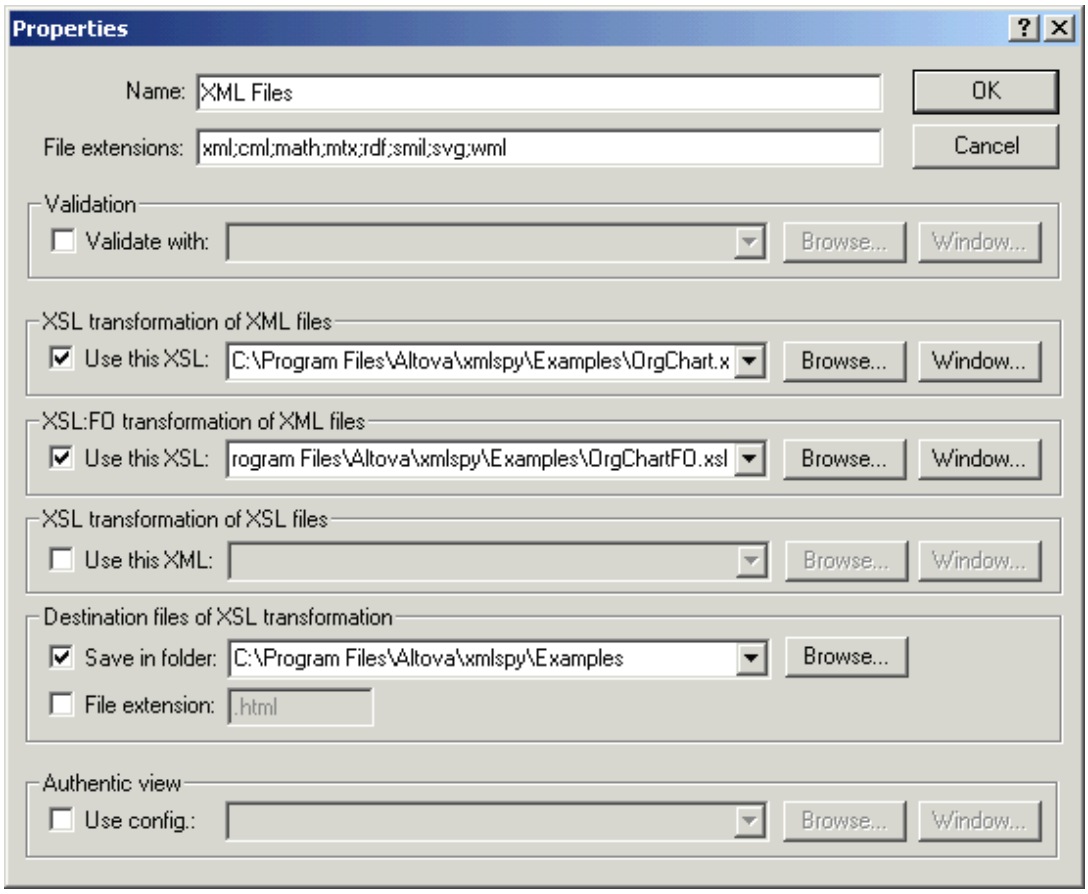
12.3.15 Properties

The **Project | Project Properties** command lets you define important settings for any of the specific folders in your project.

To define the Project Properties for a folder:

1. Right-click on the folder you want to define the properties for.
2. Select the **Properties...** command from the context menu.

Please note:
If your project file is under source control, a prompt appears asking if you want to check out the project file (*.spp). Click **OK** if you want to edit settings and be able to save them.



The files specified in the **Use this xxx** entry will take precedence over any local assignment directly within the XML file. For example, the `OrgChart.xsl` file (in the **Use this XSL** entry), will always be used when transforming any of the XML files in the **XML Files** folder. Also, such specified files for individual folders take precedence over files specified for ancestor folders.

File extensions
The File extensions help to determine the automatic file-to-folder distribution that occurs when you add new files to the project (as opposed as to one particular folder).

User ID and password for external folders
Among the properties of external folders (including external Web folders) you can save the User ID and password that might be required for accessing the server.

Validate
Define the DTD or Schema document that should be used to validate all files in the current folder (Main Pages in this example).

XSL transformation of XML files
You can define the XSL Stylesheet to be used for XSL Transformation of all files in the folder.

If you are developing XSL Stylesheets yourself, you can also assign an example XML document

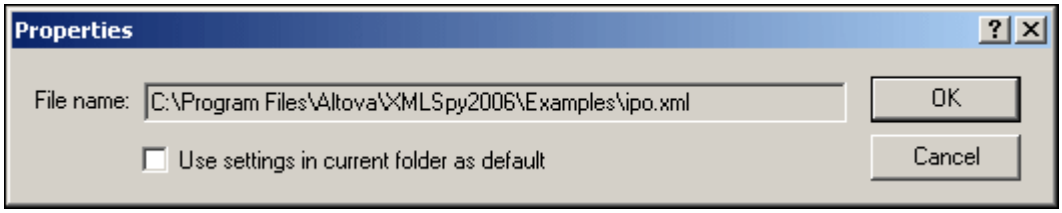
to be used to preview the XSL Stylesheet in response to an XSL Transformation command issued from the stylesheet document, instead of the XML instance document.

XSL:FO transformation of XML files
You can define the XSL Stylesheet, containing XSL:FO markup, to be used for XSL:FO Transformation of all files in the folder.

Destination files of XSL transformation
For batch XSL Transformations, you can define the destination directory the transformed files should be placed in.

If you have added one file or URL to more than one folder in your project, you can use the Properties dialog to set the default folder whose settings should be used when you choose to validate or transform the file in non-batch mode. To do this, use the **Use settings in current folder as default** check box (*see screenshot*).

To access the Properties dialog and check this check box:
1. Copy an XML file in a project to a different folder.
2. Right-click the copied file in the Project window and select **Properties** from the context menu.



Authentic View
The "Use config." option allows you to select a StyleVision Power Stylesheet (SPS file) when editing XML files using Authentic View, in the current folder. After you have associated the schema, SPS, and XML files with each other, and entered them in a project, changing the location of any of the files could cause errors among the associations.

To avoid such errors, it is best to finalize the locations of your schema, SPS, and XML files before associating them with each other and assigning them to a project.

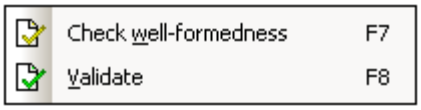
12.3.16 Most Recently Used Projects

This command displays the file name and path for the nine most recently used projects, allowing quick access to these files.

Also note, that XMLSpy can automatically open the last project that you used, whenever you start XMLSpy. (**Tools | Options | File** tab, Project | Open last project on program start).

12.4 XML Menu

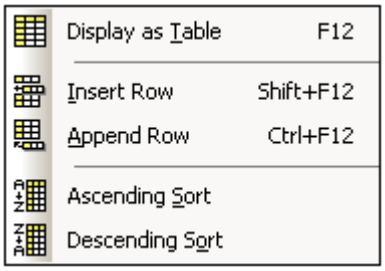
The **XML** menu contains commands commonly used when working with XML documents.



Among the most frequently used XML tasks are checks for the well-formedness of documents and validity of XML documents. Commands for these tasks are in this menu.

12.4.1 Table

The **XML** | **Table** command, though enabled in all views, can be used only in Grid View. It displays a submenu with all the commands relevant to the Database/Table View of Grid View.



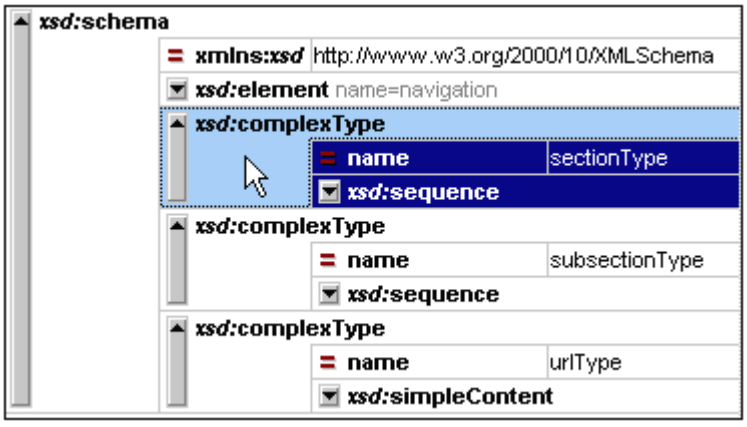
Display as Table

F12

The **XML** | **Table** | **Display as Table** command allows you to switch between the standard Grid View and Database/Table View (or Table View) of a document element. The Table View enables you to view repeated elements as a table in which the rows represent the occurrences while the columns represent child nodes (including comments, CDATA sections, and PIs).

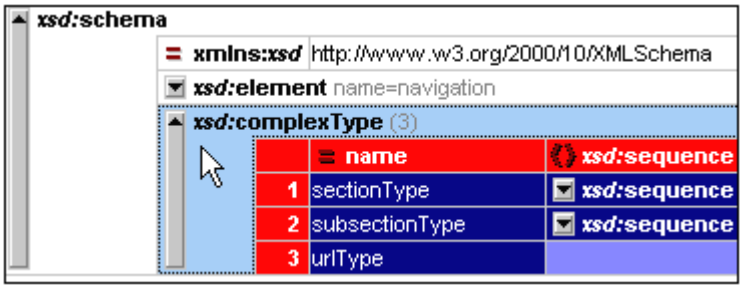
To switch to Table View:

1. Select any one occurrence of the repeating element you wish to view as a table.



2. Click **XML** | **Table** | **Display as Table** or **F12** or the toolbar icon.

© 2011 Altova GmbH



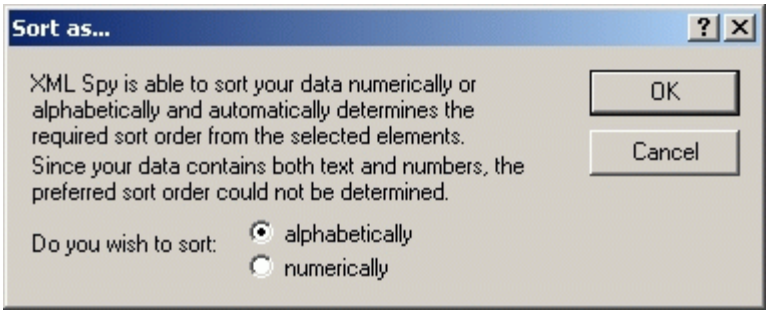
The element is displayed as a table and the toolbar icon is activated.

To switch from the Table View of a document element to the normal Grid View of that element, select the table or any of its rows or columns, and click the toolbar icon. That table element switches to Grid View.

Note: Table View colors can be set in the Colors tab of the Options dialog (**Tools | Options | Colors**)

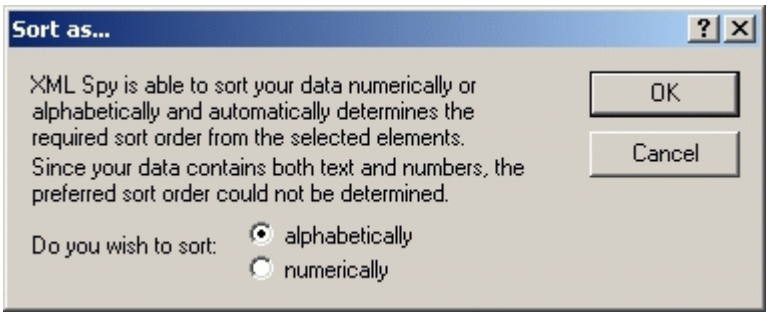
Ascending Sort

The **XML** | **Table | Ascending Sort** command is enabled in Database/Table View when a column or cell is selected. It sorts the column in either alphabetic or numeric ascending order. XMLSpy tries to automatically determine what kind of data is used in the column, and sorts on alphabetic or numeric order, as required. In case of uncertainty, you will be prompted for the sort method to use (*see screenshot*).



Descending Sort

The **XML** | **Table | Descending Sort** command is enabled in Database/Table View when a column or cell is selected. It sorts the column in either alphabetic or numeric descending order. XMLSpy tries to automatically determine what kind of data is used in the column, and sorts on alphabetic or numeric order, as required. In case of uncertainty, you will be prompted for the sort method to use (*see screenshot*).

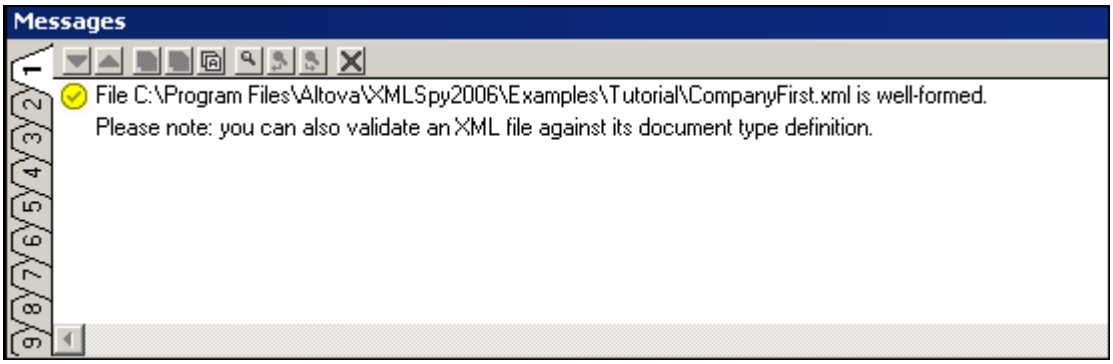


12.4.2 Check Well-Formedness

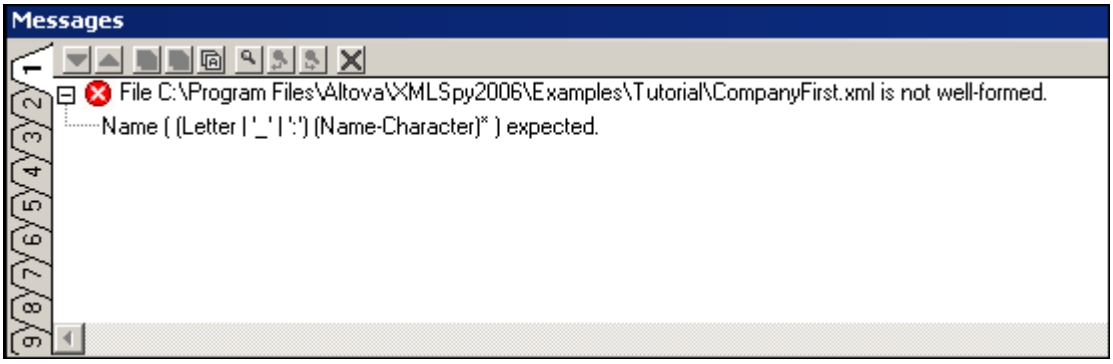
F7

The **XML** | **Check well-formedness (F7)** command checks the active document for well-formedness by the definitions of the XML 1.0 specification. Every XML document **must** be well-formed. XMLSpy checks for well-formedness whenever a document is opened or saved, or when the view is changed from Text to any other view. You can also check for well-formedness at any time while editing by using this command.

If the well-formedness check succeeds when you explicitly invoke the check, a message is displayed in the Validation window:



If an error is encountered during the well-formedness check, a corresponding error message is displayed:



Please note: The output of the Messages window has 9 tabs. The validation output is always displayed in the active tab. Therefore, you can check well-formedness in tab1 for one schema file and keep the result by switching to tab 2 before validating the next schema document

 © 2011 Altova GmbH

(otherwise tab 1 is overwritten with the validation result).

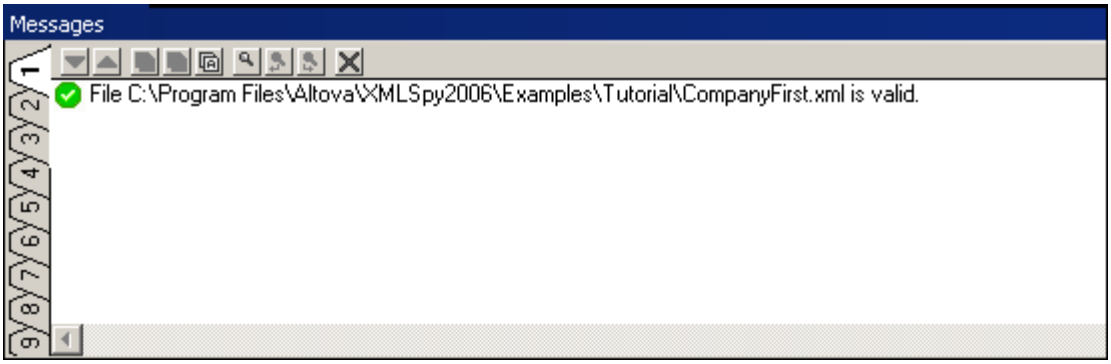
It is generally not permitted to save a malformed XML document, but XMLSpy gives you a Save Anyway option. This is useful when you want to suspend your work temporarily (in a not well-formed condition) and resume it later.

12.4.3 Validate XML

F8

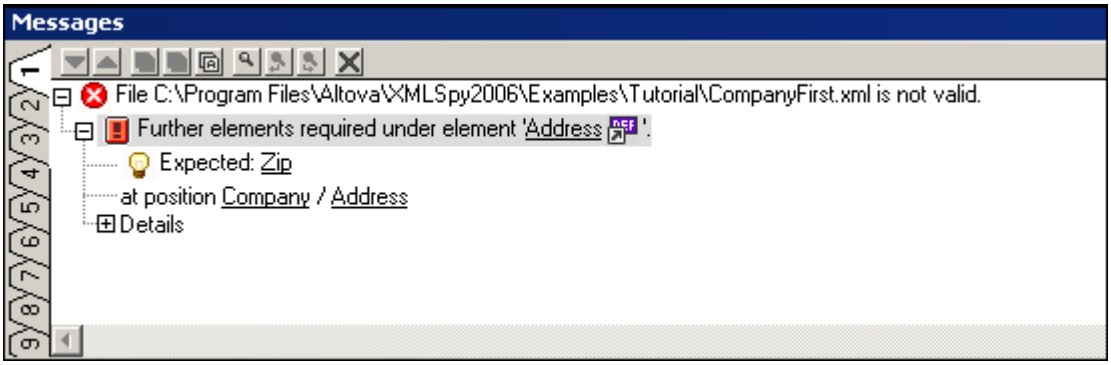
The **XML** | **Validate (F8)** command enables you to validate XML documents against DTDs, XML Schemas, and other schemas. Validation is automatically carried out when you switch from Text View to any other view. You can specify that a document be automatically validated when a file is opened or saved (**Tools | Options | File**). The **Validate** command also carries out a well-formedness check before checking validity, so there is no need to use the Check Well-Formedness command before using the **Validate** command.

If a document is valid, a successful validation message is displayed in the Validation window:



Otherwise, a message that describes the error is displayed.

Please note: You can click on the links in the error message to jump to the spot in the XML file where the error was found.

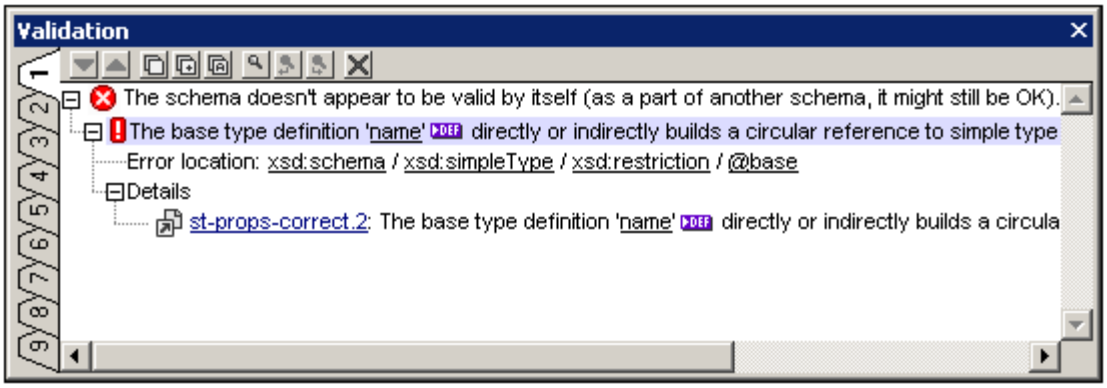


Validating XML documents
To validate an XML file, make the XML document active in the Main Window, and click **XML | Validate** or **F8**. The XML document is validated against the schema referenced in the XML file. If no reference exists, an error message is displayed in the Messages Window. As long as the XML document is open, the schema is kept in memory (see Flush Memory Cache in the DTD/

Schema menu).

Validating schema documents (DTDs and XML Schema)
XMLSpy supports major schema dialects, including DTD and XML Schema. To validate a schema document, make the document active in the Main Window, and click **XML | Validate** or **F8**.

Schema validation messages in the Validation window
If the schema (DTD or XML Schema) is valid, a successful validation message is displayed in the Validation window; the Validation window pops up at the bottom of the application window. If the schema is not valid, one or more error messages are displayed in the Validation window (*screenshot below*).



The error message is divided into three parts:

1. A description of the error. The description contains links to the relevant declarations or definitions.
2. The location of the error within the schema structure. Clicking a node in the location path highlights that node in the document.
3. Details of the error provides more information about the error and contains a link to the relevant paragraph in the relevant specification (the XML specification for DTD validation; and XML Schema specification for XML Schema validation).

Please note: When the validation is done in Text View, clicking a link in the Validation window highlights the corresponding declaration or definition in Text View. When the validation is done in Schema View, the corresponding declaration or definition is highlighted either in the design window of Schema View or in the relevant entry helper window.

Catalogs
XMLSpy supports a subset of the OASIS XML catalogs mechanism. The catalog mechanism enables XMLSpy to retrieve commonly used schemas (as well as stylesheets and other files) from local user folders. This increases the overall processing speed, enables users to work offline (that is, not connected to a network), and improves the portability of documents (because URIs need to be changed in the catalog files only.) The catalog mechanism in XMLSpy works as follows:

- XMLSpy loads a file called `RootCatalog.xml`, which contains a list of catalog files that will be looked up. You can enter as many catalog files to look up, each in a `nextCatalog` element in `RootCatalog.xml`.
- The catalog files included in `RootCatalog.xml` are looked up and the URIs are resolved according to the mappings specified in the catalog files. You should take care

 © 2011 Altova GmbH

not to duplicate mappings, as this could lead to errors.
- Two catalog files are supplied with XMLSpy. How these work is described in the section Catalogs in XMLSpy.
- The PUBLIC or SYSTEM identifier in the DOCTYPE statement of your XML file will be used for the catalog lookup. For popular schemas, the PUBLIC identifier is usually pre-defined, thus requiring only the URI in the catalog file to be changed when XML documents are used on multiple machines.

When writing your CustomCatalog.xml file (or other custom catalog file), use only the following subset of the OASIS catalog in order for XMLSpy to process the catalog correctly. Each of the elements in the supported subset can take the xml:base attribute, which is used to specify the base URI of that element.

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

Please note:

- The catalog.xml file in the %AltovaCommonFolder%\Schemas\schema folder contains references to DTDs that implement older XML Schema specifications. You should not validate your XML Schema documents against any of these schemas. The referenced DTD files are included solely to provide XMLSpy with entry helper info for editing purposes should you wish to create documents according to these older recommendations. *Also see next point*.
- If you create a custom file extension for a particular schema (for example, the .myhtml extension for (HTML) files that are to be valid according to the HTML DTD), then you can enable intelligent editing for files with these extensions by adding a line of text to CustomCatalog.xml. For the example extension mentioned, you should add the element <spy:fileExtHelper ext="myhtml" uri="schemas/xhtml/xhtml1-transitional.dtd"/> as a child of the <catalog> element. This would enable intelligent editing (auto-completion, entry helpers, etc) of .myhtml files in XMLSpy according to the XHTML 1.0 Transitional DTD.
- For more information on catalogs, see the XML Catalogs specification.

Automating validation with Altova XML 2011

AltovaXML is a free application which contains Altova's XML Validator, XSLT 1.0, XSLT 2.0, and XQuery 1.0 engines. It can be used from the command line, via a COM interface, in Java programs, and in .NET applications to validate XML documents, transform XML documents using XSLT 1.0 and 2.0 stylesheets, and execute XQuery documents.

Validation tasks can therefore be automated with the use of Altova XML. For example, you can create a batch file that calls AltovaXML to perform validation on a set of documents and sends the output to a text file. See the AltovaXML documentation for details.

12.4.4 Update Entry Helpers

The **XML** | **Update Entry Helpers** command updates the Entry Helper windows by reloading the underlying DTD or Schema. If you have modified the XML Schema or DTD that an open XML document is based upon, it is advisable to update the Entry Helpers so that the intelligent editing information reflects the changes in the schema.

© 2011 Altova GmbH

12.5 DTD/Schema Menu

The **DTD/Schema** menu contains commands that let you work efficiently with DTDs and XML Schemas.

This section contains a complete description of all the commands in this menu.

12.5.1 Assign DTD

The **DTD/Schema | Assign DTD...** command is enabled when an XML file is active. It assigns a DTD to an XML document, thus allowing the document to be validated and enabling intelligent editing for the document. The command opens the Assign File dialog to let you specify the DTD file you wish to assign. You can also select a file via a global resource or a URL (click the **Browse** button) or a file in one of the open windows in XMLSpy (click the **Window** button). Note that you can make the path of the assigned DTD file relative by clicking the Make Path Relative To... check box. When you are done, your XML document will contain a DOCTYPE declaration that references the assigned DTD. The DOCTYPE declaration will look something like this:

```
<! DOCTYPE main SYSTEM "http: //link. xmlspy. com/spyweb. dtd">
```

Please note: A DTD can be assigned to a new XML file at the time the file is created.

12.5.2 Assign Schema

The **DTD/Schema | Assign Schema...** command is enabled when an XML document is active. It assigns an XML Schema to an XML document, thus allowing the document to be validated and enabling intelligent editing for the document. The command opens the Assign File dialog to let you specify the XML Schema file you wish to assign. You can also select a file via a global resource or a URL (click the **Browse** button) or a file in one of the open windows in XMLSpy (click the **Window** button). Note that you can make the path of the assigned file relative by clicking the Make Path Relative To... check box. When you are done, your XML document will contain an XML Schema assignment with the required namespaces. The schema assignment will look something like this:

```
xmlns="http: //www. xmlspy. com/schemas/icon/orgchart"
xmlns: xsi="http: //www. w3. org/1999/XMLSchema-instance"
xsi: schemaLocation="http: //www. xmlspy. com/schemas/icon/orgchart
   http: //schema. xmlspy. com/schemas/icon/orgchart. xsd"
```

12.5.3 Go to DTD

The **DTD/Schema | Go to DTD** command opens the DTD on which the active XML document is based. If no DTD is assigned, then an error message is displayed.

12.5.4 Go to Schema

The **DTD/Schema | Go to Schema** command opens the XML Schema on which the active XML document is based. If no XML Schema is assigned, then an error message is displayed.

12.5.5 Go to Definition

The **DTD/Schema | Go to Definition** command displays the exact definition of an element or attribute in the corresponding Document Type Definition or Schema document.

To see the item definition in Schema View

- Use CTRL + Double click on the item you want to see the definition of, or
- Click the item and select menu option **DTD/Schema | Go to Definition,** or click on the icon.

  In both cases, the corresponding DTD or Schema file is opened, and the item definition is highlighted.

12.5.6 Generate XML from DB, Excel, EDI with MapForce

The **DTD/Schema | Generate XML from DB, Excel, EDI with MapForce** command launches Altova's MapForce if the application is installed. MapForce enables you to map a schema to another DTD, XML Schema, or database and to generate XML.

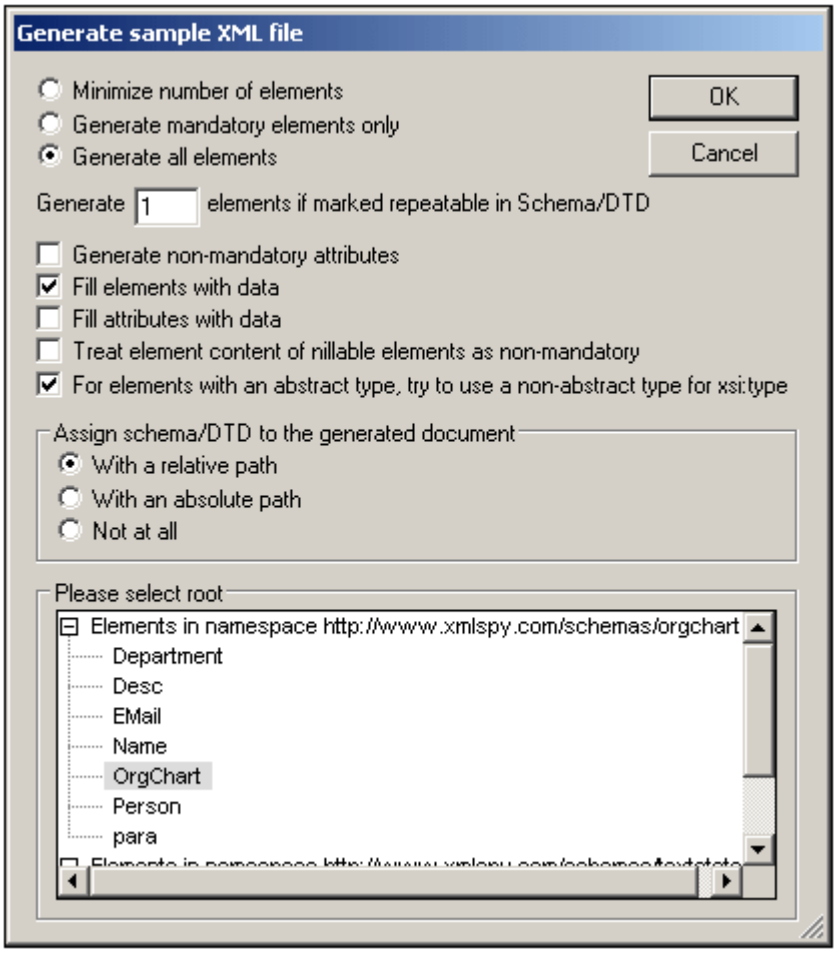
12.5.7 Design HTML/PDF/Word Output with StyleVision...

The **DTD/Schema | Design HTML/PDF Output in StyleVision...** command launches Altova's StyleVision if the application is installed. StyleVision enables you to design stylesheets for HTML, PDF, and RTF output.

12.5.8 Generate Sample XML File

The **DTD/Schema | Generate Sample XML File** command generates an XML file based on the currently active schema (DTD or XML Schema) in the main window.

© 2011 Altova GmbH



Elements to be generated
One of the following choices can be selected: (i) mandatory elements only; (ii) all elements (mandatory and non-mandatory); (iii) mandatory and minimum number of non-mandatory elements (for example, the first definition in a choice). If elements are defined as being repeatable, the number of elements to be generated (for each of these repeatable elements) can be specified.

Generate non-mandatory attributes
Activating this option generates not only mandatory attributes, but also the non-mandatory attributes, defined in the schema.

Generate X elements if marked repeatable in Schema/DTD
Activating this option generates the number of repeatable elements you enter in the text box.

Fill elements and attributes with data
Activating this option inserts the data type descriptors/values for the respective elements/attributes. For example: `Boolean = 1`, `xsd: string = string`, Max/Min inclusive = the value defined in the schema.

Nillable elements and abstract types
The contents of nillable elements can be treated as non-mandatory, and elements with an abstract type can use a non-abstract type for its `xsi: type` attribute.

Schema assignment for the generated XML file

The schema used to generate the XML file can be assigned to the generated XML file with a relative or absolute path.

Root element
If the schema contains more than one global element, these are listed, and the root element required for the sample XML file can be selected from the list.

12.5.9 Flush Memory Cache

The **DTD/Schema | Flush Memory Cache** command flushes all cached schema (DTD and XML Schema) documents from memory. To speed up validation and intelligent editing, XMLSpy caches recently used schema documents and external parsed entities in memory. Information from these cached documents is also displayed when the Go to Definition command is invoked.

Flush the memory cache if memory is tight on your system, or if you have used documents based on different schemas recently.

© 2011 Altova GmbH

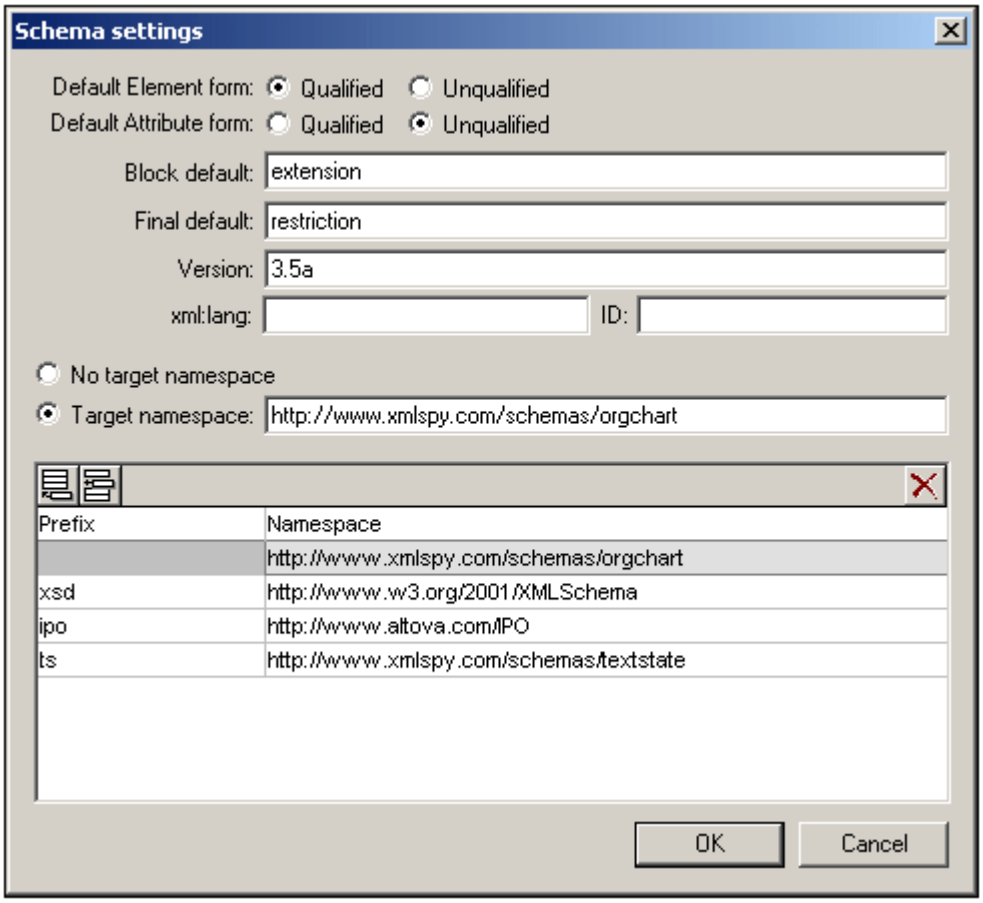
12.6 Schema Design Menu

The **Schema Design** menu enables you to design XML Schemas in a GUI. It is available when an XML Schema document is active in Schema View.

The commands available in this menu are described in this section.

12.6.1 Schema Settings

The **Schema Design** | **Schema Settings** command lets you define global settings for the active schema. These settings are attributes and their values of the XML Schema document element, `xs:schema`.



You can make the following settings in this dialog:

- The `elementFormDefault` and `attributeFormDefault` attributes of the xs:schema element can each be set to `qualified` or `unqualified`.
- The `blockDefault`, `finalDefault`, and `version` attributes can be entered in their respective fields.
- The target namespace for the XML instance document can be set by selecting the `Target Namespace` radio button, and entering the `target namespace` in the field. If you declare a target namespace, you must define that namespace for use in the schema

document. Do this by entering a namespace line in the Namespace list in the bottom pane. You can define a prefix for this namespace, or let this namespace be the default namespace (by leaving the prefix field blank as shown in the screenshot above).

The Text View of the schema settings shown in the screenshot above will look something like this:

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema
targetNamespace="http://www.xmlspy.com/schemas/Altova/orgc
hart" xmlns="http://www.xmlspy.com/schemas/Altova/orgchart"
xmlns:xsd="http://www.w3.org/2000/10/XMLSchema"
elementFormDefault="unqualified"
attributeFormDefault="unqualified" blockDefault="extension"
finalDefault="restriction" version="3.5a">
    <xsd:notation name="Altova-Orgchart"
public="http://www.xmlspy.com/schemas/Altova/orgchart"/>
    <xsd:complexType name="DivisionType">
        <xsd:sequence>
            <xsd:element name="Name" type="xsd:string">
                <xsd:annotation>
                    <xsd:documentation>Division Name
```

Please note: These settings apply to the active schema document only.

12.6.2 Configure View

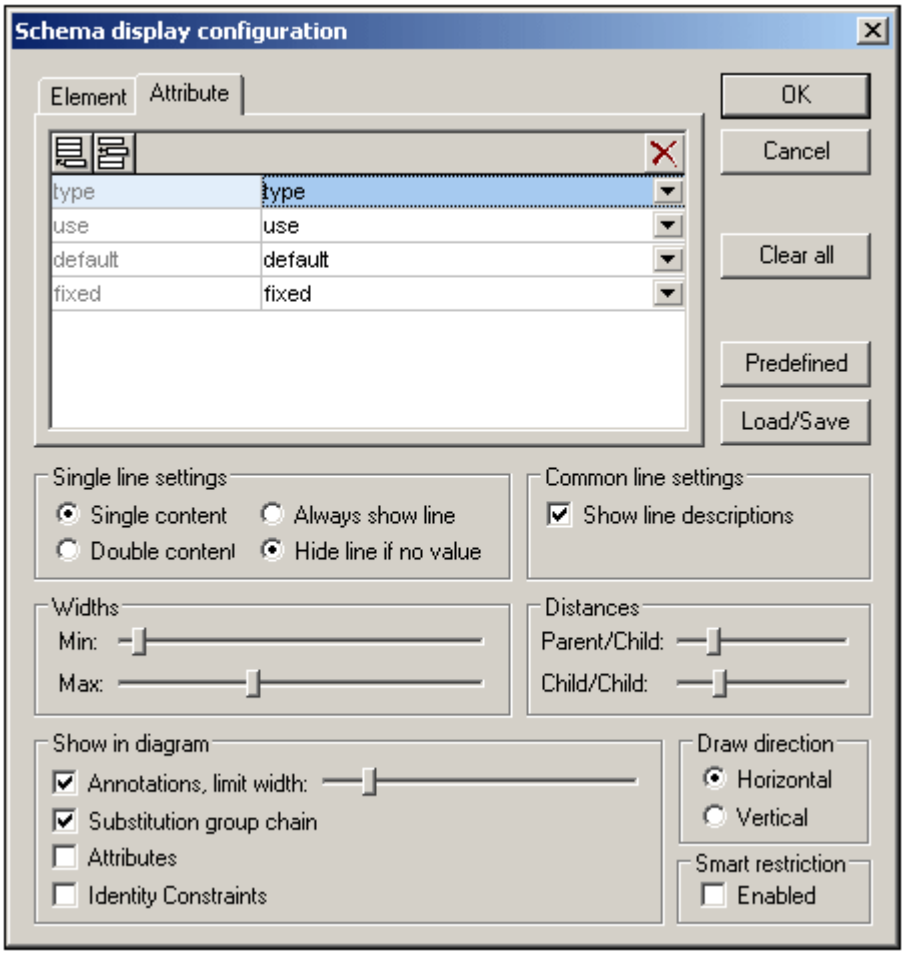
The **Schema Design | Configure view** command is active in Content Model View and allows you to configure the Content Model View. Clicking the command opens the Schema Display Configuration dialog at the bottom right of the XMLSpy window, enabling you to see the effect of your settings as you enter them in the dialog. The settings take effect when you click the **OK** button of the dialog, and apply to the Content Model View of all XML Schema files that are opened subsequently. These settings also apply to the schema documentation output and printer output.

Defining property descriptor lines for the content model
You can define what properties of elements and attributes are displayed in the Content Model View. These properties appear as grid lines in component boxes.

To define property descriptor lines:

1. Select **Schema Design | Configure view**. The Schema display configuration dialog appears.
2. In the **Element** or **Attribute** tab, click the Append or Insert icon to add a property descriptor line. The line is added in the dialog and to element boxes in the Content Model View.
3. From the combo box, select the property you want to display. *See screenshot.*
4. Repeat steps 2 and 3 for as many properties as required.

 © 2011 Altova GmbH



The Content Model View is updated, showing the defined property descriptor lines for all elements for which they exist.

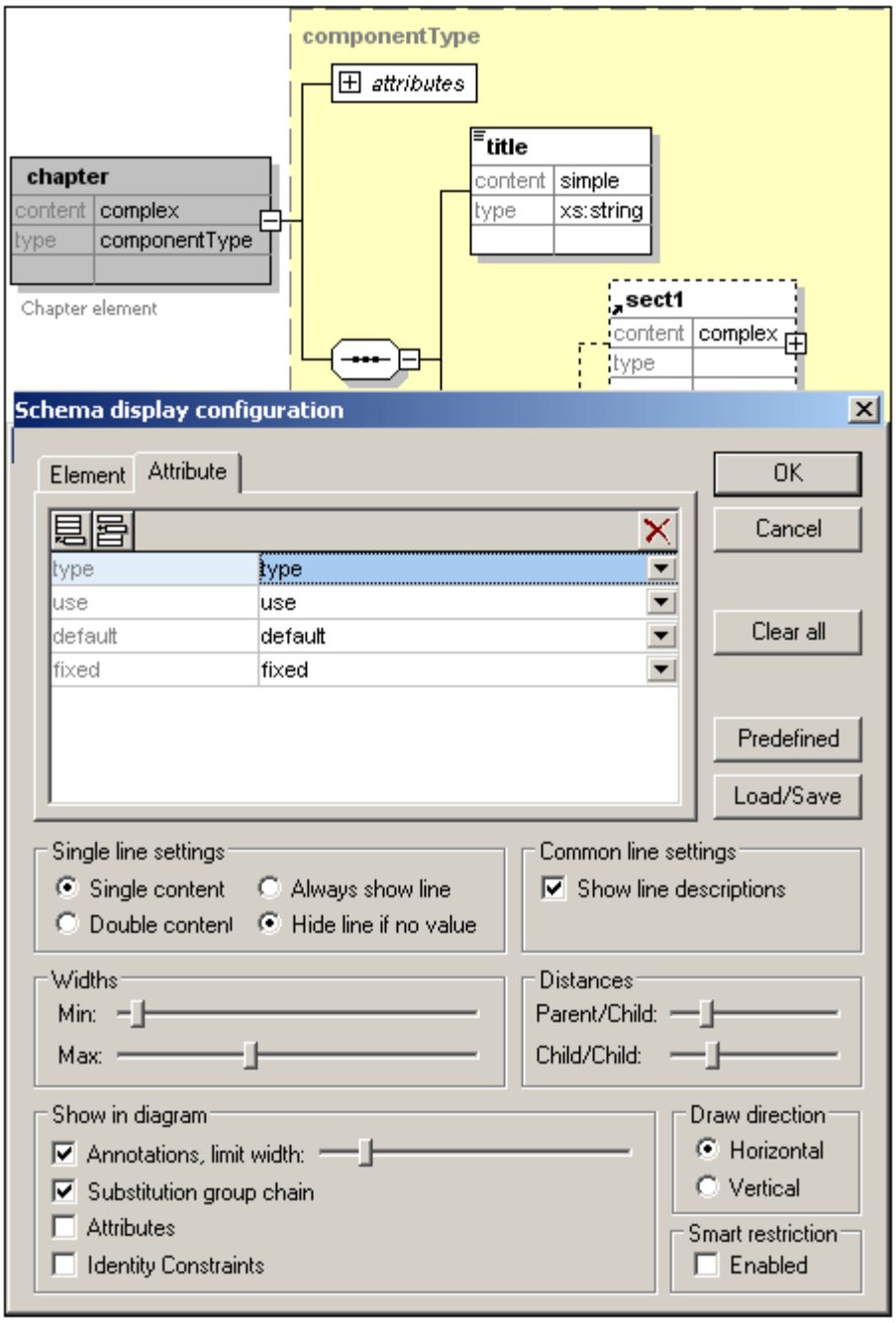
Please note:

- For attributes, the configuration you define appears only when attributes are displayed in the diagram (as opposed to them being displayed in a pane below the Content Model View).
- The configured view applies to all Content Model Views opened after the configuration is defined.

Deleting a property descriptor line from the Content Model View

To delete individual property descriptor lines, in the Schema Display Configuration dialog, select the property descriptor line you want to delete, and click the Delete icon .

Settings for configuring the Content Model View
The Content Model View can be configured using settings in the Schema Display Configuration dialog. How to define what property descriptor lines are displayed in Content Model View has been described above. The other settings are described below.



Single line settings
You can define whether a property descriptor line is to contain single or double content, and whether individual lines must appear for every element or only for elements that contain that property. Use the appropriate radio buttons to define your settings. Note that these two settings can be set for individual lines separately (select the required line and make the setting).

 © 2011 Altova GmbH

Common line settings
This option toggles the line descriptions (i.e. the name of the property) on and off.

Widths
These sliders enable you to set the minimum and maximum size of the element rectangles in Content Model View. Change the sizes if line descriptor text is not fully visible or if you want to standardize your display.

Distances
These sliders let you define the horizontal and vertical distances between various elements onscreen.

Show in diagram
The Annotations check box toggles the display of annotation text on or off, as well as the annotation text width with the slider. You can also toggle the display of the substitution groups on or off. The Attributes and Identity Constraints appear in the Content Model diagram if their check boxes are selected; otherwise they appear as tabs in a pane at the bottom of the Content Model window.

Draw direction
These options define the orientation of the element tree on screen, horizontal or vertical.

Editing the content model in the diagram itself
You can change element properties directly in the content model diagram. To do this, double-click the property you wish to edit and start entering data. If a selection is available, a drop-down list appears, from which you can select an option. Otherwise, enter a value and confirm with **Enter**.

Buttons in the Schema display configuration dialog
This dialog has the following buttons:

- The **Load/Save** button allows you to load and save the settings you make here.
- The **Predefined** button, resets the display configuration to default values.
- The **Clear all** button empties the list box of all entries.

Enabling smart restrictions
To enable smart restrictions, check the Enable Schema Restrictions check box.

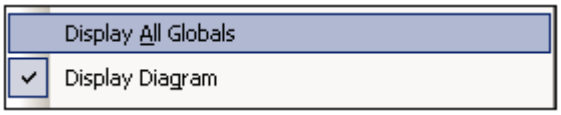
12.6.3 Zoom

The **Schema Design | Zoom** command controls the zoom factor of the Content Model View. This feature is useful if you have a large content model and wish to zoom out so that the entire content model fits in the Main Window. You can zoom between 10% and 200% of actual size.

To zoom in and out, either drag the slider or click in the entry box and enter a percentage value.

12.6.4 Display All Globals

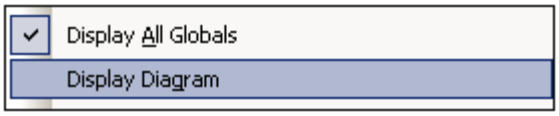
The **Schema Design | Display All Globals** command switches from Content Model View to Schema Overview to display all global components in the schema. It is a toggle with the Display Diagram command. The currently selected toggle is indicated with a check mark to its left (*see screenshot*).



Alternatively, you could use the **Display All Globals** icon at the top of the Content Model View to switch to the Schema Overview.

12.6.5 Display Diagram

The **Schema Design | Display Diagram** command switches to the Content Model View of the selected global component—if the selected component has a content model. Global components that have a content model (complex types, elements, and element groups) are indicated with the icon to its left. The Display Diagram command is a toggle with the Display All Globals command. The currently selected toggle is indicated with a check mark to its left (*screenshot below*).



Alternatively, you could use the following methods to switch to Content Model View:

- Click the icon next to the component, the content model of which you want to display.
- Double-click a component name in the Component Navigator Entry Helper (at top right).

 © 2011 Altova GmbH

12.7 XSL/XQuery Menu

The XSL Transformation language lets you specify how an XML document should be converted into other XML documents or text files. One kind of XML document that is generated with an XSLT document is an FO document, which can then be further processed to generate PDF output. XMLSpy contains built-in XSLT processors (for XSLT 1.0 and XSLT 2.0) and can link to an FO processor on your system to transform XML files and generate various kinds of outputs. The location of the FO processor must be specified in the XSL tab of the Options dialog in order to be able to use it directly from within the XMLSpy interface.

XMLSpy also has a built-in XQuery engine, which can be used to execute XQuery documents (with or without reference to an XML document).

Commands to deal with all the above transformations are accessible in the **XSL/XQuery** menu. In addition, this menu also contains commands to work with the Altova XSLT/XQuery Debugger.

12.7.1 XSL Transformation

F10

The **XSL/XQuery | XSL Transformation** command transforms an XML document using an assigned XSLT stylesheet. The transformation can be carried out using the appropriate built-in Altova XSLT Engine (Altova XSLT 1.0 Engine for XSLT 1.0 stylesheets; Altova XSLT 2.0 Engine for XSLT 2.0 stylesheets), the Microsoft-supplied MSXML module, or an external XSLT processor.

If your XML document contains a reference to an XSLT stylesheet, then this stylesheet is used for the transformation. (An XSLT stylesheet can be assigned to an XML document using the Assign XSL command.) If an XSLT stylesheet has not been assigned to an XML file, you are prompted for the XSLT stylesheet to use. You can also select a file via a global resource or a URL (click the **Browse** button) or a file in one of the open windows in XMLSpy (click the **Window** button).

Automating XSLT transformations with AltovaXML 2011

AltovaXML is a free application which contains Altova's XML Validator, XSLT 1.0, XSLT 2.0, and XQuery 1.0 engines. It can be used from the command line, via a COM interface, in Java programs, and in .NET applications to validate XML documents, transform XML documents using XSLT 1.0 and 2.0 stylesheets, and execute XQuery documents.

XSLT transformation tasks can therefore be automated with the use of AltovaXML. For example, you can create a batch file that calls AltovaXML to transform a set of documents. See the AltovaXML documentation for details.

Transformations to ZIP files

In order to enforce output to a ZIP file, including Open Office XML (OOXML) files such as `.docx`, one must specify the ZIP protocol in the file path of the output file. For example:

```
filename.zip|zip/filename.xxx
filename.docx|zip/filename.xxx
```

Note: The directory structure might need to be created before running the transformation. If

you are generating files for an Open Office XML archive, you would need to zip the archive files in order to create the top-level OOXML file (for example, . `docx`).

12.7.2 XSL-FO Transformation

Ctrl+F10

FO is an XML format that describes paged documents. An FO processor, such as the Apache XML Project's FOP, takes an FO file as input and generates PDF as output. So, the production of a PDF document from an XML document is a two-step process.

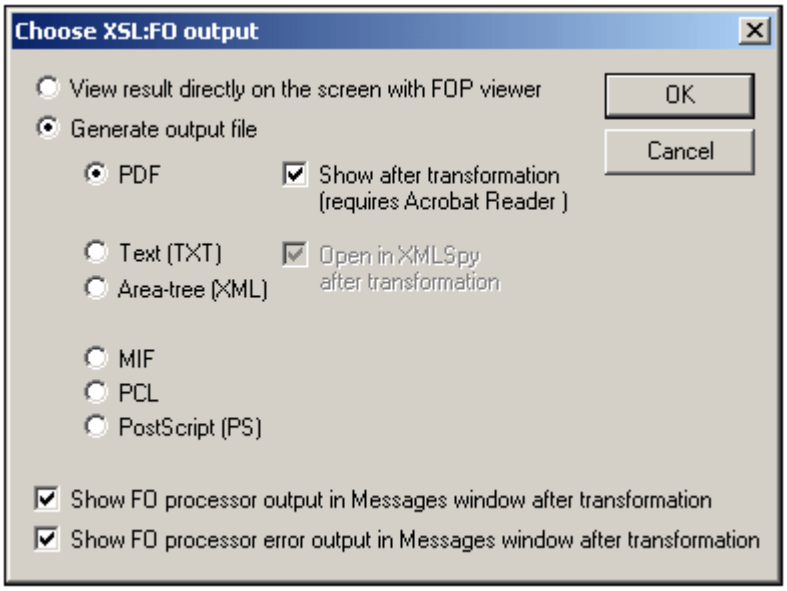
1. The XML document is transformed to an FO document using an XSLT (aka XSL-FO) stylesheet.
2. The FO document is processed by an FO processor to generate PDF (or some alternative output).

The **XSL/XQuery | XSL:FO Transformation** command transforms an XML document or an FO document to PDF.

- If the **XSL:FO Transformation** command is executed on a source XML document, then both of the steps listed above are executed, in sequence, one after the other. If the XSLT (or XSL-FO) stylesheet required to transform to FO is not referenced in the XML document, you are prompted to assign one for the transformation (*screenshot below*). Note that you can also select a file via a global resource or a URL (click the **Browse** button) or a file in one of the open windows in XMLSpy (click the **Window** button). The transformation from XML to XSL-FO is carried out by the XSLT processor specified in the of the Options dialog (**Tools | Options**). By default the selected XSLT processor is XMLSpy's built-in XSLT processor. The resultant FO document is directly processed with the FO processor specified in the of the Options dialog (**Tools | Options**).
- If the **XSL:FO Transformation** command is executed on an FO document, then the document is processed with the FO processor specified in the of the Options dialog (**Tools | Options**).

XSL:FO Transformation output
The **XSL:FO Transformation** command pops up the Choose XSL:FO Output dialog (*screenshot below*). (If the active document is an XML document without an XSLT assignment, you are first prompted for an XSLT file.)

© 2011 Altova GmbH



You can view the output of the FO processor directly on screen using FOP viewer or you can generate an output file in any one of the following formats: PDF, text, an XML area tree, MIF PCL, or PostScript. You can also switch on messages from the FO processor to show (i) the processor's standard output message in the Messages window; and (ii) the processor's error messages in the Messages window. To switch on either these two options, check the appropriate check box at the bottom of the dialog.

Please note:

- The Apache FOP processor can be downloaded free of charge using the link at the Altova Download Center. After downloading and installing FOP, you must set the path to the FOP batch file in the of the Options dialog (**Tools | Options**).

12.7.3 XSL Parameters / XQuery Variables

The **XSL/XQuery | XSL Parameters/XQuery Variables** command opens the XSLT Input Parameters/XQuery External Variables dialog (*see screenshot*). You can enter the name of one or more parameters you wish to pass to the XSLT stylesheet, or one or more external XQuery variables you wish to pass to the XQuery document, and their respective values. These parameters are used as follows in XMLSpy:

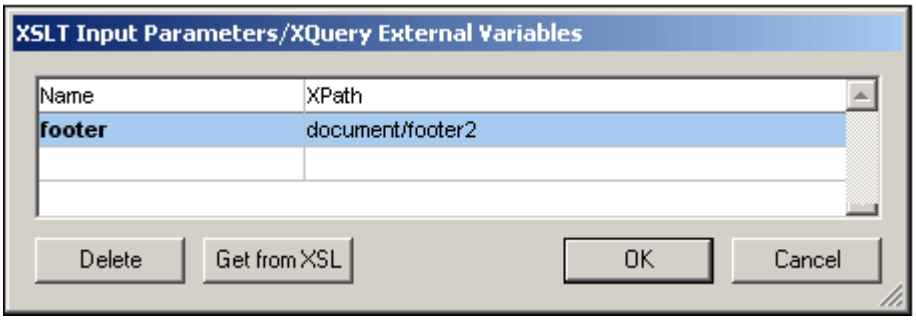
- When the **XSL Transformation** command in the XSL/XQuery menu is used to transform an XML document, the parameter values currently saved in the dialog are passed to the selected XSLT document and used for the transformation.
- When the **XQuery Execution** command in the XSL/XQuery menu is used to process an XQuery document, the XQuery external variable values currently saved in the dialog are passed to the XQuery document for the execution.

Please note: Parameters or variables that you enter in the XSLT Input Parameters/XQuery External Variables dialog are only passed on to the built-in Altova XSLT engine. Therefore, if you are using MSXML or another external engine that you have configured, these parameters are not passed to this engine.

Using XSLT Parameters

The value you enter for the parameter can be an XPath expression without quotes or a text

string delimited by quotes. If the active document is an XSLT document, the **Get from XSL** button will be enabled. Clicking this button inserts parameters declared in the XSLT into the dialog together with their default values. This enables you to quickly include declared parameters and then change their default values as required.



Please note: Once a set of parameter-values is entered in the XSLT Input Parameters/XQuery External Variables dialog, it is used for all subsequent transformations until it is explicitly deleted or the application is restarted. Parameters entered in the XSLT Input Parameters/XQuery External Variables dialog are specified at the application-level, and will be passed to the respective XSLT document for every transformation that is carried out via the IDE from that point onward. This means that:

- parameters are not associated with any particular document
- any parameter entered in the XSLT Input Parameters/XQuery External Variables dialog is erased once XMLSpy has been closed.

Usage example for XSLT parameters

In the following example, we select the required document footer from among three possibilities in the XML document (`footer1`, `footer2`, `footer3`).

```
<?xml version="1.0" encoding="UTF-8"?>
<document xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:noNamespaceSchemaLocation="C:\workarea\footers\footers.xsd">
   <footer1>Footer 1</footer1>
   <footer2>Footer 2</footer2>
   <footer3>Footer 3</footer3>
   <title>Document Title</title>
   <para>Paragraph text.</para>
   <para>Paragraph text.</para>
</document>
```

The XSLT file contains a local parameter called `footer` in the template for the root element. This parameter has a default value of `footer1`. The parameter value is instantiated subsequently in the template with a `$footer` value in the definition of the footer block.

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0"
   xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
   xmlns:fo="http://www.w3.org/1999/XSL/Format">
   ...
   <xsl:param name="footer" select="document/footer1" />
   ...
   <xsl:template match="/">
      <fo:root>
         <xsl:copy-of select="$fo:layout-master-set" />
         <fo:page-sequence master-reference="default-page"
            initial-page-number="1" format="1">
            <fo:static-content flow-name="xsl-region-after"
```

 © 2011 Altova GmbH

```
                display-align="after">
                ...
                <fo:inline color="#800000" font-size="10pt" font-weight="bold">
                   <xsl:value-of select="$footer"/>
                </fo:inline>
                ...
             </fo:static-content>
          </fo:page-sequence>
       </fo:root>
    </xsl:template>
</xsl:stylesheet>
```

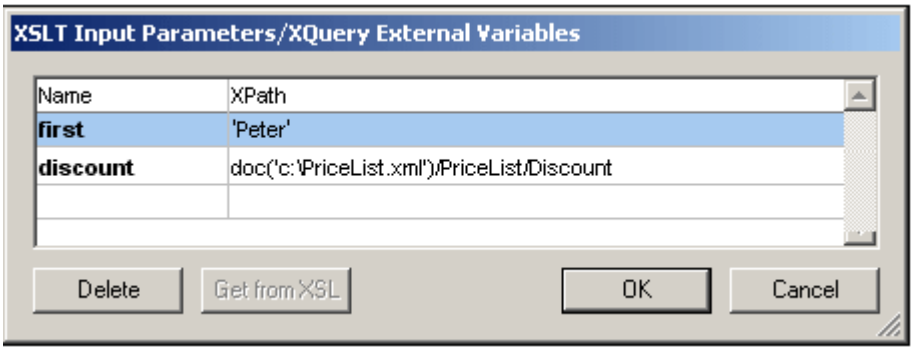
In the XSLT Input Parameters dialog, a new value for the `footer` parameter can be entered, such as the XPath: `document/footer2` (*see screenshot above*) or a text string. During transformation, this value is passed to the `footer` parameter in the template for the root element and is the value used when the footer block is instantiated.

Note:

- If you use the **XSL:FO Transformation** command (**XSL/XQuery | XSL:FO Transformation**), parameters entered in the XSLT Input Parameters/XQuery External Variables dialog are **not** passed to the stylesheet. In order for these parameters to be used in PDF output, first transform from XML to FO using the XSLT Transformation command (**XSL/XQuery | XSL Transformation**), and then transform the FO to PDF using the **XSL:FO Transformation** command (**XSL/XQuery | XSL:FO Transformation**).
- If you use an XSLT processor other than the built-in Altova XSLT Engines, parameters you enter using the Input Parameters dialog will not be passed to the external processor.

Using external XQuery variables

The value you enter for an external XQuery variable could be an XPath expression without quotes or a text string delimited by quotes. The datatype of the external variable is specified in the variable declaration in the XQuery document.



Note: Once a set of external XQuery variables are entered in the XSLT Input Parameters/XQuery External Variables dialog, they are used for all subsequent executions until they are explicitly deleted or the application is restarted. Variables entered in the XSLT Input Parameters/XQuery External Variables dialog are specified at the application-level, and will be passed to the respective XQuery document for every execution that is carried out via the IDE from that point onward. This means that:

- Variables are not associated with any particular document
- Any variable entered in the XSLT Input Parameters/XQuery External Variables dialog is erased once the application (XMLSpy) has been closed down.

Usage example for external XQuery variables
In the following example, a variable `$first` is declared in the XQuery document and is then used in the return clause of the FLWOR expression:

```
xquery version "1.0";
declare variable $first as xs:string external;
let $last := "Jones"
return concat( $first, " ", $last )
```

This XQuery returns `Peter Jones`, if the value of the external variable (entered in the XSLT Input Parameters/XQuery External Variables dialog) is `Peter`. Note the following:

- The `external` keyword in the variable declaration in the XQuery document indicates that this variable is an external variable.
- Defining the static type of the variable is optional. If a datatype for the variable is not specified in the variable declaration, then the variable value is assigned the type `xs:untypedAtomic`.
- If an external variable is declared in the XQuery document, but no external variable of that name is passed to the XQuery document, then an error is reported.
- If an external variable is declared and is entered in the XSLT Input Parameters/XQuery External Variables dialog, then it is considered to be in scope for the XQuery document being executed. If a new variable with that name is declared within the XQuery document, the new variable temporarily overrides the in-scope external variable. For example, the XQuery document below returns `Paul Jones` even though the in-scope external variable `$first` has a value of `Peter`.

  ```
  xquery version "1.0";
  declare variable $first as xs:string external;
  let $first := "Paul"
  let $last := "Jones"
  return concat( $first, " ", $last )
  ```

Note: It is not an error if an external XQuery variable (or XSLT parameter) is defined in the XSLT Input Parameters/XQuery External Variables dialog but is not used in the XQuery document. Neither is it an error if an XSLT parameter (or external XQuery variable) is defined in the XSLT Input Parameters/XQuery External Variables dialog but is not used in an XSLT transformation.

12.7.4 XQuery Execution

The **XSL/XQuery | XQuery Execution** command executes an XQuery document. It can be invoked when an XQuery or XML file is active. When invoked from an XML file, it opens a dialog asking for an XQuery file to associate with the XML file. You can also select a file via a global resource or a URL (click the **Browse** button) or a file in one of the open windows in XMLSpy (click the **Window** button).

Automating XQuery executions with AltovaXML 2011
AltovaXML is a free application which contains Altova's XML Validator, XSLT 1.0, XSLT 2.0, and XQuery 1.0 engines. It can be used from the command line, via a COM interface, in Java programs, and in .NET applications to validate XML documents, transform XML documents using XSLT 1.0 and 2.0 stylesheets, and execute XQuery documents.

 © 2011 Altova GmbH

XQuery execution tasks can therefore be automated with the use of AltovaXML. For example, you can create a batch file that calls AltovaXML to execute a set of XQuery documents. See the AltovaXML documentation for details.

12.7.5 Assign XSL

The **XSL/XQuery | Assign XSL...** command assigns an XSLT stylesheet to an XML document. Clicking the command opens a dialog to let you specify the XSLT file you want to assign. You can also select a file via a global resource or a URL (click the **Browse** button) or a file in one of the open windows in XMLSpy (click the **Window** button).

An `xml-stylesheet` processing instruction is inserted in the XML document:

```
<?xml-stylesheet type="text/xsl"
   href="C:\workarea\recursion\recursion.xslt"?>
```

Please note: You can make the path of the assigned file relative by clicking the **Make Path Relative To...** check box.

12.7.6 Assign XSL-FO

The **XSL/XQuery | Assign XSL:FO...** command assigns an XSLT stylesheet for transformation to FO to an XML document. The command opens a dialog to let you specify the XSL or XSLT file you want to assign and inserts the required processing instruction into your XML document:

```
<?xmlspyxslfo C:\Program Files\Altova\xmlspy\Examples\OrgChartFO.xsl?>
```

You can make the path of the assigned file relative by clicking the Make Path Relative To... check box. You can also select a file via a global resource or a URL (click the **Browse** button) or a file in one of the open windows in XMLSpy (click the **Window** button).

Please note: An XML document may have two XSLT files assigned to it: one for standard XSLT transformations, a second for an XSLT transformation to FO.

12.7.7 Assign Sample XML File

The **XSL/XQuery | Assign Sample XML File** command assigns an XML file to an XSLT document. The command inserts a processing instruction naming an XML file to be processed with this XSLT file when the XSL Transformation is executed on the XSLT file:

```
<?altova_samplexml C:\workarea\html2xml\article.xml?>
```

Please note: You can make the path of the assigned file relative by clicking the Make Path Relative To... check box. You can also select a file via a global resource or a URL (click the **Browse** button) or a file in one of the open windows in XMLSpy (click the **Window** button).

12.7.8 Go to XSL

The **XSL/XQuery | Go to XSL** command opens the associated XSLT document. If your XML document contains a stylesheet processing instruction (i.e. an XSLT assignment) such as this:

```
<?xml-stylesheet type="text/xsl" href="Company.xsl"?>
```

then the **Go to XSL** command opens the XSLT document in XMLSpy.

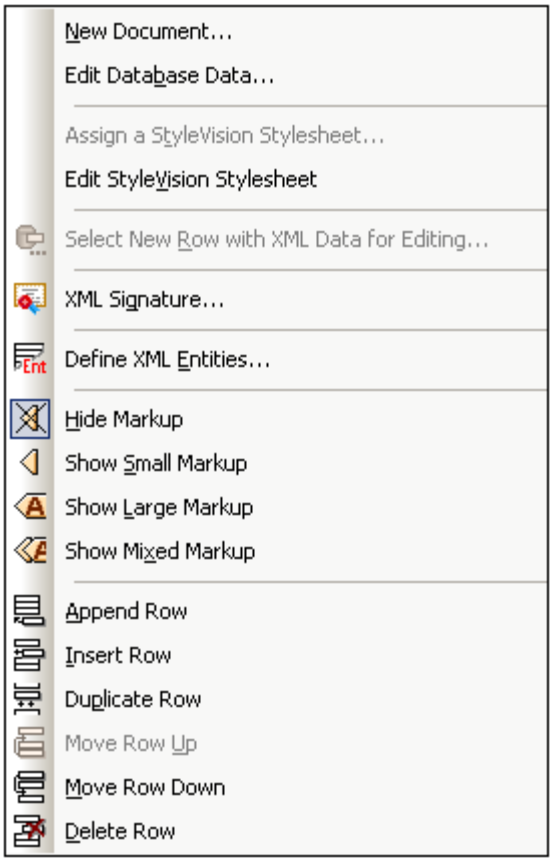
© 2011 Altova GmbH

12.8 Authentic Menu

Authentic View enables you to edit XML documents **based on StyleVision Power Stylesheets (.sps files) created in Altova's StyleVision product!** These stylesheets contain information that enables an XML file to be displayed graphically in Authentic View. In addition to containing display information, StyleVision Power Stylesheets also allow you to write data to the XML file. This data is dynamically processed using all the capability available to XSLT stylesheets and instantly produces the output in Authentic View.

Additionally, StyleVision Power Stylesheets can be created to display an editable XML view of a database. The StyleVision Power Stylesheet contains information for connecting to the database, displaying the data from the database in Authentic View, and writing back to the database.

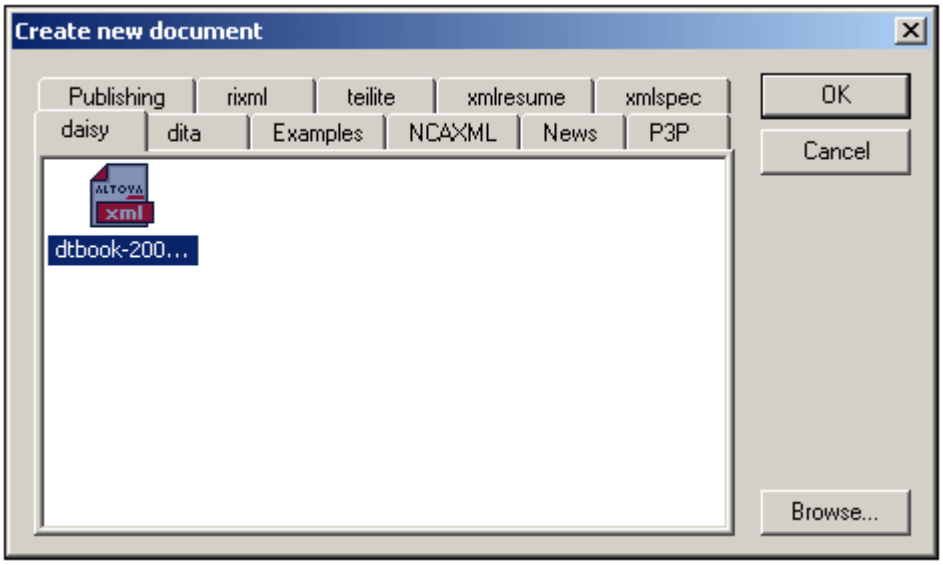
The **Authentic** menu contains commands relevant to editing XML documents in Authentic View . For a tutorial on Authentic View, see the Tutorials section.



12.8.1 New Document

The **Authentic** | **New Document...** command enables you to open a new XML document template in Authentic View. The XML document template is based on a StyleVision Power Stylesheet (. sps file), and is opened by selecting the StyleVision Power Stylesheet.

Clicking the **New Document...** command opens the Create New Document dialog.



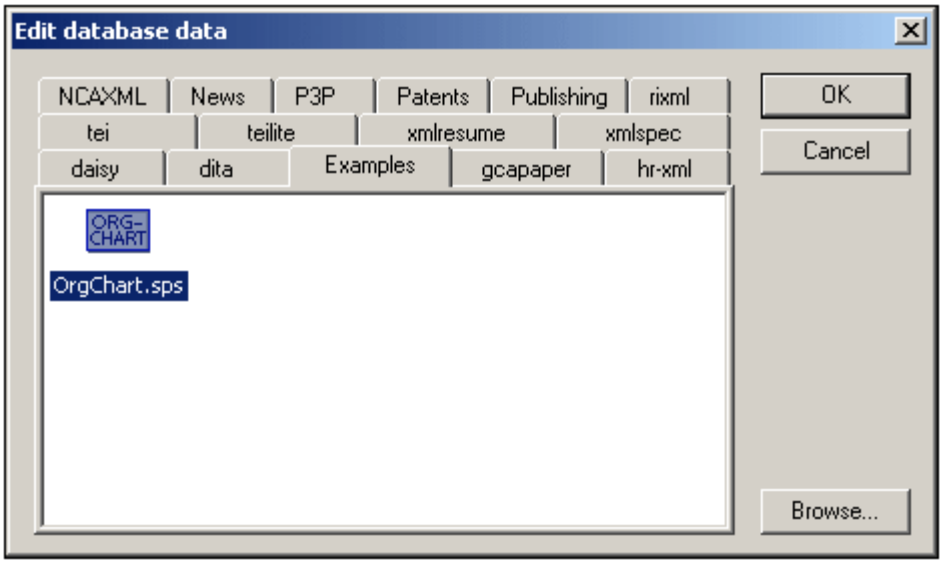
Browse for the required SPS file, and select it. This opens an XML document template in Authentic View.

Note: StyleVision Power Stylesheets are created using Altova StyleVision. The StyleVision Power Stylesheet has a Template XML File assigned to it. The data in this XML file provides the starting data of the new document template that is opened in Authentic View.

12.8.2 Edit Database Data

The **Authentic** | **Edit Database Data...** command enables you to open an editable view of a database (DB) in Authentic View. All the information about connecting to the DB and how to display the DB and accept changes to it in Authentic View is contained in a StyleVision Power Stylesheet. It is such a DB-based StyleVision Power Stylesheet that you open with the **Edit Database Data...** command. This sets up a connection to the DB and displays the DB data (through an XML lens) in Authentic View.

Clicking the **Edit Database Data...** command opens the Edit Database Data dialog.



 © 2011 Altova GmbH

Browse for the required SPS file, and select it. This connects to the DB and opens an editable view of the DB in Authentic View. The design of the DB view displayed in Authentic View is contained in the StyleVision Power Stylesheet.

Please note: If, with the **Edit Database Data...** command, you attempt to open a StyleVision Power Stylesheet that is not based on a DB or to open a DB-based StyleVision Power Stylesheet that was created in a version of StyleVision prior to the StyleVision 2005 release, you will receive an error.

Please note: StyleVision Power Stylesheets are created using Altova StyleVision.

12.8.3 Assign/Edit a StyleVision Stylesheet

Assign a StyleVision Stylesheet
The **Assign a StyleVision Stylesheet** command assigns a StyleVision Power Stylesheet (SPS) to an **XML document** to enable the viewing and editing of that XML document in Authentic View. The StyleVision Power Stylesheet that is to be assigned to the XML file must be based on the same schema as that on which the XML file is based.

To assign a StyleVision Power Stylesheet to an XML file:

1. Make the XML file the active file and select the **Authentic** | **Assign a StyleVision Stylesheet...** command.
2. The command opens a dialog box in which you specify the StyleVision Power Stylesheet file you wish to assign to the XML.
3. Click **OK** to insert the required SPS statement into your XML document. Note that you can make the path to the assigned file relative by clicking the **Make path relative to ...** check box. You can also select a file via a global resource or a URL (click the **Browse** button) or a file in one of the open windows in XMLSpy (click the **Window** button).

```
<?xml version="1.0" encoding="UTF-8"?>
<?altova_sps HTML-Orgchart.sps?>
```

In the example above, the StyleVision Power Stylesheet is called `HTML_Orgchart.sps`, and it is located in the same directory as the XML file.

Please note: Previous versions of Altova products used a processing instruction with a target or name of `xmlspysps`, so a processing instruction would look something like `<?xmlspysps HTML-Orgchart.sps?>`. These older processing instructions are still valid with Authentic View in current versions of Altova products.

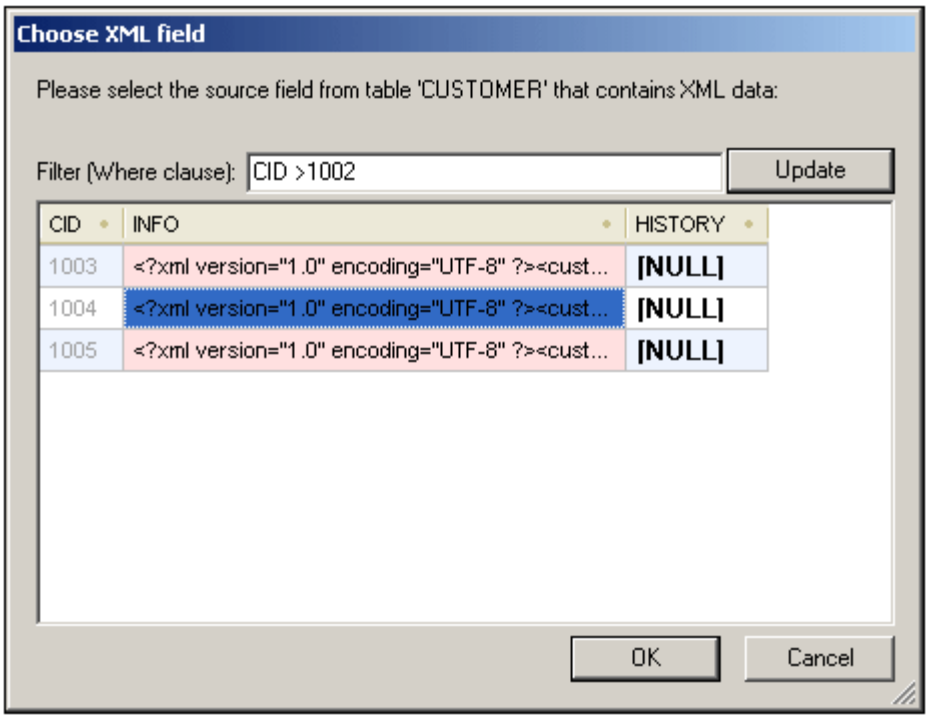
Edit StyleVision Stylesheet
The **Authentic** | **Edit StyleVision Stylesheet** command starts StyleVision and allows you to edit the StyleVision Power Stylesheet immediately in StyleVision.

12.8.4 Select New Row with XML Data for Editing

The **Select New Row with XML Data for Editing** command enables you to select a new row from the relevant table in an XML DB, such as IBM DB2. This row appears in Authentic View, can be edited there, and then saved back to the DB.

When an XML DB is used as the XML data source, the XML data that is displayed in Authentic View is the XML document contained in one of the cells of the XML data column. The **Select New Row with XML Data for Editing** command enables you to select an XML document from

another cell (or row) of that XML column. Selecting the **Select New Row...** command pops up the Choose XML Field dialog (*screenshot below*), which displays the table containing the XML column.



You can enter a filter for this table. The filter should be an SQL `WHERE` clause (just the condition, without the `WHERE` keyword, for example: `CID>1002`). Click **Update** to refresh the dialog. In the screenshot above, you can see the result of a filtered view. Next, select the cell containing the required XML document and click **OK**. The XML document in the selected cell (row) is loaded into Authentic View.

12.8.5 Define XML Entities

You can define entities for use in Authentic View, whether your document is based on a DTD or an XML Schema. Once defined, these entities are displayed in the Entities Entry Helper and in the **Insert Entity** submenu of the context menu. When you double-click on an entity in the Entities Entry Helper, that entity is inserted at the cursor insertion point.

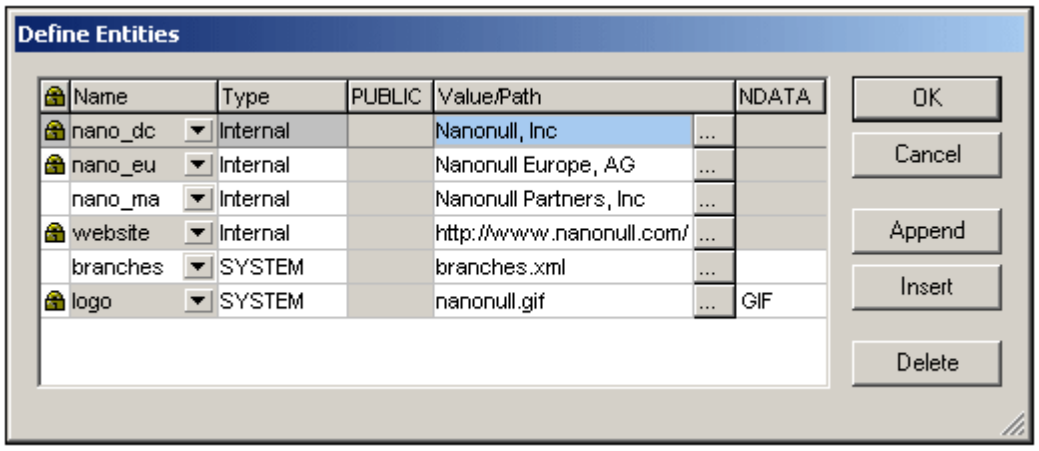
An entity is useful if you will be using a text string, XML fragment, or some other external resource in multiple locations in your document. You define the entity, which is basically a short name that stands in for the required data, in the Define Entities dialog. After defining an entity you can use it at multiple locations in your document. This helps you save time and greatly enhances maintenance.

There are two broad types of entities you can use in your document: a **parsed entity**, which is XML data (either a text string or a fragment of an XML document), or an **unparsed entity**, which is non-XML data such as a binary file (usually a graphic, sound, or multimedia object). Each entity has a name and a value. In the case of parsed entities the entity is a placeholder for the XML data. The value of the entity is either the XML data itself or a URI that points to a `.xml` file that contains the XML data. In the case of unparsed entities, the value of the entity is a URI that points to the non-XML data file.

 © 2011 Altova GmbH

To define an entity:

1. Click **Authentic | Define XML Entities...**. This opens the Define Entities dialog.



2. Enter the name of your entity in the **Name** field. This is the name that will appear in the Entities Entry Helper.
3. Enter the type of entity from the drop-down list in the **Type** field. Three types are possible. An **Internal** entity is one for which the text to be used is stored in the XML document itself. Selecting **PUBLIC** or **SYSTEM** specifies that the resource is located outside the XML file, and will be located with the use of a public identifier or a system identifier, respectively. A system identifier is a URI that gives the location of the resource. A public identifier is a location-independent identifier, which enables some processors to identify the resource. If you specify both a public and system identifier, the public identifier resolves to the system identifier, and the system identifier is used.
4. If you have selected PUBLIC as the Type, enter the public identifier of your resource in the PUBLIC field. If you have selected Internal or SYSTEM as your Type, the PUBLIC field is disabled.
5. In the **Value/Path** field, you can enter any one of the following:
   - If the entity type is Internal, enter the text string you want as the value of your entity. Do not enter quotes to delimit the entry. Any quotes that you enter will be treated as part of the text string.
   - If the entity type is SYSTEM, enter the URI of the resource or select a resource on your local network by using the **Browse** button. If the resource contains parsed data, it must be an XML file (i.e. it must have a `.xml` extension). Alternatively, the resource can be a binary file, such as a GIF file.
   - If the entity type is PUBLIC, you must additionally enter a system identifier in this field.
6. The NDATA entry tells the processor that this entity is not to be parsed but to be sent to the appropriate processor. The NDATA field should therefore be used with unparsed entities only.

Dialog features

You can append, insert, and delete entities by clicking the appropriate buttons. You can also sort entities on the alphabetical value of any column by clicking the column header; clicking once sorts in ascending order, twice in descending order. You can also resize the dialog box and the width of columns.

Once an entity is used in the XML document, it is locked and cannot be edited in the Define Entities dialog. Locked entities are indicated by a lock symbol in the first column. Locking an entity ensures that the XML document valid with respect to entities. (The document would be invalid if an entity is referenced but not defined.)

Duplicate entities are flagged.

Limitations

- An entity contained within another entity is not resolved, either in the dialog, Authentic View, or XSLT output, and the ampersand character of such an entity is displayed in its escaped form, i.e. `&`.
- External entities are not resolved in Authentic View, except in the case where an entity is an image file and it is entered as the value of an attribute which has been defined in the schema as being of type `ENTITY` or `ENTITIES`. Such entities are resolved when the document is processed with an XSLT generated from the SPS.

12.8.6 Hide Markup, Show Small/Large/Mixed Markup

The **Hide Markup** command hides markup symbols in the Authentic View.

The **Show Small Markup** command shows small markup symbols in the Authentic View.

The **Show Large Markup** command shows large markup symbols in the Authentic View.

The Show Mixed Markup command shows mixed markup symbols in Authentic View. The person who designs the StyleVision Power Stylesheet can specify either large markup, small markup, or no markup for individual elements/attributes in the document. The Authentic View user sees this customized markup in mixed markup viewing mode.

12.8.7 Append/Insert/Duplicate/Delete Row

The **Append Row** command appends a row to the current table in Authentic View.

The **Insert Row** command inserts a row into the current table in Authentic View.

The **Duplicate Row** command duplicates the current table row in Authentic View.

 © 2011 Altova GmbH

The **Delete Row** command deletes the current table row in Authentic View.

12.8.8 Move Row Up/Down

The **Move Row Up** command moves the current table row up by one row in Authentic View.

The **Move Row Down** command moves the current table row down by one row in Authentic View.

12.9 View Menu

The **View** menu controls the display of the active Main window and allows you to change the way XMLSpy displays your XML documents.

This section provides a complete description of commands in the **View** menu.

12.9.1 Text View

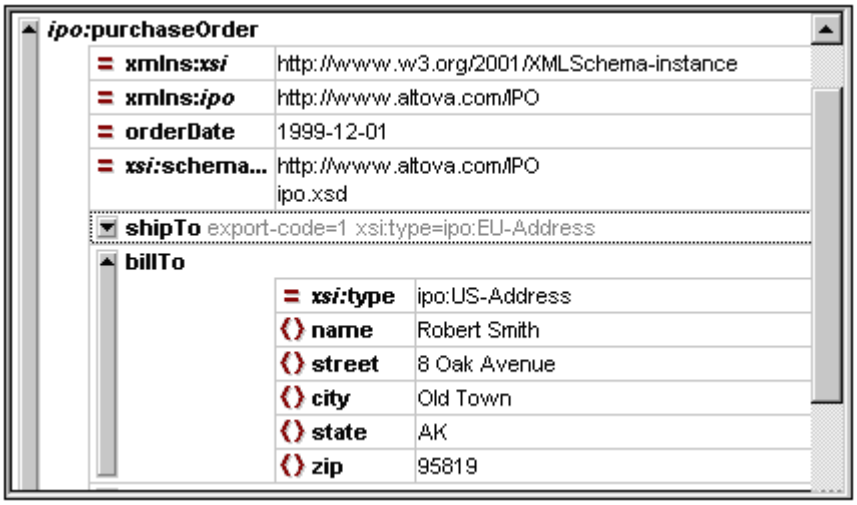
This command **switches** the current view of the document to Text View, which enables you to edit the document in its text form. It supports a number of advanced text editing features, described in detail in Text View section of this document.

Please note: You can configure aspects of the Text View using options available in the various tabs of the Options dialog (**Tools | Options**).

12.9.2 Enhanced Grid View

This command **switches** the current document into Grid View.

If the previous view was the Text View, the document is automatically checked for well-formedness.



Embedded Database/Table view
XMLSpy allows you to display recurring elements in a Database/Table View from within the Grid View. This function is available wherever the Grid View can be activated, and can be used when editing any type of XML file - XML, XSD, XSL etc.

For further information on this view, please see the Grid View section of this documentation.

12.9.3 Schema Design View

This command **switches** the current document to Schema Design View. The switch will be successful only if the document is an XML Schema document. This view is described in detail in the Schema View section of this documentation.

© 2011 Altova GmbH

12.9.4 Authentic View

This command **switches** the current document into the Authentic View.

Authentic View enables you to edit XML documents **based on StyleVision Power Stylesheet templates created in StyleVision!** The templates in StyleVision are saved as **StyleVision Power Stylesheet**s (***.sps** files), and supply all the necessary information needed by Authentic View.

To **open** a template select the **File | New** command and then click the **Select a StyleVision stylesheet...** button. Please see the Authentic View documentation for further information.

Please note: If, when you try to switch to Authentic View, you receive a message saying that a temporary (temp) file could not be created, contact your system administrator. The system administrator must change the default Security ID for "non-power users" to allow them to create folders and files.

12.9.5 Browser View

This command **switches** the current document into Browser View.

This view uses an XML-enabled browser to render the XML document using information from potential CSS or XSL style-sheets.

When switching to browser view, the document is first checked for validity, if you have selected Validate upon saving in the File tab of the Options dialog. Use the menu command **Tools | Options** to open this dialog.

For further information on this view, please see the detailed description of the various views in the Main Window section.

12.9.6 Expand

Hotkey: **"+" on the numeric keypad**

This command **expands** the selected element by one level.

The command can be used in the Grid View.

In the Grid View, the element and all its children remain selected after expansion. This allows you to expand a large element by pressing the **+** key repeatedly.

You can expand and collapse any element by clicking on the gray bar to the left of each element.

12.9.7 Collapse

Hotkey: **"-" on the numeric keypad**

This command **collapses** the selected element by one level.

The command can be used in the Grid View.

You can expand and collapse any element by clicking on the gray bar to the left of each element.

12.9.8 Expand Fully

Hotkey: **"*" on the numeric keypad**

This command **expands** all child items of the **selected element**, down to the last level of nesting.

The command can be used in the Grid View.

12.9.9 Collapse Unselected

Hotkey: **CTRL** + **"-" key on the numeric keypad**

This command allows you to focus on one element and its children, and ignore all the other surrounding elements.

The command can be used in the Grid View.

Select the item that you want to work with and choose this command to collapse all other (unselected) elements.

12.9.10 Optimal Widths

This command adjusts the widths of all columns so that each has a width that exactly accommodates in one line the longest text string in any of its cells. A maximum optimal width can be specified in the View tab of the Options dialog (**Tools | Menu**). Note that optimal widths are calculated on the basis of the visible cells of columns. This enables the optimization of the view when individual elements are collapsed or expanded.

12.9.11 Word Wrap

This command enables or disables word wrapping in the **Text view**.

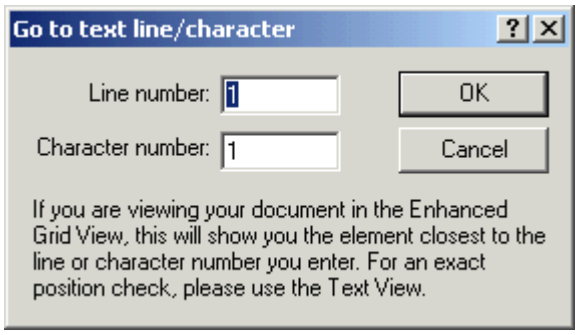
12.9.12 Go to Line/Character

Hotkey: **CTRL**+ **g**

This command goes to a **specific line number** and/or character position in an XML document in the Text view.

If you are working with an external XSLT processor (see the XSL tab on the Tools | Options dialog for details) you may often get error messages by line number and character position. XMLSpy lets you quickly navigate to that spot, using this command:

© 2011 Altova GmbH



12.9.13 Go to File

This command **opens** a document that is being **referred** to, from within the file you are currently editing.

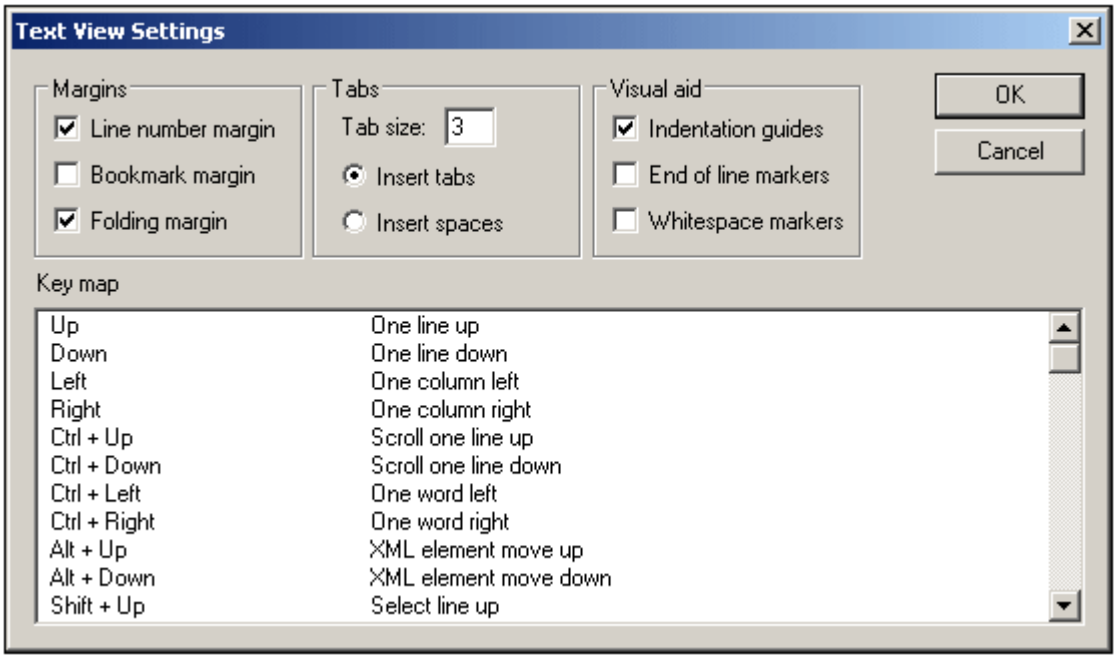
Select the file name, path name, or URL you are interested in, and choose this command from the View menu.

You can select:

-
- Some characters from within any item in the Text View.
- An enclosed string. If your text cursor is between quotes, XMLSpy will automatically use the entire string that is enclosed in the quotes.

12.9.14 Text View Settings

The **Text View Settings** command opens the Text View Settings dialog (*screenshot below*), in which you can configure Text View. The shortcut for the command is available as an icon in the Text toolbar.



Margins
In the Margins pane, the Line Number, Bookmark, and Source Folding margins can be toggled on and off. These are separate margins and display, respectively, line numbers, bookmarks, and source folding (icons to expand and collapse nodes). These settings determine whether the margins are displayed in Text View or not. Bookmark commands are in the **Edit** menu. To expand and collapse nodes, the Folding margin must be toggled on.

Tabs
The Tab pane enables you to set the tab size in terms of spaces. The radio buttons for inserting either tabs or spaces determine whether documents are displayed with tab or space indentation when pretty-printing with indentation is toggled on.

Visual Aid
The Visual Aid pane contains settings to toggle on indentation guides (dotted vertical lines that show the indentation of the text), end-of-line markers, and whitespace markers.

Key map
The key map is a list of XMLSpy shortcuts and their associated commands.

© 2011 Altova GmbH

12.10 Browser Menu

The commands in the **Browser** menu are enabled in Browser View only.

12.10.1 Back

Backspace (Alt + Left Arrow)

In Browser View, the **Back** command displays the previously viewed page. The Backspace key also achieves the same effect. The **Back** command is useful if you click a link in your XML document and want to return to your XML document.

In Schema View, the **Back** command takes you to the previously viewed component. The shortcut key is **Alt + Left Arrow**. Using the **Back** command up to 500 previously viewed positions can be re-viewed.

12.10.2 Forward

(Alt + Right Arrow)

The **Forward** command is only available once you have used the **Back** command. It moves you forward through (i) previously viewed pages in Browser View, and (ii) previous views of schema components in Schema View.

12.10.3 Stop

The **Stop** command instructs the browser to stop loading your document. This is useful if large external files or graphics are being downloaded over a slow Internet connection, and you wish to stop the process.

12.10.4 Refresh

F5

The **Refresh (F5)** command updates the Browser View by reloading the document and related documents, such as CSS and XSL stylesheets, and DTDs.

12.10.5 Fonts

The **Fonts** command allows you to select the default font size for rendering the text of your XML document. It is similar to the Font Size command in most browsers.

12.10.6 Separate Window

The **Separate Window** command opens the Browser View in a separate window, so that side-by-side viewing is possible. If you have separated the Browser View, press **F5** in editing view to automatically refresh the corresponding Browser View. To dock separate windows back into the interface, click the maximize button (at top right) of the active window.

© 2011 Altova GmbH

12.11 Tools Menu

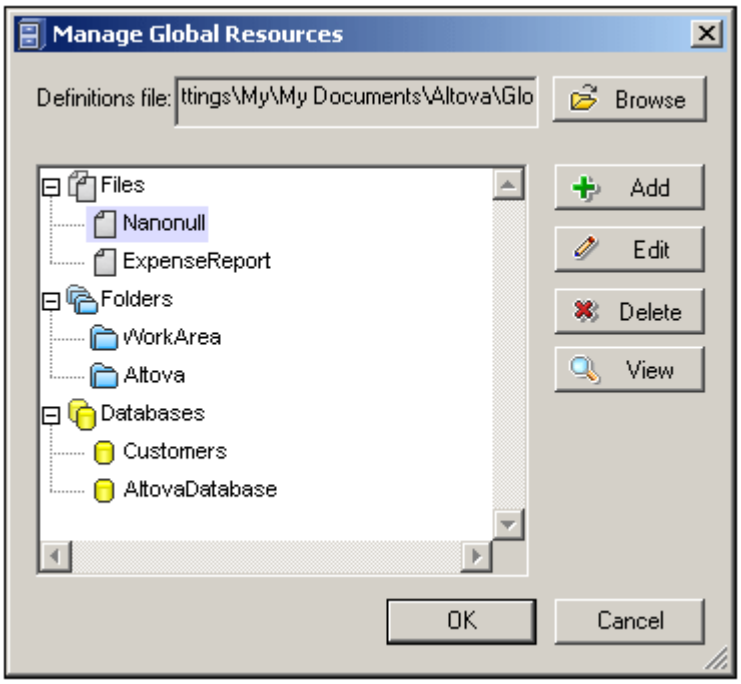
The tools menu allows you to:

-
- Compare any two files to check for differences
- Compare any two folders to check for differences
- Define global resources
- Change the active configuration for global resources in XMLSpy
- Customize your version of XMLSpy: define your own toolbars, keyboard shortcuts, menus, and macros
- Define global XMLSpy settings

12.11.1 Global Resources

The **Global Resources** command pops up the Global Resources dialog (*screenshot below*), in which you can:

- Specify the Global Resources XML File to use for global resources.
- Add file, folder, and database global resources (or aliases)
- Specify various configurations for each global resource (alias). Each configuration maps to a specific resource.

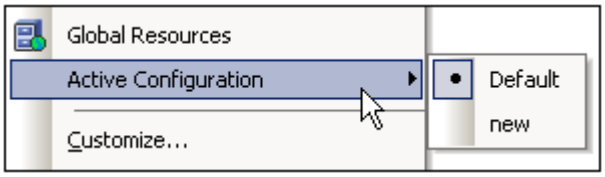


How to define global resources is described in detail in the section, Defining Global Resources.

Note: The Altova Global Resources dialog can also be accessed via the Global Resources toolbar (**Tools | Customize | Toolbars | Global Resources**).

12.11.2 Active Configuration

Mousing over the **Active Configuration** menu item rolls out a submenu containing all the configurations defined in the currently active Global Resources XML File (*screenshot below*).



The currently active configuration is indicated with a bullet. In the screenshot above the currently active configuration is `Default`. To change the active configuration, select the configuration you wish to make active.

Note: The active configuration can also be selected via the Global Resources toolbar (**Tools | Customize | Toolbars | Global Resources**).

12.11.3 Customize

The **Customize** command lets you customize XMLSpy to suit your personal needs.

Commands

The **Commands** tab allows you customize your menus or toolbars.

To add a command to a toolbar or menu:

1. Select the menu item **Tools | Customize**. The Customize dialog appears.
2. Select the **All Commands** category in the Categories list box. The available commands appear in the Commands list box.
3. Click on a command in the Commands list box and drag it to an to an existing menu or toolbar. An I-beam appears when you place the cursor over a valid position to drop the command.
4. Release the mouse button at the position you want to insert the command.

- A small button appears at the tip of mouse pointer when you drag a command. The "x" below the pointer means that the command cannot be dropped at the current cursor position.
- The "x" disappears whenever you can drop the command (over a tool bar or menu).
- Placing the cursor over a menu when dragging opens it, allowing you to insert the command anywhere in the menu.
- Commands can be placed in menus or tool bars. If you created you own toolbar you can populate it with your own commands/icons.

Please note:

You can also edit the commands in the **context menus** (right-click anywhere to open the context menu), using the same method. Click the **Menu** tab and then select the specific context menu available in the Context Menus combo box.

To delete a command or menu:

1. Select the menu item **Tools | Customize**. The Customize dialog appears.
2. Click on the menu entry or icon you want to delete, and drag with the mouse.

© 2011 Altova GmbH

3. Release the mouse button whenever the "x" icon appears below the mouse pointer. The command, or menu item, is deleted from the menu or tool bar.

Toolbars

The **Toolbars** tab allows you to activate or deactivate specific toolbars, as well as create your own specialized ones.

XMLSpy toolbars contain symbols for the most frequently used menu commands. For each symbol you get a brief "tool tip" explanation when the mouse cursor is directly over the item and the status bar shows a more detailed description of the command.

You can drag the toolbars from their standard position to any location on the screen, where they appear as a floating window. Alternatively you can also dock them to the left or right edge of the main window.

- Toolbar settings defined in Schema View and Text View are valid in those views. The Browser View toolbars are independent of all the other views.

To activate or deactivate a toolbar

- Click the check box to activate (or deactivate) the specific toolbar.

Apply changes for all views

- Check this check box at the bottom of the dialog to apply changes to all views (and not only to the current view). On clicking **Close**, all changes that were made **after** checking the check box will be applied to all views.

To create a new toolbar

1. Click the **New...** button, and give the toolbar a name in the Toolbar name dialog box.
2. Drag commands to the toolbar in the **Commands** tab of the Customize dialog box.

To reset the Menu Bar

1. Click the Menu Bar entry.
2. Click the **Reset** button, to reset the menu commands to the state they were in when XMLSpy was installed.

To reset all toolbar and menu commands

1. Click the **Reset All** button, to reset all the toolbar commands to the state they were when the program was installed. A prompt appears stating that all toolbars and menus will be reset.
2. Click **Yes** to confirm the reset.

To change a toolbar name

- Click the **Rename**... button to edit the name of the toolbar.

To delete a toolbar

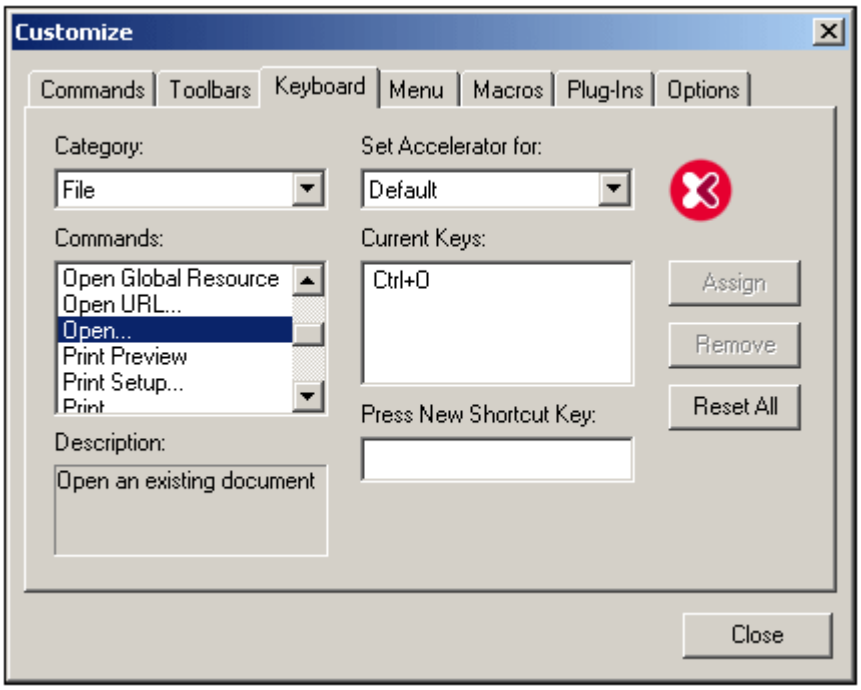
1. Select the toolbar you want to delete in the Toolbars list box.
2. Click the **Delete** button.
3. A prompt appears, asking if you really want to delete the toolbar. Click **Yes** to confirm the deletion.

Show text labels

This option displays explanatory text below toolbar icons when activated.

Keyboard

The **Keyboard** tab allows you to define (or change) keyboard shortcuts for any XMLSpy command.



To assign a new Shortcut to a command:

1. Select the All Commands category using the **Category** combo box.
2. Select the **command** you want to assign a new shortcut to, in the Commands list box
3. Click in the **Press New Shortcut Key**: text box, and press the shortcut keys that are to activate the command.
   The shortcuts appear immediately in the text box. If the shortcut was assigned previously, then that function is displayed below the text box.
4. Click the **Assign** button to assign the shortcut.
   The shortcut now appears in the Current Keys list box.
   (To **clear** this text box, press any of the control keys, **CTRL**, **ALT** or **SHIFT**).

To de-assign or delete a shortcut:

1. Click the shortcut you want to delete in the Current Keys list box.
2. Click the **Remove** button.
3. Click the **Close** button to confirm.

Set accelerator for:

Currently no function.

 © 2011 Altova GmbH

Currently assigned keyboard shortcuts:

Hotkeys by key

F1	Help Menu
F3	Find Next
F5	Refresh
F7	Check well-formedness
F8	Validate
F10	XSL Transformation
CTRL+F10	XSL:FO Transformation
F11	Step into
CTRL+F11	Step Over
Shift + F11	Step Out
Num +	Expand
Num -	Collapse
Num *	Expand fully
CTRL+Num-	Collapse unselected
CTRL + G	Goto line/char
CTRL+TAB	Switches between open documents
CTRL+F6	Cycle through open windows
Arrow keys (up / down)	Move selection bar
Esc.	Abandon edits/close dialog box
Return/Space bar	confirms a selection
Alt + F4	Closes XMLSpy
CTRL + F4	Closes active window
Alt + F, 1	Open last file
CTRL + Double click an element (Schema view)	Display element definition
CTRL + N	File New
CTRL + O	File Open
CTRL + S	File Save
CTRL + P	File Print
CTRL + A	Select All
Shift + Del	Cut (or CTRL + X)
CTRL + C	Copy
CTRL + V	Paste
CTRL + Z	Undo
CTRL + Y	Redo
Del	Delete (Delete item in Schema/)
CTRL + F	Find
F3	Find Next
CTRL + H	Replace

CTRL + I	Append Attribute
CTRL + E	In Grid View, Append Element. in Text View, Jump to Start/End Tag when cursor is in other member of the pair
CTRL + T	Append Text
CTRL + D	Append CDATA
CTRL + M	Append Comment
CTRL + SHIFT + I	Insert Attribute
CTRL + SHIFT + E	Insert Element
CTRL + SHIFT + T	Insert Text content
CTRL + SHIFT + D	Insert CDATA
CTRL + SHIFT + M	Insert Comment
CTRL + ALT + I	Add Child Attribute
CTRL + ALT + E	Add Child Element
CTRL + ALT + T	Add Child Text
CTRL + ALT + D	Add Child CDATA
CTRL + ALT + M	Add Child Comment

Hotkeys for Text View

CTRL + "+"	Zoom In
CTRL + "-"	Zoom Out
CTRL + 0	Reset Zoom
CTRL + mouse wheel forward	Zoom In
CTRL + mouse wheel back	Zoom Out

© 2011 Altova GmbH

Currently assigned keyboard shortcuts:

Hotkeys by function

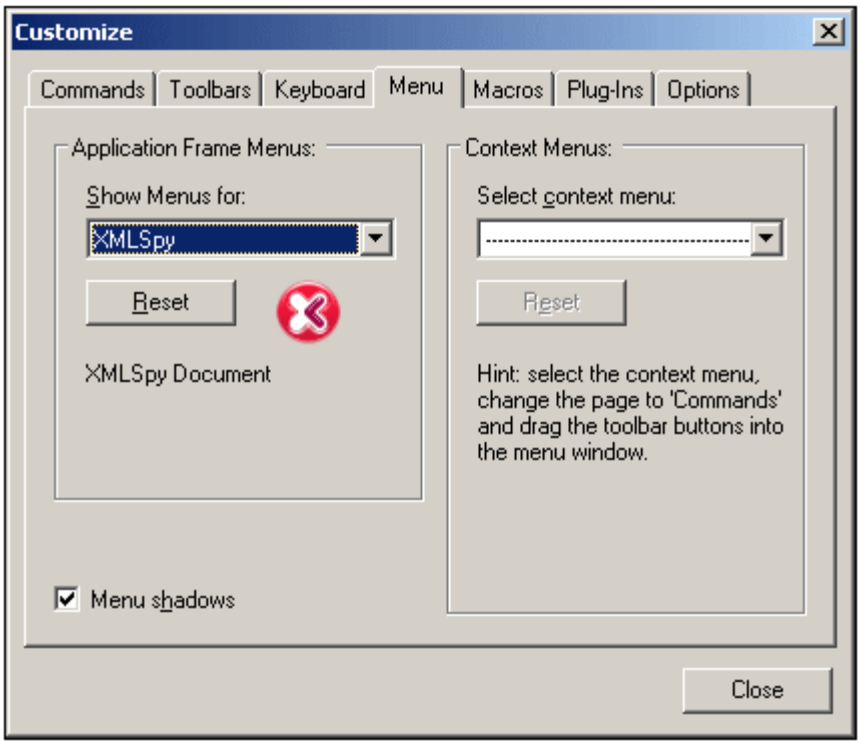
Abandon edits	Esc.
Add Child Attribute	CTRL + ALT + I
Add Child CDATA	CTRL + ALT + D
Add Child Comment	CTRL + ALT + M
Add Child Element	CTRL + ALT + E
Add Child Text	CTRL + ALT + T
Append Attribute	CTRL + I
Append CDATA	CTRL + D
Append Comment	CTRL + M
Append Element	CTRL + E (Grid View)
Append Text	CTRL + T
Check well-formedness	F7
Closes active window	CTRL + F4
Close XMLSpy	Alt + F4
Collapse	Num -
Collapse unselected	CTRL + Num-
Confirms a selection	Return / Space bar
Copy	CTRL + C
Cut	SHIFT + Del (or CTRL + X)
Cycle through windows	CTRL + TAB and CTRL + F6
Delete item	Del
Expand	Num +
Expand fully	Num *
File New	CTRL + N
File Open	CTRL + O
File Print	CTRL + P
File Save	CTRL + S
Find	CTRL + F
Find Next	F3
Goto line/char	CTRL + G
Help Menu	F1
Highlight other tag in pair when cursor is inside a start or end element tag	CTRL + E (Text View)
Insert Attribute	CTRL + SHIFT + I
Insert CDATA	CTRL + SHIFT + D
Insert Comment	CTRL + SHIFT + M
Insert Element	CTRL + SHIFT + E
Insert Text content	CTRL + SHIFT + T
Move selection bar	Arrow keys (up / down)
Open last file	Alt + F, 1
Paste	CTRL + V
Redo	CTRL + Y
Refresh	F5
Replace	CTRL + H
Select All	CTRL + A
Start Debugger/Go	Alt + F11

Step Into	F11
Step Out	Shift + F11
Step Over	CTRL + F11
To view an element definition	CTRL + Double click on an element.
Undo	CTRL + Z
Validate	F8
XSL Transformation	F10
XSL:FO Transformation	CTRL + F10

In the application, you can see a list of commands, together with their shortcuts and descriptions, in the Keyboard Map dialog (**Help | Keyboard Map**).

Menu

The **Menu** tab allows you to customize the main menu bars as well as the (popup - right click) context menus.



You can customize both the Default and XMLSpy menu bars.

The **Default** menu (*screenshot below*) is the one visible when no XML documents of any type are open in XMLSpy.

The XMLSpy menu is the menu bar visible when at least one XML document has been opened.

To customize a menu:

1. Select the menu bar you want to customize from the **Show Menus for:** combo box
2. Click the **Commands** tab, and drag the commands to the menu bar of your choice.

© 2011 Altova GmbH

To delete commands from a menu:

1. Click right on the command, or icon representing the command.
2. Select the **Delete** option from the popup menu,

or,

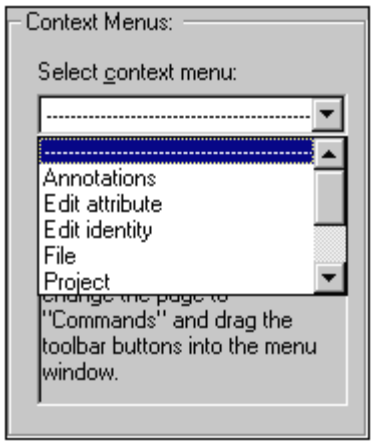
1. Select **Tools | Customize** to open the Customize dialog box.
2. Drag the command away from the menu, and drop it as soon as the check mark icon appears below the mouse pointer.

To reset either of the menu bars:

1. Select either the Default or XMLSpy entry in the **Show Menus for** combo box.
2. Click the **Reset** button just below the menu name.
   A prompt appears asking if you are sure you want to reset the menu bar.

To customize any of the Context menus (right-click menus):

1. Select the context menu from the **Select context menu** combo box. The context menu you selected appears.
2. Click the **Commands** tab, and drag the commands to the context menu.



To delete commands from a context menu:

1. Click right on the command, or icon representing the command.
2. Select the **Delete** option from the popup menu

or,

1. Select **Tools | Customize** to open the Customize dialog box.
2. Drag the command away from the context menu, and drop it as soon as the check mark icon appears below the mouse pointer.

To reset any of the context menus:

1. Select the context menu from the combo box, and
2. Click the **Reset** button just below the context menu name.
   A prompt appears asking if you are sure you want to reset the context menu.

To close a context menu window:

- Click on the **Close icon** at the top right of the title bar

or

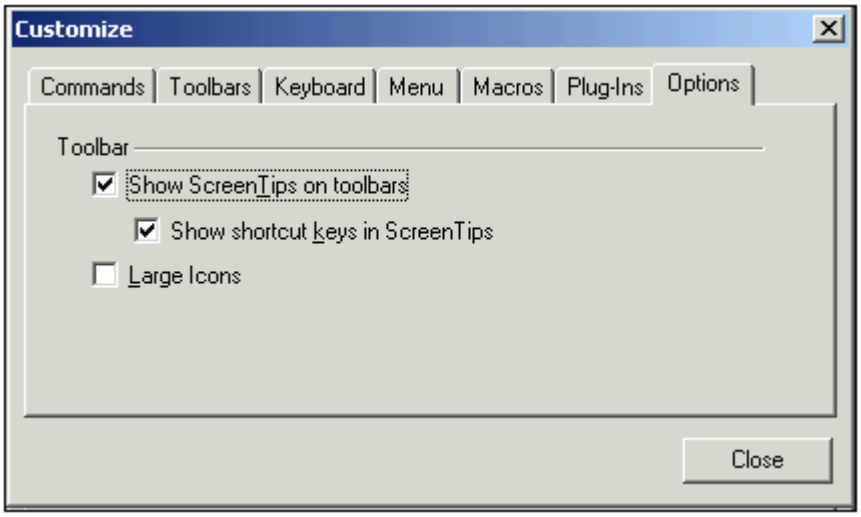
- Click the Close button of the Customize dialog box.

Menu shadows

- Click the **Menu shadows** check box, if you want all your menus to have shadows.

Options

The Options tab allows you to set general environment settings.



Toolbar

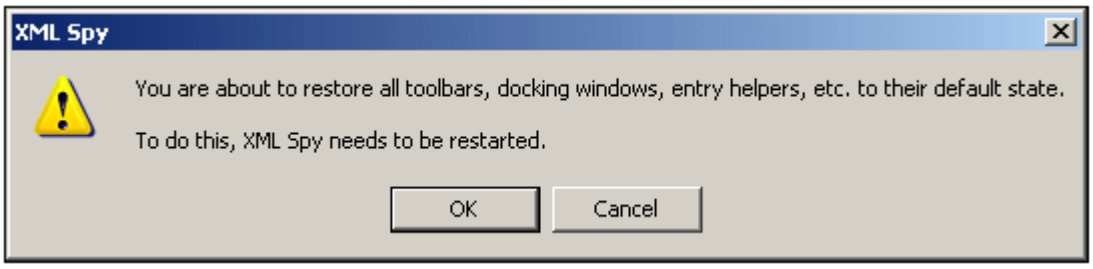
When active, the **Show ScreenTips on toolbars** check box displays a popup when the mouse pointer is placed over an icon in any of the icon bars. The popup contains a short description of the icon function, as well as the associated keyboard shortcut, if one has been assigned.

The **Show shortcut keys in ScreenTips** check box, allows you to decide if you want to have the shortcut displayed in the tooltip.

When active, the **Large icons** check box switches between the standard size icons, and larger versions of the icons.

12.11.4 Restore Toolbars and Windows

The **Restore Toolbars and Windows** command closes down XMLSpy and re-starts it with the default settings. Before it closes down a dialog pops up asking for confirmation about whether XMLSpy should be closed (*screenshot below*).



 © 2011 Altova GmbH

This command is useful if you have been resizing, moving, or hiding toolbars or windows, and would now like to have all the toolbars and windows as they originally were.

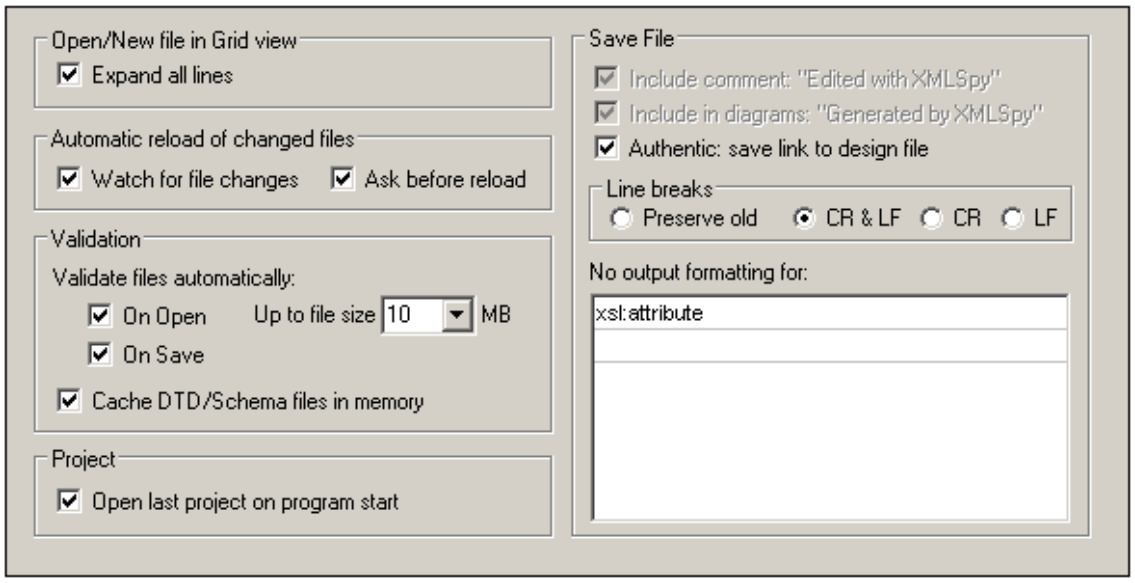
12.11.5 Options

The **Tools** | **Options** command enables you to define global application settings. These settings are specified in a tabbed dialog box and saved in the registry. They apply to all current and future document windows. The **Apply** button in the Options dialog displays the changes in the currently open documents and fixes the current settings. The changes are seen immediately in the background windows.

Each tab of the Options dialog is described in detail in this section.

File

The **File** tab defines the way XMLSpy opens and saves documents. Related settings are in the Encoding tab.



Open/New file in Grid view
You can choose to open an existing file or create a new file either in Grid View or in Text View. If you select Grid View, you can also choose to automatically expand all lines.

Automatic reload of changed files
If you are working in a multi-user environment, or if you are working on files that are dynamically generated on a server, you can watch for changes to files that are currently open in the interface. Each time XMLSpy detects a change in an open document, it will prompt you about whether you want to reload the changed file.

Validation
If you are using DTDs or schemas to define the structure of your XML documents, you can automatically check the document for validity whenever it is opened or saved. During Open and Save operations, you have the option of validating files only if the file-size is less than a size you specify in MB. If the document is not valid, an error message will be displayed. If it is valid, no message will be displayed and the operation will proceed without any notification. XMLSpy can also cache these files in memory to save any unnecessary reloading (e.g. when the schema

being referred to is accessed through a URL). If your schema location declaration uses an URL, disable the "cache DTD/Schema files in memory" option to have changes made to the schema appear immediately, and not use the cached version of the schema.

Project
When you start XMLSpy, you can open the last-used project automatically.

Save File
When saving an XML document, XMLSpy includes a short comment `<!-- Edited with XMLSpy http://www.altova.com -->` near the top of the file. This option can only be deactivated by licensed users, and takes effect when editing or saving files in the Enhanced Grid or Schema Design View.

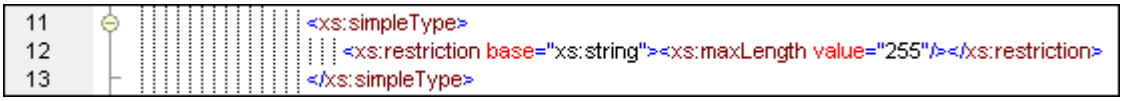
If a StyleVision Power Stylesheet is associated with an XML file, the 'Authentic: save link to design file' option will cause the link to the StyleVision Power Stylesheet to be saved with the XML file.

Line breaks
When you open a file, the character coding for line breaks in it are preserved if **Preserve old** is selected. Alternatively, you can choose to code line breaks in any of three codings: **CR&LF** (for PC), **CR** (for MacOS), or **LF** (for Unix).

No output formatting for
In Text View, the indentation of an element can be made to reflect its position in the element hierarchy (*see* **Save File**). You can, however, override this indentation for individual elements. To do this, enter the element name in the **No output formatting for** field. All elements entered in this field will be formatted such that their descendant elements have no whitespace between them (*see screenshots*).

Hierarchical indentation for all elements:

No output formatting has been specified for element `xs:restriction`:



After making the settings, click **OK** to finish and close the Options dialog.

File Types

The **File types** tab allows you to customize the behavior of XMLSpy on a per-file-type basis.

Choose a file type from the File Types list box to customize the functions for that particular file type:

Windows Explorer settings
You can define the file type description and MIME-compliant content type used by Windows Explorer and whether XMLSpy is to be the default editor for documents of this file type.

Conformance

 © 2011 Altova GmbH

XMLSpy provides specific editing and other features for various file types. The features for a file type are set by specifying the conformance in this option. XMLSpy lets you set file type to conform with XML, XQuery, ZIP, JSON, and other (text) grammars. Furthermore, XML conformance is differentiated between XML, DTD, and XML Entity file types. A large number of file types are defined with a default conformance that is appropriate for the file type. We recommend that you do not modify these settings unless you are adding a new file type or deliberately wish to set a file type to another kind of conformance.

Default view
This group lets you define the default view to be used for each file type. The screenshot above shows the Filetypes tab of the Enterprise edition. If your edition is not the Enterprise edition, it will have fewer views than shown in the screenshot.

Text View
This check box lets you set syntax-coloring for particular file types.

Disable automatic validation
This option enables you to disable automatic validation per file type. Automatic validation typically takes place when a file is opened or saved, or when a view is changed.

Save empty elements in short <E/> format
Some applications that use XML documents or output generated from XML documents may have problems understanding the short `<Element/>` form for empty elements defined in the XML 1.0 Specification. You can instruct XMLSpy to save elements in the longer (but also valid) `<Element></Element>` form.

Add new file extension
Adds a new file type to the File types list. You must then define the settings for this new file type using the other options in this tab.

Delete selected file extension
Deletes the currently selected file type and all its associated settings.

After making the settings, click **OK** to finish and close the Options dialog.

Editing

The **Editing** tab enables you to specify editing behaviour in XMLSpy.

Intelligent editing
While editing documents, XMLSpy provides intelligent editing based on these settings. You can also customize various aspects of the behavior of these Entry Helpers here. Such customization varies according to the file type. For example, the option to load entry helpers on opening the file and sorting attributes will not be applicable to DTD or XQuery documents.

Text View
The *Auto-complete* option automatically adds unambiguous structural components. For example, when the closing angular bracket of the start tag of an element is entered, then the end tag of that element is automatically added if this option is enabled.

In Text View, Auto-completion and entry helpers can be disabled if a file is bigger than the size specified in the *Disable Auto-completion...* combo box. This is useful if you wish to speed up the editing of large files and can do without the auto-completion feature and entry helpers. If the file size is bigger than that specified for this option, then the Text View context menu contains a toggle command for switching on and off Auto-completion and entry helper use. So you can always switch these editing aids on and off at any time during editing (in the event of files having

a size greater than the size specified for this option). If the value specified for this option is smaller than the size of the opened file, locations indicated in error messages will not correctly correspond to the location in Text View.

After making the settings, click **OK** to finish and close the Options dialog.

View

The **View** tab enables you to customize the XML documents presentation in XMLSpy.

Pretty-print
When you select **Edit | Pretty-Print XML Text**, the XML document will be pretty-printed with or without indentation according to whether the Use Indentation option in this dialog is checked or not. When the document is pretty-printed with indentation, it will be indented according to the settings in the Tabs pane of the Text View Settings dialog. If the document is pretty-printed, every line in the document will be displayed with zero indentation. If pretty-printing has been switched on with the *Use Indentation* setting and if you change from Text View to Grid View and back to Text View, then the document will be pretty-printed automatically.

Program logo
You can turn off the splash screen upon program startup to speed up the application.

Window title
The window title for each document window can contain either the file name only or the full path name.

After making the settings, click **OK** to finish and close the Options dialog.

Grid Fonts

The **Grid fonts** tab allows you to customize the appearance of text in Grid View.

Font face
You can select the font face and size to be used for displaying the various items in Grid View. The same fonts are also used for printing, so only TrueType fonts should be selected.

Size
Select the required size. If you want to use the same font size for all items, check the **Use the same for all** check box.

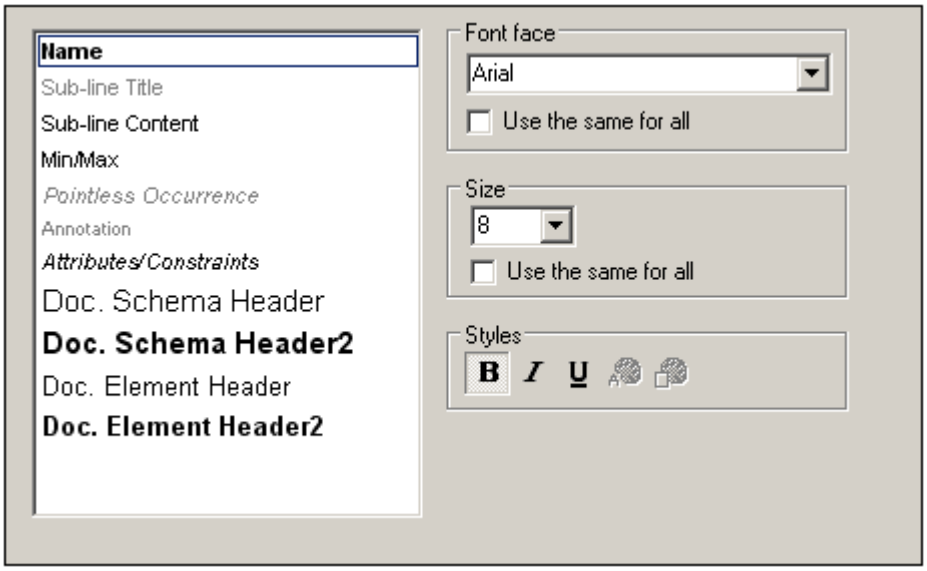
Styles
The style and color can be set using the options in this pane. The current settings are immediately reflected in the list in the left-hand pane, so you can preview the way your document will look.

After making the settings, click **OK** to finish and close the Options dialog.

Schema Fonts

The **Schema fonts** tab enables you to customize the appearance of text in Schema View.

 © 2011 Altova GmbH



Font face
You can select the font face and size to be used for displaying the various items in the Schema Design view.

Size
Select the required size. If you want to use the same font size for all items, click on the Use The Same For All" check box.

Styles
The style and color can be set using the options in this pane. The current settings are immediately reflected in the list in the left pane, so you can preview the way your document will look.

After making the settings, click **OK** to finish and close the Options dialog.

Text Fonts

The **Text fonts** tab enables you to customize the appearance of text in Text View. You can customize the appearance of of text items according to the type of text item. For example, you can color element names and attribute names differently.

The text item types are categorized into three groups:

- XML generic
- XQuery
- CSS
- JSON

To customize text fonts, do the following:

1. In the combo box at top left, select the type of document for which you wish to customize text fonts. On doing this, the text item types for that document type appear in the box below the combo box. (*In the screenshot above,* `XML generic` *has been selected as the document type.*)
2. Select the text item type you wish to customize by clicking it. (*In the screenshot above,* `Element names` *has been selected.*)
3. Set the font properties using the options in the panes on the right-hand side. You can

select the font-family, font-size, font-style, font-color, and background-color for the text. Additionally, you can also select a background color for the entire Text View.

Note: The same font, style, and size is used for all text item types. Only the text color and background color can be changed for individual text types. This enables the syntax coloring feature.

After making the settings, click **OK** to finish and close the Options dialog.

Colors

The **Colors** tab enables you to customize the background colors used in the Table View of Grid View. In the screenshot below, the colors have been changed from the default colors by clicking the palette icon next to each item and then selecting the preferred color.

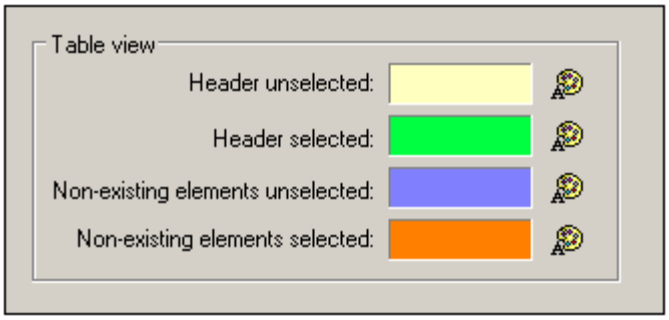
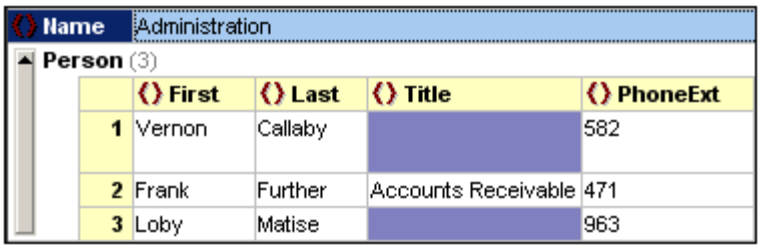
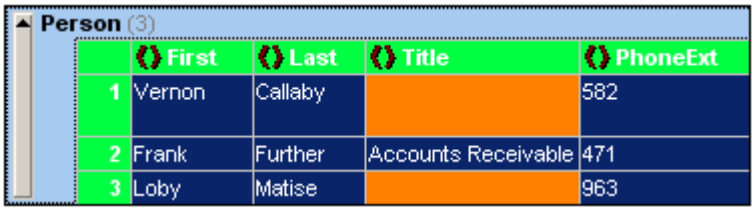


Table View

The Header unselected and Header selected options refer to the column and row headers. The screenshot below shows headers unselected; its color is as set in the dialog above.



The Header Selected color is activated when all headers are selected (*screenshot below*)—not when individual headers are selected. The screenshot below shows this using the colors defined in the dialog shown above. All headers can be selected by clicking the cell that intersects both headers or by selecting the element created as the table—or any of its ancestors.

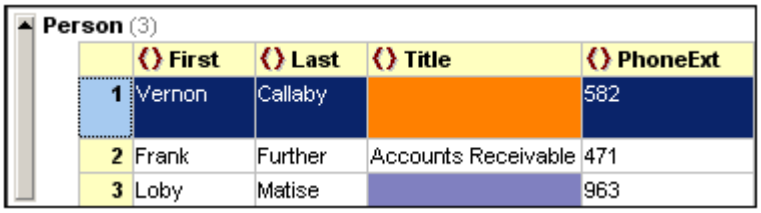


Non-existent Elements

When an element or attribute does not exist in the XML document, then it can be given different background colors when selected and unselected. This is shown in the screenshot below, in

 © 2011 Altova GmbH

which the first row is selected.



Please note: In addition to the colors you define here, XMLSpy uses the regular selection and menu color preferences set in the Display Settings in the Control Panel of your Windows installation.

After making the settings, click **OK** to finish and close the Options dialog.

Encoding

The **Encoding** tab specifies options for file encodings.

Default encoding for new XML files
The default encoding for new XML files can be set by selecting an option from the dropdown list. A new document is created with an XML declaration containing the encoding value you specify here. If a two- or four-byte encoding is selected as the default encoding (i.e. UTF-16, UCS-2, or UCS-4) you can also choose between little-endian and big-endian byte-ordering.

The encoding of existing XML files will be retained and can only be changed with the **File | Encoding** command.

Open XML files with unknown encoding as
If the encoding of an XML file cannot be determined or if the XML document has no encoding specification, the file will be opened with the encoding you select in this combo box.

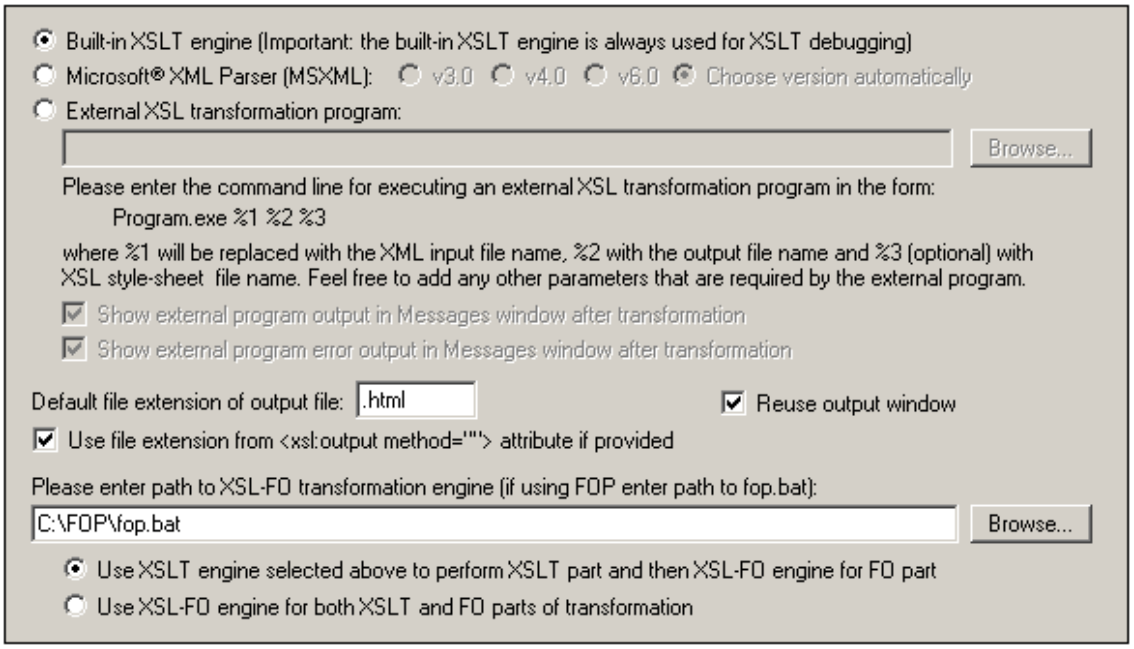
Open non-XML files in
Existing and new non-XML files are opened with the encoding you select in this combo box. You can change the encoding of the document by using the **File | Encoding** command.

BOM (Byte Order Mark)
When a document with two-byte or four-byte character encoding is saved, the document can be saved either with (i) little-endian byte-ordering and a little-endian BOM (*Always create BOM if not UTF-8*); or (ii) the detected byte-ordering and the detected BOM (*Preserve detected BOM on saving*).

After making the settings, click **OK** to finish and close the Options dialog.

XSL

The **XSL** tab (*screenshot below*) enables you to define options for XSLT transformations and XSL-FO transformations carried out from within the application.



XSLT transformations
XMLSpy contains the Altova XSLT 1.0 Engine and Altova XSLT 2.0 Engine, which you can use for XSLT transformations. The appropriate XSLT engine (1.0 or 2.0) is used (according to the value of the `version` attribute of the `xsl:stylesheet` or `xsl:transform` element).

For transforming XML documents using XSLT, you could use one of the following:

- The built-in Altova XSLT Engine (comprising the Altova XSLT 1.0 Engine and the Altova XSLT 2.0 Engine).
- The MSXML 3.0, 4.0, or 6.0 parser (which is pre-installed). If you know which version of the MSXML parser is running on your machine, you could select it; otherwise, you should let the application select the version automatically. (The *Choose version automatically* option is active by default.) In this case, the application tries to select the most recent available version.
- An external XSLT processor of your choice. You must specify the command line string for the external XSLT processor. The following variables are available for building the command line string:

  `%1` = XML document to process
  `%2` = Output file to generate
  `%3` = XSLT stylesheet to use (if the XML document does not contain a reference to a stylesheet)

  For example, the command to run a simple transformation with the Saxon (XSLT 1.0) processor is:

  ```
  saxon.exe -o output.xml input.xml stylesheet.xslt
  parameter-name=parameter-value
  ```

  To run this command from the application, select the External XSL Transformation Program radio button, and enter the following line in the text box:

  ```
  c:\saxon\saxon.exe -o %2 %1 %3 parameter-name=parameter-value
  ```

  Check the respective check boxes to show the output and error messags of the external program in the Messages Window in XMLSpy.

 © 2011 Altova GmbH

Note: The parameters set in XMLSpy's XSLT Input Parameters dialog are passed to the internal Altova XSLT Engines only. They are not passed to any other XSLT Engine that is set up as the default XSLT processor.

XSL-FO transformations
FO documents are processed using an FO processor, and the path to the executable of the FO processor must be specified in the text box for the XSL-FO transformation engine. The transformation is carried out using the **XSL/XQuery | XSL-FO Transformation** menu command. If the source file (the active document when the command is executed in the IDE) is an XSL-FO document, the FO processor is invoked for the transformation. If the source document is an XML document, an XSLT transformation is required to first convert the XML document to an XSL-FO document. This XSLT transformation can be carried out either by the XSLT engine you have specified as the default engine for the application (*see above*), or by the XSLT engine that might be built into the FO processor you have specified as the default FO processor for the application. To select between these two options, click the appropriate radio button.

After making the settings, click **OK** to finish and close the Options dialog.

Source Control

The **Source Control** tab (*screenshot below*) enables you to specify the source control provider, and the settings and default logon ID for each source control provider.

Source Control Plugin
The current source control plugin can be selected from among the currently installed source control systems. These systems are listed in the dropdown list of the combo box. After selecting the required source control, specify the login ID for it in the next text box. The **Advanced** button pops up a dialog specific to the selected source control plugin, in which you can define settings for that source control plugin. These settings are different for different source control plugins.

User preferences
A range of user preferences is available, including the following:

- Status updates can be performed in the background after a user-defined interval of time, or they can be switched off entirely. Very large source control databases could consume considerable CPU and network resources. The system can be speeded up, however, by disabling background status updates or increasing the interval between them..
- When opening and closing projects, files can be automatically checked out and checked in, respectively.
- The display of the Check Out and Check In dialogs can be suppressed.
- The **Reset** button is enabled if you have checked/activated the *Don't show this again* option in one of the dialog boxes. On clicking the **Reset** button, the *Don't show this again* prompt is re-enabled.

After making the settings, click **OK** to finish and close the Options dialog.

12.12 Window Menu

To organize the individual document windows in an XMLSpy session, the **Window** menu contains standard commands common to most Windows applications.

You can cascade the open document windows, tile them, or arrange document icons once you have minimized them. You can also switch the various Entry Helper windows on or off, or switch to an open document window directly from the menu.

12.12.1 Cascade

This command rearranges all open document windows so that they are all cascaded (i.e. staggered) on top of each other.

12.12.2 Tile Horizontally

This command rearranges all open document windows as **horizontal tiles**, making them all visible at the same time.

12.12.3 Tile Vertically

This command rearranges all open document windows as **vertical tiles**, making them all visible at the same time.

12.12.4 Project Window

This command lets you switch the Project Window on or off.

This is a dockable window. Dragging on its title bar detaches it from its current position and makes it a floating window. Click right on the title bar, to allow docking or hide the window.

12.12.5 Info Window

This command lets you switch the Info Window on or off.

This is a dockable window. Dragging on its title bar detaches it from its current position and makes it a floating window. Click right on the title bar, to allow docking or hide the window.

12.12.6 Entry Helpers

This command lets you switch all three Entry-Helper Windows on or off.

All three Entry helpers are dockable windows. Dragging on a title bar detaches it from its current position and makes it a floating window. Click right on the title bar to allow docking or hide the window.

12.12.7 Output Windows

The Output Windows are a set of tabbed output windows, such as the Messages window (which displays messages like validation results), the Find in Files window, and the XPath window (which shows XPath evaluation results). The initial setting is for them to open at below the Main Window. The Output Windows command lets you switch the Output Windows on or off.

The Output Windows window is dockable. Dragging on its title bar detaches it from its current position and makes it a floating window. Click right on the title bar to allow docking or to hide the window.

© 2011 Altova GmbH

For a complete description of Output Windows see Output Windows in the section, Text View.

12.12.8 Project and Entry Helpers

This command toggles on and off the display of the Project Window and the Entry Helpers together.

12.12.9 All On/Off

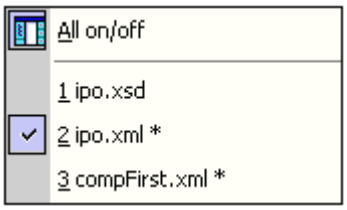
This command lets you switch all dockable windows on, or off:

- the Project Window
- the Info Window
- the three Entry-Helper Windows
- the Output Windows

This is useful if you want to hide all non-document windows quickly, to get the maximum viewing area for the document you are working on.

12.12.10 Currently Open Window List

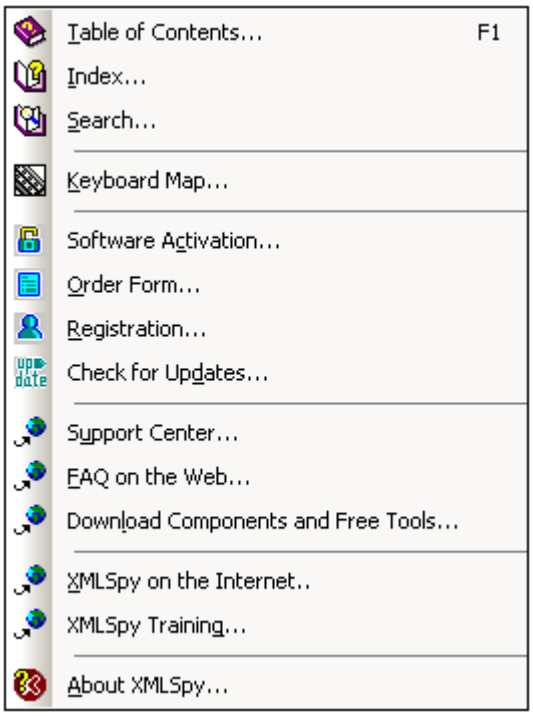
This list shows all currently open windows, and lets you quickly switch between them.



You can also use the Ctrl-TAB or CTRL F6 keyboard shortcuts to cycle through the open windows.

12.13 Help Menu

The **Help** menu contains all commands required to get help or more information on XMLSpy, as well as links to information and support pages on our web server.



The **Help** menu also contains the Registration dialog, which lets you enter your license key-code once you have purchased the product.

12.13.1 Table of Contents

The **Help | Table of contents** command displays a **hierarchical representation** of all chapters and topics contained in the online help system. Use this command to jump to the table of contents directly from within XMLSpy.

Once the help window is open, use the three tabs to toggle between the table of contents, index , and search panes. The Favorites tab lets you bookmark certain pages within the help system.

12.13.2 Index

The **Help | Index** command accesses the **keyword index** of the Online Help. You can also use the Index tab in the left pane of the online help system.

The index lists all relevant keywords and lets you navigate to a topic by double-clicking the respective keyword. If more than one topic matches the selected keyword, you are presented a list of available topics to choose from.

12.13.3 Search

The **Help | Search** command performs a **full-text search** on the entire online help system.

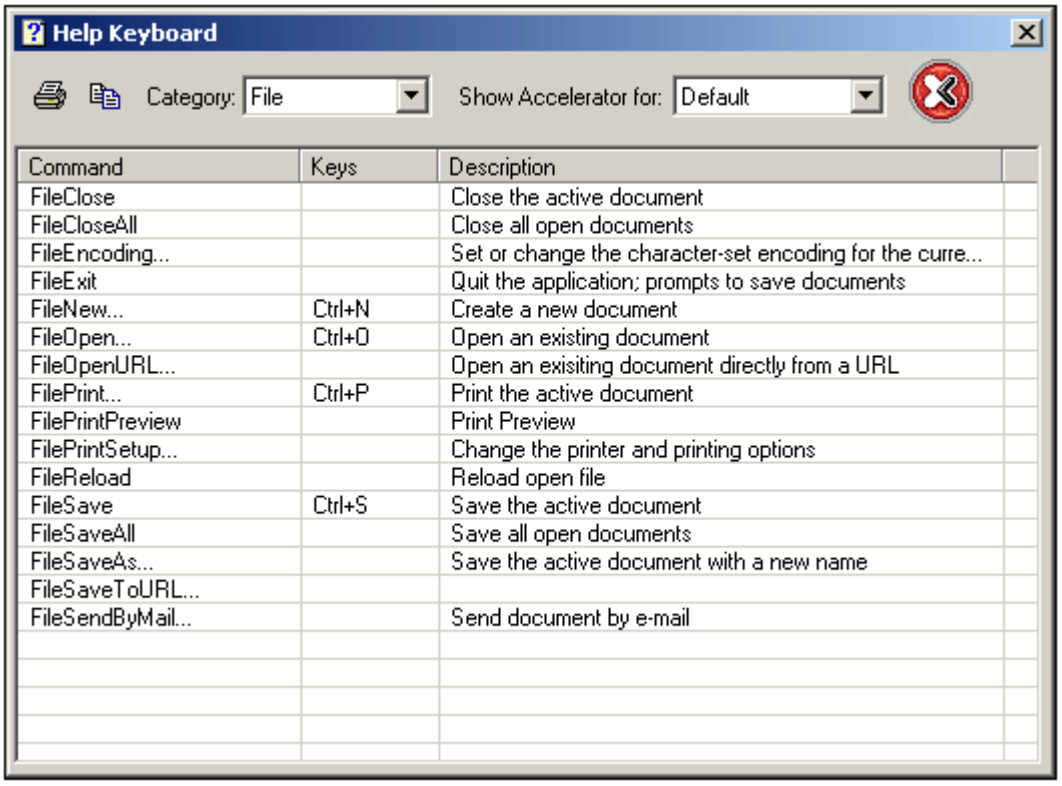
1. Enter your search term in the query field and press **Enter**.
   The online help system displays a list of available topics that contain the search term

 © 2011 Altova GmbH

you've entered.
2. Double-click on any item in the list to display the corresponding topic.

12.13.4 Keyboard Map

The **Help | Keyboard Map...** command causes an information box to be displayed that contains a menu-by-menu listing of all commands in XMLSpy. Menu commands are listed with a description and shortcut keystrokes for the command.



To view commands in a particular menu, select the menu name in the Category combo box. You can print the command by clicking the printer icon.

You should note the following points about shortcuts:

- Certain commands (and their shortcuts) are applicable only within a certain view. For example, most of the commands in the XML menu are applicable only in Grid View. Other commands (such as **File | Save** or **XML | Check Well-Formedness**) are available in multiple views.
- Other cool shortcuts: For example, **Shift+F10** brings up the context menu in Text View and Schema View; **Ctrl+E** when the cursor is inside an element start or end tag in Text View moves the cursor to the end or start tag, respectively.
- In the Keyboard tab of the Customize dialog, you can also set your own shortcuts for various menu commands.

12.13.5 Activation, Order Form, Registration, Updates

Software Activation
After you download your Altova product software, you can activate it using either a free evaluation key or a purchased permanent license key.

- ***Free evaluation key.*** When you first start the software after downloading and installing it, the Software Activation dialog will pop up. In it is a button to request a free evaluation key-code. Enter your name, company, and e-mail address in the dialog that appears, and click Request Now! The evaluation key is sent to the e-mail address you entered and should reach you in a few minutes. Now enter the key in the key-code field of the Software Activation dialog box and click **OK** to start working with your Altova product. The software will be unlocked for a period of 30 days.
- ***Permanent license key.*** The Software Activation dialog contains a button to purchase a permanent license key. Clicking this button takes you to Altova's online shop, where you can purchase a permanent license key for your product. There are two types of permanent license: single-user and multi-user. Both will be sent to you by e-mail. A *single-user license* contains your license-data and includes your name, company, e-mail, and key-code.A *multi-user license* contains your license-data and includes your company name and key-code. Note that your license agreement does not allow you to install more than the licensed number of copies of your Altova software on the computers in your organization (per-seat license). Please make sure that you enter the data required in the registration dialog exactly as given in your license e-mail.

Note: When you enter your license information in the Software Activation dialog, ensure that you enter the data exactly as given in your license e-mail. For multi-user licenses, each user should enter his or her own name in the Name field.

The Software Activation dialog can be accessed at any time by clicking the **Help | Software Activation** command.

Order Form
When you are ready to order a licensed version of the software product, you can use either the **Order license key** button in the Software Activation dialog (*see previous section*) or the **Help | Order Form** command to proceed to the secure Altova Online Shop.

Registration
The first time you start your Altova software after having activated it, a dialog appears asking whether you would like to register your product. There are three buttons in this dialog:

- **OK:** Takes you to the Registration Form
- **Remind Me Later:** Pops up a dialog in which you can select when you wish to be next reminded.
- **Cancel:** Closes the dialog and suppresses it in future. If you wish to register at a later time, you can use the **Help | Registration** command.

Check for Updates
Checks with the Altova server whether a newer version than yours is currently available and displays a message accordingly.

12.13.6 Support Center, FAQ, Downloads

Support Center
If you have any questions regarding our product, please feel free to use this command to send a query to the Altova Support Center at any time. This is the place where you'll find links to the FAQ, support form, and e-mail addresses for contacting our support staff directly.

FAQ

To help you in getting the best support possible, we are providing a list of Frequently Asked Questions (FAQ) on the Internet, that is constantly updated as our support staff encounters new issues that are raised by our customers.

Please make sure to check the FAQ before contacting our technical support team. This will allow you to get help more quickly.

We regret that we are not able to offer technical support by phone at this time, but our support staff will typically answer your e-mail requests within one business day.

If you would like to make a feature suggestion for a future version of XMLSpy or if you wish to send us any other general feedback, please use the questionnaire form.

Download components and free tools
This command is a link to the Components Download page at the Altova website, from where you can download components, free tools, and third-party add-ins that you can use in your XML development environment. Included among these are the Altova Validator, and XSLT and XQuery engines.

12.13.7 On the Internet

On the Internet
This command takes you directly to the Altova web-server http://www.altova.com where you can find out about news, product updates and additional offers from the Altova team.

Training
The **Training** command takes you to the Online Training page on the Altova website. Here you can enroll for online courses to help you develop expertise in using Altova products.

12.13.8 About

This command shows the XMLSpy splash screen and copyright information dialog box, which includes the XMLSpy logo.

Please note:
This dialog box shows the version number - to find the number of the actual build you are using, please look at the status bar, which always includes the full version and build number.

12.14 Command Line

Certain XMLSpy actions can be carried out from the command line. These commands are listed below:

Open a file
Command: `xmlspy.exe file.xml`
Action: Opens the file, `file.xml`, in XMLSpy

Note: If an XML file has an SPS file already assigned to it, then the XML file is opened in Authentic View. Otherwise, the XML file is opened in Text View. If an SPS file is not assigned, one can be assigned with the `/sps` flag (*see below*).

Open multiple files
Command: `xmlspy.exe file1.xml file2.xml`
Action: Opens the files, `file1.xml` and `file2.xml`, in XMLSpy

Assign an SPS file to an XML file for Authentic View editing
Command: `xmlspy.exe myxml.xml /sps mysps.sps`
Action: Opens the file, `myxml.xml` in Authentic View with `mysps.sps` as its SPS file. The `/sps` flag specifies that the SPS file that follows is to be used with the XML file that precedes the `/sps` flag (for Authentic View editing).

Open a new XML template file via an SPS file
Command: `xmlspy.exe mysps.sps`
Action: Opens a new XML file in Authentic View. The display will be based on the SPS and the new XML file will have a skeletal structure based on the SPS schema. The name of the newly created XML file must be assigned when saving the XML file.

Open an SPS file as an XML document in Text View
Command: `xmlspy.exe /raw mysps.sps`
Action: Opens the file `mysps.sps` as an XML document in Text View. The `/raw` flag specifies that the SPS file that follows is to be edited as an XML file.

 © 2011 Altova GmbH

Altova XMLSpy 2011

Appendices

Appendices

These appendices contain technical information about XMLSpy and important licensing information. Each appendix contains sub-sections as given below:

Engine Information

- Altova XSLT 1.0 Engine
- Altova XSLT 2.0 Engine
- Altova XQuery 1.0 Engine
- XPath 2.0 and XQuery 1.0 Functions
- Extensions

Technical Data

- OS and memory requirements
- Altova XML Parser
- Altova XSLT and XQuery Engines
- Unicode support
- Internet usage

License Information

- Electronic software distribution
- Software activation and license metering
- Copyrights
- End User License Agreement

© 2011 Altova GmbH

1 Engine Information

This section contains information about implementation-specific features of the Altova XSLT 1.0 Engine, Altova XSLT 2.0 Engine, and Altova XQuery 1.0 Engine.

1.1 XSLT 1.0 Engine: Implementation Information

The Altova XSLT 1.0 Engine is built into Altova's XMLSpy, StyleVision, Authentic, and MapForce XML products. It is also available in the free AltovaXML package. The Altova XSLT 1.0 Engine implements and conforms to the World Wide Web Consortium's XSLT 1.0 Recommendation of 16 November 1999 and XPath 1.0 Recommendation of 16 November 1999 . Limitations and implementation-specific behavior are listed below.

Limitations

- The **xsl:preserve-space** and **xsl:strip-space** elements are not supported.
- When the `method` attribute of `xsl:output` is set to HTML, or if HTML output is selected by default, then special characters in the XML or XSLT file are inserted in the HTML document directly as special characters; they are not inserted as HTML character references in the output. For instance, the character ` ` (the decimal character reference for a non-breaking space) is not inserted as ` ` in the HTML code, but directly as a non-breaking space.

Implementation's handling of whitespace-only nodes in source XML document

The XML data (and, consequently, the XML Infoset) that is passed to the Altova XSLT 1.0 Engine is stripped of boundary-whitespace-only text nodes. (A boundary-whitespace-only text node is a whitespace-only text node that occurs between two elements within an element of mixed content.) This stripping may have an effect on the value returned by the `fn:position()`, `fn:last()`, and `fn:count()` functions.

For any node selection that selects text nodes also, boundary-whitespace-only text nodes would typically also be included in the selection. However, since the XML Infoset used by the Altova engines has boundary-whitespace-only text nodes stripped from it, these nodes are not present in the XML Infoset. As a result, the size of the selection and the numbering of nodes in the selection will be different than that for a selection which included these text nodes. The `fn:position()`, `fn:last()`, and `fn:count()` functions, therefore, could produce results that are different from those produced by some other processors.

A situation in which boundary-whitespace-only text nodes are evaluated as siblings of other elements arises most commonly when `xsl:apply-templates` is used to apply templates. When the `fn:position()`, `fn:last()`, and `fn:count()` functions are used in patterns with a name test (for example, `para[3]`, which is short for `para[position()=3]`), boundary-whitespace-only nodes are irrelevant since only the named elements (`para` in the above example) are selected. (Note, however, that boundary-whitespace-only nodes **are** relevant in patterns that use the wildcard, for example, `*[10]`.)

Note: If a boundary-whitespace-only text node is required in the output, then insert the required whitespace within one of the two adjoining child elements. For example, the XML fragment:

```
<para>This is <b>bold</b> <i>italic</>.</para>
```

when processed with the XSLT template

```
<xsl:template match="para">
   <xsl:apply-templates/>
</xsl:template>
```

will produce:

```
This is bolditalic.
```

 © 2011 Altova GmbH

To get a space between `bold` and `italic` in the output, insert a space character within either the `<b>` or `<i>` elements in the XML source. For example:

```
<para>This is <b>bold</b> <i> italic</i>.</para> or
<para>This is <b>bold&#x20;</b> <i>italic</i>.</para> or
<para>This is <b>bold</b><i>&#x20;italic</i>.</para>
```

When any of the `para` elements above is processed with the same XSLT template given above, it will produce:

```
This is bold italic.
```

1.2 XSLT 2.0 Engine: Implementation Information

The Altova XSLT 2.0 Engine is built into Altova's XMLSpy, StyleVision, Authentic, and MapForce XML products. It is also available in the free AltovaXML package. This section describes the engine's implementation-specific aspects of behavior. It starts with a section giving general information about the engine, and then goes on to list the implementation-specific behavior of XSLT 2.0 functions.

For information about implementation-specific behavior of XPath 2.0 functions, see the section, XPath 2.0 and XQuery 1.0 Functions.

1.2.1 General Information

The Altova XSLT 2.0 Engine conforms to the World Wide Web Consortium's (W3C's) XSLT 2.0 Recommendation of 23 January 2007. Note the following general information about the engine.

Backwards Compatibility

The Altova XSLT 2.0 Engine is backwards compatible. The only time the backwards compatibility of the XSLT 2.0 Engine comes into play is when using the XSLT 2.0 Engine of Altova XML to process an XSLT 1.0 stylesheet. Note that there could be differences in the outputs produced by the XSLT 1.0 Engine and the backwards-compatible XSLT 2.0 Engine.

In all other Altova products, the backwards-compatibility issue never arises. This is because these products automatically select the appropriate engine for the transformation. For example, consider that in XMLSpy you specify that a certain XML document be processed with an XSLT 1.0 stylesheet. When the transformation command is invoked, XMLSpy automatically selects the XSLT 1.0 Engine of XMLSpy to carry out the transformation.

Note: The stylesheet version is specified in the `version` attribute of the `stylesheet` or `transform` element of the stylesheet.

Namespaces

Your XSLT 2.0 stylesheet should declare the following namespaces in order for you to be able to use the type constructors and functions available in XSLT 2.0. The prefixes given below are conventionally used; you could use alternative prefixes if you wish.

Namespace Name	Prefix	Namespace URI
XML Schema types	`xs:`	`http://www.w3.org/2001/XMLSchema`
XPath 2.0 functions	`fn:`	`http://www.w3.org/2005/xpath-functions`

Typically, these namespaces will be declared on the `xsl:stylesheet` or `xsl:transform` element, as shown in the following listing:

```
<xsl:stylesheet version="2.0"
   xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
   xmlns:xs="http://www.w3.org/2001/XMLSchema"
   xmlns:fn="http://www.w3.org/2005/xpath-functions"
   ...
</xsl:stylesheet>
```

The following points should be noted:

 © 2011 Altova GmbH

- The Altova XSLT 2.0 Engine uses the XPath 2.0 and XQuery 1.0 Functions namespace (listed in the table above) as its **default functions namespace**. So you can use XPath 2.0 and XSLT 2.0 functions in your stylesheet without any prefix. If you declare the XPath 2.0 Functions namespace in your stylesheet with a prefix, then you can additionally use the prefix assigned in the declaration.
- When using type constructors and types from the XML Schema namespace, the prefix used in the namespace declaration must be used when calling the type constructor (for example, `xs:date`).
- With the CRs of 23 January 2007, the `untypedAtomic` and duration datatypes (`dayTimeDuration` and `yearMonthDuration`), which were formerly in the XPath Datatypes namespace (typically prefixed `xdt:`) have been moved to the XML Schema namespace.
- Some XPath 2.0 functions have the same name as XML Schema datatypes. For example, for the XPath functions `fn:string` and `fn:boolean` there exist XML Schema datatypes with the same local names: `xs:string` and `xs:boolean`. So if you were to use the XPath expression `string('Hello')`, the expression evaluates as `fn:string('Hello')` —not as `xs:string('Hello')`.

Schema-awareness
The Altova XSLT 2.0 Engine is schema-aware.

Whitespace in XML document
By default, the Altova XSLT 2.0 Engine strips all boundary whitespace from boundary-whitespace-only nodes in the source XML document. The removal of this whitespace affects the values that the `fn:position()`, `fn:last()`, `fn:count()`, and `fn:deep-equal()` functions return. For more details, see Whitespace-only Nodes in XML Document in the XPath 2.0 and XQuery 1.0 Functions section.

Note: If a boundary-whitespace-only text node is required in the output, then insert the required whitespace within one of the two adjoining child elements. For example, the XML fragment:

```
<para>This is <b>bold</b> <i>italic</>.</para>
```

when processed with the XSLT template

```
<xsl:template match="para">
   <xsl:apply-templates/>
</xsl:template>
```

will produce:

```
This is bolditalic.
```

To get a space between `bold` and `italic` in the output, insert a space character within either the `<b>` or `<i>` elements in the XML source. For example:

```
<para>This is <b>bold</b> <i> italic</>.</para>  or
<para>This is <b>bold&#x20;</b> <i>italic</>.</para>  or
<para>This is <b>bold</b><i>&#x20;italic</>.</para>
```

When such an XML fragment is processed with the same XSLT template given above, it will produce:

```
This is bold italic.
```

XSLT 2.0 elements and functions

Limitations and implementation-specific behavior of XSLT 2.0 elements and functions are listed in the section XSLT 2.0 Elements and Functions.

XPath 2.0 functions
Implementation-specific behavior of XPath 2.0 functions is listed in the section XPath 2.0 and XQuery 1.0 Functions.

1.2.2 XSLT 2.0 Elements and Functions

Limitations
The **xsl:preserve-space** and **xsl:strip-space** elements are not supported.

Implementation-specific behavior
Given below is a description of how the Altova XSLT 2.0 Engine handles implementation-specific aspects of the behavior of certain XSLT 2.0 functions.

xsl:result-document
Additionally supported encodings are: `x-base16tobinary` and `x-base64tobinary`.

function-available
The function tests for the availability of in-scope functions (XSLT 2.0, XPath 2.0, and extension functions).

unparsed-text
The `href` attribute accepts (i) relative paths for files in the base-uri folder, and (ii) absolute paths with or without the `file://` protocol. Additionally supported encodings are: `x-binarytobase16` and `x-binarytobase64`.

unparsed-text-available
The `href` attribute accepts (i) relative paths for files in the base-uri folder, and (ii) absolute paths with or without the `file://` protocol. Additionally supported encodings are: `x-binarytobase16` and `x-binarytobase64`.

Note: The following encoding values, which were implemented in earlier versions of AltovaXML are now deprecated: `base16tobinary`, `base64tobinary`, `binarytobase16` and `binarytobase64`.

 © 2011 Altova GmbH

1.3 XQuery 1.0 Engine: Implementation Information

The Altova XQuery 1.0 Engine is built into Altova's XMLSpy and MapForce XML products. It is also available in the free AltovaXML package. This section provides information about implementation-defined aspects of behavior.

Standards conformance

The Altova XQuery 1.0 Engine conforms to the World Wide Web Consortium's (W3C's) XQuery 1.0 Recommendation of 23 January 2007. The XQuery standard gives implementations discretion about how to implement many features. Given below is a list explaining how the Altova XQuery 1.0 Engine implements these features.

Schema awareness

The Altova XQuery 1.0 Engine is **schema-aware**.

Encoding

The UTF-8 and UTF-16 character encodings are supported.

Namespaces

The following namespace URIs and their associated bindings are pre-defined.

Namespace Name	Prefix	Namespace URI
XML Schema types	`xs:`	`http://www.w3.org/2001/XMLSchema`
Schema instance	`xsi:`	`http://www.w3.org/2001/XMLSchema-instance`
Built-in functions	`fn:`	`http://www.w3.org/2005/xpath-functions`
Local functions	`local:`	`http://www.w3.org/2005/xquery-local-functions`

The following points should be noted:

- The Altova XQuery 1.0 Engine recognizes the prefixes listed above as being bound to the corresponding namespaces.
- Since the built-in functions namespace listed above is the default functions namespace in XQuery, the `fn:` prefix does not need to be used when built-in functions are invoked (for example, `string("Hello")` will call the `fn:string` function). However, the prefix `fn:` can be used to call a built-in function without having to declare the namespace in the query prolog (for example: `fn:string("Hello")`).
- You can change the default functions namespace by declaring the `default function namespace` expression in the query prolog.
- When using types from the XML Schema namespace, the prefix `xs:` may be used without having to explicitly declare the namespaces and bind these prefixes to them in the query prolog. (Example: `xs:date` and `xs:yearMonthDuration`.) If you wish to use some other prefix for the XML Schema namespace, this must be explicitly declared in the query prolog. (Example: `declare namespace alt = "http://www.w3.org/2001/XMLSchema"; alt:date("2004-10-04")`.)
- Note that the `untypedAtomic`, `dayTimeDuration`, and `yearMonthDuration` datatypes have been moved, with the CRs of 23 January 2007, from the XPath Datatypes namespace to the XML Schema namespace, so: `xs:yearMonthDuration`.

If namespaces for functions, type constructors, node tests, etc are wrongly assigned, an error is

reported. Note, however, that some functions have the same name as schema datatypes, e.g. `fn:string` and `fn:boolean`. (Both `xs:string` and `xs:boolean` are defined.) The namespace prefix determines whether the function or type constructor is used.

XML source document and validation
XML documents used in executing an XQuery document with the Altova XQuery 1.0 Engine must be well-formed. However, they do not need to be valid according to an XML Schema. If the file is not valid, the invalid file is loaded without schema information. If the XML file is associated with an external schema and is valid according to it, then post-schema validation information is generated for the XML data and will be used for query evaluation.

Static and dynamic type checking
The static analysis phase checks aspects of the query such as syntax, whether external references (e.g. for modules) exist, whether invoked functions and variables are defined, and so on. No type checking is done in the static analysis phase. If an error is detected in the static analysis phase, it is reported and the execution is stopped.

Dynamic type checking is carried out at run-time, when the query is actually executed. If a type is incompatible with the requirement of an operation, an error is reported. For example, the expression `xs:string("1") + 1` returns an error because the addition operation cannot be carried out on an operand of type `xs:string`.

Library Modules
Library modules store functions and variables so they can be reused. The Altova XQuery 1.0 Engine supports modules that are stored in **a single external XQuery file**. Such a module file must contain a `module` declaration in its prolog, which associates a target namespace. Here is an example module:

```
module namespace libns="urn:module-library";
declare variable $libns:company := "Altova";
declare function libns:webaddress() { "http://www.altova.com" };
```

All functions and variables declared in the module belong to the namespace associated with the module. The module is used by importing it into an XQuery file with the `import module` statement in the query prolog. The `import module` statement only imports functions and variables declared directly in the library module file. As follows:

```
import module namespace modlib = "urn:module-library" at
   "modulefilename.xq";
if    ($modlib:company = "Altova")
then   modlib:webaddress()
else   error("No match found.")
```

External functions
External functions are not supported, i.e. in those expressions using the `external` keyword, as in:

```
declare function hoo($param as xs:integer) as xs:string external;
```

Collations
The default collation is the Unicode codepoint collation. No other collation is currently supported. Comparisons, including the `fn:max` function, are based on this collation.

 © 2011 Altova GmbH

Character normalization
No character normalization form is supported.

Precision of numeric types

- The `xs:integer` datatype is arbitrary-precision, i.e. it can represent any number of digits.
- The `xs:decimal` datatype has a limit of 20 digits after the decimal point.
- The `xs:float` and `xs:double` datatypes have limited-precision of 15 digits.

XQuery Instructions Support
The `Pragma` instruction is not supported. If encountered, it is ignored and the fallback expression is evaluated.

XQuery Functions Support
For information about implementation-specific behavior of XQuery 1.0 functions, see the section, XPath 2.0 and XQuery 1.0 Functions.

1.4 XPath 2.0 and XQuery 1.0 Functions

XPath 2.0 and XQuery 1.0 functions are evaluated by:

- the **Altova XPath 2.0 Engine**, which (i) is a component of the Altova XSLT 2.0 Engine, and (ii) is used in the XPath Evaluator of Altova's XMLSpy product to evaluate XPath expressions with respect to the XML document that is active in the XMLSpy interface.
- the **Altova XQuery 1.0 Engine**.

This section describes how XPath 2.0 and XQuery 1.0 functions are handled by the Altova XPath 2.0 Engine and Altova XQuery 1.0 Engine. Only those functions are listed, for which the behavior is implementation-specific, or where the behavior of an individual function is different in any of the three environments in which these functions are used (that is, in XSLT 2.0, in XQuery 1.0, and in the XPath Evaluator of XMLSpy). Note that this section does not describe how to use these functions. For more information about the usage of functions, see the World Wide Web Consortium's (W3C's) XQuery 1.0 and XPath 2.0 Functions and Operators Recommendation of 23 January 2007.

1.4.1 General Information

Standards conformance

- The Altova XPath 2.0 Engine implements the World Wide Web Consortium's (W3C's) XPath 2.0 Recommendation of 23 January 2007. The Altova XQuery 1.0 Engine implements the World Wide Web Consortium's (W3C's) XQuery 1.0 Recommendation of 23 January 2007. The XPath 2.0 and XQuery 1.0 functions support in these two engines is compliant with the XQuery 1.0 and XPath 2.0 Functions and Operators Recommendation of 23 January 2007.
- The Altova XPath 2.0 Engine conforms to the rules of XML 1.0 (Fourth Edition) and XML Namespaces (1.0).

Default functions namespace

The default functions namespace has been set to comply with that specified in the standard. Functions can therefore be called without a prefix.

Boundary-whitespace-only nodes in source XML document

The XML data (and, consequently, the XML Infoset) that is passed to the Altova XPath 2.0 Engine and Altova XQuery 1.0 Engine is stripped of boundary-whitespace-only text nodes. (A boundary-whitespace-only text node is a child whitespace-only text node that occurs between two elements within an element of mixed content.) This stripping has an effect on the value returned by the `fn:position()`, `fn:last()`, `fn:count()`, and `fn:deep-equal()` functions.

For any node selection that selects text nodes also, boundary-whitespace-only text nodes would typically also be included in the selection. However, since the XML Infoset used by the Altova engines has boundary-whitespace-only text nodes stripped from it, these nodes are not present in the XML Infoset. As a result, the size of the selection and the numbering of nodes in the selection will be different than that for a selection which included these text nodes. The `fn:position()`, `fn:last()`, `fn:count()`, and `fn:deep-equal()` functions, therefore, could produce results that are different from those produced by some other processors.

A situation in which boundary-whitespace-only text nodes are evaluated as siblings of other elements arises most commonly when `xsl:apply-templates` is used to apply templates. When the `fn:position()`, `fn:last()`, and `fn:count()` functions are used in patterns with

 © 2011 Altova GmbH

a name test (for example, `para[3]`, which is short for `para[position()=3]`), boundary-whitespace-only nodes are irrelevant since only the named elements (`para` in the above example) are selected. (Note, however, that boundary-whitespace-only nodes **are** relevant in patterns that use the wildcard, for example, `*[10]`.)

Numeric notation
On output, when an `xs:double` is converted to a string, scientific notation (for example, `1.0E12`) is used when the absolute value is less than 0.000001 or greater than 1,000,000. Otherwise decimal or integer notation is used.

Precision of `xs:decimal`
The precision refers to the number of digits in the number, and a minimum of 18 digits is required by the specification. For division operations that produce a result of type `xs:decimal`, the precision is 19 digits after the decimal point with no rounding.

Implicit timezone
When two `date`, `time`, or `dateTime` values need to be compared, the timezone of the values being compared need to be known. When the timezone is not explicitly given in such a value, the implicit timezone is used. The implicit timezone is taken from the system clock, and its value can be checked with the `fn:implicit-timezone()` function.

Collations
Only the Unicode codepoint collation is supported. No other collations can be used. String comparisons, including for the `fn:max` and `fn:min` functions, are based on this collation.

Namespace axis
The namespace axis is deprecated in XPath 2.0. Use of the namespace axis is, however, supported. To access namespace information with XPath 2.0 mechanisms, use the `fn:in-scope-prefixes()`, `fn:namespace-uri()` and `fn:namespace-uri-for-prefix()` functions.

Static typing extensions
The optional static type checking feature is not supported.

1.4.2 Functions Support

The table below lists (in alphabetical order) the implementation-specific behavior of certain functions. The following general points should be noted:

- In general, when a function expects a sequence of one item as an argument, and a sequence of more than one item is submitted, then an error is returned.
- All string comparisons are done using the Unicode codepoint collation.
- Results that are QNames are serialized in the form `[prefix:]localname`.

Function Name	Notes

base-uri	• If external entities are used in the source XML document and if a node in the external entity is specified as the input node argument of the base-uri() function, it is still the base URI of the including XML document that is used—not the base URI of the external entity. • The base URI of a node in the XML document can be modified using the xml:base attribute.
collection	• The argument is a relative URI that is resolved against the current base URI. • If the resolved URI identifies an XML file, then this XML file is treated as a catalog which references a collection of files. This file must have the form: <collection> <doc href="uri-1" /> <doc href="uri-2" /> <doc href="uri-3" /> </collection> The files referenced by the href attributes are loaded, and their document nodes are returned as a sequence. • If the resolved URI does not identify an XML file with the catalog structure described above, then the argument string (in which wildcards such as ? and * are allowed) is used as a search string. XML files with names that match the search expression are loaded, and their document nodes are returned as a sequence. See examples below. • XSLT example: The expression collection("c:\MyDocs*.xml")//Title returns a sequence of all DocTitle elements in the .xml files in the MyDocs folder. • XQuery example: The expression {for $i in collection(c:\MyDocs*.xml) return element doc{base-uri($i)}} returns the base URIs of all the .xml files in the MyDocs folder, each URI being within a doc element. • The default collection is empty.

Function Name	Notes
count	• See note on whitespace in the General Information section.
current-date, current-dateTime, current-time	• The current date and time is taken from the system clock. • The timezone is taken from the implicit timezone provided by the evaluation context; the implicit timezone is taken from the system clock. • The timezone is always specified in the result.
deep-equal	• See note on whitespace in the General Information section.

© 2011 Altova GmbH

doc	• An error is raised only if no XML file is available at the specified location or if the file is not well-formed. The file is validated if a schema is available. If the file is not valid, the invalid file is loaded without schema information.
id	• In a well-formed but invalid document that contains two or more elements having the same ID value, the first element in document order is returned.
in-scope-prefixes	• Only default namespaces may be undeclared in the XML document. However, even when a default namespace is undeclared on an element node, the prefix for the default namespace, which is the zero-length string, is returned for that node.
last	• See note on whitespace in the General Information section.
lower-case	• The Unicode character set is supported.
normalize-unicode	• The normalization forms NFC, NFD, NFKC, and NFKD are supported.

Function Name	Notes
position	• See note on whitespace in the General Information section.
resolve-uri	• If the second, optional argument is omitted, the URI to be resolved (the first argument) is resolved against the base URI from the static context, which is the URI of the XSLT stylesheet or the base URI given in the prolog of the XQuery document. • The relative URI (the first argument) is appended after the last "/" in the path notation of the base URI notation. • If the value of the first argument is the zero-length string, the base URI from the static context is returned, and this URI includes the file name of the document from which the base URI of the static context is derived (e.g. the XSLT or XML file).
static-base-uri	• The base URI from the static context is the base URI of the XSLT stylesheet or the base URI specified in the prolog of the XQuery document. • When using XPath Evaluator in the XMLSpy IDE, the base URI from the static context is the URI of the active XML document.
upper-case	• The Unicode character set is supported.

1.5 Extensions

Altova Engines (XSLT 1.0, XSLT 2.0, and XQuery 1.0), which are used in a number of Altova products, support the use of a set of AltovaXML-specific custom extension functions.

1.5.1 Altova Extension Functions

Altova extension functions are in the namespace **http://www.altova.com/xslt-extensions** and are indicated in this section with the prefix **altova:** , which is assumed to be bound to the namespace given above.

The following extension functions are supported in the current version of your Altova product in the manner described below. However, note that in future versions of your product, support for one or more of these functions might be discontinued or the behavior of individual functions might change. Consult the documentation of future releases for information about support for Altova extension functions in that release.

General functions

- altova:evaluate()
- altova:distinct-nodes()
- altova:encode-for-rtf()
- altova:xbrl-labels()
- altova:xbrl-footnotes()
- altova:generate-auto-number()
- altova:reset-auto-number()
- altova:get-temp-folder()

General Functions

The following extension functions are supported in the current version of your Altova product in the manner described below. However, note that in future versions of your product, support for one or more of these functions might be discontinued or the behavior of individual functions might change. Consult the documentation of future releases for information about support for Altova extension functions in that release.

- altova:evaluate()
- altova:distinct-nodes()
- altova:encode-for-rtf()
- altova:xbrl-labels()
- altova:xbrl-footnotes()
- altova:generate-auto-number()
- altova:reset-auto-number()
- altova:get-temp-folder()

altova:evaluate()

The `altova:evaluate()` function takes an XPath expression, passed as a string, as its mandatory argument. It returns the output of the evaluated expression.

```
altova:evaluate( XPathExp as xs:string)
```

For example:

```
altova:evaluate('//Name[1]')
```

In the example above, note that the expression `//Name[1]` is passed as a string by enclosing it in single quotes. The `altova:evaluate` function returns the contents of the first `Name` element in the document.

The `altova:evaluate` function can take additional (optional) arguments. These arguments are, respectively, the values of variables with the names `p1`, `p2`, `p3`... `pN` that can be used in the XPath expression.

```
altova:evaluate( XPathExp as xs:string [, p1value ... pNvalue])
```

where

- the variable names must be of the form `pX`, `X` being an integer
- the sequence of the function's arguments, from the second argument onwards corresponds to the sequence of variables named `p1` to `pN`. So the second argument will be the value of the variable `p1`, the third argument that of the variable `p2`, and so on.
- The variable values must be of type `item*`

For example:

```
<xsl:variable name="xpath" select="'$p3, $p2, $p1'" />
<xsl:value-of select="altova:evaluate( $xpath, 10, 20, 'hi' )" />
```

Outputs "`hi 20 10`*"*

In the above listing, notice the following:

- The second argument of the `altova:evaluate` expression is the value assigned to the variable `$p1`, the third argument that assigned to the variable `$p2`, and so on.
- Notice that the fourth argument of the function is a string value, indicated by its being enclosed in quotes.
- The `select` attribute of the `xs:variable` element supplies the XPath expression. Since this expression must be of type `xs:string`, it is enclosed in single quotes.

The following examples further illustrate usage:

```
<xsl:variable name="xpath" select="'$p1'" />
<xsl:value-of select="altova:evaluate( $xpath, //Name[1] )" />
```

Outputs value of the first `Name` *element.*

```
<xsl:variable name="xpath" select="'$p1'" />
<xsl:value-of select="altova:evaluate( $xpath, '//Name[1]' )" />
```

Outputs "`//Name[1]`*"*

The `altova:evaluate()` extension function is useful in situations where an XPath expression in the XSLT stylesheet contains one or more parts that must be evaluated dynamically. For example, consider a situation in which a user enters his request for the sorting criterion and this criterion is stored in the attribute `UserReq/@sortkey`. In the stylesheet, you could then have the expression :

```
<xsl:sort select="altova:evaluate(../UserReq/@sortkey)" order="ascending"/
>
```

The `altova:evaluate()` function reads the `sortkey` attribute of the `UserReq` child element of the parent of the context node. Say the value of the `sortkey` attribute is `Price`, then `Price` is returned by the `altova:evaluate()` function and becomes the value of the `select` attribute:

```
<xsl:sort select="Price" order="ascending"/>
```

If this `sort` instruction occurs within the context of an element called `Order`, then the `Order` elements will be sorted according to the values of their `Price` children. Alternatively, if the value of `@sortkey` were, say, `Date`, then the `Order` elements would be sorted according to the values of their `Date` children. So the sort criterion for `Order` is selected from the `sortkey` attribute at runtime. This could not have been achieved with an expression like:

```
<xsl:sort select="../UserReq/@sortkey" order="ascending"/>
```

In the case shown above, the sort criterion would be the `sortkey` attribute itself, not `Price` or `Date` (or any other current content of `sortkey`).

Variables can be used in the `altova:evaluate()` extension function as shown in the examples below:

- Static variables: `<xsl:value-of select="$i3, $i2, $i1" />`
  Outputs the values of three variables.

- Dynamic XPath expression with dynamic variables:
  `<xsl:variable name="xpath" select="'$p3, $p2, $p1'" />`
  `<xsl:value-of select="altova:evaluate( $xpath, 10, 20, 30 )" />`
  Outputs "30 20 10"

- Dynamic XPath expression with no dynamic variable:
  `<xsl:variable name="xpath" select="'$p3, $p2, $p1'" />`
  `<xsl:value-of select="altova:evaluate( $xpath )" />`
  Outputs error: No variable defined for $p3.

Note: The static context includes namespaces, types, and functions—but not variables—from the calling environment. The base URI and default namespace are inherited.

altova:distinct-nodes()

The `altova:distinct-nodes()` function takes a set of one or more nodes as its input and returns the same set minus nodes with duplicate values. The comparison is done using the XPath/XQuery function `fn:deep-equal`.

```
altova:distinct-nodes( $arg as node()* ) as node()*
```

altova:encode-for-rtf()

The `altova:encode-for-rtf()` function converts the input string into code for RTF.

```
altova:encode-for-rtf( $inputstr as xs:string?,
$preserveallwhitespace as xs:boolean,
$preservenewlines as xs:boolean) as xs:string
```

Whitespace and new lines will be preserved according to the boolean value specified for their respective parameters.

altova:xbrl-labels()

The `altova:xbrl-labels()` function takes two input arguments: a node name and the taxonomy file location containing the node. The function returns the XBRL labels associated

 © 2011 Altova GmbH

with the input node.

```
altova:xbrl-labels( $name as xs:QName, $file as xs:string )  as node()*
```

altova:xbrl-footnotes()
The `altova:footnotes()` function takes a node as its input argument and returns the set of XBRL footnote nodes referenced by the input node.

```
altova:footnotes( $arg as node() )  as node()*
```

altova:generate-auto-number(id as xs:string, start-with as xs:integer, increment as xs:integer, reset-on-change as xs:string)
Generates a series of numbers having the specified ID. The start integer and the increment is specified.

altova:reset-auto-number(id as xs:string)
This function resets the auto-numbering of the auto-numbering series specified with the ID argument. The series is reset to the start integer of the series (see `altova:generate-auto-number` above).

altova:get-temp-folder as xs:string
Gets the temporary folder.

2 Technical Data

This section contains useful background information on the technical aspects of your software. It is organized into the following sections:

- OS and Memory Requirements
- Altova XML Parser
- Altova XSLT and XQuery Engines
- Unicode Support
- Internet Usage

© 2011 Altova GmbH

2.1 OS and Memory Requirements

Operating System
Altova software applications are:

- 32-bit Windows applications for Windows XP, Windows Server 2003 and 2008, Windows Vista, and Windows 7, or
- 64-bit Windows applications for Windows Vista and Windows 7

Memory
Since the software is written in C++ it does not require the overhead of a Java Runtime Environment and typically requires less memory than comparable Java-based applications. However, each document is loaded fully into memory so as to parse it completely and to improve viewing and editing speed. The memory requirement increases with the size of the document.

Memory requirements are also influenced by the unlimited Undo history. When repeatedly cutting and pasting large selections in large documents, available memory can rapidly be depleted.

2.2 Altova XML Parser

When opening any XML document, the application uses its built-in validating parser (the Altova XML Parser) to check for well-formedness, validate the document against a schema (if specified), and build trees and Infosets. The Altova XML Parser is also used to provide intelligent editing help while you edit documents and to dynamically display any validation error that may occur.

The built-in Altova XML Parser implements the Final Recommendation of the W3C's XML Schema specification. New developments recommended by the W3C's XML Schema Working Group are continuously being incorporated in the Altova Parser, so that Altova products give you a state-of-the-art development environment.

© 2011 Altova GmbH

2.3 Altova XSLT and XQuery Engines

Altova products use the Altova XSLT 1.0 Engine, Altova XSLT 2.0 Engine, and Altova XQuery 1.0 Engines. Documentation about implementation-specific behavior for each engine is in the section Engine Information, in Appendix 1 of the product documentation, should that engine be used in the product.

These three engines are also available in the AltovaXML package, which can be downloaded from the Altova website free of charge. Documentation for using the engines is available with the AltovaXML package.

2.4 Unicode Support

Unicode is the 16-bit character-set (extendable to 32-bit) defined by the Unicode Consortium. It provides a unique number for every character,

- no matter what the platform,
- no matter what the program,
- no matter what the language.

Fundamentally, computers just deal with numbers. They store letters and other characters by assigning a number for each one. Before Unicode was invented, there were hundreds of different encoding systems for assigning these numbers. No single encoding could contain enough characters: for example, the European Union alone requires several different encodings to cover all its languages. Even for a single language like English, no single encoding was adequate for all the letters, punctuation, and technical symbols in common use.

These encoding systems used to conflict with one another. That is, two encodings used the same number for two different characters, or different numbers for the same character. Any given computer (especially servers) needs to support many different encodings; yet whenever data is passed between different encodings or platforms, that data always runs the risk of corruption.

Unicode is changing all that!
Unicode provides a unique number for every character, no matter what the platform, no matter what the program, and no matter what the language. The Unicode Standard has been adopted by such industry leaders as Apple, HP, IBM, JustSystems, Microsoft, Oracle, SAP, Sun, Base and many others.

Unicode is required by modern standards such as XML, Java, ECMAScript (JavaScript), LDAP, CORBA 3.0, WML, etc., and is the official way to implement ISO/IEC 10646. It is supported in many operating systems, all modern browsers, and many other products. The emergence of the Unicode Standard, and the availability of tools supporting it, are among the most significant recent global software technology trends.

Incorporating Unicode into client-server or multi-tiered applications and web sites offers significant cost savings over the use of legacy character sets. Unicode enables a single software product or a single web site to be targeted across multiple platforms, languages and countries without re-engineering. It allows data to be transported through many different systems without corruption.

2.4.1 Windows XP

Altova's XML products provide full Unicode support. To edit an XML document, you will also need a font that supports the Unicode characters being used by that document.

Please note that most fonts only contain a very specific subset of the entire Unicode range and are therefore typically targeted at the corresponding writing system. Consequently you may encounter XML documents that contain "unprintable" characters, because the font you have selected does not contain the required glyphs. Therefore it can sometimes be very useful to have a font that covers the entire Unicode range - especially when editing XML documents from all over the world.

The most universal font we have encountered is a typeface called Arial Unicode MS that has been created by Agfa Monotype for Microsoft. This font contains over 50,000 glyphs and covers the entire set of characters specified by the Unicode 2.1 standard. It needs 23MB and is included with Microsoft Office 2000.

 © 2011 Altova GmbH

We highly recommend that you install this font on your system and use it with the application if you are often editing documents in different writing systems. This font is not installed with the "Typical" setting of the Microsoft Office setup program, but you can choose the Custom Setup option to install this font.

In the `/Examples` folder in your application folder you will also find a new XHTML file called `Unicode-UTF8.html` that contains the sentence "When the world wants to talk, it speaks Unicode" in many different languages ("Wenn die Welt miteinander spricht, spricht sie Unicode") and writing-systems (世界的に話すなら、Unicode です。) - this line has been adopted from the 10th Unicode conference in 1997 and is a beautiful illustration of the importance of Unicode for the XML standard. Opening this file will give you a quick impression on what is possible with Unicode and what writing systems are supported by the fonts available on your PC installation.

2.4.2 Right-to-Left Writing Systems

Please note that even under Windows NT 4.0 any text from a right-to-left writing-system (such as Hebrew or Arabic) is not rendered correctly except in those countries that actually use right-to-left writing-systems. This is due to the fact that only the Hebrew and Arabic versions of Windows NT contains support for rendering and editing right-to-left text on the operating system layer.

2.5 Internet Usage

Altova applications will initiate Internet connections on your behalf in the following situations:

- If you click the "Request evaluation key-code" in the Registration dialog (**Help | Software Activation**), the three fields in the registration dialog box are transferred to our web server by means of a regular http (port 80) connection and the free evaluation key-code is sent back to the customer via regular SMTP e-mail.
- If you use the URL mode of the Open dialog box to open a document directly from a URL (**File | Open | Switch to URL**), that document is retrieved through a http (port 80) connection. (*This functionality is available in XMLSpy and Authentic Desktop.*)
- If you open an XML document that refers to an XML Schema or DTD and the document is specified through a URL, it is also retrieved through a http (port 80) connection once you validate the XML document. This may also happen automatically upon opening a document if you have instructed the application to automatically validate files upon opening in the File tab of the Options dialog (**Tools | Options**). (*This functionality is available in XMLSpy and Authentic Desktop.*)
- If you are using the Send by Mail... command (**File | Send by Mail**) in XMLSpy, the current selection or file is sent by means of any MAPI-compliant mail program installed on the user's PC.
- As part of Software Activation and LiveUpdate as further described in this manual and the Altova Software License Agreement.

 © 2011 Altova GmbH

3 License Information

This section contains:

- Information about the distribution of this software product
- Information about the intellectual property rights related to this software product
- The End User License Agreement governing the use of this software product

Please read this information carefully. It is binding upon you since you agreed to these terms when you installed this software product.

3.1 Electronic Software Distribution

This product is available through electronic software distribution, a distribution method that provides the following unique benefits:

- You can evaluate the software free-of-charge before making a purchasing decision.
- Once you decide to buy the software, you can place your order online at the Altova website and immediately get a fully licensed product within minutes.
- When you place an online order, you always get the latest version of our software.
- The product package includes a comprehensive integrated onscreen help system. The latest version of the user manual is available at www.altova.com (i) in HTML format for online browsing, and (ii) in PDF format for download (and to print if you prefer to have the documentation on paper).

30-day evaluation period
After downloading this product, you can evaluate it for a period of up to 30 days free of charge. About 20 days into this evaluation period, the software will start to remind you that it has not yet been licensed. The reminder message will be displayed once each time you start the application. If you would like to continue using the program after the 30-day evaluation period, you have to purchase an Altova Software License Agreement, which is delivered in the form of a key-code that you enter into the Software Activation dialog to unlock the product. You can purchase your license at the online shop at the Altova website.

Helping Others within Your Organization to Evaluate the Software
If you wish to distribute the evaluation version within your company network, or if you plan to use it on a PC that is not connected to the Internet, you may only distribute the Setup programs, provided that they are not modified in any way. Any person that accesses the software installer that you have provided, must request their own 30-day evaluation license key code and after expiration of their evaluation period, must also purchase a license in order to be able to continue using the product.

For further details, please refer to the Altova Software License Agreement at the end of this section.

© 2011 Altova GmbH

3.2 Software Activation and License Metering

As part of Altova's Software Activation, the software may use your internal network and Internet connection for the purpose of transmitting license-related data at the time of installation, registration, use, or update to an Altova-operated license server and validating the authenticity of the license-related data in order to protect Altova against unlicensed or illegal use of the software and to improve customer service. Activation is based on the exchange of license related data such as operating system, IP address, date/time, software version, and computer name, along with other information between your computer and an Altova license server.

Your Altova product has a built-in license metering module that further helps you avoid any unintentional violation of the End User License Agreement. Your product is licensed either as a single-user or multi-user installation, and the license-metering module makes sure that no more than the licensed number of users use the application concurrently.

This license-metering technology uses your local area network (LAN) to communicate between instances of the application running on different computers.

Single license
When the application starts up, as part of the license metering process, the software sends a short broadcast datagram to find any other instance of the product running on another computer in the same network segment. If it doesn't get any response, it will open a port for listening to other instances of the application.

Multi license
If more than one instance of the application is used within the same LAN, these instances will briefly communicate with each other on startup. These instances exchange key-codes in order to help you to better determine that the number of concurrent licenses purchased is not accidentally violated. This is the same kind of license metering technology that is common in the Unix world and with a number of database development tools. It allows Altova customers to purchase reasonably-priced concurrent-use multi-user licenses.

We have also designed the applications so that they send few and small network packets so as to not put a burden on your network. The TCP/IP ports (2799) used by your Altova product are officially registered with the IANA (*see* http://www.isi.edu/in-notes/iana/assignments/port-numbers *for details*) and our license-metering module is tested and proven technology.

If you are using a firewall, you may notice communications on port 2799 between the computers that are running Altova products. You are, of course, free to block such traffic between different groups in your organization, as long as you can ensure by other means, that your license agreement is not violated.

You will also notice that, if you are online, your Altova product contains many useful functions; these are unrelated to the license-metering technology.

3.3 Intellectual Property Rights

The Altova Software and any copies that you are authorized by Altova to make are the intellectual property of and are owned by Altova and its suppliers. The structure, organization and code of the Software are the valuable trade secrets and confidential information of Altova and its suppliers. The Software is protected by copyright, including without limitation by United States Copyright Law, international treaty provisions and applicable laws in the country in which it is being used. Altova retains the ownership of all patents, copyrights, trade secrets, trademarks and other intellectual property rights pertaining to the Software, and that Altova's ownership rights extend to any images, photographs, animations, videos, audio, music, text and "applets" incorporated into the Software and all accompanying printed materials. Notifications of claimed copyright infringement should be sent to Altova's copyright agent as further provided on the Altova Web Site.

Altova software contains certain Third Party Software that is also protected by intellectual property laws, including without limitation applicable copyright laws as described in detail at http://www.altova.com/legal_3rdparty.html.

All other names or trademarks are the property of their respective owners.

© 2011 Altova GmbH

3.4 Altova End User License Agreement

THIS IS A LEGAL DOCUMENT -- RETAIN FOR YOUR RECORDS

ALTOVA® END USER LICENSE AGREEMENT

Licensor:
Altova GmbH
Rudolfsplatz 13a/9
A-1010 Wien
Austria

Important - Read Carefully. Notice to User:

This End User License Agreement ("Software License Agreement") is a legal document between you and Altova GmbH ("Altova"). It is important that you read this document before using the Altova-provided software ("Software") and any accompanying documentation, including, without limitation printed materials, 'online' files, or electronic documentation ("Documentation"). By clicking the "I accept" and "Next" buttons below, or by installing, or otherwise using the Software, you agree to be bound by the terms of this Software License Agreement as well as the Altova Privacy Policy ("Privacy Policy") including, without limitation, the warranty disclaimers, limitation of liability, data use and termination provisions below, whether or not you decide to purchase the Software. You agree that this agreement is enforceable like any written agreement negotiated and signed by you. If you do not agree, you are not licensed to use the Software, and you must destroy any downloaded copies of the Software in your possession or control. You may print a copy of this Software License Agreement as part of the installation process at the time of acceptance. Alternatively, you may go to our Web site at http://www.altova.com/eula to download and print a copy of this Software License Agreement for your files and http://www.altova.com/privacy to review the Privacy Policy.

1. SOFTWARE LICENSE

(a) License Grant.

(i) Upon your acceptance of this Software License Agreement Altova grants you a non-exclusive, non-transferable (except as provided below), limited license, without the right to grant sublicenses, to install and use a copy of the Software on one compatible personal computer or workstation up to the Permitted Number of computers. Subject to the limitations set forth in Section 1(c), you may install and use a copy of the Software on more than one of your compatible personal computers or workstations if you have purchased a Named User license. The Permitted Number of computers and/or users shall be determined and specified at such time as you elect to purchase the Software. During the evaluation period, hereinafter defined, only a single user may install and use the software on one (1) personal computer or workstation. If you have licensed the Software as part of a suite of Altova software products (collectively, the "Suite") and have not installed each product individually, then the Software License Agreement governs your use of all of the software included in the Suite.

(ii) If you have licensed SchemaAgent, then the terms and conditions of this Software License Agreement apply to your use of the SchemaAgent server software ("SchemaAgent Server") included therein, as applicable, and you are licensed to use SchemaAgent Server solely in connection with your use of Altova Software and solely for the purposes described in the accompanying documentation.

(iii) If you have licensed Software that enables users to generate source code, your license to install and use a copy of the Software as provided herein permits you to generate source code based on (i) Altova Library modules that are included in the Software (such

generated code hereinafter referred to as the "Restricted Source Code") and (ii) schemas or mappings that you create or provide (such code as may be generated from your schema or mapping source materials hereinafter referred to as the "Unrestricted Source Code"). In addition to the rights granted herein, Altova grants you a non-exclusive, non-transferable, limited license to compile the complete generated code (comprised of the combination of the Restricted Source Code and the Unrestricted Source Code) into executable object code form, and to use, copy, distribute or license that executable. You may not distribute or redistribute, sublicense, sell, or transfer the Restricted Source Code to a third-party unless said third-party already has a license to the Restricted Source Code through their separate agreement with Altova. Notwithstanding anything to the contrary herein, you may not distribute, incorporate or combine with other software, or otherwise use the Altova Library modules or Restricted Source Code, or any Altova intellectual property embodied in or associated with the Altova Library modules or Restricted Source Code, in any manner that would subject the Restricted Source Code to the terms of a copyleft, free software or open source license that would require the Restricted Source Code or Altova Library modules source code to be disclosed in source code form. Altova reserves all other rights in and to the Software. With respect to the feature(s) of UModel that permit reverse-engineering of your own source code or other source code that you have lawfully obtained, such use by you does not constitute a violation of this Agreement. Except as otherwise expressly permitted in Section 1(i) reverse engineering of the Software is strictly prohibited as further detailed therein.

(iv) In the event Restricted Source Code is incorporated into executable object code form, you will include the following statement in (1) introductory splash screens, or if none, within one or more screens readily accessible by the end-user, and (2) in the electronic and/or hard copy documentation: "Portions of this program were developed using Altova® [name of Altova Software, e.g. MapForce® 2011] and include libraries owned by Altova GmbH, Copyright © 2007-2011 Altova GmbH (www.altova.com)."

(b) Server Use. You may install one (1) copy of the Software on a computer file server within your internal network solely for the purpose of downloading and installing the Software onto other computers within your internal network up to the Permitted Number of computers in a commercial environment only. If you have licensed SchemaAgent, then you may install SchemaAgent Server on any server computer or workstation and use it in connection with your Software. No other network use is permitted, including without limitation using the Software either directly or through commands, data or instructions from or to a computer not part of your internal network, for Internet or Web-hosting services or by any user not licensed to use this copy of the Software through a valid license from Altova.

If you have purchased Concurrent User Licenses as defined in Section 1(d), and subject to limits set forth therein, you may install a copy of the Software on a terminal server (Microsoft Terminal Server, Citrix Metaframe, etc.), application virtualization server (Microsoft App-V, Citrix XenApp, VMWare ThinApp, etc.) or virtual machine environment within your internal network for the sole and exclusive purpose of permitting individual users within your organization to access and use the Software through a terminal server, application virtualization session, or virtual machine environment from another computer provided that the total number of users that access or use the Software concurrently at any given point in time on such network, virtual machine or terminal server does not exceed the Permitted Number; and provided that the total number of users authorized to use the Software through the terminal server, application virtualization session, or virtual machine environment does not exceed six (6) times the Permitted Number of users. Accordingly, the limitations set forth in Section 1(d) regarding the number of installations and the requirement that the usage be on the same physical network shall not apply to terminal server, application virtualization session, or virtual machine environments. Altova makes no warranties or representations about the performance of Altova software in a terminal server, application virtualization session, or virtual machine environment and the foregoing are expressly excluded from the limited warranty in Section 5 hereof and technical support is not available with respect to issues arising from use in such environments.

(c) Named Use. If you have licensed the "Named User" version of the software, you may install the Software on up to five (5) compatible personal computers or workstations of which you are the primary user thereby allowing you to switch from one computer to the other as necessary provided that only one (1) instance of the Software will be used by you as the Named User at any given time. If you have purchased multiple Named User licenses, each individual Named User will receive a separate license key code.

(d) Concurrent Use. If you have licensed a "Concurrent-User" version of the Software, you may install the Software on any compatible computers in a commercial environment only, up to ten (10) times the Permitted Number of users, provided that only the Permitted Number of users actually use the Software at the same time and further provided that the computers on which the Software is installed are on the same physical computer network. The Permitted Number of concurrent users shall be delineated at such time as you elect to purchase the Software licenses. Each separate physical network or office location requires its own set of separate Concurrent User Licenses for those wishing to use the Concurrent User versions of the Software in more than one location or on more than one network, all subject to the above Permitted Number limitations and based on the number of users using the Software. If a computer is not on the same physical network, then a locally installed user license is required. Home User restrictions and limitations with respect to the Concurrent User licenses used on home computers are set forth in Section 1(f).

(e) Backup and Archival Copies. You may make one (1) backup and one (1) archival copy of the Software, provided your backup and archival copies are not installed or used on any computer and further provided that all such copies shall bear the original and unmodified copyright, patent and other intellectual property markings that appear on or in the Software. You may not transfer the rights to a backup or archival copy unless you transfer all rights in the Software as provided under Section 3.

(f) Home Use (Personal and Non-Commercial). In order to further familiarize yourself with the Software and allow you to explore its features and functions, you, as the primary user of the computer on which the Software is installed for commercial purposes, may also install one copy of the Software on only one (1) home personal computer (such as your laptop or desktop) solely for your personal and non-commercial ("HPNC") use. This HPNC copy may not be used in any commercial or revenue-generating business activities, including without limitation, work-from-home, teleworking, telecommuting, or other work-related use of the Software. The HPNC copy of the Software may not be used at the same time on a home personal computer as the Software is being used on the primary computer.

(g) Key Codes, Upgrades and Updates. Prior to your purchase and as part of the registration for the thirty (30) day evaluation period, as applicable, you will receive an evaluation key code. You will receive a purchase key code when you elect to purchase the Software from either Altova GmbH or an authorized reseller. The purchase key code will enable you to activate the Software beyond the initial evaluation period. You may not re-license, reproduce or distribute any key code except with the express written permission of Altova. If the Software that you have licensed is an upgrade or an update, then the latest update or upgrade that you download and install replaces all or part of the Software previously licensed. The update or upgrade and the associated license keys does not constitute the granting of a second license to the Software in that you may not use the upgrade or updated copy in addition to the copy of the Software that it is replacing and whose license has terminated.

(h) Title. Title to the Software is not transferred to you. Ownership of all copies of the Software and of copies made by you is vested in Altova, subject to the rights of use granted to you in this Software License Agreement. As between you and Altova, documents, files, stylesheets, generated program code (including the Unrestricted Source Code) and schemas that are authored or created by you via your utilization of the Software, in accordance with its Documentation and the terms of this Software License Agreement, are your property unless they are created using Evaluation Software, as defined in Section 4 of this Agreement, in which

case you have only a limited license to use any output that contains generated program code (including Unrestricted Source Code) such as Java, C++, C#, VB.NET or XSLT and associated project files and build scripts, as well as generated XML, XML Schemas, documentation, UML diagrams, and database structures only for the thirty (30) day evaluation period.

(i) Reverse Engineering. Except and to the limited extent as may be otherwise specifically provided by applicable law in the European Union, you may not reverse engineer, decompile, disassemble or otherwise attempt to discover the source code, underlying ideas, underlying user interface techniques or algorithms of the Software by any means whatsoever, directly or indirectly, or disclose any of the foregoing, except to the extent you may be expressly permitted to decompile under applicable law in the European Union, if it is essential to do so in order to achieve operability of the Software with another software program, and you have first requested Altova to provide the information necessary to achieve such operability and Altova has not made such information available. Altova has the right to impose reasonable conditions and to request a reasonable fee before providing such information. Any information supplied by Altova or obtained by you, as permitted hereunder, may only be used by you for the purpose described herein and may not be disclosed to any third party or used to create any software which is substantially similar to the expression of the Software. Requests for information from users in the European Union with respect to the above should be directed to the Altova Customer Support Department.

(j) Other Restrictions. You may not loan, rent, lease, sublicense, distribute or otherwise transfer all or any portion of the Software to third parties except to the limited extent set forth in Section 3 or as otherwise expressly provided. You may not copy the Software except as expressly set forth above, and any copies that you are permitted to make pursuant to this Software License Agreement must contain the same copyright, patent and other intellectual property markings that appear on or in the Software. You may not modify, adapt or translate the Software. You may not, directly or indirectly, encumber or suffer to exist any lien or security interest on the Software; knowingly take any action that would cause the Software to be placed in the public domain; or use the Software in any computer environment not specified in this Software License Agreement.

You will comply with applicable law and Altova's instructions regarding the use of the Software. You agree to notify your employees and agents who may have access to the Software of the restrictions contained in this Software License Agreement and to ensure their compliance with these restrictions.

(k) THE SOFTWARE IS NEITHER GUARANTEED NOR WARRANTED TO BE ERROR-FREE NOR SHALL ANY LIABILITY BE ASSUMED BY ALTOVA IN THIS RESPECT. NOTWITHSTANDING ANY SUPPORT FOR ANY TECHNICAL STANDARD, THE SOFTWARE IS NOT INTENDED FOR USE IN OR IN CONNECTION WITH, WITHOUT LIMITATION, THE OPERATION OF NUCLEAR FACILITIES, AIRCRAFT NAVIGATION, COMMUNICATION SYSTEMS, AIR TRAFFIC CONTROL EQUIPMENT, MEDICAL DEVICES OR LIFE SUPPORT SYSTEMS, MEDICAL OR HEALTH CARE APPLICATIONS, OR OTHER APPLICATIONS WHERE THE FAILURE OF THE SOFTWARE OR ERRORS IN DATA PROCESSING COULD LEAD TO DEATH, PERSONAL INJURY OR SEVERE PHYSICAL OR ENVIRONMENTAL DAMAGE. YOU AGREE THAT YOU ARE SOLELY RESPONSIBLE FOR THE ACCURACY AND ADEQUACY OF THE SOFTWARE AND ANY DATA GENERATED OR PROCESSED BY THE SOFTWARE FOR YOUR INTENDED USE AND YOU WILL DEFEND, INDEMNIFY AND HOLD ALTOVA, ITS OFFICERS AND EMPLOYEES HARMLESS FROM ANY 3RD PARTY CLAIMS, DEMANDS, OR SUITS THAT ARE BASED UPON THE ACCURACY AND ADEQUACY OF THE SOFTWARE IN YOUR USE OR ANY DATA GENERATED BY THE SOFTWARE IN YOUR USE.

2. INTELLECTUAL PROPERTY RIGHTS

Acknowledgement of Altova's Rights. You acknowledge that the Software and any copies

that you are authorized by Altova to make are the intellectual property of and are owned by Altova and its suppliers. The structure, organization and code of the Software are the valuable trade secrets and confidential information of Altova and its suppliers. The Software is protected by copyright, including without limitation by United States Copyright Law, international treaty provisions and applicable laws in the country in which it is being used. You acknowledge that Altova retains the ownership of all patents, copyrights, trade secrets, trademarks and other intellectual property rights pertaining to the Software, and that Altova's ownership rights extend to any images, photographs, animations, videos, audio, music, text and "applets" incorporated into the Software and all accompanying printed materials. You will take no actions which adversely affect Altova's intellectual property rights in the Software. Trademarks shall be used in accordance with accepted trademark practice, including identification of trademark owners' names. Trademarks may only be used to identify printed output produced by the Software, and such use of any trademark does not give you any right of ownership in that trademark. XMLSpy®, Authentic®, StyleVision®, MapForce®, UModel®, DatabaseSpy®, DiffDog®, SchemaAgent®, SemanticWorks®, MissionKit®, Markup Your Mind®, Nanonull™, and Altova® are trademarks of Altova GmbH. (registered in numerous countries). Unicode and the Unicode Logo are trademarks of Unicode, Inc. Windows, Windows XP, Windows Vista, and Windows 7 are trademarks of Microsoft. W3C, CSS, DOM, MathML, RDF, XHTML, XML and XSL are trademarks (registered in numerous countries) of the World Wide Web Consortium (W3C); marks of the W3C are registered and held by its host institutions, MIT, INRIA and Keio. Except as expressly stated above, this Software License Agreement does not grant you any intellectual property rights in the Software. Notifications of claimed copyright infringement should be sent to Altova's copyright agent as further provided on the Altova Web Site.

3. LIMITED TRANSFER RIGHTS

Notwithstanding the foregoing, you may transfer all your rights to use the Software to another person or legal entity provided that: (a) you also transfer each of this Software License Agreement, the Software and all other software or hardware bundled or pre-installed with the Software, including all copies, updates and prior versions, and all copies of font software converted into other formats, to such person or entity; (b) you retain no copies, including backups and copies stored on a computer; (c) the receiving party secures a personalized key code from Altova; and (d) the receiving party accepts the terms and conditions of this Software License Agreement and any other terms and conditions upon which you legally purchased a license to the Software. Notwithstanding the foregoing, you may not transfer education, pre-release, or not-for-resale copies of the Software.

4. PRE-RELEASE AND EVALUATION PRODUCT ADDITIONAL TERMS

If the product you have received with this license is pre-commercial release or beta Software ("Pre-release Software"), then this Section applies. In addition, this section applies to all evaluation and/or demonstration copies of Altova software ("Evaluation Software") and continues in effect until you purchase a license. To the extent that any provision in this section is in conflict with any other term or condition in this Software License Agreement, this section shall supersede such other term(s) and condition(s) with respect to the Pre-release and/or Evaluation Software, but only to the extent necessary to resolve the conflict. You acknowledge that the Pre-release Software is a pre-release version, does not represent final product from Altova, and may contain bugs, errors and other problems that could cause system or other failures and data loss. CONSEQUENTLY, THE PRE-RELEASE AND/OR EVALUATION SOFTWARE IS PROVIDED TO YOU **"AS-IS" WITH NO WARRANTIES FOR USE OR PERFORMANCE,** AND ALTOVA DISCLAIMS ANY WARRANTY OR LIABILITY OBLIGATIONS TO YOU OF ANY KIND, WHETHER EXPRESS OR IMPLIED. WHERE LEGALLY LIABILITY CANNOT BE EXCLUDED FOR PRE-RELEASE AND/OR EVALUATION SOFTWARE, BUT IT MAY BE LIMITED, ALTOVA'S LIABILITY AND THAT OF ITS SUPPLIERS SHALL BE LIMITED TO THE SUM OF FIFTY DOLLARS (USD $50) IN TOTAL. If the Evaluation Software has a time-out feature, then the software will cease operation after the conclusion of the designated evaluation period. Upon such expiration date, your license will expire unless otherwise extended. Your

license to use any output created with the Evaluation Software that contains generated program code (including Unrestricted Source Code) such as Java, C++, C, VB.NET or XSLT and associated project files and build scripts as well as generated XML, XML Schemas, documentation, UML diagrams, and database structures terminates automatically upon the expiration of the designated evaluation period but the license to use such output is revived upon your purchase of a license for the Software that you evaluated and used to create such output. Access to any files created with the Evaluation Software is entirely at your risk. You acknowledge that Altova has not promised or guaranteed to you that Pre-release Software will be announced or made available to anyone in the future, that Altova has no express or implied obligation to you to announce or introduce the Pre-release Software, and that Altova may not introduce a product similar to or compatible with the Pre-release Software. Accordingly, you acknowledge that any research or development that you perform regarding the Pre-release Software or any product associated with the Pre-release Software is done entirely at your own risk. During the term of this Software License Agreement, if requested by Altova, you will provide feedback to Altova regarding testing and use of the Pre-release Software, including error or bug reports. If you have been provided the Pre-release Software pursuant to a separate written agreement, your use of the Software is governed by such agreement. You may not sublicense, lease, loan, rent, distribute or otherwise transfer the Pre-release Software. Upon receipt of a later unreleased version of the Pre-release Software or release by Altova of a publicly released commercial version of the Software, whether as a stand-alone product or as part of a larger product, you agree to return or destroy all earlier Pre-release Software received from Altova and to abide by the terms of the license agreement for any such later versions of the Pre-release Software.

5. LIMITED WARRANTY AND LIMITATION OF LIABILITY

(a) Limited Warranty and Customer Remedies. Altova warrants to the person or entity that first purchases a license for use of the Software pursuant to the terms of this Software License Agreement that (i) the Software will perform substantially in accordance with any accompanying Documentation for a period of ninety (90) days from the date of receipt, and (ii) any support services provided by Altova shall be substantially as described in Section 6 of this agreement. Some states and jurisdictions do not allow limitations on duration of an implied warranty, so the above limitation may not apply to you. To the extent allowed by applicable law, implied warranties on the Software, if any, are limited to ninety (90) days. Altova's and its suppliers' entire liability and your exclusive remedy shall be, at Altova's option, either (i) return of the price paid, if any, or (ii) repair or replacement of the Software that does not meet Altova's Limited Warranty and which is returned to Altova with a copy of your receipt. This Limited Warranty is void if failure of the Software has resulted from accident, abuse, misapplication, abnormal use, Trojan horse, virus, or any other malicious external code. Any replacement Software will be warranted for the remainder of the original warranty period or thirty (30) days, whichever is longer. This limited warranty does not apply to Evaluation and/or Pre-release Software.

(b) No Other Warranties and Disclaimer. THE FOREGOING LIMITED WARRANTY AND REMEDIES STATE THE SOLE AND EXCLUSIVE REMEDIES FOR ALTOVA OR ITS SUPPLIER'S BREACH OF WARRANTY. ALTOVA AND ITS SUPPLIERS DO NOT AND CANNOT WARRANT THE PERFORMANCE OR RESULTS YOU MAY OBTAIN BY USING THE SOFTWARE. EXCEPT FOR THE FOREGOING LIMITED WARRANTY, AND FOR ANY WARRANTY, CONDITION, REPRESENTATION OR TERM TO THE EXTENT WHICH THE SAME CANNOT OR MAY NOT BE EXCLUDED OR LIMITED BY LAW APPLICABLE TO YOU IN YOUR JURISDICTION, ALTOVA AND ITS SUPPLIERS MAKE NO WARRANTIES, CONDITIONS, REPRESENTATIONS OR TERMS, EXPRESS OR IMPLIED, WHETHER BY STATUTE, COMMON LAW, CUSTOM, USAGE OR OTHERWISE AS TO ANY OTHER MATTERS. TO THE MAXIMUM EXTENT PERMITTED BY APPLICABLE LAW, ALTOVA AND ITS SUPPLIERS DISCLAIM ALL OTHER WARRANTIES AND CONDITIONS, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, SATISFACTORY QUALITY,

INFORMATIONAL CONTENT OR ACCURACY, QUIET ENJOYMENT, TITLE AND NON-INFRINGEMENT, WITH REGARD TO THE SOFTWARE, AND THE PROVISION OF OR FAILURE TO PROVIDE SUPPORT SERVICES. THIS LIMITED WARRANTY GIVES YOU SPECIFIC LEGAL RIGHTS. YOU MAY HAVE OTHERS, WHICH VARY FROM STATE/JURISDICTION TO STATE/JURISDICTION.

(c) Limitation of Liability. TO THE MAXIMUM EXTENT PERMITTED BY APPLICABLE LAW EVEN IF A REMEDY FAILS ITS ESSENTIAL PURPOSE, IN NO EVENT SHALL ALTOVA OR ITS SUPPLIERS BE LIABLE FOR ANY SPECIAL, INCIDENTAL, DIRECT, INDIRECT OR CONSEQUENTIAL DAMAGES WHATSOEVER (INCLUDING, WITHOUT LIMITATION, DAMAGES FOR LOSS OF BUSINESS PROFITS, BUSINESS INTERRUPTION, LOSS OF BUSINESS INFORMATION, OR ANY OTHER PECUNIARY LOSS) ARISING OUT OF THE USE OF OR INABILITY TO USE THE SOFTWARE OR THE PROVISION OF OR FAILURE TO PROVIDE SUPPORT SERVICES, EVEN IF ALTOVA HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. IN ANY CASE, ALTOVA'S ENTIRE LIABILITY UNDER ANY PROVISION OF THIS SOFTWARE LICENSE AGREEMENT SHALL BE LIMITED TO THE AMOUNT ACTUALLY PAID BY YOU FOR THE SOFTWARE PRODUCT. Because some states and jurisdictions do not allow the exclusion or limitation of liability, the above limitation may not apply to you. In such states and jurisdictions, Altova's liability shall be limited to the greatest extent permitted by law and the limitations or exclusions of warranties and liability contained herein do not prejudice applicable statutory consumer rights of person acquiring goods otherwise than in the course of business. The disclaimer and limited liability above are fundamental to this Software License Agreement between Altova and you.

(d) Infringement Claims. Altova will indemnify and hold you harmless and will defend or settle any claim, suit or proceeding brought against you by a third party that is based upon a claim that the content contained in the Software infringes a copyright or violates an intellectual or proprietary right protected by United States or European Union law ("Claim"), but only to the extent the Claim arises directly out of the use of the Software and subject to the limitations set forth in Section 5 of this Agreement except as otherwise expressly provided. You must notify Altova in writing of any Claim within ten (10) business days after you first receive notice of the Claim, and you shall provide to Altova at no cost such assistance and cooperation as Altova may reasonably request from time to time in connection with the defense of the Claim. Altova shall have sole control over any Claim (including, without limitation, the selection of counsel and the right to settle on your behalf on any terms Altova deems desirable in the sole exercise of its discretion). You may, at your sole cost, retain separate counsel and participate in the defense or settlement negotiations. Altova shall pay actual damages, costs, and attorney fees awarded against you (or payable by you pursuant to a settlement agreement) in connection with a Claim to the extent such direct damages and costs are not reimbursed to you by insurance or a third party, to an aggregate maximum equal to the purchase price of the Software. If the Software or its use becomes the subject of a Claim or its use is enjoined, or if in the opinion of Altova's legal counsel the Software is likely to become the subject of a Claim, Altova shall attempt to resolve the Claim by using commercially reasonable efforts to modify the Software or obtain a license to continue using the Software. If in the opinion of Altova's legal counsel the Claim, the injunction or potential Claim cannot be resolved through reasonable modification or licensing, Altova, at its own election, may terminate this Software License Agreement without penalty, and will refund to you on a pro rata basis any fees paid in advance by you to Altova. THE FOREGOING CONSTITUTES ALTOVA'S SOLE AND EXCLUSIVE LIABILITY FOR INTELLECTUAL PROPERTY INFRINGEMENT. This indemnity does not apply to infringements that would not be such, except for customer-supplied elements.

6. SUPPORT AND MAINTENANCE

Altova offers multiple optional "Support & Maintenance Package(s)" ("SMP") for the version of Software product edition that you have licensed, which you may elect to purchase in addition to your Software license. The Support Period, hereinafter defined, covered by such SMP shall be delineated at such time as you elect to purchase a SMP. Your rights with respect to support

and maintenance as well as your upgrade eligibility depend on your decision to purchase SMP and the level of SMP that you have purchased:

(a) If you have not purchased SMP, you will receive the Software AS IS and will not receive any maintenance releases or updates. However, Altova, at its option and in its sole discretion on a case by case basis, may decide to offer maintenance releases to you as a courtesy, but these maintenance releases will not include any new features in excess of the feature set at the time of your purchase of the Software. In addition, Altova will provide free technical support to you for thirty (30) days after the date of your purchase (the "Support Period" for the purposes of this paragraph 6(a), and Altova, in its sole discretion on a case by case basis, may also provide free courtesy technical support during your thirty (30) day evaluation period. Technical support is provided via a Web-based support form only, and there is no guaranteed response time.

(b) If you have purchased SMP, then solely for the duration of its delineated Support Period, **you are eligible to receive the version of the Software edition** that you have licensed and all maintenance releases and updates for that edition that are released during your Support Period. For the duration of your SMP's Support Period, you will also be eligible to receive upgrades to the comparable edition of the next version of the Software that succeeds the Software edition that you have licensed for applicable upgrades released during your Support Period. The specific upgrade edition that you are eligible to receive based on your Support Period is further detailed in the SMP that you have purchased. Software that is introduced as separate product is not included in SMP. Maintenance releases, updates and upgrades may or may not include additional features. In addition, Altova will provide Priority Technical Support to you for the duration of the Support Period. Priority Technical Support is provided via a Web-based support form only and Altova will make commercially reasonable efforts to respond via e-mail to all requests within forty-eight (48) hours during Altova's business hours (MO-FR, 8am UTC – 10pm UTC, Austrian and US holidays excluded) and to make reasonable efforts to provide work-arounds to errors reported in the Software.

During the Support Period you may also report any Software problem or error to Altova. If Altova determines that a reported reproducible material error in the Software exists and significantly impairs the usability and utility of the Software, Altova agrees to use reasonable commercial efforts to correct or provide a usable work-around solution in an upcoming maintenance release or update, which is made available at certain times at Altova's sole discretion.

If Altova, in its discretion, requests written verification of an error or malfunction discovered by you or requests supporting example files that exhibit the Software problem, you shall promptly provide such verification or files, by email, telecopy, or overnight mail, setting forth in reasonable detail the respects in which the Software fails to perform. You shall use reasonable efforts to cooperate in diagnosis or study of errors. Altova may include error corrections in maintenance releases, updates, or new major releases of the Software. Altova is not obligated to fix errors that are immaterial. Immaterial errors are those that do not significantly impact use of the Software as determined by Altova in its sole discretion. Whether or not you have purchased the Support & Maintenance Package, technical support only covers issues or questions resulting directly out of the operation of the Software and Altova will not provide you with generic consultation, assistance, or advice under any circumstances.

Updating Software may require the updating of software not covered by this Software License Agreement before installation. Updates of the operating system and application software not specifically covered by this Software License Agreement are your responsibility and will not be provided by Altova under this Software License Agreement. Altova's obligations under this Section 6 are contingent upon your proper use of the Software and your compliance with the terms and conditions of this Software License Agreement at all times. Altova shall be under no obligation to provide the above technical support if, in Altova's opinion, the Software has failed due to the following conditions: (i) damage caused by the relocation of the software to another location or CPU; (ii) alterations, modifications or attempts to change the Software without Altova 's written approval; (iii) causes external to the Software, such as natural disasters, the failure or

fluctuation of electrical power, or computer equipment failure; (iv) your failure to maintain the Software at Altova's specified release level; or (v) use of the Software with other software without Altova's prior written approval. It will be your sole responsibility to: (i) comply with all Altova-specified operating and troubleshooting procedures and then notify Altova immediately of Software malfunction and provide Altova with complete information thereof; (ii) provide for the security of your confidential information; (iii) establish and maintain backup systems and procedures necessary to reconstruct lost or altered files, data or programs.

7. SOFTWARE ACTIVATION, UPDATES AND LICENSE METERING

(a) License Metering. Altova has a built-in license metering module that helps you to avoid any unintentional violation of this Software License Agreement. Altova may use your internal network for license metering between installed versions of the Software.

(b) Software Activation. **Altova's Software may use your internal network and Internet connection for the purpose of transmitting license-related data at the time of installation, registration, use, or update to an Altova-operated license server and validating the authenticity of the license-related data in order to protect Altova against unlicensed or illegal use of the Software and to improve customer service. Activation is based on the exchange of license related data between your computer and the Altova license server. You agree that Altova may use these measures and you agree to follow any applicable requirements. You further agree that use of license key codes that are not or were not generated by Altova and lawfully obtained from Altova, or an authorized reseller as part of an effort to activate or use the Software violates Altova's intellectual property rights as well as the terms of this Software License Agreement. You agree that efforts to circumvent or disable Altova's copyright protection mechanisms or license management mechanism violate Altova's intellectual property rights as well as the terms of this Software License Agreement. Altova expressly reserves the rights to seek all available legal and equitable remedies to prevent such actions and to recover lost profits, damages and costs.**

(c) LiveUpdate. Altova provides a new LiveUpdate notification service to you, which is free of charge. Altova may use your internal network and Internet connection for the purpose of transmitting license-related data to an Altova-operated LiveUpdate server to validate your license at appropriate intervals and determine if there is any update available for you.

(d) Use of Data. The terms and conditions of the Privacy Policy are set out in full at http://www.altova.com/privacy and are incorporated by reference into this Software License Agreement. By your acceptance of the terms of this Software License Agreement and/or use of the Software, you authorize the collection, use and disclosure of information collected by Altova for the purposes provided for in this Software License Agreement and/or the Privacy Policy. Altova has the right in its sole discretion to amend this provision of the Software License Agreement and/or Privacy Policy at any time. You are encouraged to review the terms of the Privacy Policy as posted on the Altova Web site from time to time.

(e) Notice to European Users. Please note that the information as described in paragraph 7(d) above may be transferred outside of the European Economic Area, for purposes of processing, analysis, and review, by Altova, Inc., a company located in Beverly, Massachusetts, U.S.A., or its subsidiaries or Altova's subsidiaries or divisions, or authorized partners, located worldwide. You are advised that the United States uses a sectoral model of privacy protection that relies on a mix of legislation, governmental regulation, and self-regulation. You are further advised that the Council of the European Union has found that this model does not provide "adequate" privacy protections as contemplated by Article 25 of the European Union's Data Directive. (Directive 95/46/EC, 1995 O.J. (L 281) 31). Article 26 of the European Union's Data Directive allows for transfer of personal data from the European Union to a third country if the individual has unambiguously given his consent to the transfer of personal information, regardless of the third country's level of protection. By agreeing to this

Software License Agreement, you consent to the transfer of all such information to the United States and the processing of that information as described in this Software License Agreement and the Privacy Policy.

8. TERM AND TERMINATION

This Software License Agreement may be terminated (a) by your giving Altova written notice of termination; (b) by Altova, at its option, giving you written notice of termination if you commit a breach of this Software License Agreement and fail to cure such breach within ten (10) days after notice from Altova; or (c) at the request of an authorized Altova reseller in the event that you fail to make your license payment or other monies due and payable. In addition the Software License Agreement governing your use of a previous version that you have upgraded or updated of the Software is terminated upon your acceptance of the terms and conditions of the Software License Agreement accompanying such upgrade or update. Upon any termination of the Software License Agreement, you must cease all use of the Software that this Software License Agreement governs, destroy all copies then in your possession or control and take such other actions as Altova may reasonably request to ensure that no copies of the Software remain in your possession or control. The terms and conditions set forth in Sections 1(h), 1(i), 1(j), 1(k), 2, 5(b), 5(c), 5(d), 7(d), 7(e), 9, 10 and 11 survive termination as applicable.

9. RESTRICTED RIGHTS NOTICE AND EXPORT RESTRICTIONS

The Software was developed entirely at private expense and is commercial computer software provided with **RESTRICTED RIGHTS**. Use, duplication or disclosure by the U.S. Government or a U.S. Government contractor or subcontractor is subject to the restrictions set forth in this Agreement and as provided in FAR 12.211 and 12.212 (48 C.F.R. §12.211 and 12.212) or DFARS 227. 7202 (48 C.F.R. §227-7202) as applicable. Consistent with the above as applicable, Commercial Computer Software and Commercial Computer Documentation licensed to U.S. government end users only as commercial items and only with those rights as are granted to all other end users under the terms and conditions set forth in this Software License Agreement. Manufacturer is Altova GmbH, Rudolfsplatz, 13a/9, A-1010 Vienna, Austria/EU. You may not use or otherwise export or re-export the Software or Documentation except as authorized by United States law and the laws of the jurisdiction in which the Software was obtained. In particular, but without limitation, the Software or Documentation may not be exported or re-exported (i) into (or to a national or resident of) any U.S. embargoed country or (ii) to anyone on the U.S. Treasury Department's list of Specially Designated Nationals or the U.S. Department of Commerce's Table of Denial Orders. By using the Software, you represent and warrant that you are not located in, under control of, or a national or resident of any such country or on any such list.

10. THIRD PARTY SOFTWARE

The Software may contain third party software which requires notices and/or additional terms and conditions. Such required third party software notices and/or additional terms and conditions are located at our Website at http://www.altova.com/legal_3rdparty.html and are made a part of and incorporated by reference into this Agreement. By accepting this Agreement, you are also accepting the additional terms and conditions, if any, set forth therein.

11. GENERAL PROVISIONS

If you are located in the European Union and are using the Software in the European Union and not in the United States, then this Software License Agreement will be governed by and construed in accordance with the laws of the Republic of Austria (excluding its conflict of laws principles and the U.N. Convention on Contracts for the International Sale of Goods) and you expressly agree that exclusive jurisdiction for any claim or dispute with Altova or relating in any way to your use of the Software resides in the Handelsgericht, Wien (Commercial Court, Vienna) and you further agree and expressly consent to the exercise of personal jurisdiction in

the Handelsgericht, Wien (Commercial Court, Vienna) in connection with any such dispute or claim.

If you are located in the United States or are using the Software in the United States then this Software License Agreement will be governed by and construed in accordance with the laws of the Commonwealth of Massachusetts, USA (excluding its conflict of laws principles and the U.N. Convention on Contracts for the International Sale of Goods) and you expressly agree that exclusive jurisdiction for any claim or dispute with Altova or relating in any way to your use of the Software resides in the federal or state courts of the Commonwealth of Massachusetts and you further agree and expressly consent to the exercise of personal jurisdiction in the federal or state courts of the Commonwealth of Massachusetts in connection with any such dispute or claim.

If you are located outside of the European Union or the United States and are not using the Software in the United States, then this Software License Agreement will be governed by and construed in accordance with the laws of the Republic of Austria (excluding its conflict of laws principles and the U.N. Convention on Contracts for the International Sale of Goods) and you expressly agree that exclusive jurisdiction for any claim or dispute with Altova or relating in any way to your use of the Software resides in the Handelsgericht, Wien (Commercial Court, Vienna) and you further agree and expressly consent to the exercise of personal jurisdiction in the Handelsgericht Wien (Commercial Court, Vienna) in connection with any such dispute or claim. This Software License Agreement will not be governed by the conflict of law rules of any jurisdiction or the United Nations Convention on Contracts for the International Sale of Goods, the application of which is expressly excluded.

This Software License Agreement contains the entire agreement and understanding of the parties with respect to the subject matter hereof, and supersedes all prior written and oral understandings of the parties with respect to the subject matter hereof. Any notice or other communication given under this Software License Agreement shall be in writing and shall have been properly given by either of us to the other if sent by certified or registered mail, return receipt requested, or by overnight courier to the address shown on Altova's Web site for Altova and the address shown in Altova's records for you, or such other address as the parties may designate by notice given in the manner set forth above. This Software License Agreement will bind and inure to the benefit of the parties and our respective heirs, personal and legal representatives, affiliates, successors and permitted assigns. The failure of either of us at any time to require performance of any provision hereof shall in no manner affect such party's right at a later time to enforce the same or any other term of this Software License Agreement. This Software License Agreement may be amended only by a document in writing signed by both of us. In the event of a breach or threatened breach of this Software License Agreement by either party, the other shall have all applicable equitable as well as legal remedies. Each party is duly authorized and empowered to enter into and perform this Software License Agreement. If, for any reason, any provision of this Software License Agreement is held invalid or otherwise unenforceable, such invalidity or unenforceability shall not affect the remainder of this Software License Agreement, and this Software License Agreement shall continue in full force and effect to the fullest extent allowed by law. The parties knowingly and expressly consent to the foregoing terms and conditions.

Last updated: 2011-06-08

Index

A

Activating the software, 325
Active configuration,
for global resources, 304
Alias,
see Global Resources, 177
Altova Engines,
in Altova products, 351
Altova extension functions,
chart functions (see chart functions), 344
general functions, 344
Altova extensions,
chart functions (see chart functions), 344
Altova Global Resources,
see under Global Resources, 177
Altova products, 17
Altova support, 17
Altova XML Parser,
about, 350
Altova XSLT 1.0 Engine,
limitations and implementation-specific behavior, 332
Altova XSLT 2.0 Engine,
general information about, 334
information about, 334
Append,
row (in Authentic View), 294
Apply, 313
Assign,
shortcut to a command, 306
Assigning StyleVision Power Stylesheet to XML file, 291
atomization of nodes,
in XPath 2.0 and XQuery 1.0 evaluation, 340
Attribute preview, 316
Attribute values,
entering in Authentic View, 143
AttributeFormDefault,
settings in Schema Design View, 275
Attributes entry helper,
in Authentic View, 90
Attributes of schema components,
defining in Schema View, 54
Authentic menu, 289
dynamic table editing, 86
markup display, 86
Authentic View,
adding nodes, 138
applying elements, 138
CDATA sections in, 141
clearing elements, 138
context menu, 136
context menus, 94
data entry devices in, 141
displaying markup tags, 136
document display, 88
editing data in an XML DB, 291
editing DB data in, 290
editing XML in, 106
entering attribute values, 143
entering data in, 141
entities in, 141
entry helpers, 136
entry helpers in, 90
formating text in, 86
inserting entities in, 143
inserting nodes, 138
interface, 85
main window in, 88
markup display in, 86, 88
opening an XML document in, 135
opening new XML file in, 289
overview of GUI, 85
paste as XML/Text, 94
printing an XML document from, 144
removing nodes, 138
special characters in, 141
SPS Tables, 150
switching to, 297
tables (SPS and XML), 150
tables in, 138
toolbar icons, 86
tutorial, 134
usage of important features, 146
usage of XML tables, 151
XML table icons, 155
XML tables, 151
Authentic View template, 135
Authentic XML, 132
Auto-complete,

Auto-complete,
text view enable/disable, 315
Auto-hiding windows, 9
Automatic validation, 314

B

Back,
in Schema View, 84
Background,
status updates, 208
status updates - increase interval, 194
Background Information, 348
backwards compatibility,
of XSLT 2.0 Engine, 334
Base type,
modifying, 70
Big-endian, 319
Bookmark, 324
Bookmark margin, 299
Bookmarks,
inserting and removing, 236
navigating, 236
Bookmarks in Text View, 44
Browser, 316
View, 297
Browser menu, 301
Browser View, 97, 301
back, 301
font size, 301
forward, 301
refresh content, 301
separate window, 301
stop loading page, 301

C

Carriage return key,
see Enter key, 167
Cascade,
Window, 322
Catalog,
Oasis XML, 267
Catalogs, 116
CDATA sections,
inserting in Authentic View, 146
Changing view,
to Authentic View, 86
Chapters, 324
Character,
position, 298
character entities,
in HTML output of XSLT transformation, 332
character normalization,
in XQuery document, 337
Character-Set,
encoding, 319
Code page, 316
Collapse,
unselected, 298
collations,
in XPath 2.0, 340
in XQuery document, 337
Color, 316, 317
tab, 318
table, 318
Command,
add to toolbar/menu, 304
context menu, 310
delete from menu, 310
reset menu, 310
Command line, 328
Commands,
listing in key map, 325
Commenting in and out,
in XML documents in Text View., 103
Commenting XML text in and out, 236
Component Navigator, 71
Compositor,
in Schema View, 58
Configurations,
of a global resource, 178
Configurations in global resources, 187
Configure view,
dialog for Content Model View, 276
Constraints, 74
Content Model View, 58
configuring, 276
Content models,
of schema components, 58
Context menu,
commands, 310

© 2011 Altova GmbH

Context menus,
in Authentic View, 94
Copy command, 231
Copy XPath, 234
Copy XPointer, 234
Copyright information, 327, 355
count() function,
in XPath 1.0, 332
count() function in XPath 2.0,
see fn:count(), 340
CPU,
load - increase background status updates, 194
CR&LF, 313
CSS, 170
auto-completion, 173
document outline, 173
Info window, 173
properties, 173
syntax coloring, 173
CSS Info window, 173
CustomCatalog, 267
Customization, 16
Customize,
context menu, 310
menu, 310
toolbar/menu commands, 304
Cut command, 231
CVS, 194

D

Databases,
editing in Authentic View, 290
see also DB, 157
datatypes,
in XPath 2.0 and XQuery 1.0, 340
Date Picker,
using in Authentic View, 163
Dates,
changing manually, 164
DB,
creating queries, 158
editing in Authentic View, 157, 162
filtering display in Authentic View, 158
navigating tables in Authentic View, 157
parameters in DB queries, 158
queries in Authentic View, 157
deep-equal() function in XPath 2.0,
see fn:deep-equal(), 340
Default,
encoding, 319
menu, 310
Default editor, 314
default functions namespace,
for XPath 2.0 and XQueyr 1.0 expressions, 340
in XSLT 2.0 stylesheets, 334
Default view,
setting in Main Window, 314
Delete,
command from context menu, 310
command from toolbar, 304
icon from toolbar, 304
row (in Authentic View), 294
shortcut, 306
toolbar, 305
Delete command, 231
Derived types,
modifying base type of, 70
Deriving a schema type, 78
Diffdog,
configure for differencing, 208
Differencing,
configuring Diffdog, 208
Display all globals, 280
Display diagram, 280
Display Settings, 318
Distribution,
of Altova's software products, 355, 356, 358
Dockable window, 322
Docking windows, 9
Documents in Main Window, 10
DTD,
assigning to XML document, 271
generate outline XML file from, 272
generating from XML Schema (Enterprise and Professional editions), 115
go to definition in from XML document, 272
go to from XML document, 271
menu commands related to, 271
DTD/Schema menu, 271
DTDs, 113, 313, 314
converting to XML Schemas (Enterprise and Professional editions), 114

© 2011 Altova GmbH

DTDs, 113, 313, 314
editing in Grid View (Enterprise and Professional editions), 114
editing in Text View, 114
generating XML document from, 114
Duplicate,
row (in Authentic View), 294
Dynamic (SPS) tables in Authentic View,
usage of, 150
Dynamic tables,
editing, 86

E

Edit menu, 231
Edited with XMLSPY, 313
Editing in Text View, 47
Editing views, 41
element type,
specifying in XML document, 24
ElementFormDefault,
settings in Schema Design View, 275
Elements entry helper,
in Authentic View, 90
E-mail,
sending files with, 228
Empty elements, 314
Empty lines,
in XML documents in Text View, 103
encoding,
default, 319
in XQuery document, 337
of files, 222
End User License Agreement, 355, 359
End-of-line markers, 299
Engine information, 331
Enhanced Grid View, 296
Enter key,
effects of using, 167
Entities,
defining in Authentic View, 146, 164
in XML Schema-based XML, 101
inserting in Authentic View, 143, 146
Entities entry helper,
in Authentic View, 90
Entry Helper, 19
in Grid View, 32
Entry helpers, 13
for XML documents, 108
for XQuery, 127
in Schema View, 71
toggling display on and off, 323
updating, 270
Entry helpers in Text View, 49
Entry-Helper, 322, 323
Evaluation key,
for your Altova software, 325
Evaluation period,
of Altova's software products, 355, 356, 358
Example files,
tutorial, 18
Examples,
location of installed files, 16
Expand,
fully, 298
Explorer, 314
Extension functions for XSLT and XQuery, 344
external functions,
in XQuery document, 337
External parsed entites, 314
External XSL processor, 319

F

Favorites, 324
File,
closing, 222
creating new, 215
default encoding, 319
encoding, 222
opening, 218
opening options, 313
printing options, 229
saving, 223
sending by e-mail, 228
tab, 313
File extensions,
customizing, 267
for XQuery files, 126
setting extensions as file type, 116
File menu, 215
File paths,

© 2011 Altova GmbH

File paths,
inserting in XMl document, 103
File types, 314
Files,
adding to source control, 248
most recently used, 230
Find,
and replace text in document, 236
text in document, 235
Floating windows, 9
fn:base-uri in XPath 2.0,
support in Altova Engines, 341
fn:collection in XPath 2.0,
support in Altova Engines, 341
fn:count() in XPath 2.0,
and whitespace, 340
fn:current-date in XPath 2.0,
support in Altova Engines, 341
fn:current-dateTime in XPath 2.0,
support in Altova Engines, 341
fn:current-time in XPath 2.0,
support in Altova Engines, 341
fn:data in XPath 2.0,
support in Altova Engines, 341
fn:deep-equal() in XPath 2.0,
and whitespace, 340
fn:id in XPath 2.0,
support in Altova Engines, 341
fn:idref in XPath 2.0,
support in Altova Engines, 341
fn:index-of in XPath 2.0,
support in Altova Engines, 341
fn:in-scope-prefixes in XPath 2.0,
support in Altova Engines, 341
fn:last() in XPath 2.0,
and whitespace, 340
fn:lower-case in XPath 2.0,
support in Altova Engines, 341
fn:normalize-unicode in XPath 2.0,
support in Altova Engines, 341
fn:position() in XPath 2.0,
and whitespace, 340
fn:resolve-uri in XPath 2.0,
support in Altova Engines, 341
fn:static-base-uri in XPath 2.0,
support in Altova Engines, 341
fn:upper-case in XPath 2.0,
support in Altova Engines, 341
Folding margin, 299
Font,
schema, 316
Schema Documentation, 316
Font size,
in Browser View, 301
Fonts in Text View, 42
Formatting in Text View, 42
Forward,
in Schema View, 84
Full-text,
search, 324
functions,
see under XSLT 2.0 functions, 336
XPath 2.0 and XQuery 1.0, 340

G

Global,
settings, 313
Global components,
creating in Schema Overview, 54
Global resources, 177
active configuration for, 304
changing configurations, 187
copying configurations, 183
defining, 178, 303
defining file-type, 180
defining folder-type, 182
toolbar activation, 305
using, 184, 187
using file-type and folder-type, 184
Global Resources XML File, 178
Go to File, 299
Go to line/char, 298
Grammar, 314
Graphics formats,
in Authentic View, 166
Gray bar, 297
Grid fonts, 316
Grid view, 51, 296, 297
appending elements and attributes, 32
editing XML documents in (Enterprise and Professional editions), 105
entry helpers in, 52
switching to Table View, 264

© 2011 Altova GmbH

Grid view, 51, 296, 297
using Entry Helpers, 32
GUI description, 9

H

Help,
contents, 324
index, 324
key map, 325
search, 324
Help menu, 324
Help system, 324
Hide, 322, 323
Hide markup, 86, 88
Hide markup (in Authentic View), 294
Hotkey, 306
HTML, 170
HTML documents,
editing, 171
Info window, 171
HTML Info window, 171

I

Icon,
add to toolbar/menu, 304
show large, 312
Identity constraints, 74
defining in Schema View, 54
Image formats,
in Authentic View, 166
implementation-specific behavior,
of XSLT 2.0 functions, 336
implicit timezone,
and XPath 2.0 functions, 340
Indentation,
in Text View, 235
Indentation guides, 299
Indentation in Text View, 42
Index,
help, 324
Info,
window, 19
Info Window, 13, 322, 323
for CSS documents, 173
for HTML documents, 171
Insert,
row (in Authentic View), 294
Installation,
location of example files, 16
Installing,
version control systems, 201
Intelligent Editing, 315
Internet, 326, 327
Internet usage,
in Altova products, 354

J

JSON, 170

K

Key map, 325
Keyboard shortcut, 306
Key-codes,
for your Altova software, 325

L

Large markup (in Authentic View), 294
last() function,
in XPath 1.0, 332
last() function in XPath 2.0,
see fn:last(), 340
Legal information, 355
library modules,
in XQuery document, 337
License, 359
information about, 355
License metering,
in Altova products, 357
Licenses,
for your Altova software, 325
Line,
go to, 298

© 2011 Altova GmbH

Line length,
word wrap in text view, 298
Line margin, 299
Line numbering in Text View, 44
Line-breaks, 313
Links,
following in Authentic View, 146
Little-endian, 319

M

Main window, 10, 19
MainCatalog, 267
MapForce, 272
Markup,
in Authentic View, 86, 88
Markup (in Authentic View),
hide, 294
show small/large/mixed, 294
Maximum cell width, 316
Memory,
storage of schema information, 274
Memory requirements, 349
Menu,
add/delete command, 304
Authentic, 289
customize, 310
Default/XMLSPY, 310
delete commands from, 310
DTD/Schema, 271
Edit, 231
Help, 324
Project, 238
Schema Design, 275
Tools, 303
View, 296
Window, 322
XML, 264
XSL/XQuery, 281
Menu Bar, 15
Menu Browser, 301
Messages Window, 14
Microsoft® SharePoint® Server, 258
MIME, 314
Mixed markup (in Authentic View), 294
Mostly recently used files,
list of, 230
Move up/down,
row (Authentic View, 295
MS Visual Source Safe, 194
MSXML, 319
Multi-user, 313

N

namespaces,
in XQuery document, 337
in XSLT 2.0 stylesheet, 334
settings in Schema Design View, 275
Navigation history, 84
New file,
creating, 215
New XML document,
creating, 22
Non-XML files, 314

O

OASIS,
XML catalog, 267
Open,
file, 218
Opening options,
file, 313
Optimal Widths, 298, 316
Ordering Altova software, 325
OS,
for Altova products, 349
Output formatting, 313
Output windows,
toggling display on and off, 322
Overview, 19

P

Parameters,
in DB queries, 158
passing to stylesheet via interface, 283
Parser,

© 2011 Altova GmbH

Parser,
built into Altova products, 350
XSLT, 319
Paste,
as Text, 146
as XML, 146
Paste As,
Text, 94
XML, 94
Paste command, 231
PDF,
transforming to in XMLSpy, 123
Platforms,
for Altova products, 349
Position,
Character, 298
Line, 298
position() function,
in XPath 1.0, 332
position() function in XPath 2.0,
see fn:position(), 340
Presentation, 316
Pretty-print,
in Text View, 235
Print setup, 229
Printing,
from Authentic View, 144
Printing options, 229
Program settings, 313
Project,
properties, 261
window, 19
Project management in XMLSpy, 38
Project menu, 238
Project Window, 11, 322, 323
toggling display on and off, 323
Projects,
adding active files to, 255
adding external folders to, 255
adding external Web folders to, 258
adding files to, 254
adding folders to, 255
adding global resources to, 254
adding related files to, 255
adding to source control, 248
adding URL to, 254
batch processing with, 192
benefits of using, 192
closing, 240
creating new, 239
how to create and edit, 189
location of installed files, 16
most recently used, 263
naming, 189
opening, 240
overview, 238
overview of, 188
properties of, 189
reloading, 240
saving, 189, 240
using, 192
Projects in XMLSpy,
benefits of, 38
how to create, 38
Provider,
source control, 194
PUBLIC,
identifier - catalog, 267
PVCS Version Manager, 194

Q

QName serialization,
when returned by XPath 2.0 functions, 341
Queries,
for DB display in Authentic View, 158

R

Redo command, 231
Registering your Altova software, 325
Registry,
settings, 313
Regular expressions,
in search string, 235
Reload, 313
Reloading,
changed files, 222
Repeated elements, 315
Replace,
text, 235
text in document, 236

© 2011 Altova GmbH

Repositories, 194
Reset,
menu commands, 310
shortcut, 306
toolbar & menu commands, 305
Resources,
increase - background status updates, 194
Return key,
see Enter key, 167
RichEdit 3.0, 317
Right-to-left writing systems, 353
Row,
append (in Authentic View), 294
delete (in Authentic View), 294
duplicate (in Authentic View), 294
insert (in Authentic View), 294
move up/down, 295

S

Saving files,
encoding of, 222
Schema,
also see XML Schema, 271
Design view, 296
Documentation font, 316
settings, 313
Schema Design menu, 275
Schema Design View,
Display all globals, 280
Display diagram, 280
zoom feature, 279
Schema editing,
annotations, 54, 58
comments, 54, 58
content model, 58
processing instructions, 54, 58
Schema fonts, 316
Schema Overview, 54
schema validation of XML document,
for XQuery, 337
Schema View, 54
entry helpers, 71
moving back and forward, 84
schema-awareness,
of XPath 2.0 and XQuery Engines, 340
Schemas,
in memory, 274
Scripts in XSLT/XQuery,
see under Extension functions, 344
Search,
help, 324
see Find, 236
Select All command, 235
Settings, 16, 313
SharePoint® Server, 258
Shortcut, 306
assigning/deleting, 306
show in tooltip, 312
Show, 322, 323
Show large markup, 86, 88
Show mixed markup, 86, 88
Show small arkup, 88
Show small markup, 86
Side-by-side, 316
Size, 317
Small markup (in Authentic View), 294
Smart Restrictions, 78
Software product license, 359
Source control, 194, 321
add to source control, 248
changing provider, 253
checking in, 246
checking out, 245
enabling, disabling, 242
get latest version, 243
getting files, 243
getting folders, 244
open project, 241
properties, 252
refresh status, 253
removing from, 249
sharing from, 249
show differences, 251
show history, 250
supported providers, 240
undo check out, 247
Source control manager, 253
Source folding in Text View, 44
Speed up,
increase - background status updates, 194
Splash screen, 316
SPP file locations, 238
SPS,

© 2011 Altova GmbH

SPS,
assigning to new XML file, 215
SPS file,
assigning to XML file, 291
SPS tables,
editing dynamic tables, 86
SPS tables in Authentic View,
usage of, 150
StarTeam, 194
Static (SPS) tables in Authentic View,
usage of, 150
Status,
background updates, 208
Status Bar, 15
Structured text, 315
Style, 316, 317
Stylesheet PI, 287
StyleVision, 272
for editing StyleVision Power Stylesheet, 291
StyleVision Power Stylesheet,
assigning to XML file, 291
editing in StyleVision, 291
Support Center, 326
Support options, 17
Syntax coloring,
for XQuery, 127
Syntax-coloring, 314, 316

T

Tab characters, 313
Tab size,
and pretty-printing, 299
setting, 299
Table,
build automatically, 314
colors, 318
Table command,
in Grid View, 264
Table of contents, 324
Table View, 315
and switching to Grid View, 264
sorting columns, 265
Tables,
editing dynamic (SPS) tables, 86
in Authentic View, 138
Tables in Authentic View,
icons for editing XML tables, 155
usage of, 150
using SPS (static and dynamic) tables, 150
using XML tables, 151
Technical Information, 348
Technical Support, 326
Template files,
for new documents, 215
Template folder, 18
Template XML File,
in Authentic View, 135
Templates,
of XML documents in Authentic View, 289
Text,
editing in Authentic View, 146
find and replace, 236
finding in document, 235
font, 317
formatting in Authentic View, 146
pretty-printing, 235
Text view, 42, 296
and commenting in XML documents, 103
and empty lines in XML documents, 103
auto-complete enable/disable, 315
bookmarks in, 44
editing in, 25
Entry helpers in, 49
font properties, 42
formatting of text, 42
indentation, 42
indentation in, 44
intelligent editing features, 47
line numbering in, 44
schema fonts, 316
source folding in, 44
special editing features for XML documents, 103
word-wrapping, 42
Text View Settings dialog, 299
Tile,
horizontally, 322
vertically, 322
Toggle, 322, 323
Toolbar, 15
activate/deactivate, 305
add command to, 304
create new, 305
reset toolbar & menu commands, 305

© 2011 Altova GmbH

Toolbar, 15
show large icons, 312
Tools menu, 303
Tooltip,
show, 312
show shortcuts in, 312
Topic,
view on TOC, 324
Transformation,
see XSLT transformation, 282
Turn off automatic validation, 314
Tutorial,
example files, 18
goals, 18
Tutorials,
location of installed files, 16
type,
extension in XML document, 24

U

UCS-2, 319
Undo command, 231
Unicode,
support in Altova products, 352
Unicode support,
in Altova products, 352, 353
Unselected, 298
Update,
background status updates, 208
Update Entry Helpers command, 270
URL,
sending by e-mail, 228
User interface description, 9
User Manual, 3, 6
User Reference, 214
UTF-16, 319

V

Validating,
XML documents, 29
Validating XML documents, 101
Validation, 16, 267
assigning DTD to XML document, 271
assigning XML Schema to XML document, 271
Validation messages, 14
Validator,
in Altova products, 350
Version control,
Diffdog differencing editor, 208
installation procedures, 201
Version Number, 327
View,
Browser view, 297
Collapse, 297, 298
Enhance Grid view, 296
Expand, 297, 298
Go to File, 299
Go to line/char, 298
Optimal widths, 298
Schema Design view, 296
Text view, 296
View menu, 296

W

Watch for changes, 313
Web Server, 326, 327
Well-formedness check, 266
for XML document, 29
Well-formedness of XML documents, 101
whitespace handling,
and XPath 2.0 functions, 340
whitespace in XML document,
handling by Altova XSLT 2.0 Engine, 334
Whitespace markers, 299
whitespace nodes in XML document,
and handling by XSLT 1.0 Engine, 332
Window,
Cascade, 322
Entry-Helper, 322, 323
Info, 322, 323
Open, 323
Project, 322, 323
Tile horizontally, 322
Tile vertically, 322
Window menu, 322
Windows,
auto-hiding, 9

© 2011 Altova GmbH

Windows,
floating, docking, tabbing, 9
managing display of, 9
overview, 19
support for Altova products, 349
Word wrap,
enable/disable, 298
Word-wrapping in Text View, 42
Wrap,
word wrap enable/disable, 298

X

XInclude, 231
inserting in Grid View, 231
inserting in Text View, 231
inserting in XML document, 103
XML,
Oasis catalog, 267
XML DB,
loading new data row into Authentic View, 291
loading new XML data row, 157
XML document,
assigning to XSLT stylesheet, 287
creating new, 22
editing in Text View, 25
generating from DTD, 114
generating from XML Schema (Enterprise and Professional editions), 115
opening in Authentic View, 135
XML document creation,
tutorial, 22
XML documents, 98
and commenting in Text View, 103
and empty lines in Text View, 103
and XPath expression of a node, 103
and XQuery, 110
assigning schemas (incl. DTDs), 101
automatic validation, 99
automating XQuery executions of, 110
automating XSLT transformations of, 110
checking validity of, 29
checking well-formedness, 101
default views of, 99
editing in Authentic View, 106
editing in Grid View (Enterprise and Professional editions), 105
encoding of, 112
evaluating XPath expressions on, 112
generating schemas from, 112
importing and exporting text, 112
inserting file paths in, 103
inserting XInclude, 103
opening, 99
saving, 99
searching and replacing in, 112
Text View editing features for, 103
transforming with XSLT, 110
validating, 101
XML file,
generate from DTD or XML Schema, 272
XML menu, 264
XML Parser,
about, 350
XML Schema,
also see Schema, 271
assigning to XML document, 271
configuring Content Model View, 276
generate outline XML file from, 272
generating from DTD (Enterprise and Professional editions), 114
go to definition in from XML document, 272
go to from XML document, 272
menu commands related to, 271
namespaces settings in Schema Design View, 275
settings in Schema Design View, 275
XML Schema tutorial, 20
XML Schemas, 113
and global resources, 101
converting to DTD (Enterprise and Professional editions), 115
editing in Grid View (Enterprise and Professional editions), 115
editing in Schema View (Enterprise and Professional editions), 115
editing in Text View, 115
generating XML document from, 115
plus DTDs, 101
XML tables in Authentic View,
icons for editing, 155
usage of, 151
xml:base, 82
and XInclude, 231
xml:id, 82
xml:lang, 82
xml:space, 82

© 2011 Altova GmbH

XML-Conformance, 314
XMLSPY, 214, 327
features, 17
help, 17
XMLSpy Enterprise Edition,
user manual, 3
XML-Text, 315
XPath,
generating of a node in an XML document, 103
XPath 2.0 functions,
general information about, 340
implementation information, 340
see under fn: for specific functions, 340
XPath functions support,
see under fn: for individual functions, 341
XPath of selected node in XML document,
copying to the clipborad, 234
XPath to selected node, 85
XPointer,
generating of a node in an XML document, 103
XPointer of selected node in XML document,
copying to the clipborad, 234
XPointers, 231
XQuery,
document validation, 130
editing in Text View, 125
engine information, 331
entry helpers, 127
execution, 130
Extension functions, 344
intelligent editing features, 129
opening file, 126
passing variables to the XQuery document, 283
syntax coloring, 127
XQuery 1.0 Engine,
information about, 337
XQuery 1.0 functions,
general information about, 340
implementation information, 340
see under fn: for specific functions, 340
XQuery Execution, 286
XQuery files,
setting file extensions in XMLSpy, 126
XQuery processor,
in Altova products, 351
xs:QName,
also see QName, 341
xsi:type,
usage, 24
XSL,
see XSLT, 288
XSL transformation,
see XSLT, 34
XSL/XQuery menu, 281
XSL:FO,
and XSLT transformations, 123
xsl:preserve-space, 332
xsl:strip-space, 332
XSLT, 298
and batch transformations, 123
auto-completion in Text View, 121
documents, 121
engine information, 331
entry helpers for, 121
Extension functions, 344
functionality in XMLSpy, 121
modifying in XMLSpy, 36
processor, 319
transformations in XMLSpy, 123
validating, 121
XSLT 1.0 Engine,
limitations and implementation-specific behavior, 332
XSLT 2.0 Engine,
general information about, 334
information about, 334
XSLT 2.0 functions,
implementation-specific behavior of, 336
see under fn: for specific functions, 336
XSLT 2.0 stylesheet,
namespace declarations in, 334
XSLT parameters,
passing to stylesheet via interface, 283
XSLT processors,
in Altova products, 351
XSLT stylesheet,
assigning to XML document, 287
assigning XML document to, 287
opening, 288
XSLT stylesheet for FO,
assigning to XML document, 287
XSLT transformation, 281
assigning XSLT file, 34
in XMLSpy, 35
to FO, 282
to PDF, 282
tutorial, 34

© 2011 Altova GmbH

Z

Zoom feature,
in Schema Design View, 279
Zooming in Text View, 44

© 2011 Altova GmbH