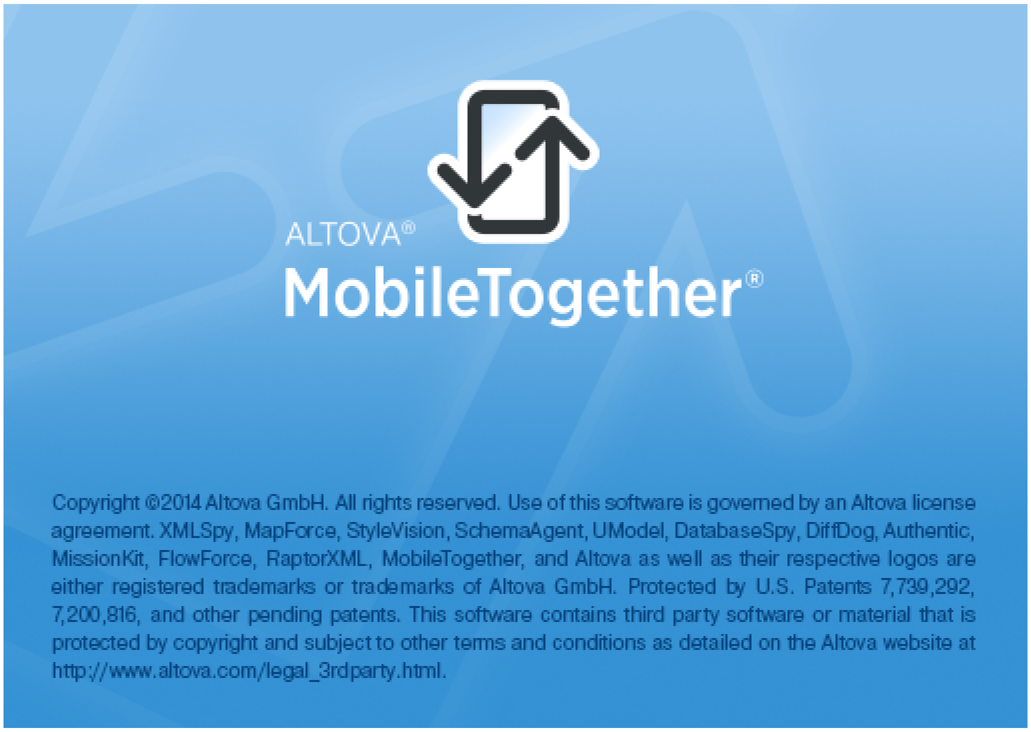
User and Reference Manual



ALTOVA®

MobileTogether®

Altova MobileTogether Designer User & Reference Manual

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: 2014

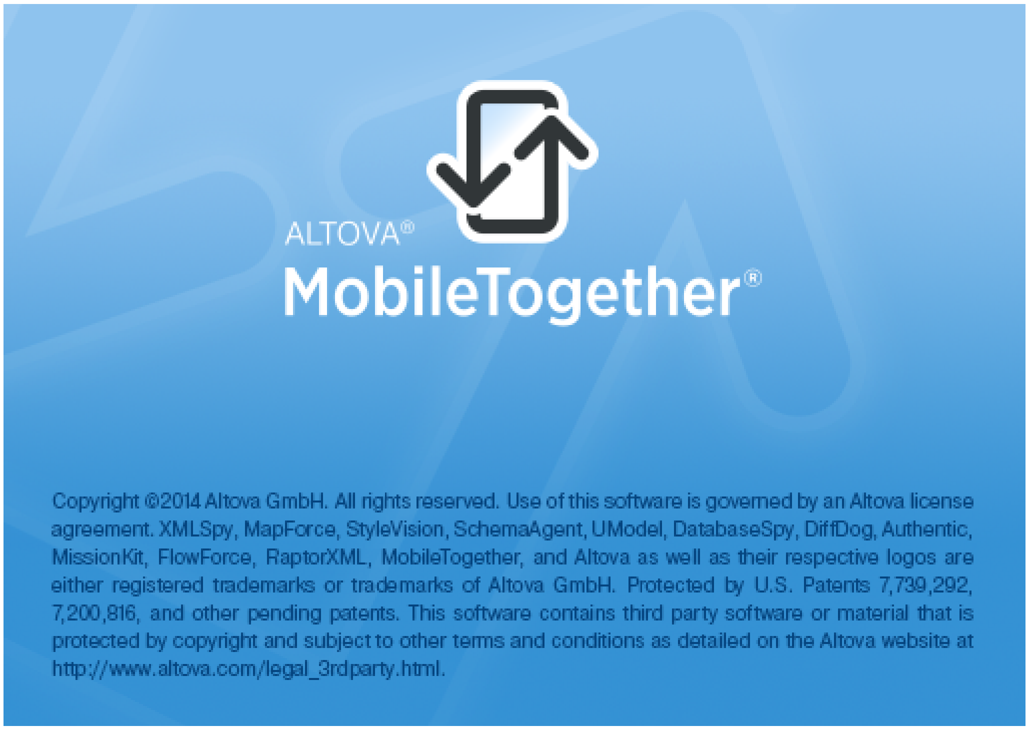
Table of Contents

Chapter 1

Altova MobileTogether Designer

1 Altova MobileTogether Designer

MobileTogether Designer is an entirely free-to-use product for Windows machines that establishes your mobile solutions precisely the way you want them. With an easy-to-comprehend approach, you employ drag-and-drop functionality to create elegant mobile solutions. MobileTogether Designer comes equipped with a complete mobile simulator so that you can instantly simulate your mobile solution in the designer. You can also run the mobile solution directly on your mobile device to view your project in real-time.



MobileTogether Designer Video Demos
At the Altova website you can view **video demos** that show how to use MobileTogether Designer to build a variety of MobileTogether solutions. These videos provide a fast introduction to the powerful features of MobileTogether Designer.

This documentation
This documentation is the user manual of MobileTogether Designer. It is organized into the following sections:

- Introduction
- Tutorials
- The User Interface
- The Project

- Data Sources
- Page Design
- Databases
- Altova Global Resources
- Simulation
- Menu Commands
- Appendices

Current version: 1.0 beta 2

Last updated: 09-15-2014

Chapter 2

Introduction

2 Introduction

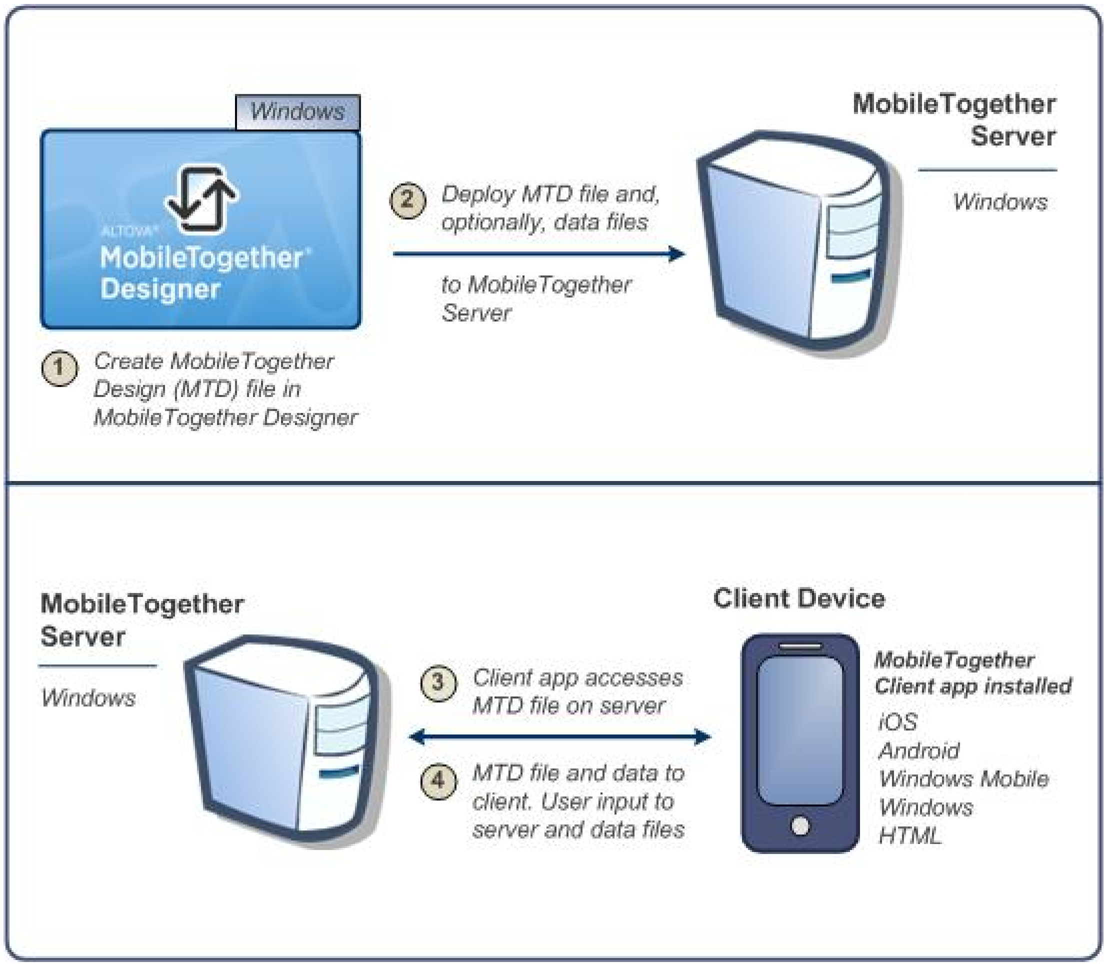
This section provides an overview of MobileTogether and MobileTogether Designer. It contains the following sections:

- MobileTogether Overview
- Terminology Q&A
- Design Steps

2.1 MobileTogether Overview

MobileTogether consists of the following modules:

- *MobileTogether Designer*, in which MobileTogether solutions for mobile clients (MTD files with the extension `.mtd`) are created. These MobileTogether solutions are then uploaded to MobileTogether Server.
- *MobileTogether Server*, which serves MobileTogether solutions to mobile clients.
- *MobileTogether Client* apps (for iOS, Android, Windows Phone 8, Windows RT, Windows Metro, web clients, web-based smartphones/tablets), on which the end user receives and interacts with MobileTogether solutions (**`.mtd`** files) delivered by MobileTogether Server.


Windows
ALTOVA®
MobileTogether®
Designer
1 Create MobileTogether Design (MTD) file in MobileTogether Designer
2 Deploy MTD file and, optionally, data files
to MobileTogether Server
MobileTogether Server
Windows
MobileTogether Server
Windows
3 Client app accesses MTD file on server
4 MTD file and data to client. User input to server and data files
Client Device
MobileTogether Client app installed
iOS
Android
Windows Mobile
Windows
HTML

System requirements

▾ **MobileTogether Designer**

Windows	XP (SP2 for x64, SP3 for x86), Vista, 7, 8, or newer

Windows Server	2008 R2 or newer

MobileTogether Server

Windows	XP (SP2 for x64, SP3 for x86), Vista, 7, 8, or newer
Windows Server	2008 R2 or newer

MobileTogether Client

iOS	iPhone, iPad
Android	Galaxy S, Nexus 7, Note 3, Sony Xperia U, and more
Windows Mobile	Windows Phone 8
Windows RT, Metro	For touch-enabled PCs and tablet computers
HTML	For any other mobile devices

2.2 Terminology Q&A

How does the MobileTogether system work?

- In MobileTogether Designer, you create MobileTogether Design files (MTD files), which have the file extension **`.mtd`**.
- These files are deployed to a MobileTogether Server, from where they are served to the mobile client device as MobileTogether solutions.
- The data files that are used to populate the design template/s in the MTD file may reside at their original locations or can be deployed to MobileTogether Server together with the MTD file.
- On the mobile client, the end user can view reports presented in a format defined in the MTD file. End users can also update data files from their mobile client devices (via MobileTogether solutions).

What's in an MTD file and in a MobileTogether project?

- An MTD file is a native MobileTogether Designer document.
- Each MTD file contains one MobileTogether project.
- A MobileTogether project consists of one or more pages. A page is what the end user sees on the mobile client device.
- If there is more than one page in the MTD file, then these pages are connected to one another in a simple sequence, with the first page leading to the next, and so on, till the last page is reached. (The order can be changed by setting page actions when page or control events are triggered.)
- Sub-pages can also be defined, and these can be accessed from within main pages with the `GoToSubpage` action.

What does a page consist of?

- A page consists of page controls (also called 'controls' for short), formatted for viewing on the mobile client device and set up for user-interaction.
- Each control has different properties. These properties define associated content, formatting, and action/s to perform when an event of a control (control event, for short) is triggered.
- For each page, a set of data sources can be defined in the Page Sources Pane of that page.
- The content associated with a control can come from one (or more) of these data sources. Such data is accessed using the XPath or XQuery language.
- Via its controls, therefore, a page presents data to the end user and can accept modifications to its data sources.

What are the different kinds of events and actions in a design?

- *Control events and their actions:* Each control on a page can have events that trigger actions you can specify. For example, the combo box control has the `OnFinishEditing` event, which occurs when an item from the dropdown list of the combo box is selected. This event can be defined by you to trigger a desired action, such as changing data as a result of the combo box selection.
- *Page events and their actions:* The page itself (as a single entity) can be associated with events that trigger actions. For example, `OnPageLoad` is a page event. This event can be defined by you to trigger a desired action, such as loading data into the page from a certain data file.

2.3 Design Steps

Given below is a broad outline, in steps, of how to create a MobileTogether Design file (MTD file).

1. Create a new MTD file
   Each MTD file represents a project consisting of one or more pages in a simple sequence. When a new MTD file is created, it has one default page that has no data source. You can add data sources to the default page, and you can add more pages to the project (*see the points below*). Create a new MTD file with the **File | New** command. The file is created in memory and must be saved with the **File | Save** command to store it on disk. Define Project Properties and specify other project-related settings.

2. Add data sources for the page (page sources)
   Each page is assigned data sources, from which it obtains the data that will be displayed in the page. The data sources of a page are added via the Page Sources Pane and are shown there as a tree of nodes. Data from these nodes is used by the controls in the page design, for display or for processing that typically leads to some kind of display (such as charts or images). Nodes in data source trees are addressed using XPath expressions. Client data input can also be saved back to the data sources if desired. See the section Page Sources for details.

3. Add controls to the page, and define their properties and event-actions
   Page controls are added to a page from the Controls Pane. Each control has a set of properties (defined in the Styles & Properties Pane) and data (from the data source trees) associated with it. A control can also have one or more predefined events. You can specify the action/s to be performed when a control event is triggered. For example, a button control has the event `OnButtonPressed`, and this event can have an `Open URL` action associated with it. For more information, see the sections Events and Actions. Additionally, pages also have events, and you can specify actions to perform when a page event is triggered. For example, when a page is loaded (a page event), an action can be specified that loads data from a specified XML file into a given page data source.

4. If required, add more pages to the project and design these
   Additional pages can be added to the initial page. A new page can be added as a top page or a sub page by clicking the **Add Page** icon in the Page Pane's toolbar. The sequence of top pages in the Pages Pane determines the sequence of the workflow.

5. Create a flow between top pages and sub pages
   You can further structure the solution's workflow by using sub pages. These are accessed from within top pages with the `GoToSubpage` action (of control or page events). Other page-related actions provide for more movement between pages.

6. Optionally, add additional design and user-related functionality to the project
   After all the pages have been added and the structure of the workflow has been finalized, you can revise your page designs and workflow. Any additional design components or actions can be inserted in the project now.

7. Run a simulation of the MobileTogether solution
   You can test the design by running a workflow simulation within MobileTogether Designer. The simulation shows (in MobileTogether Designer itself) how the workflow will be executed on the client device. Select **Project | Simulate Workflow** or press **F5** to start the simulation.

8. Deploy the MTD file to MobileTogether Server
   After making final changes and re-testing the file, save it, and then deploy it to MobileTogether Server. The MobileTogether solution is now ready to be accessed by mobile client devices.

For detailed information about the following topics, see their respective sections:

- MobileTogether Designer Projects
- Page Sources
- Page Design
- Databases

Chapter 3

Tutorials

3 Tutorials

This section contains tutorials that will guide you through the basic steps needed to create a MobileTogether design, as well as help you to understand more advanced MobileTogether features. The QuickStart Tutorial takes you through the basics, all the way up to deploying the MTD file and associated files on MobileTogether Server. The Database-And-Charts tutorial uses a finished design file to explain how to work with databases and how to create charts.

Finished design files, as well as the data sources and image files used in the tutorials, are available in the *(My) Documents* folder: `Altova\MobileTogetherDesigner1\MobileTogetherDesignerExamples\Tutorials`.

We strongly recommend that you create the folder **`C:\MobileTogether`** and then copy save all the files that are in the Tutorials folder to the **`C:\MobileTogether`** folder. This is because the supplied design files (**`.mtd`** files) use absolute URLs to target resources in the **`C:\MobileTogether`** folder. It is therefore best to save all the tutorial files together in the **`C:\MobileTogether`** folder.

After you have completed the tutorials, you could open the more advanced example design files (**`.mtd`** files) in the the `MobileTogetherDesignerExamples` folder. These examples show how various features of MobileTogether Designer can be used, and so can be used as reference points for your own designs.

File paths in Windows XP, Windows Vista, Windows 7, and Windows 8
File paths given in this documentation will not be the same for all operating systems. You should note the following correspondences:

- *(My) Documents folder:* Located by default at the following locations. Example files are located in a sub-folder of this folder.

Windows XP	`C:\Documents and Settings\<username>\My Documents`
Windows Vista, Windows 7/8	`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 7/8	`C:\Program Files\Altova\`
32 bit Version on 64-bit OS	`C:\Program Files (x86)\Altova\`

File URLs in example files
File URLs in example files might be absolute URLs and, in such cases, therefore will probably not correspond to the directory structures of your work environment. If you use the supplied example files, make sure that file URLs in
them—and any XPath expressions that build such URLs—point to the correct file locations on your machine or network.

3.1 QuickStart (Part 1)

This QuickStart tutorial guides you through the basic steps of creating a MobileTogether design for a MobileTogether solution. The solution displays the splash screen of an Altova product which the end user selects in a combo box. The tutorial shows you how to set up a page, its data source, how to add controls, and make the display of the splash screen conditional upon end-user input. It also explains validation, how to deploy the design file to MobileTogether Server, and how to run simulations. After you have finished with this QuickStart tutorial, you should have a good understanding of the broad working principles of MobileTogether designs. You will then be ready to tackle more complex designs.

This tutorial is organized into the following sections, each of which deals with one key aspect of creating a MobileTogether design.

- Create a New Design
- Set Up a Page
- Add a Page Data Source
- Format the Design
- Add a Control: Combo Box
- Add a Control: Image
- Define Control Actions
- Validate the Project
- Run a Simulation
- Deploy to Server

The tutorial files
The files for this tutorial are located in your *(My) Documents* MobileTogether folder: **`MobileTogetherDesignerExamples\Tutorials\`**. You can save the splash screen image files that are used in the tutorial to any other location and use them from there if you like.

- The file you will end with should be similar to: **`QuickStart01.mtd`**
- The image files used in the tutorial are the **`*.bmp`** files in the folder

Create a New Design

In this part, you will:

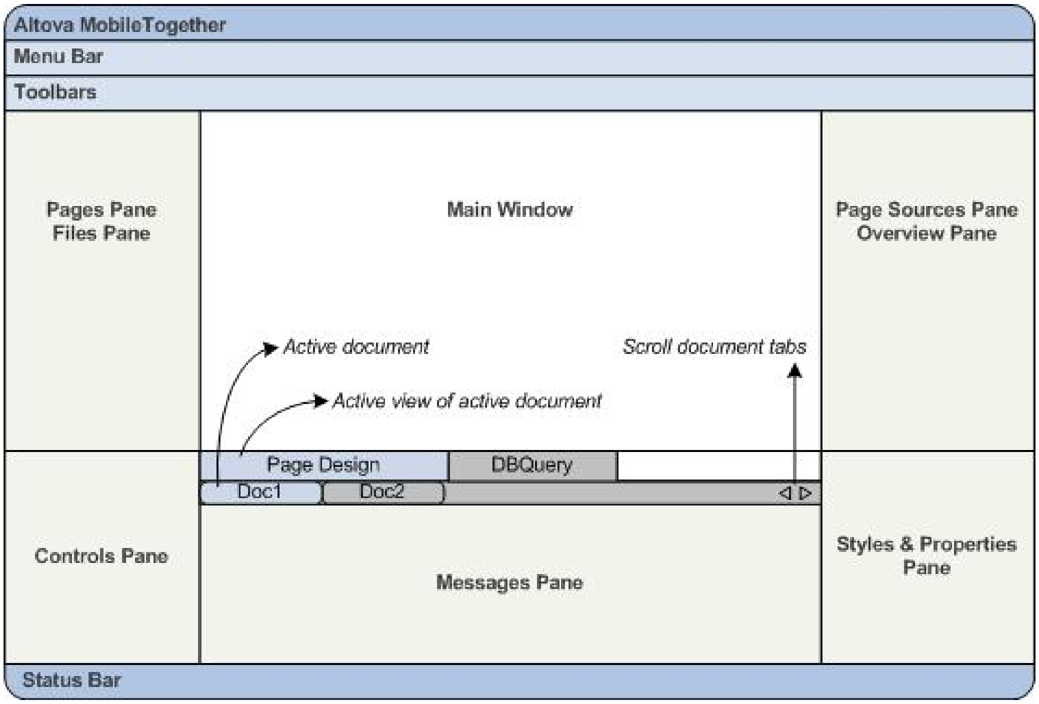
- Create a new MobileTogether design
- Save the design to file
- Set up the design's preview properties: the preview device and magnification level
- Familiarize yourself with the GUI of MobileTogether Designer

---

A MobileTogether design is created in an MTD (MobileTogether Design) file, which has the **`.mtd`** extension and is native to MobileTogether Designer.

To create a new MobileTogether design, do the following:

1. Click **File | New**. This opens a new design file in the main window of the GUI (*figure below*). It will have a default page called `New Page1`.



2. The new design will be in a tab called `New Design1`. Click **File | Save**, and save the file with any name to any location you like. Make sure, however, that you save your design file in the same folder as the splash screen images. (Since your design, when complete, will show the splash screens of Altova products, you could, for example, call it `AltovaSplashScreens.mtd`.) The new file name, which will have a `.mtd` extension, replaces `New Design1` in the file's tab. The file name also appears in the application's title bar.
3. In the Main toolbar of MobileTogether Designer, you can select your **preview device** and set the **magnification** of the design. Select the options you want. Since mobile devices have different background colors and dimensions, selecting an appropriate preview device

will help you visualize your design better. You can change the preview device at any time if you wish to visualize the design on another device. Note that these settings will also be applied to simulations.

4. Familiarize yourself with the Main Window and its tabs, and with the different panes that are located around the main window. The user interface is described in the section, The User Interface.
5. Switch between the Page Design and DB Query tabs of the Main Window, and see how the contents of the panes change.

Set Up a Page

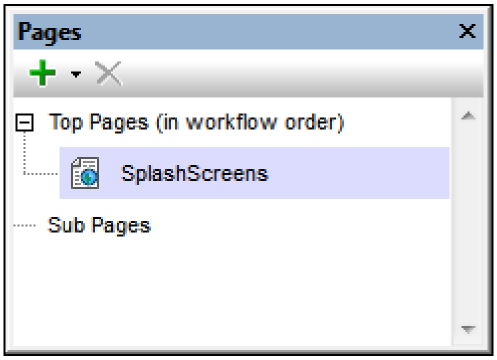
A MobileTogether design can consist of one or more pages. A page is what the end-user sees on the mobile client device. If there are multiple pages in a design, the page sequence is defined by their sequential order in the Pages Pane. Within a page, controls can be set to go to sub pages (that typically contain reusable modules). In this part of the tutorial, in order to keep things simple, we will create a project that has only one page. For more information about pages, see the description of the Pages Pane.

We are building a design that has one page. In this part, you will:

- Give the default page a name
- Add a label control so as to display a title for the page
- Format this label

Page name and the label control
Give the default page of the new design a name as follows:

1. In the Pages Pane (*screenshot below*), double-click `New Page1` and rename it to **`SplashScreens`**.



2. From the Controls Pane, drag and drop the Label control into the design. The control will be placed at the top of the page and will contain a red exclamation mark. On hovering over the exclamation mark, a message is displayed, warning that the label has no content.
3. The content of the label can be static or dynamic. If dynamic, the content can be taken from an XPath expression or from a node in one of the page's data sources. (No data source has been defined for the `SplashScreens` page as yet. This will be done in the next part.) Enter static content by double-clicking the label and entering the text, **`Altova Splash Screens`**, and then pressing **Enter**.
4. In the Table toolbar, click **Center** to center the text. Then click **Text Color** and/or **Background Color** (also in the Table toolbar) if you wish to set these properties.
5. With the Label control selected, the label's properties are displayed in the Styles & Properties Pane. If you like, modify any of the label's formatting properties available in this pane, such as the margin-bottom property to create empty space below the label.

Add a Page Data Source

You have so far created a page and given it a title. In this part, you will:

- Add an empty XML data source for this page
- Since the data source has neither structure or content, create structure and content directly in the Page Sources Pane
- Set a default XPath context node

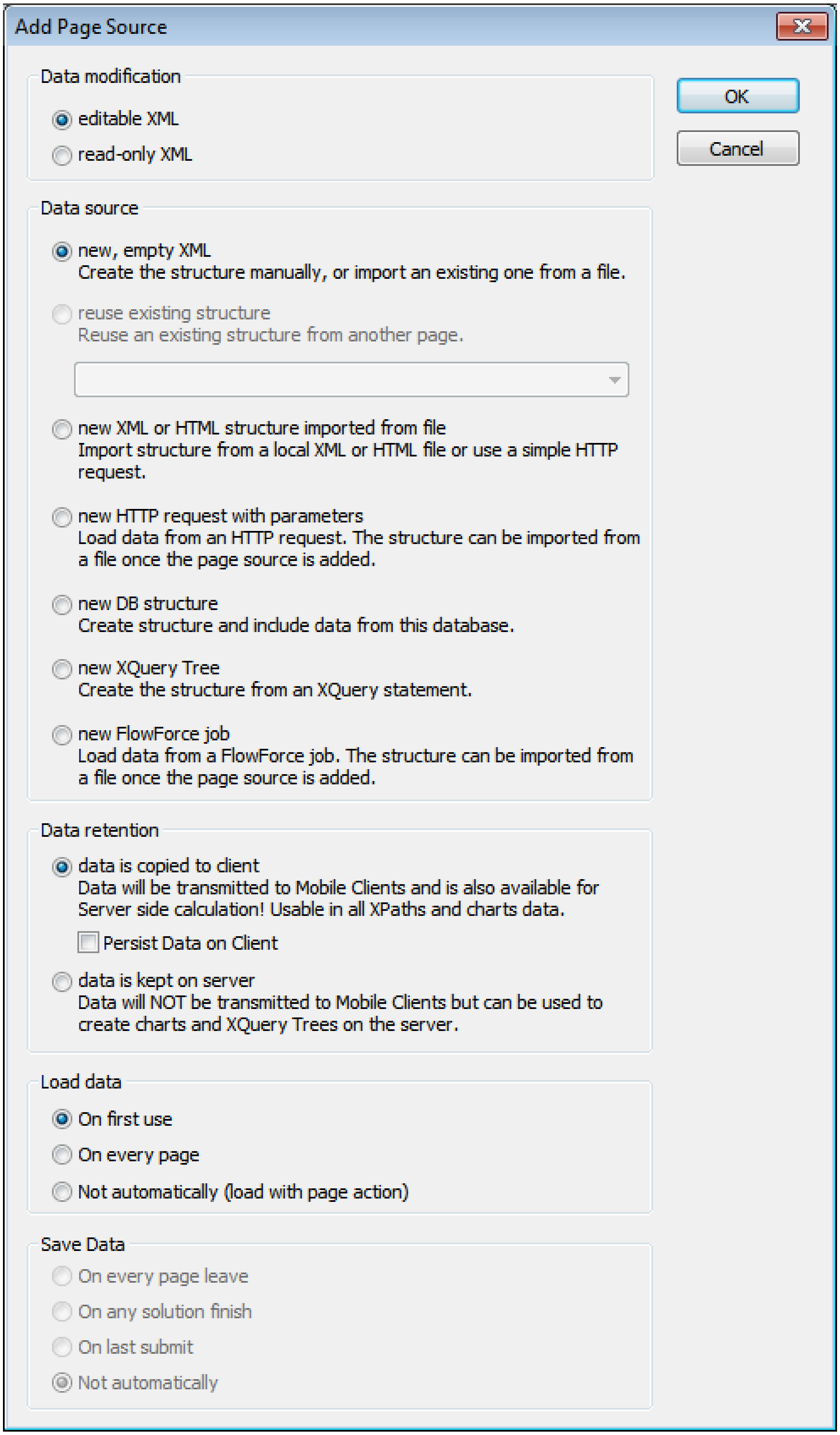
⊟ *Buttons in this section*

+	**Add Source**

A page can have one or more data sources from where data can be obtained. Data sources are defined in the Page Sources Pane and can be specified to be read-only or editable. They can be external sources, or can be created directly in the Page Sources Pane and contain static data. The types of data sources that can be used are described in the section Data Sources. In this tutorial, we will add one type of data source, an editable empty XML source, for which we will then create a structure and content directly in the Page Sources Pane.

To add the data source for the page, do the following:

1. Click the **Add Source** button in the toolbar of the Page Sources Pane to display the Add Page Source dialog (*screenshot below*).



Add Page Source

Data modification

● editable XML

○ read-only XML

Data source

● new, empty XML
Create the structure manually, or import an existing one from a file.

○ reuse existing structure
Reuse an existing structure from another page.

○ new XML or HTML structure imported from file
Import structure from a local XML or HTML file or use a simple HTTP request.

○ new HTTP request with parameters
Load data from an HTTP request. The structure can be imported from a file once the page source is added.

○ new DB structure
Create structure and include data from this database.

○ new XQuery Tree
Create the structure from an XQuery statement.

○ new FlowForce job
Load data from a FlowForce job. The structure can be imported from a file once the page source is added.

Data retention

● data is copied to client
Data will be transmitted to Mobile Clients and is also available for Server side calculation! Usable in all XPaths and charts data.

□ Persist Data on Client

○ data is kept on server
Data will NOT be transmitted to Mobile Clients but can be used to create charts and XQuery Trees on the server.

Load data

● On first use

○ On every page

○ Not automatically (load with page action)

Save Data

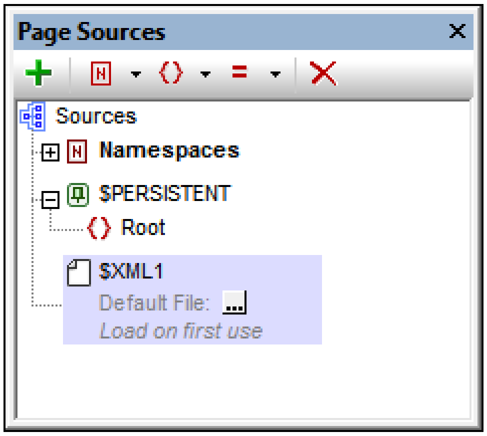
○ On every page leave

○ On any solution finish

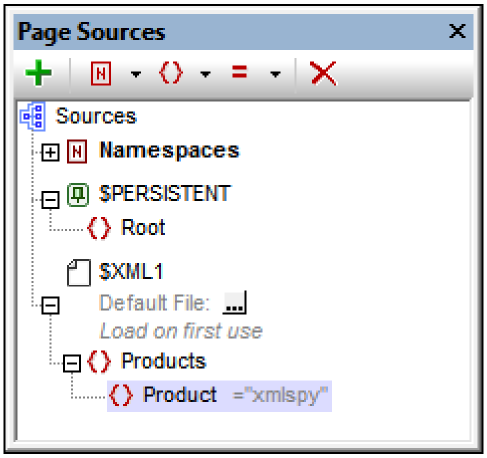
○ On last submit

● Not automatically

2. Select *Editable XML* and *New, empty XML*, and click **OK**. (When a data source is created as editable, data in it can be modified.) A root node called `$XML1` is added to the Page Sources tree in the Page Sources Pane (*screenshot below*).



3. Notice that the XML tree with the root node `$XML1` is empty. Right-click `$XML1` and select **Add Child | Element,** then enter the element name, `Products`.
4. Add a child element to `Products` (by right-clicking it and selecting **Add Child | Element**), and name the child element `Product`. If you wish to rename the elements, double-click the element names and edit them.
5. Right-click the `Product` element, select **Ensure exists before page load (fixed value)**, then enter **`xmlspy`** (*see screenshot below*). This will be the `Product` element's default content when the page loads.



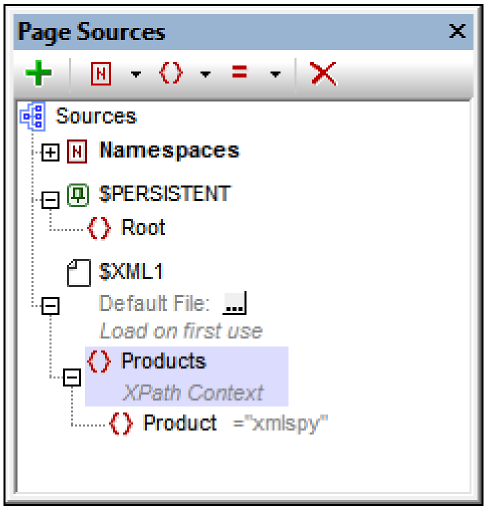
Notice that there is no default XML file assigned to `$XML1`. A default file provides the data that goes into the nodes of the data source's tree. The data provided by a default file can be overridden by data that is manually entered in a tree node. In our case, there is a single data node: **`$XML1/Products/Product`**, and it has the string **`xmlspy`** as its default content. The XML data structure would be:

```
<Products>
   <Product>xmlspy</Product>
</Products>
```

Set a default XPath context node

Each page has a default XPath context node, which is used as the context node for relative XPath expressions defined on that page. The default XPath context node is indicated in the Page Sources Pane with the label *XPath Context* (*see screenshot below*). Note that each page has its own default XPath context node.

We will change the default XPath context node to the **`Products`** node. Do this by right-clicking **`Products`** and selecting **Set As Default XPath Context**. The element **`Products`** will be set as the default XPath context node—this is indicated by the node being labelled *XPath Context* (*see screenshot below*).



Note: If you wish, in your XPath expression, to reference a node in a data source tree other than that of the default XPath context node, use an absolute path to that node, starting with the root node of that tree. For example: **`$XML2/SomeRootElement/SomeOtherElement`**.

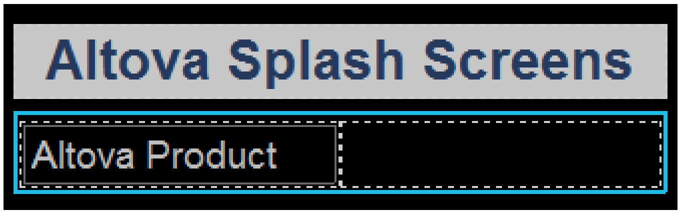
Format the Design

The design page now has a name (SplashScreens), a display title (Altova Splash Screens), and a data source ($XML1). In this part, you will:

- Add a Table control for presentation purposes
- Add formatting: the horizontal line control, spacing, and color
- Copy-paste controls to other parts of the design

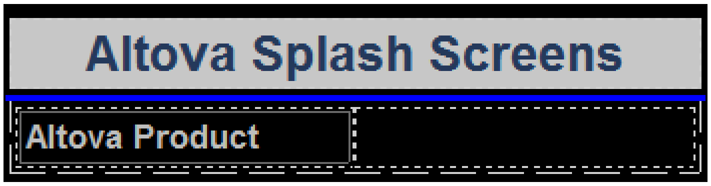
Add a table
We will add a table for presentation purposes.

1. From the Controls Pane, drag the Table control into the design and drop it below the label.
2. In the New Table dialog that appears, specify that the table should have two static columns and one static row, and then click **OK**. The table is created with a single row and two columns.
3. From the Controls Pane, drag and drop the Label control into the left-hand cell, and give it a static text value of **`Altova Product`** (*see screenshot below, which shows the table highlighted*).



Add a horizontal line, spacing, and color
Add a horizontal line to separate the title from the table as follows:

1. From the Controls Pane, drag and drop the Horizontal Line control between the title and the table (*see screenshot below; the horizontal line is dark blue*).



2. With the line selected in the design, in the Styles & Properties Pane, set the color and width of the line.
3. Copy the line, by selecting it and pressing **Ctrl+C**, and paste it below the table (using **Ctrl+V**). The line will be copied with all the properties that you defined for it.

4. You can change the background color of the label, the table, and individual table cells. Try this by placing the cursor in the respective components, and selecting different background colors in the Styles & Properties Pane.
5. Select the label control and set `Margin Top` and `Margin Bottom` properties in the Styles & Properties Pane.
6. Try copying different components to various parts of the design.

Add a Control: Combo Box

We will add a combo box to the right-hand cell of the table you created in the previous section. We will then define the properties of the combo box. A combo box needs two important property definitions:

- *The combo box values:* These values will populate the dropdown list of the combo box and provide the options the user can choose (*visible entries*). Each visible entry will have a corresponding XML value that will go into the associated XML node.
- *The associated XML node:* This data source node is synced with the combo box selection. It receives its value from the combo box selection, but its initial value determines the initial combo box selection.

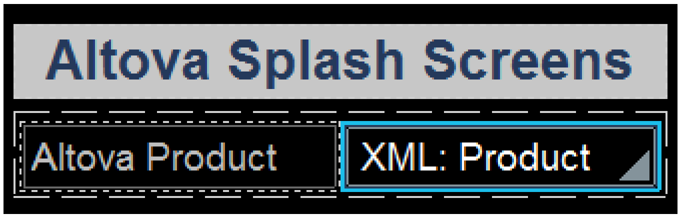
⊟ *Buttons in this section*

[...] **Additional Dialog**

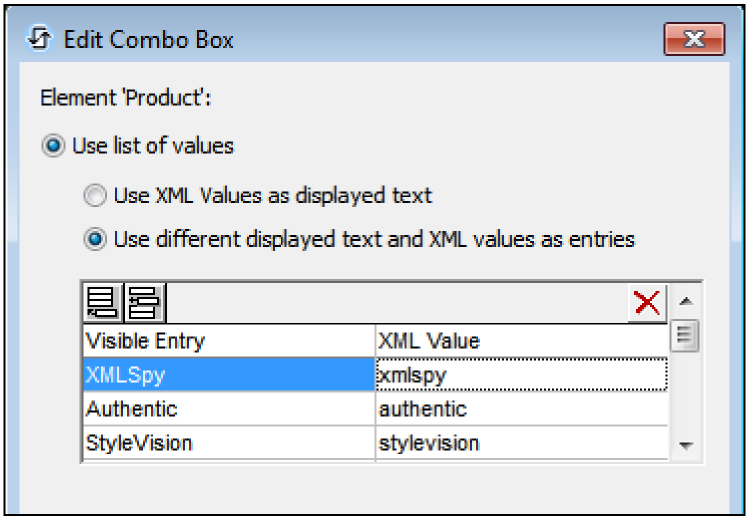
Add a combo box

Add the combo box and define its properties as follows:

1. From the Controls Pane, drag and drop the Combo Box control into the right-hand cell of the table (*see screenshot below*).



2. Drag the **`Product`** node from the Page Sources Pane, and drop it on to the combo box. This associates the **`Product`** node with the combo box, and creates what we call a **page source link**. When the end user selects an option from the combo box, the XML value of the selected option will be passed to the **`Product`** node and will become the **`Product`** node's content.
3. To define the dropdown list items of the combo box, select the combo box, and, in the Styles & Properties Pane, click the **Additional Dialog** button of the **`Combo Box Entry Values`** property. The Edit Combo Box dialog (*screenshot below*) appears.



Edit Combo Box

Element 'Product':

● Use list of values

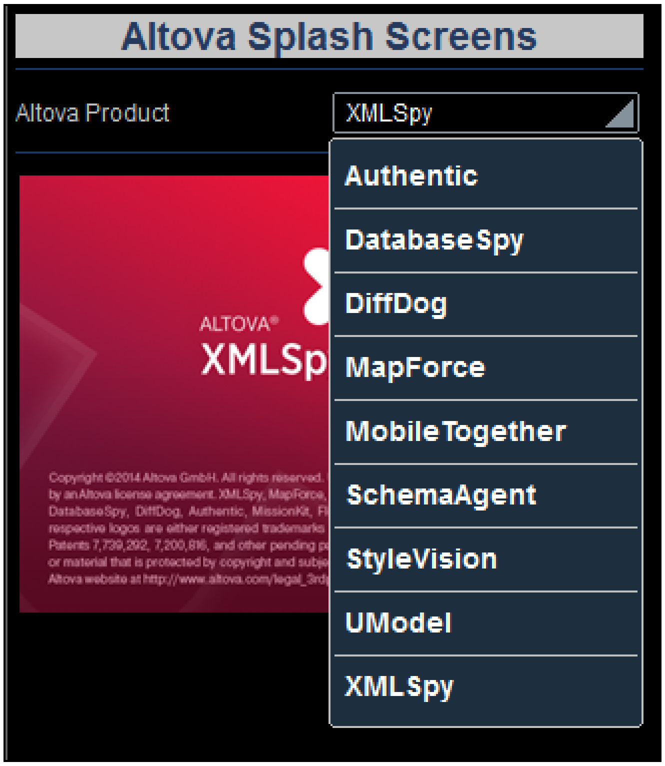
○ Use XML Values as displayed text

● Use different displayed text and XML values as entries

Visible Entry	XML Value
XMLSpy	xmlspy
Authentic	authentic
StyleVision	stylevision

4. Select *Use list of values*, and then *Use different text and values*, and enter values for the visible entry (in the dropdown list) and for the corresponding XML value (which is what will be written into the `Product` node). When we add the Image control, you will find out why we want the XML values lowercased. (Hint: The image names are lowercased, for example, `xmlspy.bmp`.) You can add up to nine Altova products to this list in any order you like: `XMLSpy, Authentic, StyleVision, MapForce, DiffDog, DatabaseSpy, MobileTogether, SchemaAgent, UModel`.
5. Select the *Sort values* check box at the bottom of the dialog to sort the list when it is displayed, and click **OK** to finish.

When the completed solution is executed or a simulation of the completed project is run, the dropdown list of the combo box (*screenshot below*) will display the values listed in the *Visible Entry* column definitions of the combo box (*see screenshot above*).



Altova Splash Screens

Altova Product: XMLSpy

The objective of our project is this: When the end user selects an entry from the dropdown list, the XML value corresponding to that entry (*see screenshot of the combo box definition above*) will be passed to the `Product` node of the data source. Note that the default content of `Product` is `xmlspy` (defined when the node was created). We can now use the value in the `Product` node to build the URL of the splash screen image to display, which we will now do in the next part of the tutorial, Add a Control: Image.

Add a Control: Image

We will now add an image to the design. This image will be the splash screen of the Altova product that the end user selects in the combo box (*see previous section*). An image has one important property: the `Image URL` property, which selects the image to display. The URL we will build is a relative URL that looks for the image file in the same folder as the design file. You could, of course, use an absolute file URL or an image at an Internet location.

In this part, you will:

- Add an image control
- Define the `Image URL` property using an XPath expression that allows the URL to change dynamically with the combo box selection. The images referred to in this tutorial are located in the folder, `MobileTogetherDesignerExamples\Tutorials`, located in your *(My) Documents folder*. If you like you can save the images to another folder.

⊟ *Buttons in this section*

X PATH **XPath**

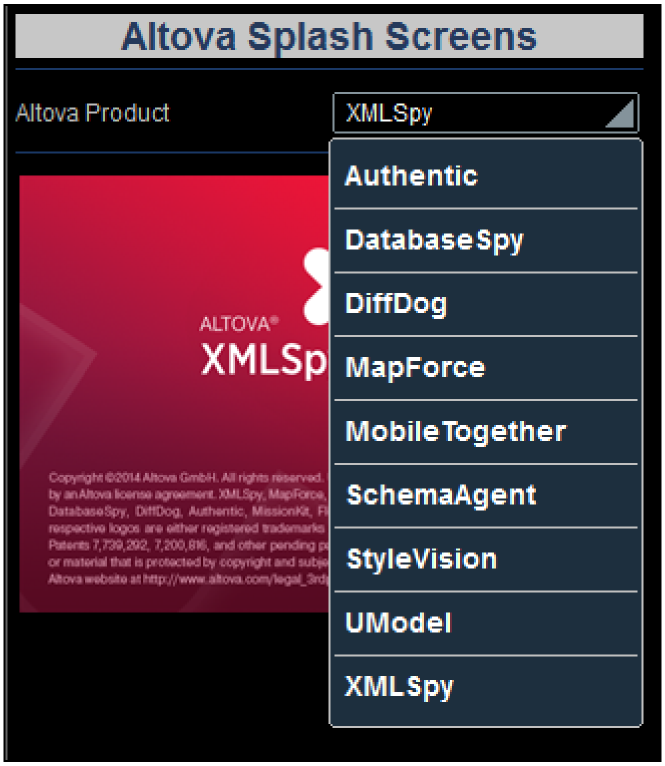
Add an image

Add the image and define its properties as follows:

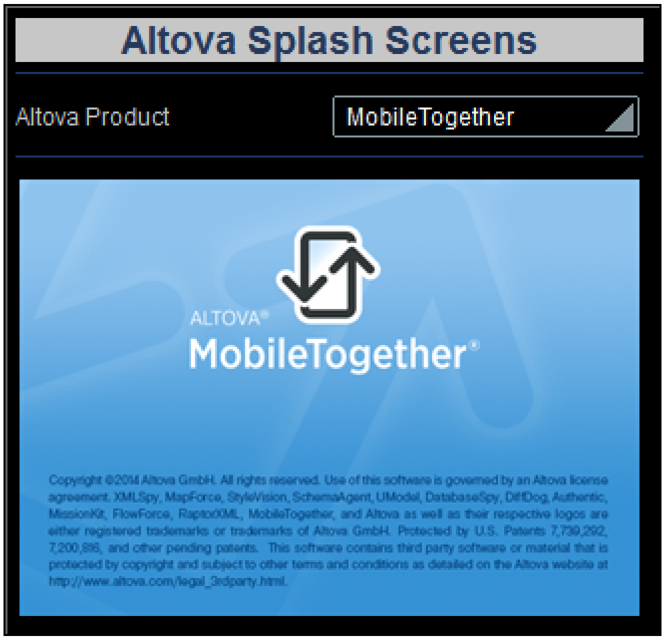
1. From the Controls Pane, drag and drop the Image control below the table.
2. In the Styles & Properties Pane, name the image by setting its `Name` property to `Image: SplaschScreen`.
3. Select the **`Image URL`** property, and click the **XPath** button in the toolbar of the Styles & Properties Pane. This displays the Edit XPath/XQuery Expression dialog.
4. Enter the XPath expression: `concat(Product, '.bmp')`, and click **OK**.

This XPath expression produces a **relative URL** that locates a **`.bmp`** file in the same folder as the design file. You should specify the location that is correct for you, that is, the location where the image files have been saved. If you need to, you can use an absolute URL (*see example below*). The filename (for example, `xmlspy`) is obtained from the `Product` node, which in turn gets its content from the selection that the end user makes in the combo box. The default value of the **`Product`** node was set to **`xmlspy`**, so the XPath expression and starting image URL will be as below:

This XPath expression:	**`concat(Product, '.bmp')`**
Produces this absolute URL:	**`xmlspy.bmp`**



When the end user selects a product from the dropdown list of the combo box (*see screenshot above*), say **`MobileTogether`**, the splash screen of MobileTogether will be displayed (*screenshot below*). This is because the XML value corresponding to the MobileTogether selection is **`mobiletogether`**. This XML value is passed to the **`Product`** node, and it is used in the XPath expression to dynamically build the relative image URL: **`mobiletogether.bmp`**.



To ensure that the image is reloaded whenever a combo box selection is made, a *Reload* action must be set on the combo box. How to do this is described in the next section, Define Control Actions.

Define Control Actions

Control actions define what action a control takes in response to an event such as a button press or combo box selection. Actions that can be taken include: updating data nodes, reloading or saving page sources, or executing database commands.

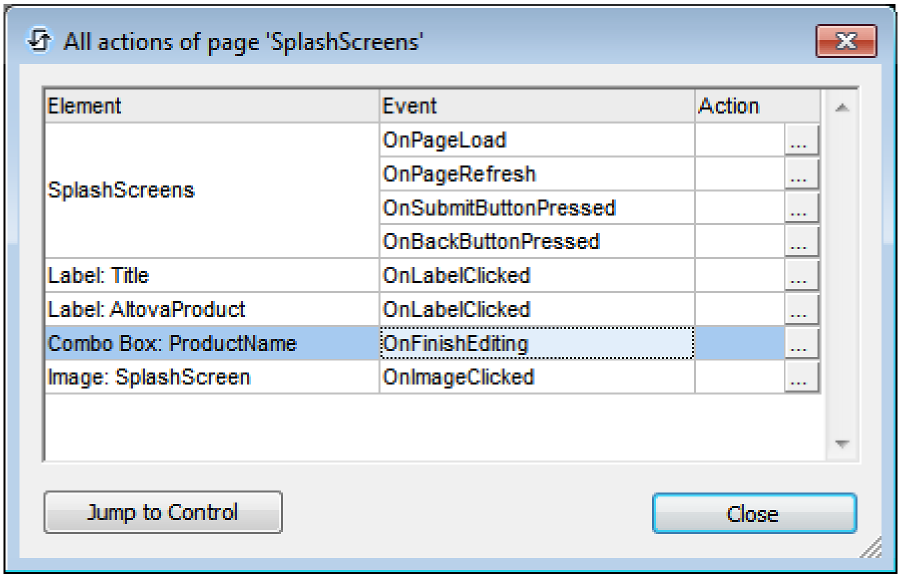
In this part, you will:

- Look at all the page actions and control actions defined for the page
- Add a combo box action that updates the image whenever the combo box is edited

---

Overview of page actions

To go to an overview of all actions of the `SplashScreens` page, click **Page | Actions Overview**. The Page Actions Overview dialog (*screenshot below*) appears. It displays all the events and their actions as currently defined for the page. The display includes page events and control events, and their respective actions. The `SplashScreens` element in the screenshot below refers to the page; all the other elements are the different controls of the page design. You can see that there is currently no action defined for any event.

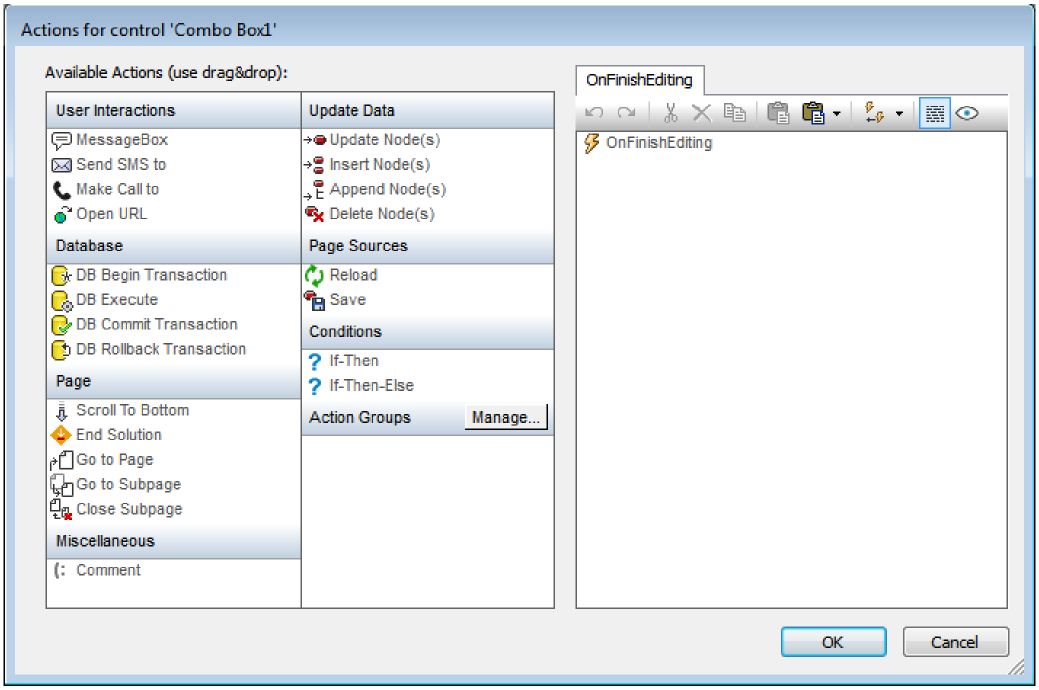


Element	Event	Action
SplashScreens	OnPageLoad	...
	OnPageRefresh	...
	OnSubmitButtonPressed	...
	OnBackButtonPressed	...
Label: Title	OnLabelClicked	...
Label: AltovaProduct	OnLabelClicked	...
Combo Box: ProductName	OnFinishEditing	...
Image: SplashScreen	OnImageClicked	...

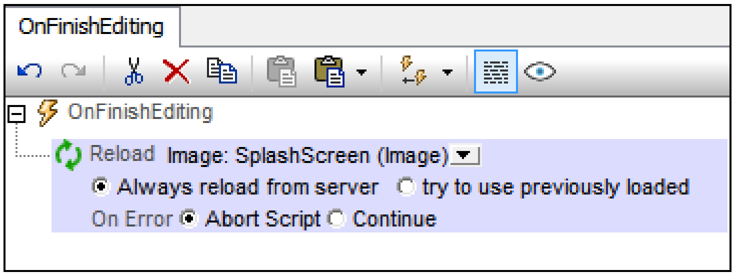
Define the Reload action on the combo box control

Defining a control action consists of two parts: (i) selecting the control event that triggers the action; and (ii) specifying the action to perform when the event occurs. In our example, we want to do the following: When the end user makes a combo box selection, the image must be reloaded. This is because we want the image URL to be re-evaluated with the new value of the `Product` node (selected in the combo box). So, when combo box editing is finished, we want the *Reload* action to reload the image. Define this combo box event-and-action as follows.

1. Right-click the **`XML:Product`** combo box and select **Control Actions on OnFinishEditing**. This opens the Control Actions dialog for the combo box (*screenshot below*). There is only one combo box event: **`OnFinishEditing`** (*see the right-hand-side pane*). If additional events were available they would be shown as additional tabs of the right-hand pane. All the available actions for events are listed, in groups, in the left-hand-side pane.

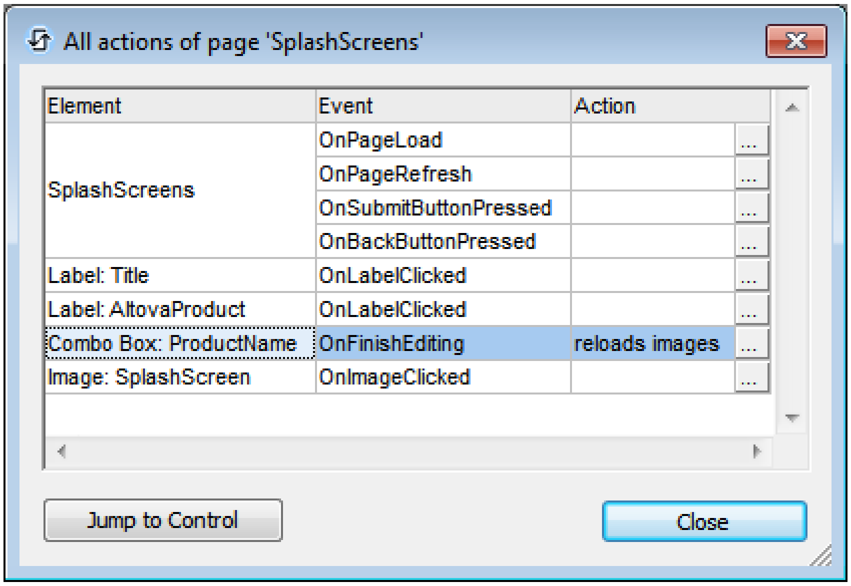


2. Drag the **`Reload`** action from the Page Sources group and drop it in the **`OnFinishEditing`** tab (*screenshot below*). This specifies that a reload action must be carried out when the combo box is edited.
3. Click the drop-down arrow (next to `$WORKFLOWDATA1`), select `Image: SplashScreen`, and click **OK**. This specifies that the image control will be updated when the combo box value is changed.



Reload Image: SplashScreen (Image)
● Always reload from server ○ try to use previously loaded
On Error ● Abort Script ○ Continue

If you now check the Page Actions Overview dialog, you will see that a *Reloads Image* action is defined for the combo box's `OnFinishEditing` event.



Element	Event	Action
SplashScreens	OnPageLoad	
	OnPageRefresh	
	OnSubmitButtonPressed	
	OnBackButtonPressed	
Label: Title	OnLabelClicked	
Label: AltovaProduct	OnLabelClicked	
Combo Box: ProductName	OnFinishEditing	reloads images
Image: SplashScreen	OnImageClicked	

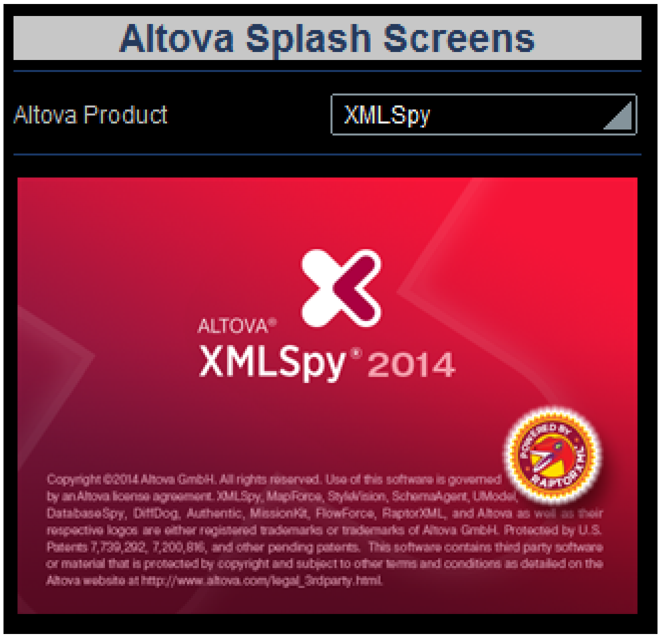
Validate the Project

Now that the design of the page is complete, you should validate the project (**Project | Validate**) to check for errors. On validating, you will get a message saying that there are no errors or warnings.

- Warnings indicate design issues that might need your attention, but these issues are not fatal and will not prevent the solution from running.
- An error message, on the other hand—as opposed to a warning—would be fatal and should be remedied.

Run a Simulation

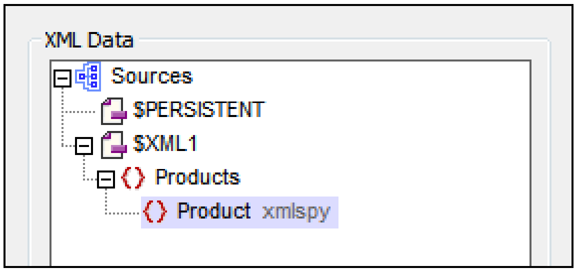
You can run a simulation of the project workflow directly within MobileTogether Designer. The simulation device will be the device currently selected in the preview device combo box of the Main toolbar. You can change the preview device to see the simulation on different devices. To run the simulation, select **Project | Simulate Workflow** or **F5**. This opens the simulator and starts the simulation.



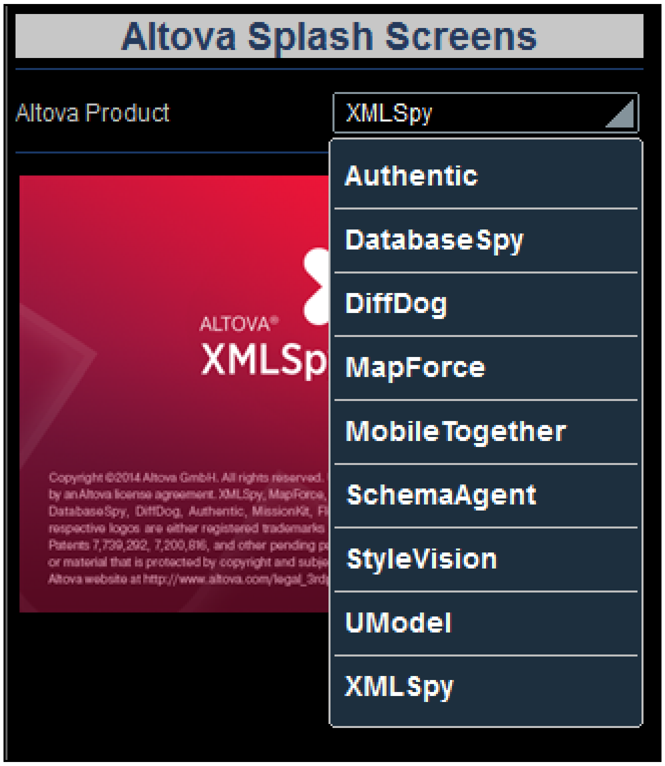
Altova Product: XMLSpy

Copyright ©2014 Altova GmbH. All rights reserved. Use of this software is governed by an Altova license agreement. XMLSpy, MapForce, StyleVision, SchemaAgent, UModel, DatabaseSpy, DiffDog, Authentic, MissionKit, FlowForce, RaptorXML, and Altova as well as their respective logos are either registered trademarks or trademarks of Altova GmbH. Protected by U.S. Patents 7,739,292, 7,200,816, and other pending patents. This software contains third 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.

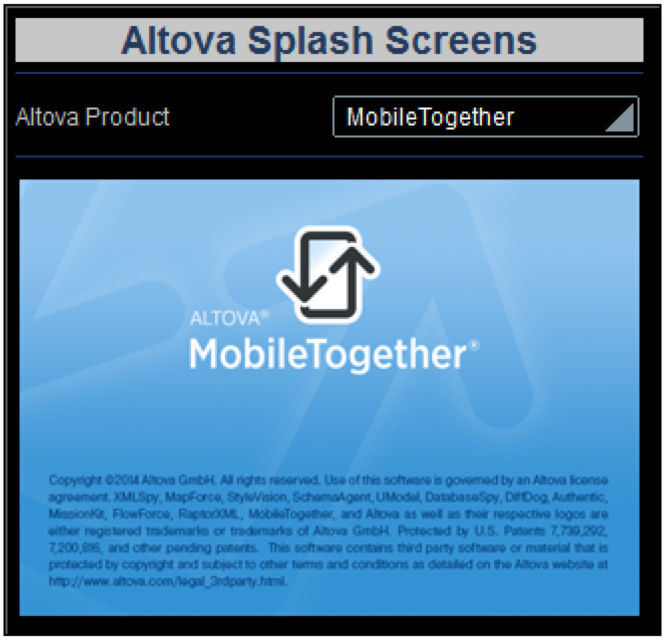
The XMLSpy splash screen is displayed because the default value of **`Product`** was set to **`xmlspy`**. You can see this in the *XML Data* pane on the right if you expand the `$XML1` node (*screenshot below*).



Now, in the simulation, click the combo box to display its dropdown list (*screenshot below*).

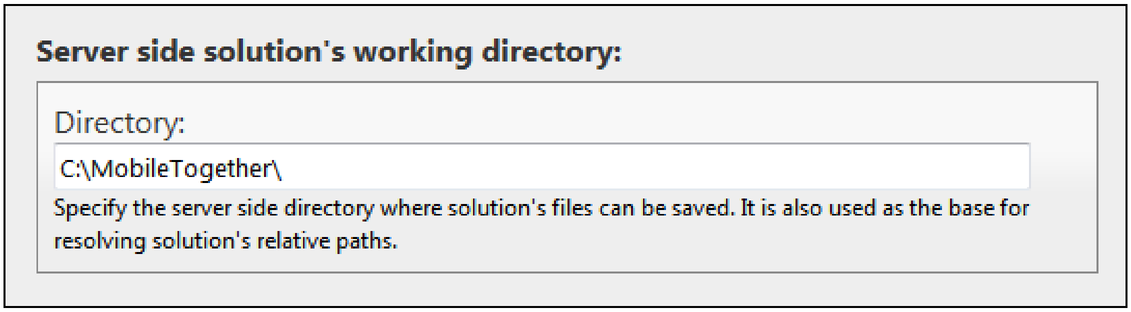


Select a product from the list and notice how the image and value of the `Product` node changes. (Note that, for iOS devices, you will need to press **Set** to confirm selections.)



After you have finished, click **Close** or press **Esc** to close the simulator.

Running a server simulation
A server simulation uses MobileTogether Server to run the simulation (**Project | Use Server for Workflow Simulation**). In the Web UI of MobileTogether Server, you must set the solution's working directory (**Settings | Server Side Solution's Working Directory**, *see screenshot below*). All relative paths in the design will be resolved relative to the directory specified in this setting. In order for server simulation to work correctly, enter the path of the directory where your referenced files are saved.



Server side solution's working directory:

Directory: C:\MobileTogether\

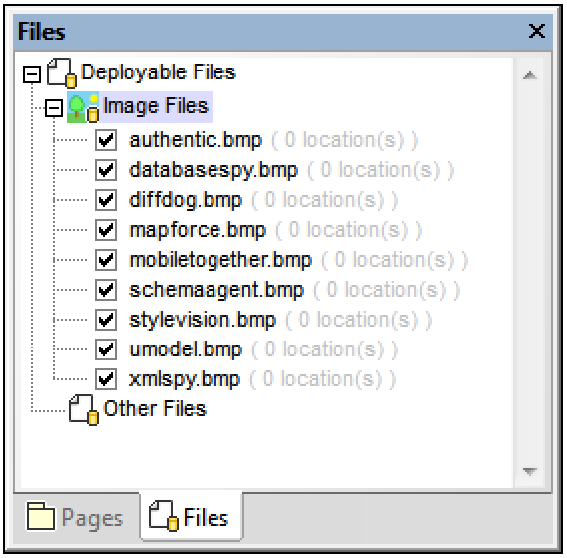
Specify the server side directory where solution's files can be saved. It is also used as the base for resolving solution's relative paths.

Deploy to Server

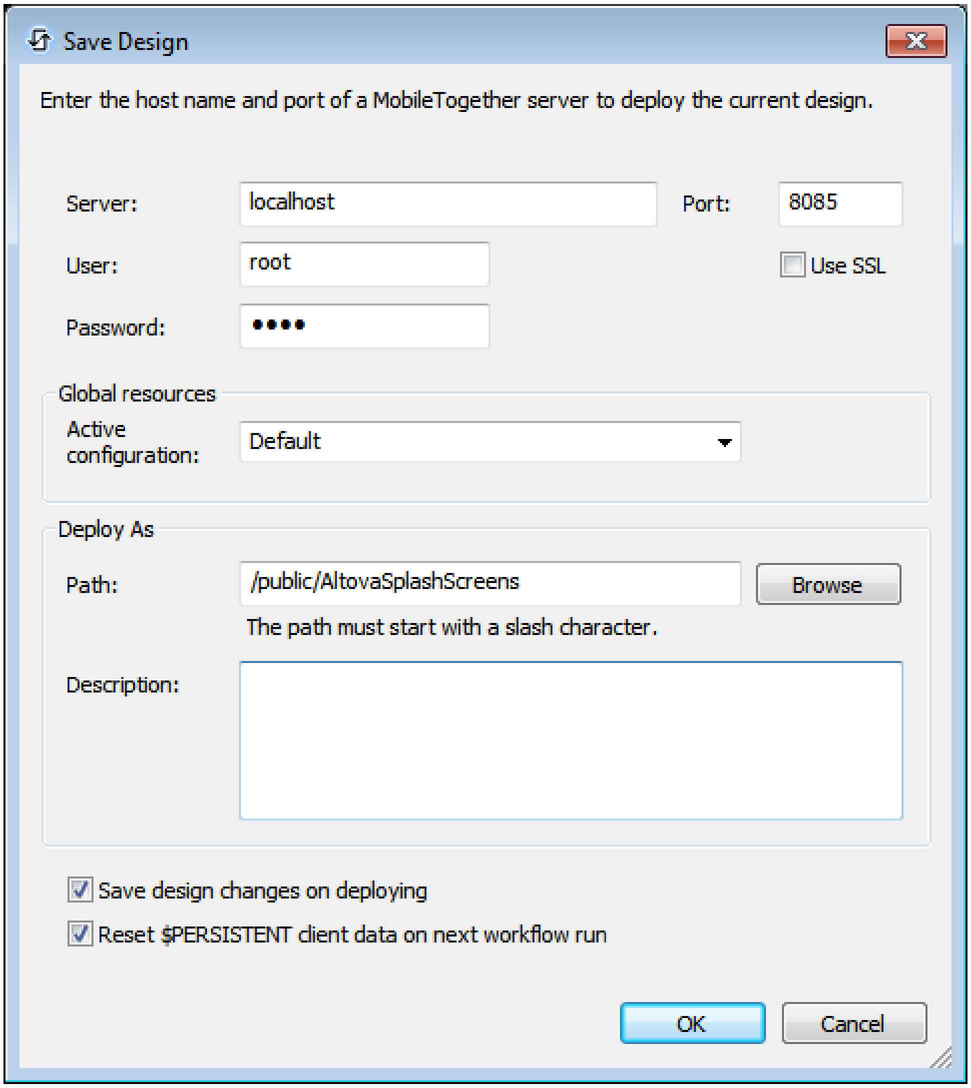
After you have successfully validated the project and tested the simulations, you are ready to deploy the project to MobileTogether Server. When the project is deployed to the server, it becomes a solution that a MobileTogether Client app can run. In order to deploy the project to MobileTogether Server, you will need to access the server as a user having the privilege, *Save Workflow from Designer*. The access rights of users of MobileTogether Server are defined in the Web UI of MobileTogether Server. See the MobileTogether Server user manual for information about setting user privileges.

Deploying the project to MobileTogether Server
Deploy the project and its associated (image) files to MobileTogether Server as follows:

1. In the Files Pane (*screenshot below*), right-click Image Files, and, in the context menu that appears, select **Add File**.


Files
Deployable Files
Image Files
☑ authentic.bmp (0 location(s))
☑ databasespy.bmp (0 location(s))
☑ diffdog.bmp (0 location(s))
☑ mapforce.bmp (0 location(s))
☑ mobiletogether.bmp (0 location(s))
☑ schemaagent.bmp (0 location(s))
☑ stylevision.bmp (0 location(s))
☑ umodel.bmp (0 location(s))
☑ xmlspy.bmp (0 location(s))
Other Files
Pages
Files

2. Browse for the image file and add it to the Image Files list.
3. Repeat Steps 1 and 2 for all the image files that are required.
4. Select **File | Deploy to MobileTogether Server**. This displays the Save Design dialog (*screenshot below*).



Save Design

Enter the host name and port of a MobileTogether server to deploy the current design.

Server: localhost

Port: 8085

User: root

□ Use SSL

Password: ••••

Global resources

Active configuration: Default

Deploy As

Path: /public/AltovaSplashScreens

The path must start with a slash character.

Description:

☑ Save design changes on deploying

☑ Reset $PERSISTENT client data on next workflow run

5. Enter the MobileTogether Server address and administrator port (default `HTTP` port is `8085`, `HTTPS` is `8086`).
6. Enter the user name and password with which you want to access MobileTogether Server. Note that this user must have the *Save Workflow from Designer* privilege in order for deployment to be successful. See the MobileTogether Server user manual for information about user privileges.
7. In the *Deploy As* | *Path* field, enter the name under which you wish to deploy this solution on the server. For example, in the screenshot above, the workflow will be saved in the **`/public`** container and will have the name **`AltovaSplashScreens`**. On MobileTogether Server, you will see only the one entry: **`AltovaSplashScreens`**. The image files will be contained in this entity and will not be visible as separate files.
8. Click **OK** when done. The project and the files selected for deployment are now deployed to MobileTogether Server as the workflow, **`AltovaSplashScreens`**.

That's it!

3.2 QuickStart (Part 2)

The second part of this QuickStart tutorial continues where Part 1 left off. It focuses on using an external XML file as its data source. You will learn how to generate a dynamic dropdown list for the combo box from the data in the XML file, how to create dynamic links to website pages, and how to save data back to the XML file.

This tutorial is organized into the following sections, each of which deals with one key aspect of creating a MobileTogether design.

- Load Data from a File
- Change Page Source Link
- Run a Simulation
- Use File Data for Combo Box Entries
- Set Data File as Default File
- Create Dynamic Links to Web Pages
- Save Data Back to File

The tutorial files
The files for this tutorial are located in your *(My) Documents* MobileTogether folder: **`MobileTogetherDesignerExamples\Tutorials\`**. The `Tutorials` folder also contains the splash screen images referred to in the tutorial. You can save these images and the XML file to any other location and use them from there if you like.

- The file to start with is the file you created in Part 1. Alternatively, you can use the supplied design file, **`QuickStart01.mtd`**.
- The file you will end with should be similar to **`QuickStart02.mtd`**
- The XML data file is **`AltovaProducts.xml`**
- The image files are: **`*.bmp`**

Load Data from a File

In this part, you will:

- Specify that the data for the solution be taken from an XML file when the solution page is loaded. The file, located in the Tutorials folder, is called **`AltovaProducts.xml`** and its listing is given below. It has a similar structure to the data source created in Part 1, with the only difference being that there is one new element: **`Selection`**.
- Modify the tree structure of the data source to match the different structure of the XML data file.

Listing of the XML file, AltovaProducts.xml

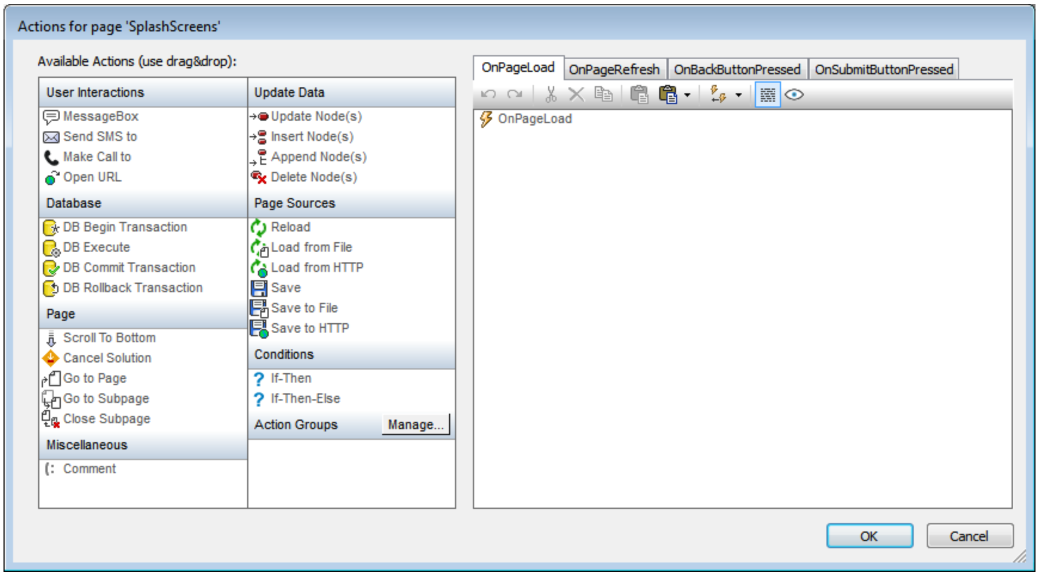
Located in the MobileTogether folder of the *(My) Documents folder*: **`MobileTogetherDesignerExamples\Tutorials\AltovaProducts.xml`**.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Products>
<Selection></Selection>
<Product>XMLSpy</Product>
<Product>MapForce</Product>
<Product>StyleVision</Product>
<Product>MobileTogether</Product>
<Product>DatabaseSpy</Product>
<Product>DiffDog</Product>
<Product>SchemaAgent</Product>
<Product>UModel</Product>
<Product>Authentic</Product>
</Products>
```

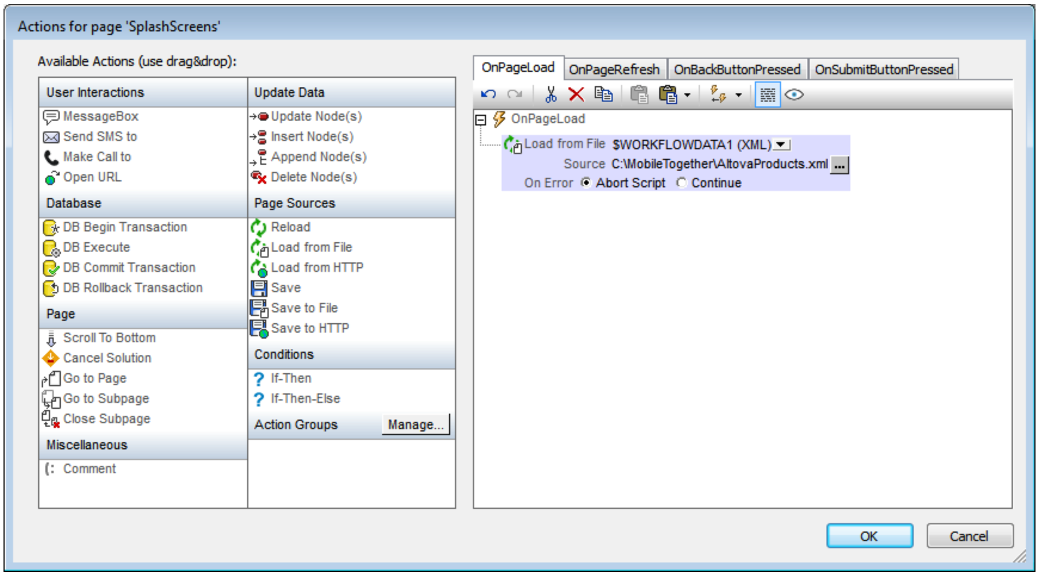
Specify data file to use on page load

To specify that data for the page data source be taken from an XML file, do the following:

1. Open **`QuickStart01.mtd`**, which is located in the MobileTogether folder of the *(My) Documents folder*: **`MobileTogetherDesignerExamples\Tutorials\`**.
2. Click **Page | Page Actions**. This displays the Page Actions dialog (*screenshot below*).



3. Drag and drop the *Load from File* action into the tab of the `OnPageLoad` event.
4. Click the **Additional Dialog** button of the **`Source`** entry. This displays the Specify File dialog.
5. Select *Absolute/Relative Path* and browse for the file, **`AltovaProducts.xml`**.
6. You will be asked whether you want to deploy this file together with the design file to MobileTogether Server. Click **OK**. The file will be set as the data file to load when the page is loaded (*screenshot below*).

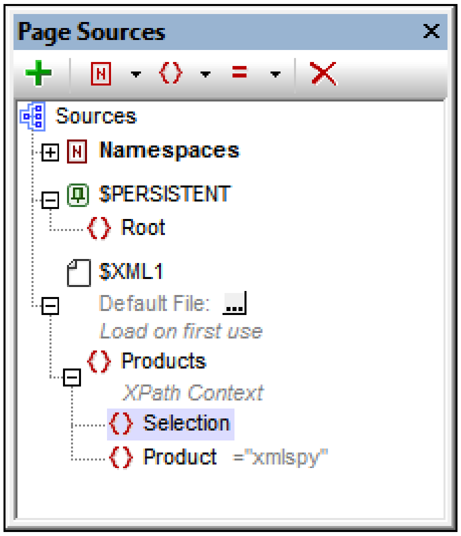


OnPageLoad
Load from File $WORKFLOWDATA1 (XML)
Source C:\MobileTogether\AltovaProducts.xml
On Error ● Abort Script ○ Continue

7. Click **OK** to finish.

Modify the data structure of the page source

The XML data file has an extra **`Selection`** element. So, in order for the XML tree to hold the data from this element, we will now add a **`Selection`** element to the XML tree of the data source in the Page Sources Pane (*see screenshot below and listing above*). Add the **`Selection`** element to the tree by right-clicking either `Products` or `Product` and selecting, respectively, **Add Child** or **Append**, and then **Element**. Rename the element to **`Selection`** by double-clicking the element and then editing the name.



We will not add any default value for **`Selection`**. This is because, when the page is loaded, we want the data for the page to come from the file `AltovaProducts.xml`. It was the action we defined for the `OnPageLoad` event of this page (*see above*). If we were to set a default value for **`Selection`**, then this default value would override the value obtained from the **`Selection`** node in `AltovaProducts.xml`. So with no default values assigned in the Page Sources Pane, when the page is loaded, the **`Selection`** node will be empty. This is because the **`Selection`** node in `AltovaProducts.xml` is empty (*see the file listing above*). We will test this in simulations later in the tutorial.

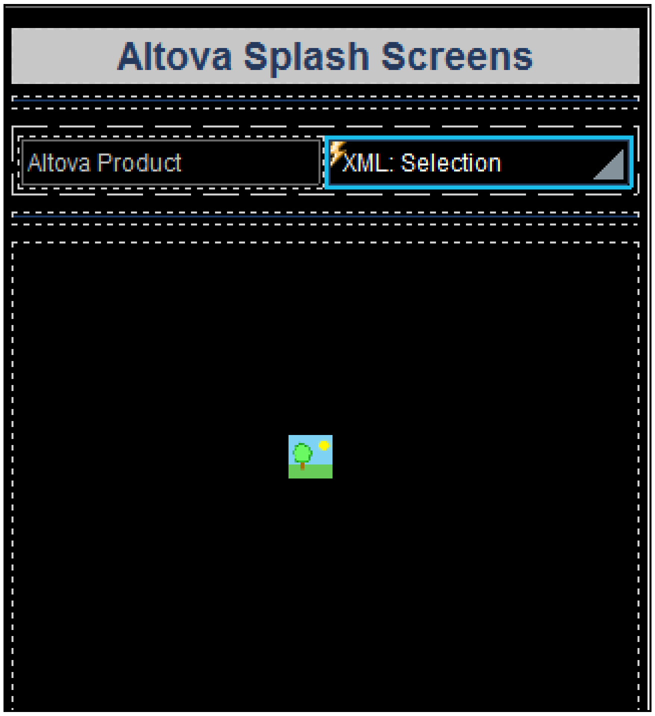
Change Page Source Link

In this part, you will:

- Change the page source link of the combo box
- Save the XPath expression of the image

Change page source link of the combo box
Each page control can have a **page source link**, which is a node in one of the data sources. The link is made by dragging the data source node from the Page Sources Pane onto the control in the design. Essentially, a page source link transfers data from the XML node to the control. But how the data in the XML node relates exactly to the control depends on the type of control it is. For example: A combo box selection updates its page source link—an XML node—and that value is reflected in the combo box display, whereas the page source link of an image provides the URL of the image. When you hover over a control, the popup indicates how the page source link will be used, for example, as an XML node to edit (for combo boxes), or as a data originator (for images).

We will change the page source link of the combo box, from **`Product`** to **`Selection`**. Do this by dragging the **`Selection`** node from the Page Sources Pane onto the combo box control (*see screenshot below*).



We do this because we want to put the end user's combo box selection in the **`Selection`** element rather than the **`Product`** element. The reasons for wanting this are:

- There are multiple sibling **`Product`** elements in the file `AltovaProducts.xml`, each of which contains data we do not want to change.
- If the page source link were `Product`, only the first `Product` element (`Product[1]`) would

be updated with the combo box selection. This is undesirable.
- The best solution would be to store the end user selection in a separate element.

After changing the page source link, the combo box selection will update the **`Selection`** node—and not the **`Product`** node.

Change XPath expression of the image URL

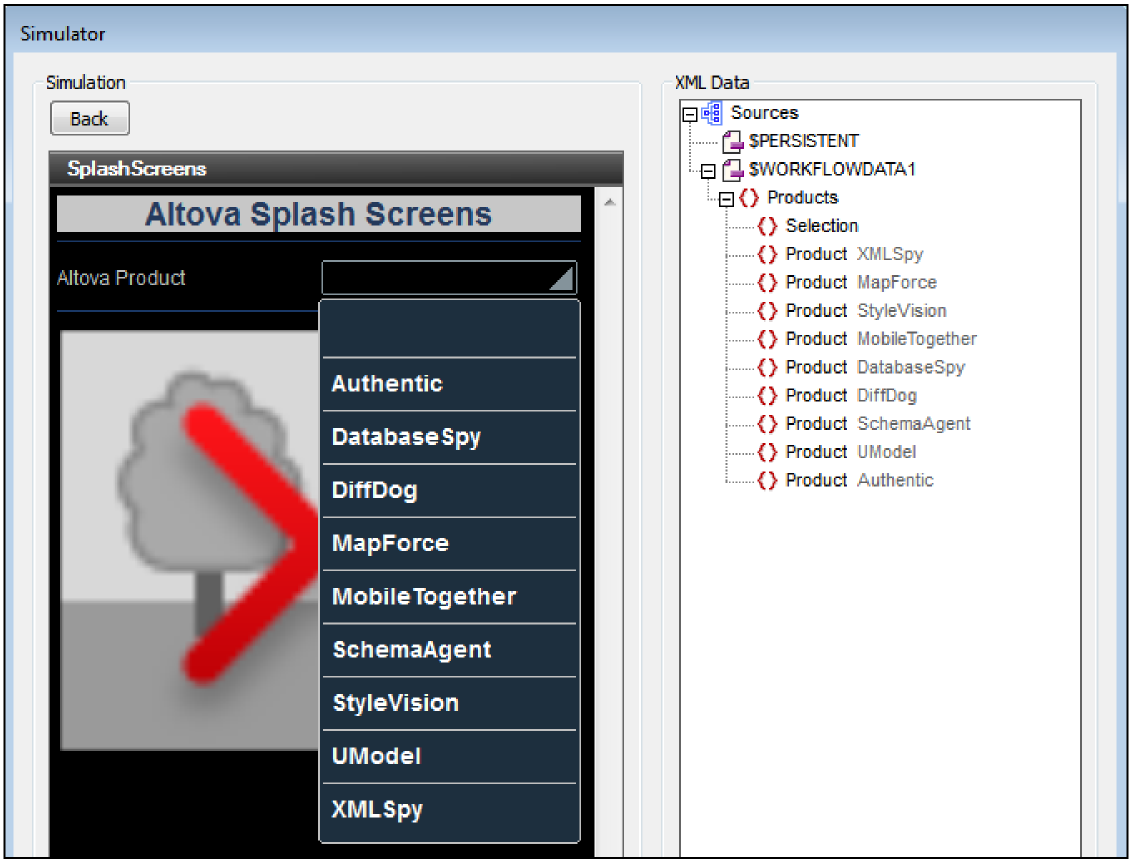
Since the XML value of the combo box selection goes into the **`Selection`** node, this node must be used in the XPath expression that constructs the image URL. In the design, select the image, and click the **XPath** button of the **`Image URL`** property (in the Styles & Properties Pane). In the Edit XPath/XQuery Expression Dialog that appears, modify the XPath expression so that **`Product`** is replaced by **`Selection`**. For example:

If you have:	`concat(Product, '.bmp')`
Change it to:	`concat(Selection, '.bmp')`

This XPath expression uses the end user's combo box selection (now stored in the **`Selection`** node) to generate the image file name. The file name is also the relative path to the file from the location of the design file (since both files are in the same folder).

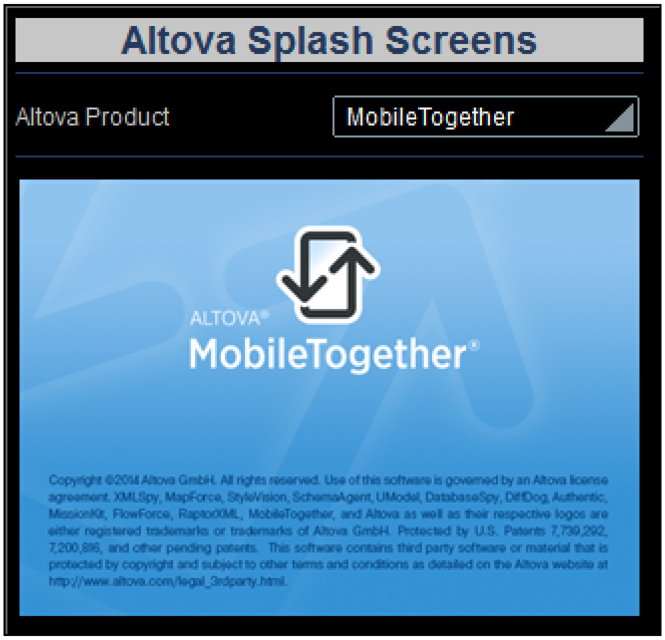
Run a Simulation

If you run a simulation now (**Project | Simulate Workflow** or **F5**), you will see the following:



What you see	Reason
`Selection` node is empty	Value comes from the empty **`Selection`** node in `AltovaProducts.xml`, the file that is loaded on page open
Combo box empty	Because the **`Selection`** node is empty
Dropdown list has empty entry	Empty entry added as a result of current combo box selection (`=empty`)
No splash screen displayed	Image URL is built using the empty **`Selection`** node

If you now select a product from the dropdown list (for example, `MobileTogether`), the splash screen of that product will be displayed (*screenshot below*).



Altova Splash Screens

Altova Product: MobileTogether

Copyright ©2014 Altova GmbH. All rights reserved. Use of this software is governed by an Altova license agreement. XMLSpy, MapForce, StyleVision, SchemaAgent, UModel, DatabaseSpy, DiffDog, Authentic, MissionKit, FlowForce, RaptorXML, MobileTogether, and Altova as well as their respective logos are either registered trademarks or trademarks of Altova GmbH. Protected by U.S. Patents 7,739,292, 7,200,816, and other pending patents. This software contains third 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 is because:

1. A combo box selection (of, say, **`MobileTogether`**) puts the corresponding value (`mobiletogether`) into the **`Selection`** node.
2. The value in the **`Selection`** node is used to correctly construct the image URL.

Notice also that the combo box, from now onwards, no longer has the empty entry in its dropdown list. This is because the **`Selection`** node is not empty and because the list of defined combo box entries, therefore, does not have an empty entry.

After you have finished, click **Close** or press **Esc** to close the simulator.

Use File Data for Combo Box Entries

In this part, you will:

- Use the data tree structure to generate the combo box entries
- Run a simulation to test the effect of the change

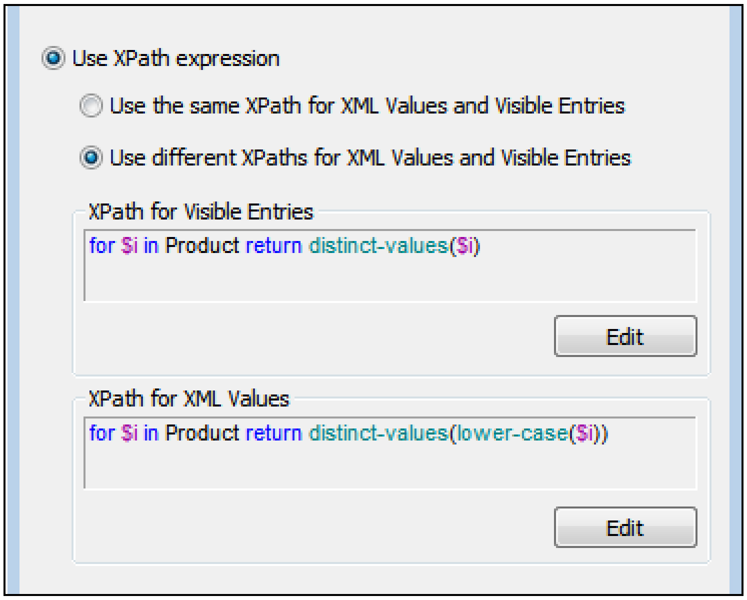
⊟ *Listing of the XML file, AltovaProducts.xml*
Located in MobileTogether folder of *(My) Documents folder*:
`MobileTogetherDesignerExamples\Tutorials\AltovaProducts.xml`.

```
<?xml version="1.0" encoding="UTF-8"?>
<Products>
<Selection></Selection>
<Product>XMLSpy</Product>
<Product>MapForce</Product>
<Product>StyleVision</Product>
<Product>MobileTogether</Product>
<Product>DatabaseSpy</Product>
<Product>DiffDog</Product>
<Product>SchemaAgent</Product>
<Product>UModel</Product>
<Product>Authentic</Product>
</Products>
```

Edit the combo box entries

Edit the combo box entries as follows:

1. Select the combo box, and, in the Styles & Properties Pane, click the Additional Dialog button of the **`Combo Box Entry Values`** property. The Edit Combo Box dialog (*screenshot below*) appears.



● Use XPath expression

○ Use the same XPath for XML Values and Visible Entries

● Use different XPaths for XML Values and Visible Entries

XPath for Visible Entries

```
for $i in Product return distinct-values($i)
```

XPath for XML Values

```
for $i in Product return distinct-values(lower-case($i))
```

2. Select *Use XPath expression*, and then *Use different XPath for XML Values and Visible Entries*.
3. Enter the XPath expressions for *Visible Entries* and *XML Values* as shown in the screenshot above.
4. Select the *Sort values* check box at the bottom of the dialog to sort the list when it is displayed.
5. Click **OK** to finish.

Remember that the **`Products`** node was defined as the default XPath context node for this page. The XPath **`for`** expression iterates over the `Product` child nodes of `Products` (the context node) and returns a sequence of all the unique distinct values, alphabetically sorted. In the case of the *XML Values* sequence, the values are transformed to lowercase before being filtered for uniqueness. These two sequences are the entries of the dropdown list (*Visible Entries*) and their corresponding XML values (*XML Values*). The advantage of using the data tree structure to build the combo box entries and a data source file to load data is that combo box entries are generated dynamically from the data source file; they are not hard-coded as items of a list in the design. Consequently, if a new product is added to the XML file, it will automatically appear as an entry in the dropdown list.

Run a simulation

When you run a simulation, it will run in exactly the same way as when the combo box entries were defined as a list (*see the previous section*, Run a Simulation). The only difference will be that the entries of the dropdown list will be the values of the `Product` elements in **`AltovaProducts.xml`** (*see listing above*). When an entry from the dropdown list is selected, the corresponding (lowercase) XML value will be entered in the **`Selection`** node, and the image URL will be correctly evaluated.

Change data in the data source file

Make the following two changes in the data source file, `AltovaProducts.xml` (*listing above*):

- Add a lowercase product name to the **`Selection`** node as shown in the listing below
- Remove a few `Product` elements from the file, or add some `Product` elements to the file, and change the order of the `Product` elements

```
<?xml version="1.0" encoding="UTF-8"?>
<Products>
  <Selection>databasespy</Selection>
  <Product>XMLSpy</Product>
    ...
    ...
  <Product>DatabaseSpy</Product>
</Products>
```

After making these changes, save the file and run a simulation. The initial splash screen will be that of the product in the **`Selection`** node. Also, the dropdown list of the combo box will not have any empty entry, and the number of entries in the dropdown list will be equal to the number of unique `Product` elements in the XML file.

Set Data File as Default File

In this part, you will:

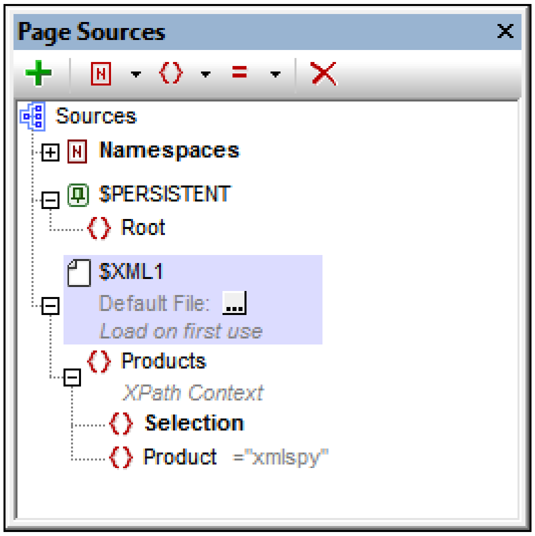
- Specify a default file for the page data source
- Run a simulation

⊟ *Buttons in this section*

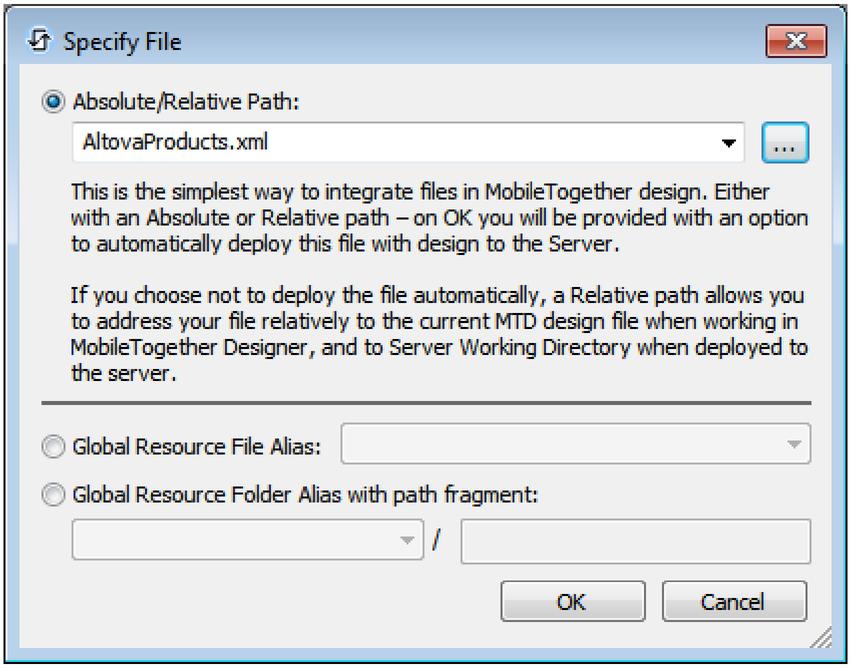
...	**Additional Dialog**

Specify a default file for the page data source
The data that goes into a data source can be specified by selecting a default file for the data source. Do this as follows:

1. Click the **Additional Dialog** button of the **`$XML1`** default file (*screenshot below*).



2. Select *Absolute/Relative Path* and browse for the file **`AltovaProducts.xml`**.



● Absolute/Relative Path:

AltovaProducts.xml

This is the simplest way to integrate files in MobileTogether design. Either with an Absolute or Relative path – on OK you will be provided with an option to automatically deploy this file with design to the Server.

If you choose not to deploy the file automatically, a Relative path allows you to address your file relatively to the current MTD design file when working in MobileTogether Designer, and to Server Working Directory when deployed to the server.

○ Global Resource File Alias:

○ Global Resource Folder Alias with path fragment:

3. On clicking **OK**, you will be prompted about whether you want to deploy the file to the server. Click **No**. The file will be added as the default file, and its data will populate the data source tree.
4. Click **Page | Page Actions** to open the Page Actions dialog.
5. In the `OnPageLoad` tab, select the Load from File entry and delete it. This is because the *Load from File* action is now redundant since the file `AltovaProducts.xml` has been specified as the default file of the page's only data source.
6. Run a simulation to test whether the default file is being used.

Create Dynamic Links to Web Pages

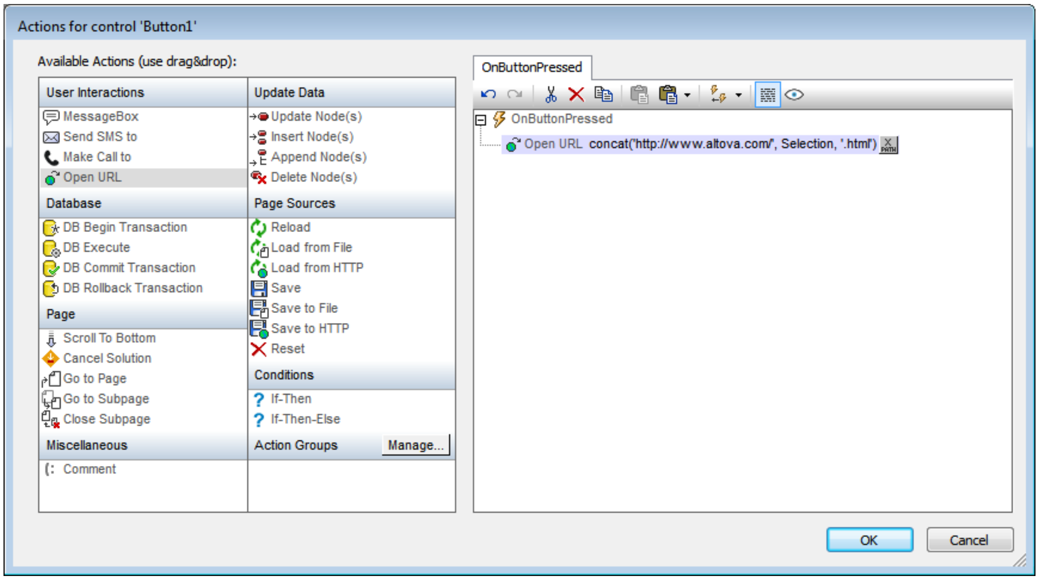
In this part, you will:

- Add a button that links dynamically to a web page (using an XPath expression)
- Run a simulation

Add a button that links to a web page

We will now add a button that links to the product description page of the product selected in the combo box. Do this as follows:

1. Drag the button control from the Controls Pane and drop it below the image (*see the simulator screenshot below*).
2. Enter the text *Go to Product Description*.
3. Right-click the button and select **Control Actions for OnButtonPressed**.
4. In the Actions dialog that appears (*screenshot below*), drag the *Open URL* action into the `OnButtonPressed` tab.
5. Click the **XPath** button, and, in the Edit XPath/XQuery Expression dialog that appears, enter the XPath expression: `concat('http://www.altova.com/', Selection, '.html')`

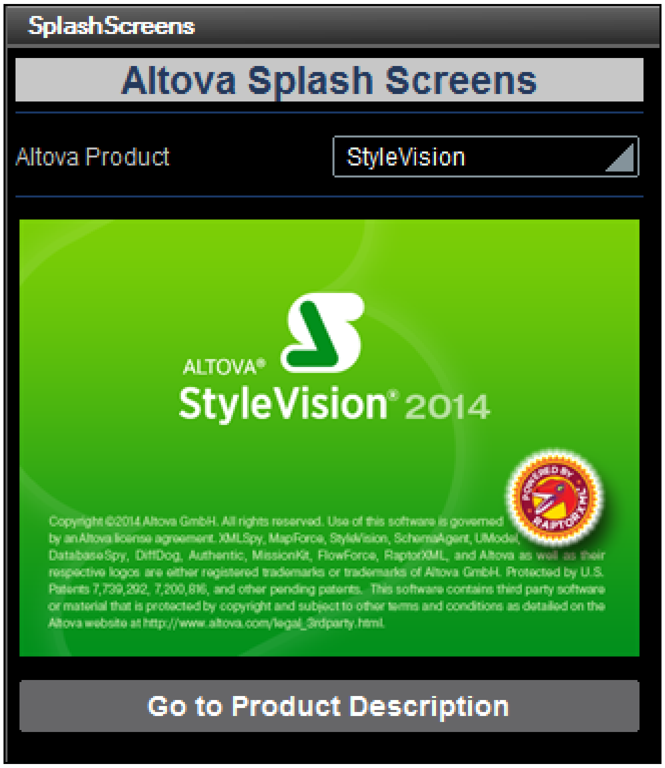


Open URL concat('http://www.altova.com/', Selection, '.html')

6. Click **OK** to finish, and save the file.

Run a simulation

Run a simulation by clicking **F5** or **Project | Simulate Workflow**. When the simulation starts, select a product in the combo box and then click the **Go to Product Description** button (*see screenshot below*). This will take you to the product description page on the Altova website.



Altova Splash Screens

Altova Product: StyleVision

Copyright ©2014 Altova GmbH. All rights reserved. Use of this software is governed by an Altova license agreement. XMLSpy, MapForce, StyleVision, SchemaAgent, UModel, DatabaseSpy, DiffDog, Authentic, MissionKit, FlowForce, RaptorXML, and Altova as well as their respective logos are either registered trademarks or trademarks of Altova GmbH. Protected by U.S. Patents 7,739,292, 7,200,816, and other pending patents. This software contains third 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.

Save Data Back to File

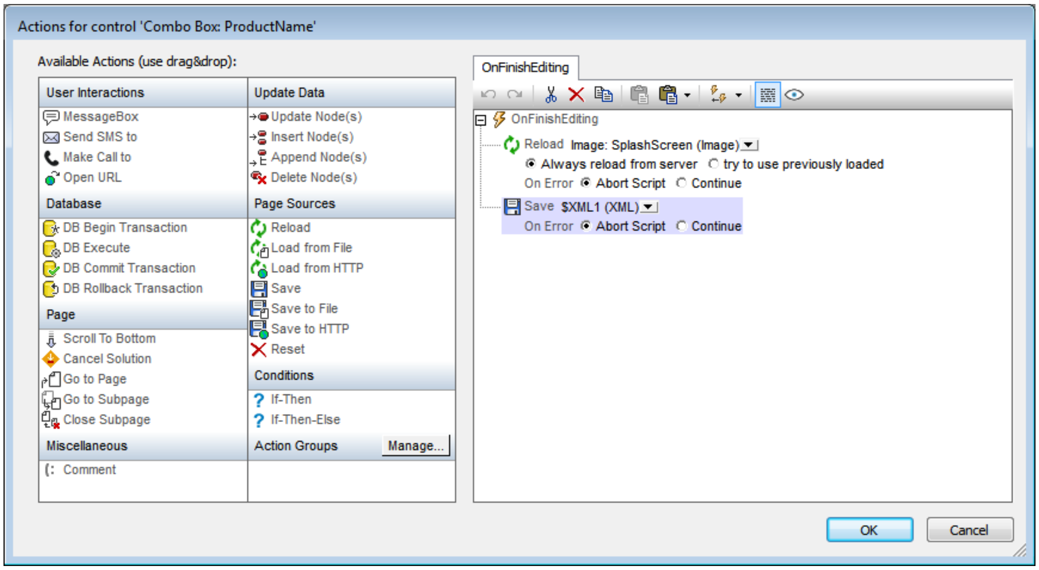
In this part, you will:

- Save changed data back to file using a control action
- Run a simulation to test the success of the *Save to File* action

Save data to file after editing combo box

You can specify that a change made by editing the combo box is saved back to file. Since the combo box has a page source link to the `Selection` element, the combo box selection will be saved to this element. To specify that the change is saved back to the `Selection` element in the default file, we will add the *Save* action to the **`OnFinishEditing`** event of the combo box. Do this as follows:

1. Right-click the combo box and select **Control Actions for OnFinishEditing** from the context menu.
2. This displays the Control Actions dialog for the combo box, which already has the *Reload* action that targets the image.
3. Drag the *Save* action below the *Reload* action, and drop (*see screenshot below*).



OnFinishEditing
Reload Image: SplashScreen (Image)
● Always reload from server ○ try to use previously loaded
On Error ● Abort Script ○ Continue
Save $XML1 (XML)
On Error ● Abort Script ○ Continue

4. Click **OK** to finish, and then save the file.
5. To test whether the change is saved to the default file, open the default file in an editor, run a simulation, select a combo box entry, and then reload the default file in the editor. The new combo box selection will appear as content of the `Selection` element in the default file.

Note: If you wish to save to some file other than the default file, use the *Save to File* action (instead of the *Save* action). If you wish to save to a web page, use the *Save to HTTP action*. (In this case, you will also need to provide the authentication details that allow the user to modify the page at the HTTP URL.)

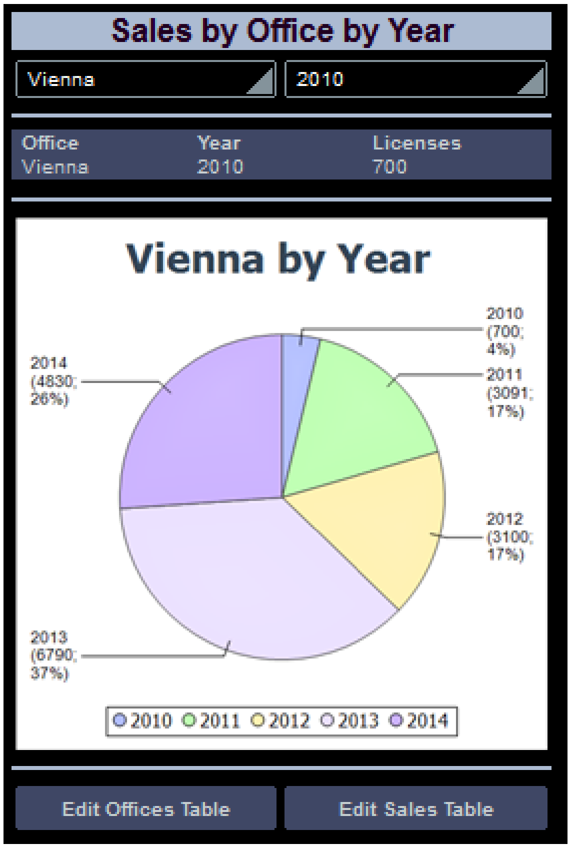
That's it!

3.3 Database-And-Charts

This Database-And-Charts tutorial (`DBAndCharts.mtd`) shows you how to work with databases (DBs) and charts. It explains how to:

- Set up DB tables as data sources so that DB table data can be displayed and edited
- How to display DB data based on end user choices
- How to create charts that are based on the DB data

The screenshot below shows the first page of the `DBAndCharts.mtd` solution. The end user can select the office and year for which sales are to be displayed. These selections are made in combo boxes at the top of the design. The total sales for that year are then displayed in the *Licenses* column of the results table below the combo boxes. Whenever a new office or year is selected in either of the combo box, the results table is automatically updated to reflect the new selection. Additionally, a pie chart showing the sales, by year, for the selected office is displayed. Each slice represents a year, with its sales volume and the percentage of all sales till now that year represents. Whenever a new office is selected, a pie chart showing the statistics of that particular office is displayed.



Sales by Office by Year

Vienna | 2010

Office	Year	Licenses
Vienna	2010	700

Below the chart are two buttons that each take you to a page in which the relevant DB table data can be edited.

The tutorial files

The files for this tutorial are located in your *(My) Documents* MobileTogether folder: **`MobileTogetherDesignerExamples\Tutorials\`**.

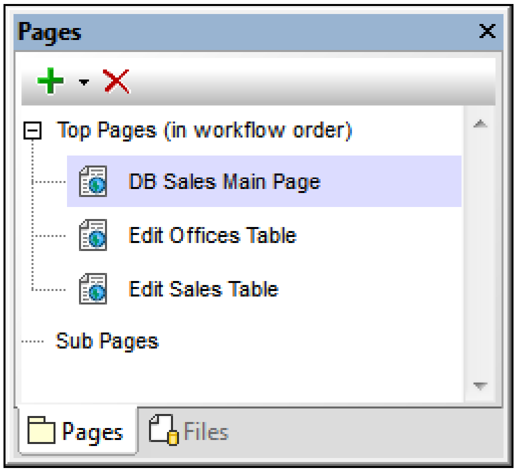
- The design file is **`DBAndCharts.mtd`**. Open it and read the descriptions in the tutorial to see how the design was built and how it works.
- The MS Access database, **`OfficeSales_DB.mdb`**, contains the tables used as the data sources of the design.

The Project Structure

Open DBAndCharts.mtd and validate it (**Project | Validate**) to check that the file correctly connects to the Access DB, OfficeSales_DB.mdb. If there is a connection error, make sure you correct this before proceeding. (See the section, Data Sources of the Main Page, for more information.)

As shown in the Pages Pane (*screenshot below*), the project consists of three top pages:

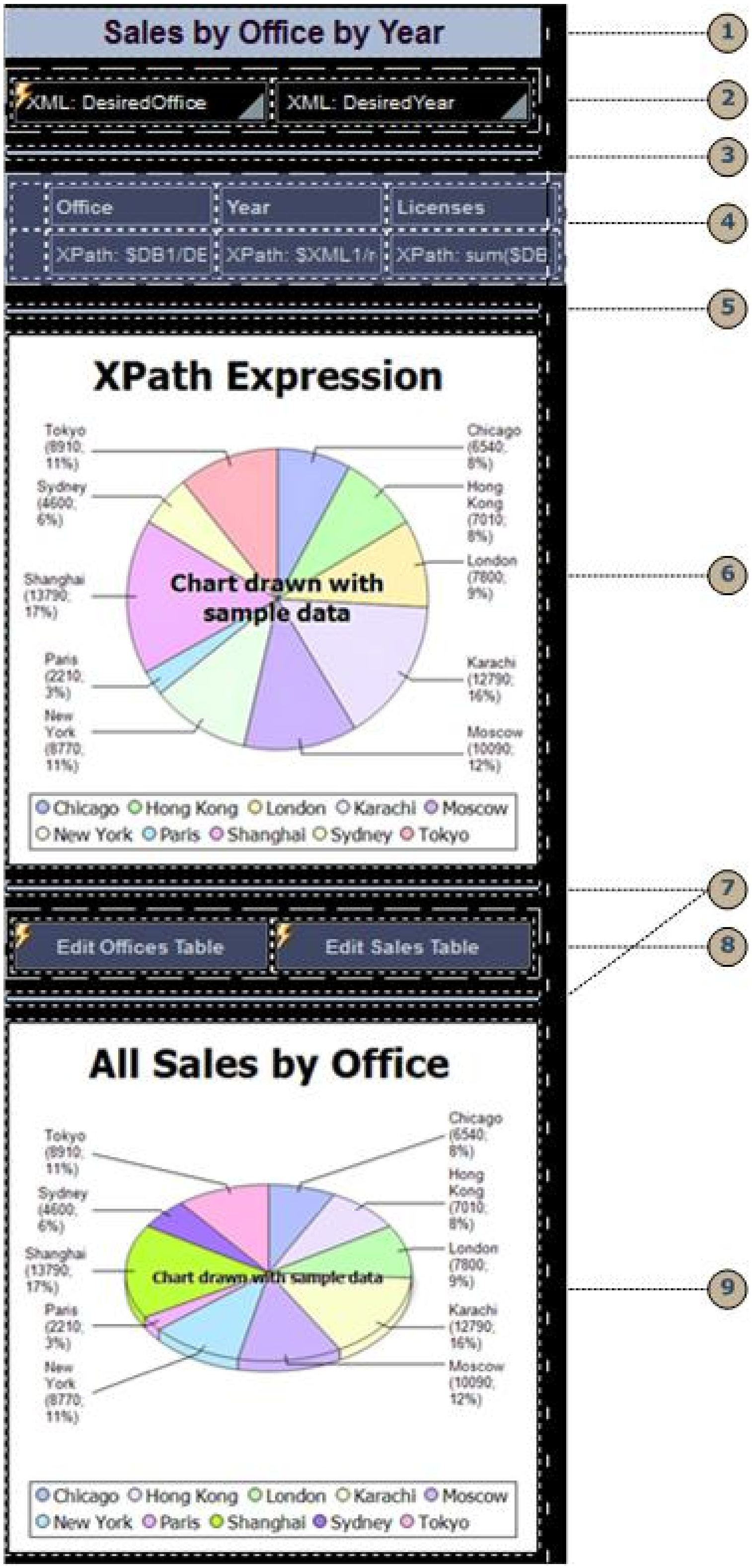
- *DB Sales Main Page:* This is the start page. It displays the DB data and has two buttons that go, respectively, to the other two top pages.
- *Edit Offices Table:* Is arrived at via a button click from the main page and enables editing of the DB's Offices table.
- *Edit Sales Table:* Is arrived at via a button click from the main page and enables editing of the DB's Sales table.



When the solution runs, note that it is the first top page in the list above, *DB Sales Main Page*, that is loaded in the client app.

The Main Page

The design of the *DB Sales Main Page* is show below. Its components are numbered in the callouts and are described below.



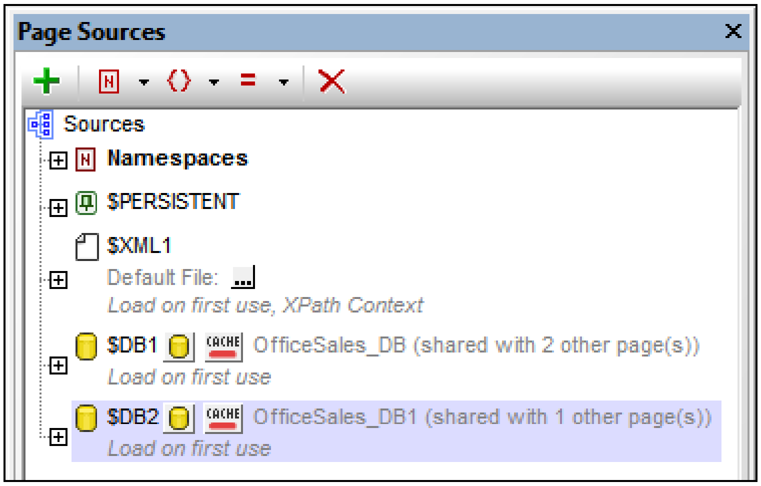
All the components are controls that have been dragged from the Controls Pane and dropped into the design. Each has then been assigned properties in the Styles & Properties Pane. For controls that need to be associated with data from the data sources, the appropriate page source link has been assigned by dragging the data source node onto the control. The combo boxes and buttons additionally have actions associated with their events. Actions are assigned in the Actions dialog for the control, which is accessed by right-clicking the control and selecting the **Control Actions for...** command.

1	A label control to display the title of the page; style properties applied
2	Combo boxes for end user selection of `Office` and `Year`. *See detailed description*
3 5 7	A horizontal line control as layout component; style properties applied
4	Table control with cells that contain DB data. *See detailed description*
6 9	Chart controls that display DB data in the form of charts. *See detailed description*
8	Button controls with `OnButtonPressed` actions that go to *Edit Offices* and *Edit Sales* pages

The *DB Sales Main Page* has an action defined for its `OnPageLoad` event (**Page | Page Actions**) that updates a data source node. This action is explained in the next section, Data Sources of the Main Page.

Data Sources of the Main Page

The main page has three data sources: **`$XML1`**, **`$DB1`**, and **`$DB2`**. These are displayed and managed in the Page Sources Pane (*see screenshot below*).

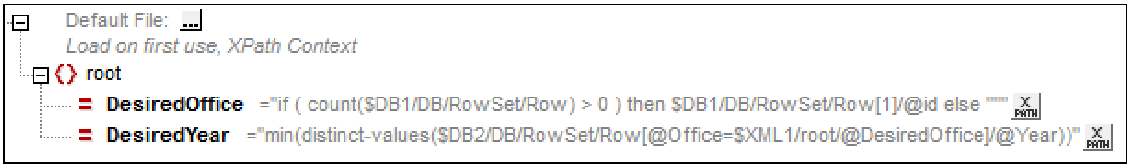


Page Sources

Sources
Namespaces
$PERSISTENT
$XML1
Default File: ...
Load on first use, XPath Context
$DB1 OfficeSales_DB (shared with 2 other page(s))
Load on first use
$DB2 OfficeSales_DB1 (shared with 1 other page(s))
Load on first use

The XPath context node of the page is the root node **`$XML1`**. This means that all XPath expressions on this page have `$XML1` as their context node. To locate a node in any of the other trees (`$DB1` and `$DB2`, which are the root nodes of these trees) start the XPath locator path with the respective root node.

The first data source: `$XML1`
This data source was created as an editable empty XML. The root node **`$XML1`** contains a root element (**`root`**), which has two attributes (**`DesiredOffice`** and **`DesiredYear`**). The root node, `$XML1`, was set (via its context menu) as the XPath context node for the page two. No default file is set, so no data is imported into the tree.

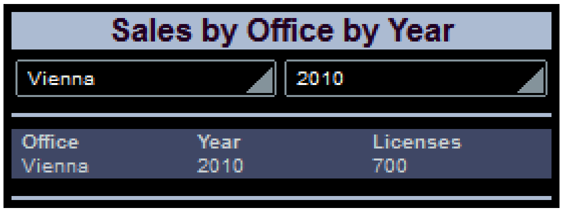


Default File: ...
Load on first use, XPath Context
root
DesiredOffice ="if (count($DB1/DB/RowSet/Row) > 0) then $DB1/DB/RowSet/Row[1]/@id else """
DesiredYear ="min(distinct-values($DB2/DB/RowSet/Row[@Office=$XML1/root/@DesiredOffice]/@Year))"

This data source (`$XML1`) has been created to hold the end user's combo box selections:

- The `DesiredOffice` attribute holds the end user's `Office` selection
- The `DesiredYear` attribute holds the end user's `Year` selection

In order to hold the data selected in the combo boxes, the two attribute nodes are associated with the combo boxes as page source links. Each of the two page source links is made by dragging the attribute node onto the respective comb box (*see simulator screenshot below*).



Sales by Office by Year

Vienna

2010

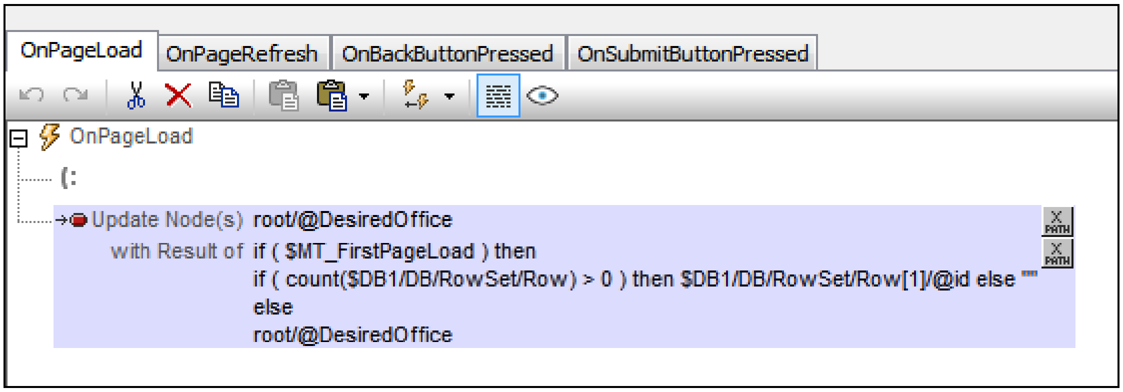
Office	Year	Licenses
Vienna	2010	700

Each of the nodes has been given an initial value when the page loads (via the context menu command **Ensure Exists before Page Load (XPath Value)**). This is because the value of the node appears in the associated combo box, and we want the combo box to have an initial selection (*see simulator screenshot above*). The XPath expressions that provide the initial values are:

- For `@DesiredOffice`: `if (count($DB1/DB/RowSet/Row) > 0) then $DB1/DB/RowSet/Row[1]/@id else ""`
  If there is one or more records in `$DB1`, sets the `@id` value of the first record as the value of `@DesiredOffice`. If there is no record, sets the empty string as the value of `@DesiredOffice`.

- For `@DesiredYear`: `min(distinct-values($DB2/DB/RowSet/Row[@Office=$XML1/root/@DesiredOffice]/@Year))`
  In `$DB2`, selects all the records of the office selected in `@DesiredOffice`, collects the unique years from these records, and then selects the year with the minimum numerical value.

Additionally, we have specified that the **`@DesiredOffice`** node is correctly filled whenever the main page loads. This is done with an *Update Node* action on the **`OnPageLoad`** event of the main page (**Page | Page Actions**).

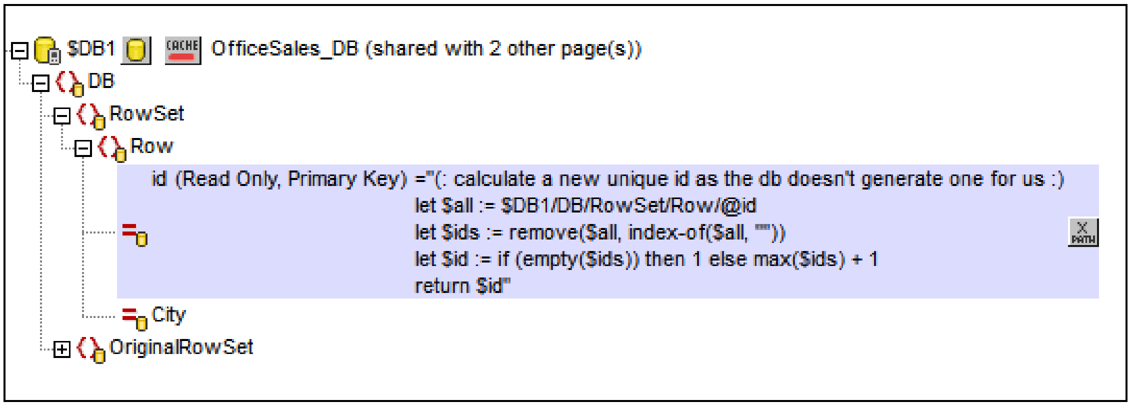


```
Update Node(s)  root/@DesiredOffice
with Result of  if ( $MT_FirstPageLoad ) then
                if ( count($DB1/DB/RowSet/Row) > 0 ) then $DB1/DB/RowSet/Row[1]/@id else ""
                else
                root/@DesiredOffice
```

The action updates the **`@DesiredOffice`** node. If this is the first loading of the page, then the ID of the first office is passed as the content of **`@DesiredOffice`**. Otherwise the value is what is already present in **`@DesiredOffice`**. The result of this is that during an execution, the value in **`@DesiredOffice`** is not changed, but the value is initialized whenever the page is loaded for the first time.

The second data source: `$DB1`

The second data source (**`$DB1`**) is the `Offices` table in the MS Access database, **`OfficeSales_DB.mdb`**. The data for this data tree comes from the DB's `Office` table.



```
$DB1    OfficeSales_DB (shared with 2 other page(s))
  DB
    RowSet
      Row
        id (Read Only, Primary Key) ="(: calculate a new unique id as the db doesn't generate one for us :)
                                      let $all := $DB1/DB/RowSet/Row/@id
                                      let $ids := remove($all, index-of($all, ""))
                                      let $id := if (empty($ids)) then 1 else max($ids) + 1
                                      return $id"
        City
    OriginalRowSet
```

The Offices table has two columns (**`id`** and **`City`**), which are represented in the data tree as attributes of the `Row` element, which itself corresponds to a row in the DB table. Since the `id` column is the primary key and values in it cannot be changed, we cannot edit this column. However, we need to create `id` values for new rows. We automate this by writing an XQuery expression to generate the `id` value for every new row that is created. The XQuery expression is inserted by using the context menu command, **Ensure Exists before Page Load (XPath Value)**:

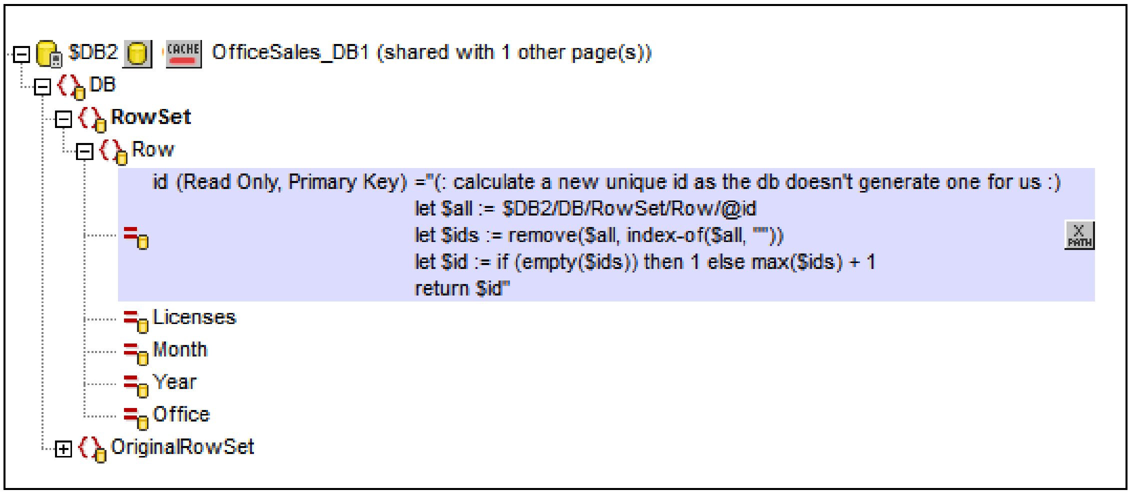
```
let $all := $DB1/DB/RowSet/Row/@id
let $ids := remove($all, index-of($all, ""))
let $id := if (empty($ids)) then 1 else max($ids) + 1
return $id
```

Note that the **`id`** value is the unique ID number of the office, whereas the **`City`** value is the name of the city in which the office is. This is important because while it is the **`id`** that is used to uniquely identify an office (via the `$XML1/root/@DesiredOffice` node), it is the name of the city that we use to identify an office to the end user.

An **`OriginalRowSet`** node must be created (via the context menu) if any node in the data source is to be edited. This is required so that `OriginalRowSet` holds the original data while **`RowSet`** holds the current (edited) data. The two sets of data (original and edited) are required so that MobileTogether Designer can tell the difference between what is new, updated, and deleted, and can make the necessary changes at the right time. It is also required so that it can create new primary keys with the XQuery **`let`** statement. When the database is updated, the updated data becomes the new original data and is entered in the `OriginalRowSet` node.

The third data source: `$DB2`

The second data source (**`$DB1`**) is the Sales table in the MS Access database, **`OfficeSales_DB.mdb`**. The data for this data tree comes from the DB's Sales table.



$DB2 OfficeSales_DB1 (shared with 1 other page(s))
DB
RowSet
Row
id (Read Only, Primary Key) ="(: calculate a new unique id as the db doesn't generate one for us :)
let $all := $DB2/DB/RowSet/Row/@id
let $ids := remove($all, index-of($all, ""))
let $id := if (empty($ids)) then 1 else max($ids) + 1
return $id"
Licenses
Month
Year
Office
OriginalRowSet

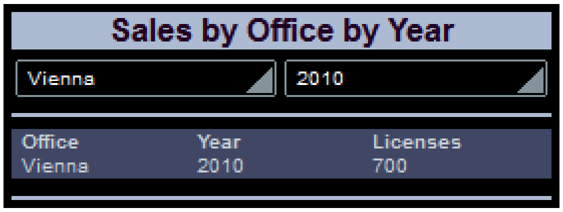
Each row in the Sales table has five columns (**id** , **Licenses**, **Month**, **Year**, and **Office**). The DB table row corresponds to the **Row** element in the data source tree. The table's columns correspond to the attributes of the **Row** element. The `id` attribute has an XQuery expression to generate the `id` value for every new row that is created. The XQuery expression is inserted by using the context menu command, **Ensure Exists before Page Load (XPath Value)**:

```
let $all := $DB1/DB/RowSet/Row/@id
let $ids := remove($all, index-of($all, ""))
let $id := if (empty($ids)) then 1 else max($ids) + 1
return $id
```

An **OriginalRowSet** node must be created (via the context menu) if any node in the data source is to be edited. This is required so that `OriginalRowSet` holds the original data while **RowSet** holds the current (edited) data.

The Combo Boxes

The two combo boxes at the top of the page are used to accept end user selections and to display data based on these selections. The screenshot below shows the combo boxes when the solution is run; the tabular report below the combo boxes is based on the combo box selections.

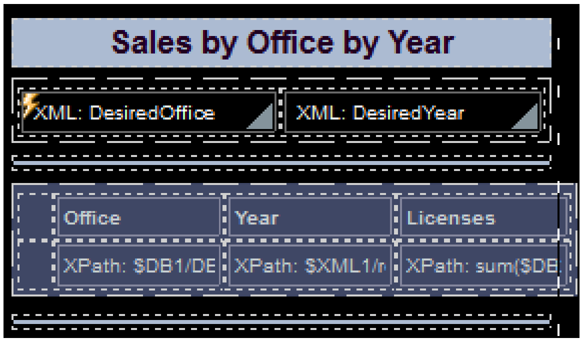


Sales by Office by Year

Vienna | 2010

Office	Year	Licenses
Vienna	2010	700

The next screenshot shows the combo boxes in the design. The combo boxes have been placed in separate cells of a table for layout purposes.



Sales by Office by Year

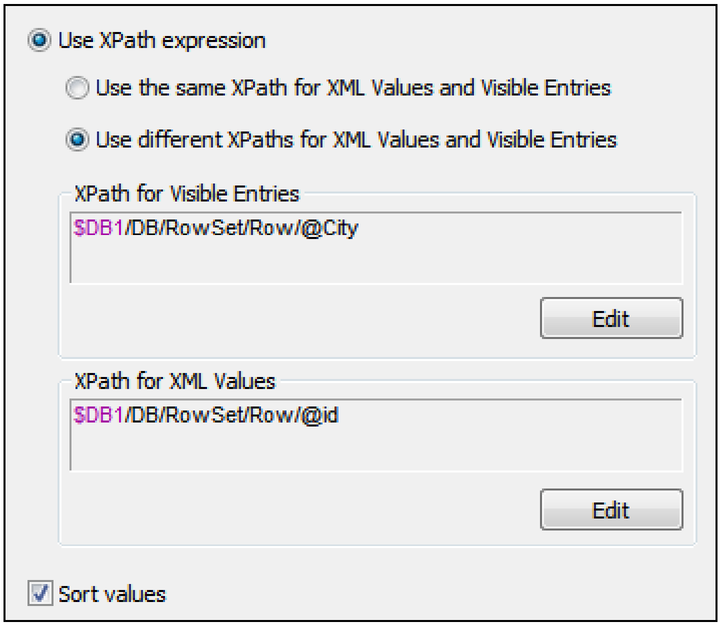
XML: DesiredOffice | XML: DesiredYear

Office	Year	Licenses
XPath: $DB1/DE	XPath: $XML1/r	XPath: sum($DB

The `DesiredOffice` combo box

The following settings have been made:

- A page source link is made between the combo box and the `$XML1/root/@DesiredOffice` node by dragging the node onto the combo box. This serves to pass the combo box selection to the node and the value of the node to the combo box.
- The items in the dropdown list of the combo box are defined in the Edit Combo Box dialog (*screenshot below*), which is accessed via the **Additional Dialog** button of the `Combo Box Entry Values` property (in the Styles & Properties Pane).



● Use XPath expression

○ Use the same XPath for XML Values and Visible Entries

● Use different XPaths for XML Values and Visible Entries

XPath for Visible Entries

```
$DB1/DB/RowSet/Row/@City
```

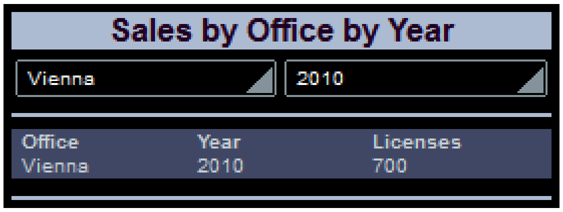
XPath for XML Values

```
$DB1/DB/RowSet/Row/@id
```

☑ Sort values

Notice that the values in the dropdown list are taken from the **`$DB1/DB/RowSet/Row/@City`** node (that is, the names of the cities). But the value that goes into the `$XML1/root/@DesiredOffice` node (because of the page source link) is taken from the **`$DB1/DB/RowSet/Row/@id`** node. Since the *Sort Values* check box has been selected the items of the dropdown list will be sorted.

- An *UpdateNode* action has been set for the **`OnFinishEditing`** event. Right-click the combo box and select **Control Actions for OnFinishEditing** to display the action definition. The node to be updated is **`root/@DesiredYear`**. The update value is provided by an XPath expression: `min(distinct-values($DB2/DB/RowSet/Row[@Office=$XML1/root/@DesiredOffice]/@Year))`. This expression selects all the records of the office selected in the combo box, then collects the unique years from these records, and finally selects the year with the minimum numerical value.



Sales by Office by Year

Vienna | 2010

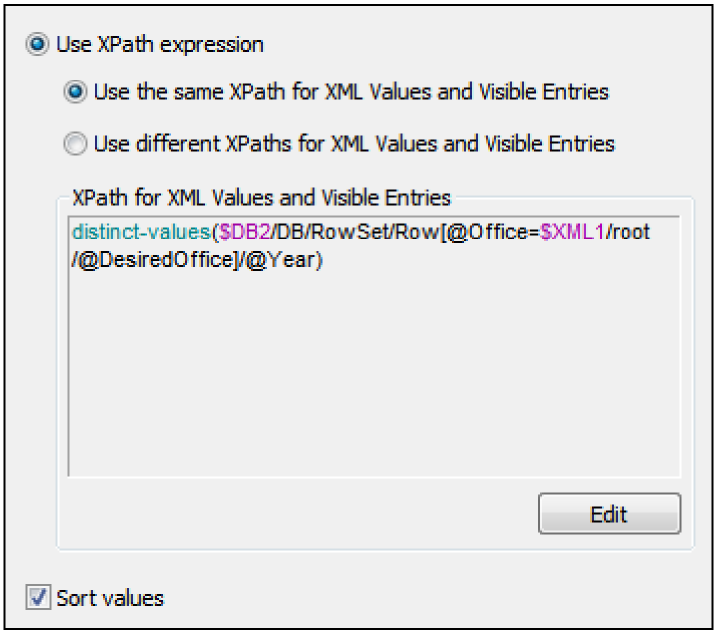
Office	Year	Licenses
Vienna	2010	700

So, when Vienna is selected in the first combo box (as in the screenshot above), all the records in **`$DB2`** with `@Office='Vienna'` were searched and a sequence of the unique years in these records was created. The year with the minimum numerical value—in this case **`2010`**—is passed to the node to be updated—in this case `$XML1/root/@DesiredOffice`. Since this node is the page source link of the second combo box (the `@DesiredYear` combo box), this combo box now displays the minimum year value—in this case, **`2010`**.

The `DesiredYear` combo box
The following settings have been made:

- A page source link is made between the combo box and the `$XML1/root/@DesiredYear` node by dragging the node onto the combo box. This serves to pass the combo box selection to the node and the value of the node to the combo box.

- The items in the dropdown list of the combo box are defined in the Edit Combo Box dialog (*screenshot below*), which is accessed via the **Additional Dialog** button of the `Combo Box Entry Values` property (in the Styles & Properties Pane).



● Use XPath expression

● Use the same XPath for XML Values and Visible Entries

○ Use different XPaths for XML Values and Visible Entries

XPath for XML Values and Visible Entries

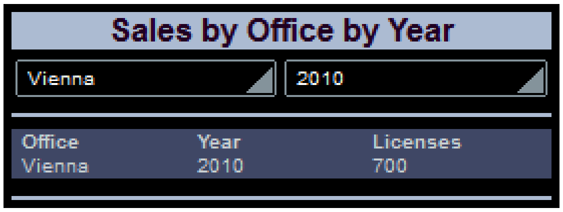
```
distinct-values($DB2/DB/RowSet/Row[@Office=$XML1/root
/@DesiredOffice]/@Year)
```

☑ Sort values

Notice that both the values in the dropdown list as well s those that will be passed to the XML node are the same. they are the sequence of all the unique years in which the selected office has recorded sales. Since the *Sort Values* check box has been selected the items of the dropdown list will be sorted.

The Tabular Report

The tabular report is displayed in the table below the two combo boxes. When the end user selects the office and year for which a report is required, the tabular report displays the total sales of that year (in terms of number of licenses). The screenshot below shows the page when the solution is run.

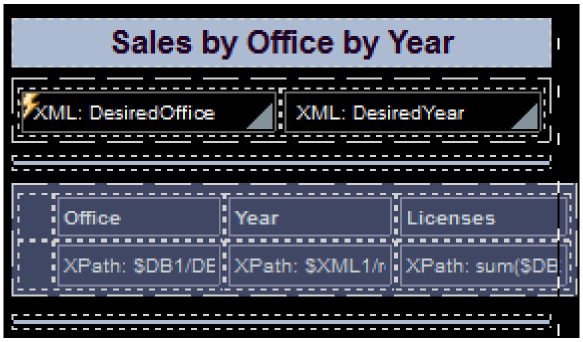


Sales by Office by Year

Vienna | 2010

Office	Year	Licenses
Vienna	2010	700

The next screenshot shows the tabular report in the design. The table consists of two rows and four columns, with the first column being used to provide padding. Each of the remaining six cells contains a label with a text value that is either directly entered text or is calculated by an XPath expression. See each label's `Text` property in the Styles & Properties Pane.



Sales by Office by Year

XML: DesiredOffice | XML: DesiredYear

Office	Year	Licenses
XPath: $DB1/DE	XPath: $XML1/r	XPath: sum($DB

The XPath expressions are as follows:

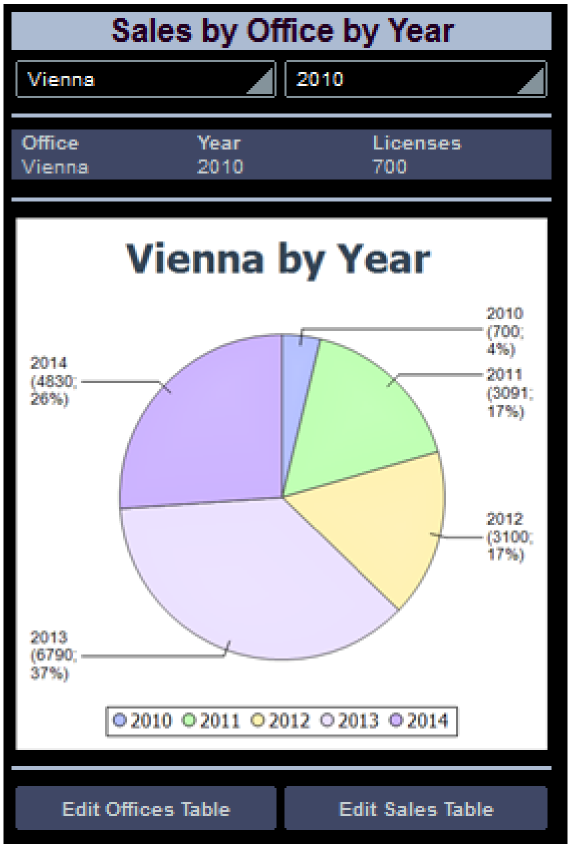
- *DesiredOffice:* Is taken from **`$DB1`**. It is the **`@City`** value of the **`Row`** with an **`@id`** equal to the **`id`** value of the combo box selection.
  `$DB1/DB/RowSet/Row[@id=$XML1/root/@DesiredOffice]/@City`

- *DesiredYear:* Is taken from **`$XML1`**. It is the value of the **`@DesiredYear`**. The year is either selected by the end user in the combo box, or is the minimum of all unique sales years for that office.
  `$XML1/root/@DesiredYear`

- *Licenses Sold:* Is taken from **`$DB2`**. Sums up all **`@Licenses`** values of the **`Row`** elements with **`@Office`** and **`@Year`** attributes equal to the values of the combo box selections. (Note that the **`@Office`** values in **`$DB2`** are the ID values of the offices, not their city names.)
  `sum($DB2/DB/RowSet/Row[@Office= $XML1/root/@DesiredOffice][@Year= $XML1/root/@DesiredYear]/@Licenses)`

The Charts

There are two charts in the design. The first chart shows the yearly breakdown of all sales of the office selected in the combo box (*see the simulator screenshot below*).



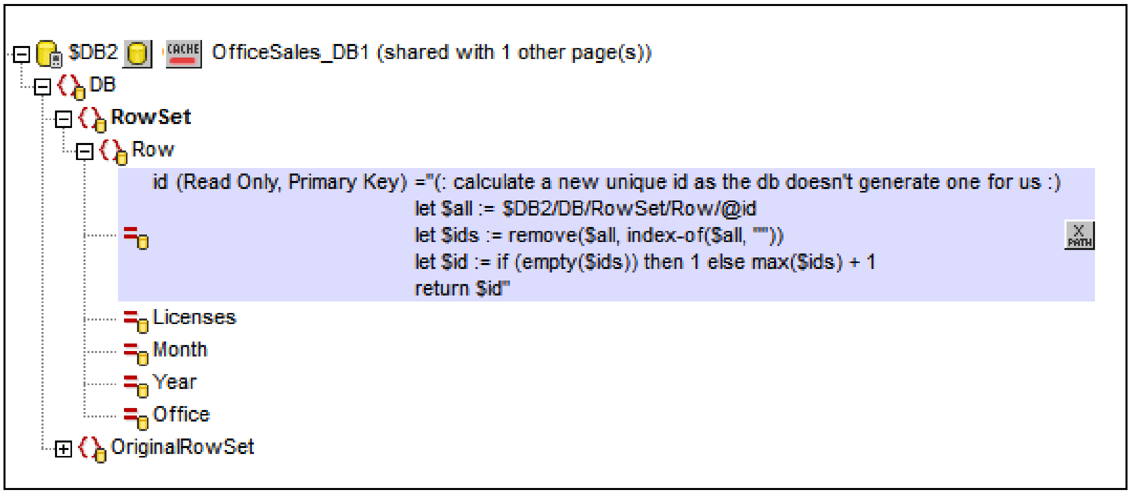
Sales by Office by Year

Vienna | 2010

Office	Year	Licenses
Vienna	2010	700

XPath context node

The main chart definitions are what goes on the X and Y axes. Since these are determined by XPath expressions it is important to correctly select the XPath context node for the chart. For the context node, it is best to select the immediate parent of the nodeset that will be used for the X and Y axes. Since we are going to use data from the Sales data table, we will use the **`$DB2`** tree for creating the chart (*screenshot below*). And since our nodeset for both axes will consist of the **`Row`** element, we select **`RowSet`** as the XPath context node. We do this by dragging the **`RowSet`** node onto the chart. The node is displayed bold, indicating that it is a page source link.



```
$DB2  OfficeSales_DB1 (shared with 1 other page(s))
  DB
    RowSet
      Row
        id (Read Only, Primary Key) ="(: calculate a new unique id as the db doesn't generate one for us :)
                                     let $all := $DB2/DB/RowSet/Row/@id
                                     let $ids := remove($all, index-of($all, ""))
                                     let $id := if (empty($ids)) then 1 else max($ids) + 1
                                     return $id"
        Licenses
        Month
        Year
        Office
    OriginalRowSet
```

Defining the chart axes

We are now ready to define the chart's axes. Open the chart's Chart Configuration dialog (*screenshot below*) by either double-clicking the chart or clicking the **Additional Dialog** button of the `Chart Settings` property (in the Styles & Properties Pane). Note that the chart type is a pie chart.

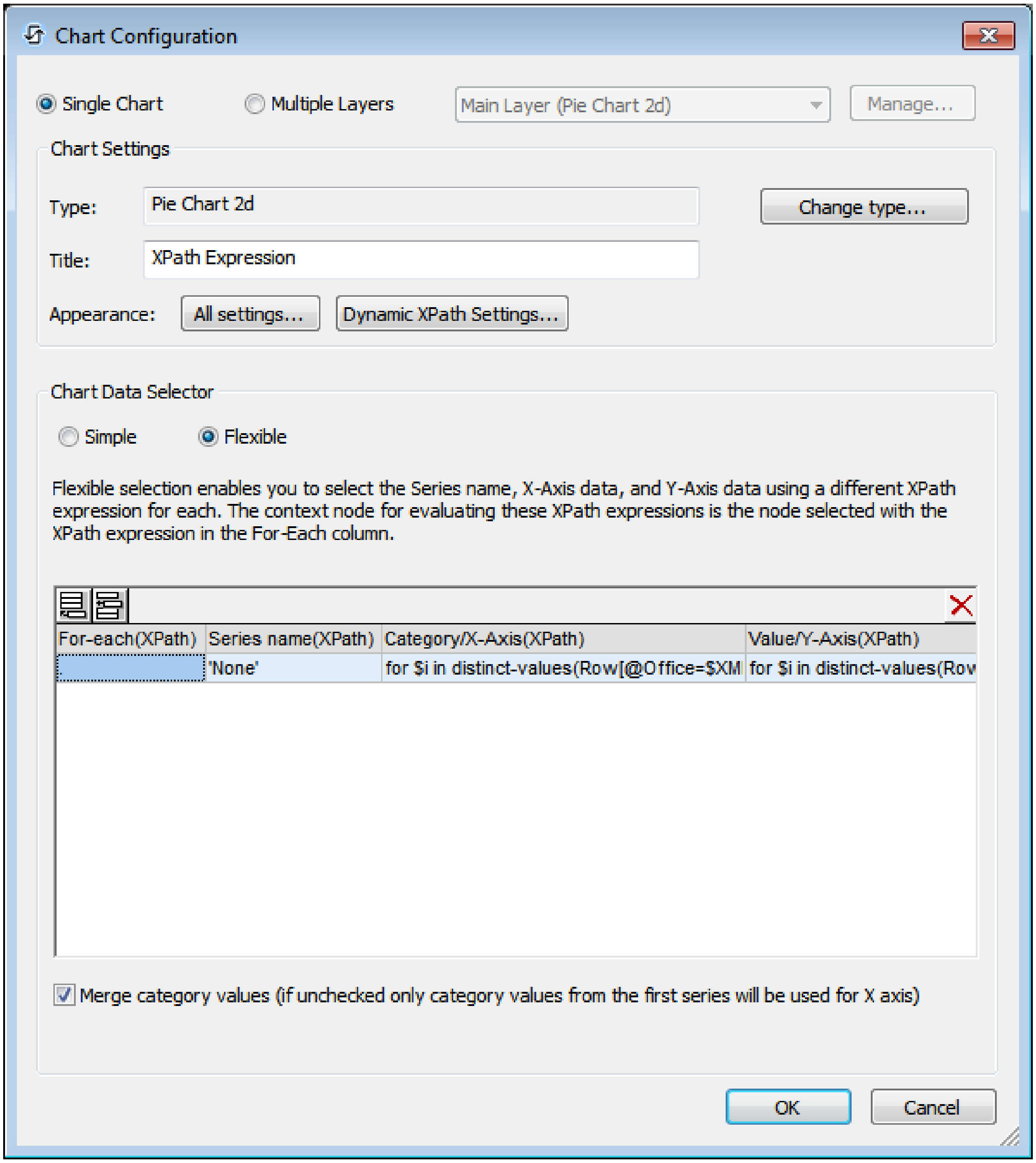


Chart Configuration

● Single Chart ○ Multiple Layers Main Layer (Pie Chart 2d) Manage...

Chart Settings

Type: Pie Chart 2d Change type...

Title: XPath Expression

Appearance: All settings... Dynamic XPath Settings...

Chart Data Selector

○ Simple ● Flexible

Flexible selection enables you to select the Series name, X-Axis data, and Y-Axis data using a different XPath expression for each. The context node for evaluating these XPath expressions is the node selected with the XPath expression in the For-Each column.

For-each(XPath)	Series name(XPath)	Category/X-Axis(XPath)	Value/Y-Axis(XPath)
	'None'	for $i in distinct-values(Row[@Office=$XM	for $i in distinct-values(Rov

☑ Merge category values (if unchecked only category values from the first series will be used for X axis)

OK Cancel

For pie charts, we need two series (for the X and Y axes). The *Flexible* option is ideal for defining the axes for two series. The *For-each* setting select the current node (`RowSet`). We define the following XPath expressions for the two axes:

- *X-Axis:* Creates a sequence of the unique years during which the selected office recorded sales.

  ```
  for $i in distinct-values(Row[@Office=$XML1/root/@DesiredOffice]/@Year)
  return $i
  ```

- *Y-Axis:* For the selected office and for each of its unique years, sums up the sales (stored in its `@Licenses` attribute)

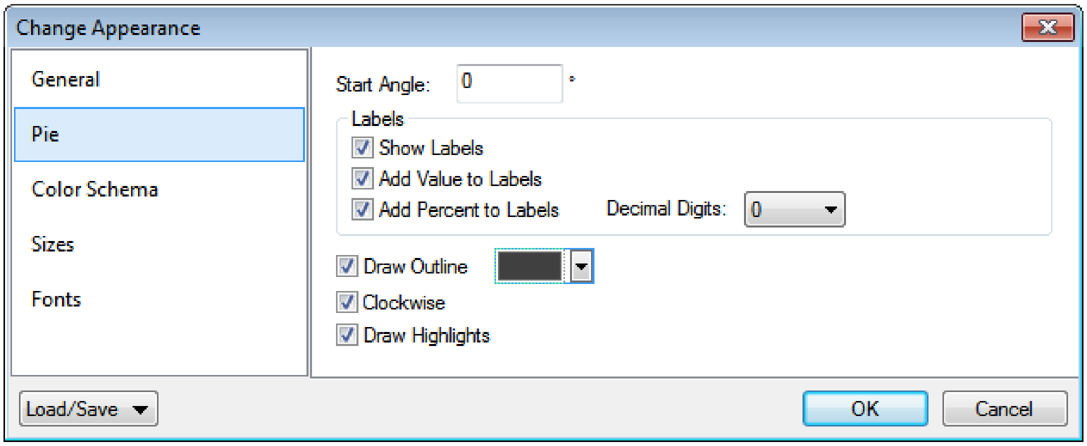
  ```
  for $i in distinct-values(Row[@Office=$XML1/root/@DesiredOffice]/@Year)
  ```

```
return sum(Row[@Office= $XML1/root/@DesiredOffice][@Year=$i]/@Licenses)
```

Additional definitions

Additionally, the following settings were made:

- In the Chart Configuration dialog, click **Dynamic XPath Settings** and set the title using an XPath expression. This enables the selected office to be displayed in the title.
- In the Chart Configuration dialog, click **All Settings**. In the Change Appearance dialog that appears, select *Pie* and select *Add Value to Labels* and *Add Percent to Labels*.



Start Angle: 0 °

Labels

☑ Show Labels

☑ Add Value to Labels

☑ Add Percent to Labels   Decimal Digits: 0

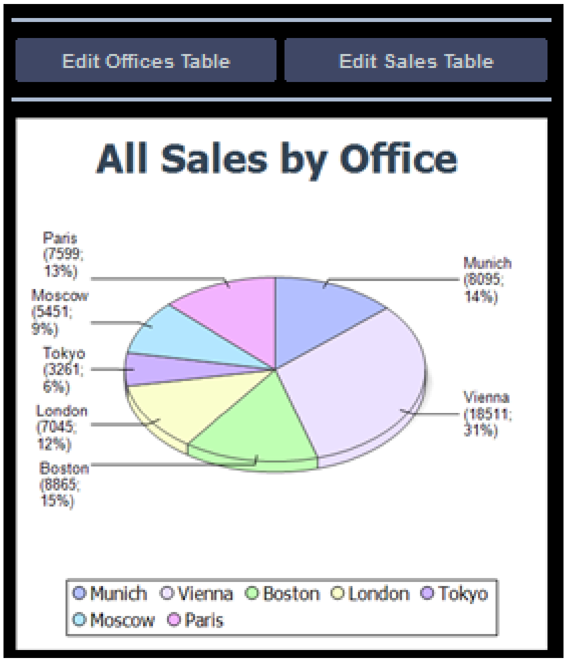
☑ Draw Outline

☑ Clockwise

☑ Draw Highlights

The second chart

The second chart is similar to the first, but is a 3D pie chart (*screenshot below*). It shows the sales of each office over all years as a part of total sales over all years.



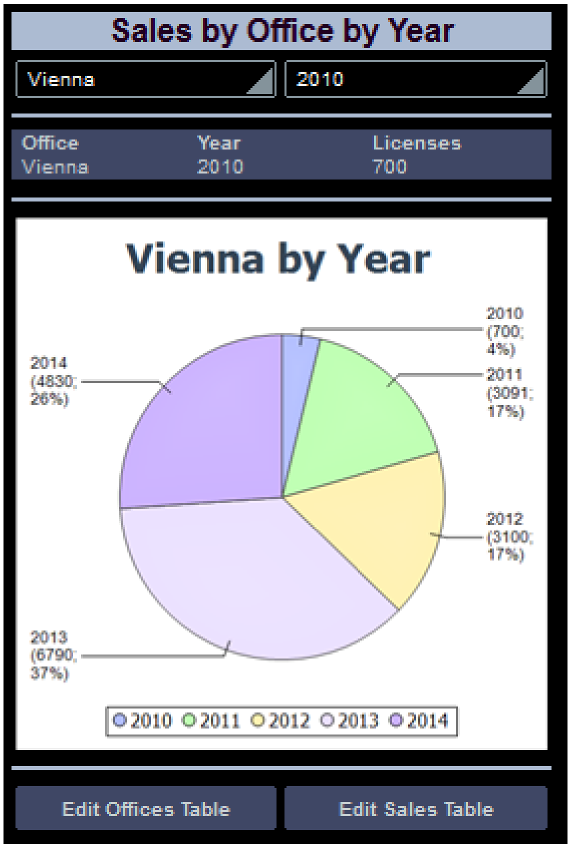
The XPath expressions are as follows:

- *X-Axis:* Creates a sequence of the city names of offices (not IDs), with city names being taken from **`$DB1`**.
  `for $i in distinct-values(Row/@Office) return $DB1/DB/RowSet/Row[@id=$i]/@City`

- *Y-Axis:* For the selected office, sums up the sales (stored in its `@Licenses` attribute)
  `for $i in distinct-values(Row/@Office) return sum(Row[@Office=$i]/@Licenses)`

Edit Offices Table

The Edit Offices table has been created on a separate top page. When the solution runs, this page is accessed from the main page (*screenshot below left*). Clicking the button **Edit Offices Table** loads the Edit Offices table (*screenshot below right*). The Offices table has seven rows, each of which has a non-editable ID column, an editable City column, and a Delete control (*see screenshot below right*). Additionally, there is an Append Row control below the last row, a **Submit** button in the *Edit Offices Table* bar, and a **Back** button to go back to the previous page (the main page in this case).

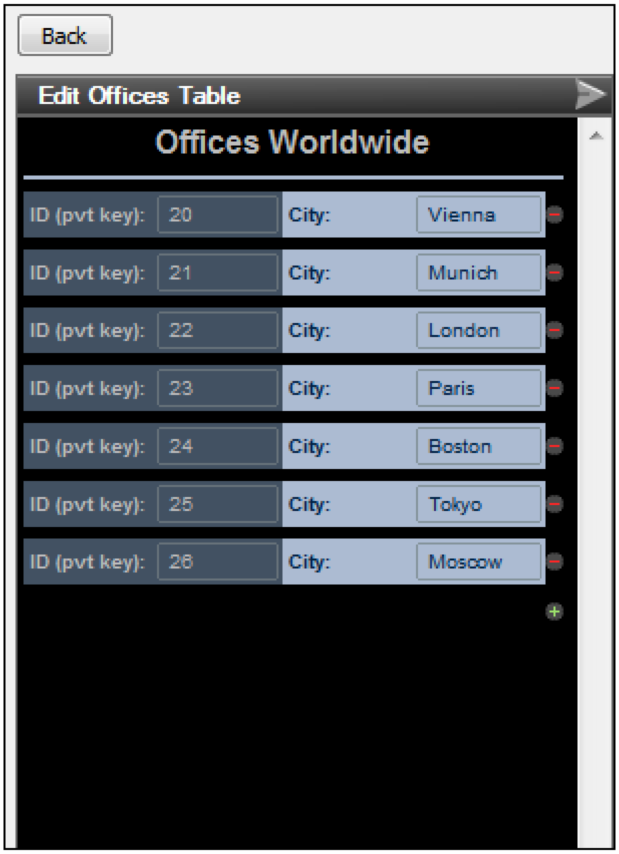


Sales by Office by Year

Vienna

2010

Office	Year	Licenses
Vienna	2010	700



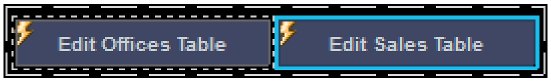
Back

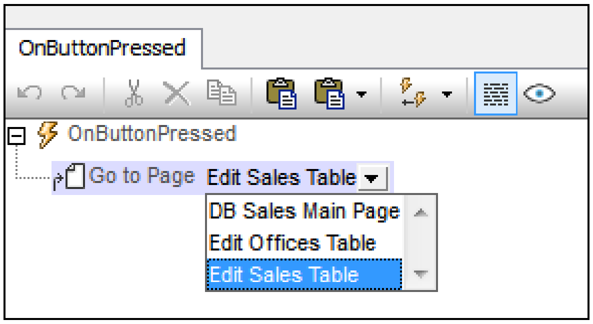
Edit Offices Table

Offices Worldwide

ID (pvt key):	20	City:	Vienna
ID (pvt key):	21	City:	Munich
ID (pvt key):	22	City:	London
ID (pvt key):	23	City:	Paris
ID (pvt key):	24	City:	Boston
ID (pvt key):	25	City:	Tokyo
ID (pvt key):	26	City:	Moscow

In the design, the **Edit** buttons (*first screenshot below*) have both been assigned the *Go to Page* action on their respective **OnButtonPressed** events (right-click the button and select **Control Actions for OnButtonPressed**). These *Go to Page* actions (*second screenshot below*) load the respective target pages.





OnButtonPressed

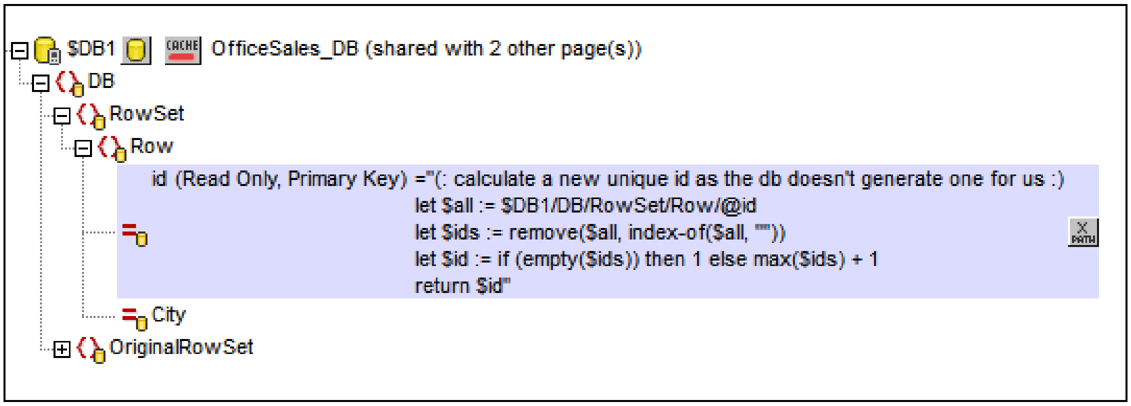
OnButtonPressed
Go to Page Edit Sales Table

DB Sales Main Page
Edit Offices Table
Edit Sales Table

Creating the editable Offices table

The Offices table in the DB has the structure displayed in the data tree of **$DB1** (*screenshot* *below*). Since the **@id** attribute is the primary key, it cannot be changed. This means that when a new record is appended, the end user cannot enter an @id value via the solution. The @id value must be generated automatically using an XQuery expression. The XQuery expression is inserted by using the context menu command, **Ensure Exists before Page Load (XPath Value)**:

```
let $all := $DB1/DB/RowSet/Row/@id
let $ids := remove($all, index-of($all, ""))
let $id := if (empty($ids)) then 1 else max($ids) + 1
return $id
```



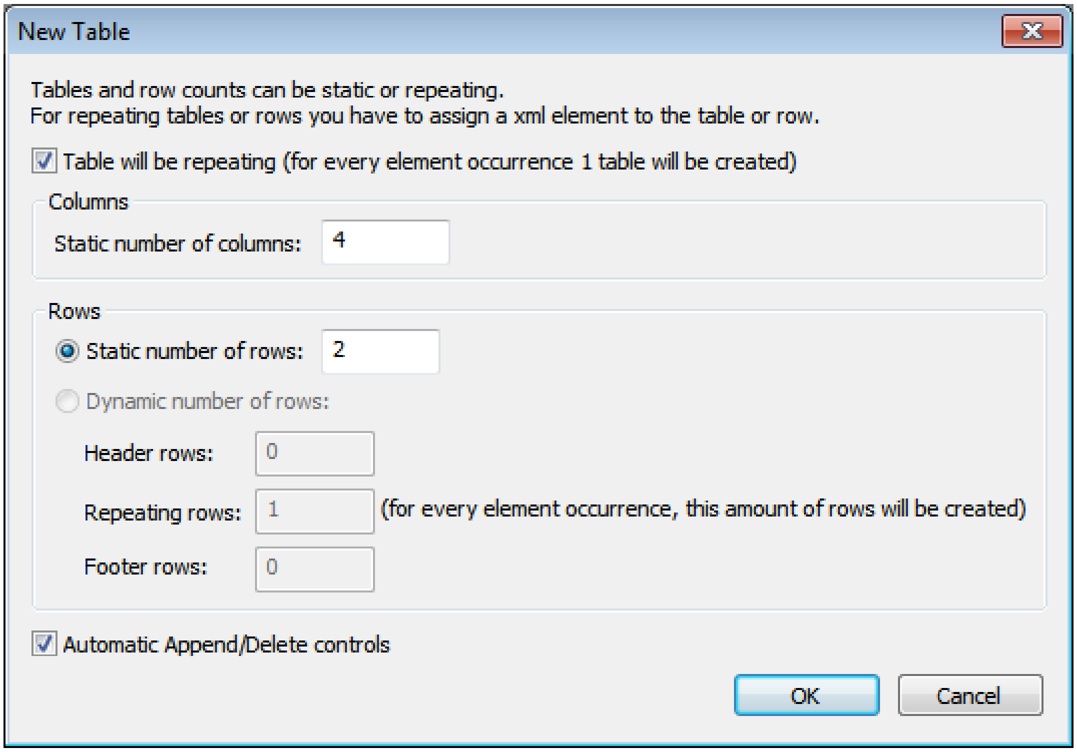
```
id (Read Only, Primary Key) ="(: calculate a new unique id as the db doesn't generate one for us :)
                               let $all := $DB1/DB/RowSet/Row/@id
                               let $ids := remove($all, index-of($all, ""))
                               let $id := if (empty($ids)) then 1 else max($ids) + 1
                               return $id"
```

In the design, we will do the following:

To do...	How
Display all (Office) rows	Add a repeating table, with the Office row as the repeating element
Include controls for deletion and addition of rows	When adding the table, enable the automatic inclusion of Delete/Append controls
Enable editing of @City values	Add an Edit Field control that has a page source link to @City
Save changes back to DB	Add a *Save* action to the page's OnSubmitButtonPressed event; Also, right-click $DB1 and toggle on **Create OriginalRowSet**
Go back to the main page	Add a *Go to Page* action to the page's OnBackButtonPressed event

⊟ *Add a repeating table that has Append/Delete controls*

On dragging the table control from the Controls Pane and dropping it in the design, the New table dialog appears (*screenshot below*).



New Table

Tables and row counts can be static or repeating.
For repeating tables or rows you have to assign a xml element to the table or row.

☑ Table will be repeating (for every element occurrence 1 table will be created)

Columns

Static number of columns: 4

Rows

● Static number of rows: 2

○ Dynamic number of rows:

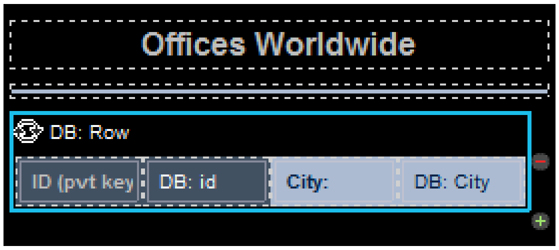
Header rows: 0

Repeating rows: 1 (for every element occurrence, this amount of rows will be created)

Footer rows: 0

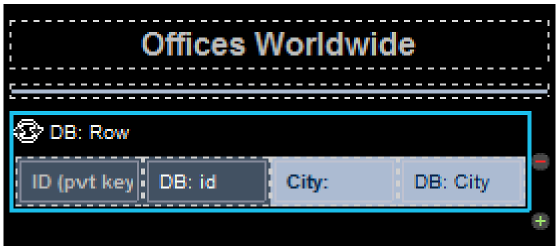
☑ Automatic Append/Delete controls

Specify that the table will be repeating, enter the number of columns (4) and rows (1), select the *Automatic Append/Delete Controls* check box, and click **OK**. Labels are added to the first three cells of the row, as shown in the screenshot below. A page source link to the @id node of $DB1 is created for the second label (DB:id).



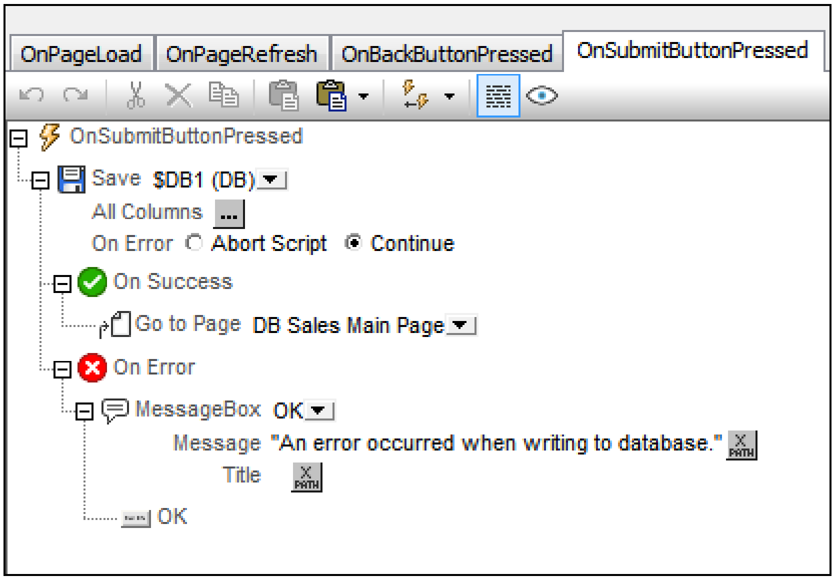
⊟ *Enable editing of @City in $DB1*

An edit field control is added to the fourth cell and a page source link to the @City node of $DB1 is created for it (DB:City). We use an edit field control in this cell because we want the end user to be able edit @City values; all the other cells have label controls.



⊟ *Page actions: 'Save' and 'Go to page'*

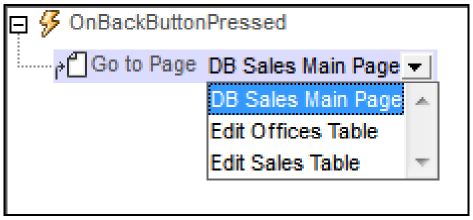
Click **Page | Page Actions** to open the Page Actions dialog (*screenshot below*).



OnSubmitButtonPressed
Save $DB1 (DB)
All Columns ...
On Error ○ Abort Script ● Continue
On Success
Go to Page DB Sales Main Page
On Error
MessageBox OK
Message "An error occurred when writing to database."
Title
OK

Actions are defined for the following events:

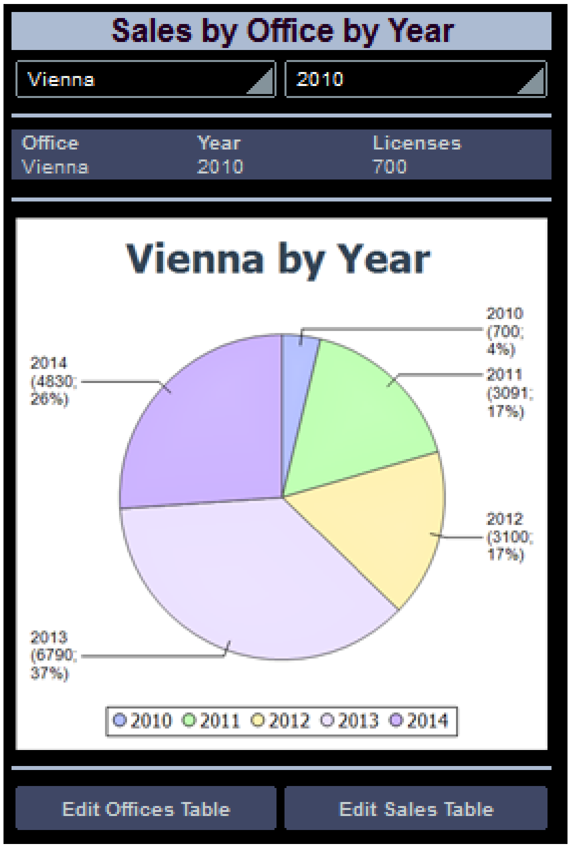
- **`OnSubmitButtonPressed`**: Saves all columns of the page to the DB (`$DB1`) and goes back to the main page.
- **`OnBackButtonPressed`**: Goes back to the main page.


OnBackButtonPressed
Go to Page DB Sales Main Page

The tree of the data source must also include an `OriginalRowSet` element, which is a copy of the `RowSet` element. Original data is saved in the `OriginalRowSet` element, so that the columns of the `RowSet` element can be edited. The `OriginalRowSet` element is updated with the new value only when the data is saved back to the DB.

Edit Sales Table

The Edit Sales table, like the Edit Offices Table, has been created on a separate top page. When the solution runs, this page is accessed from the main page (*screenshot below left*). Clicking the button **Edit Sales Table** loads the Edit Sales table (*screenshot below right*). The Sales table has multiple rows, each of which has a non-editable (sales item) ID column, editable Office, Month, Year, and Licenses columns, and a Delete control (*see screenshot below right*). Additionally, there is an Append Row control below the last row, a **Submit** button in the *Edit Sales Table* bar, and a **Back** button to go back to the previous page (the main page in this case).

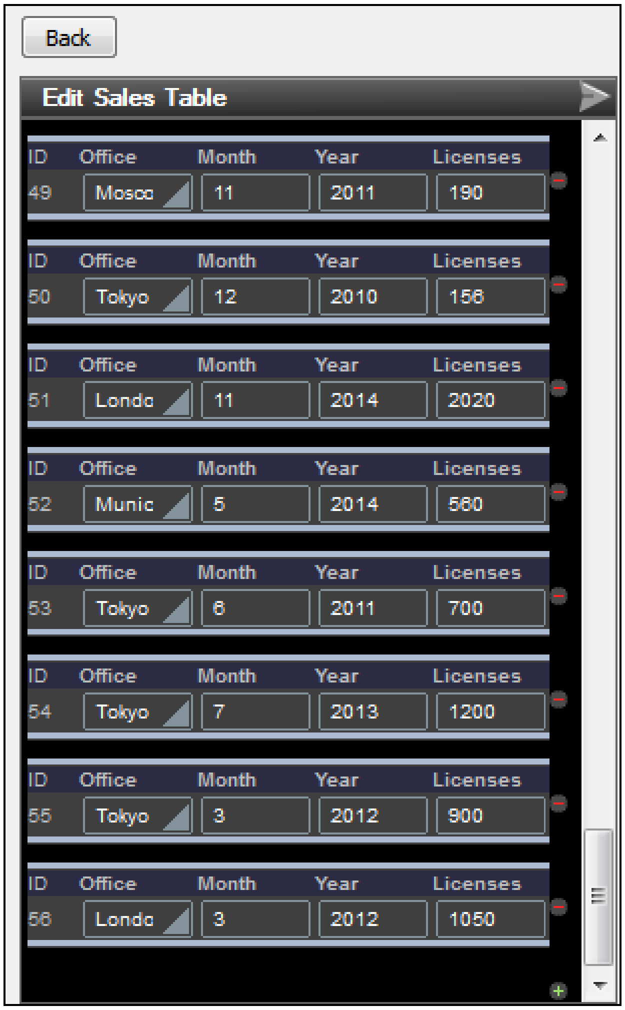


Sales by Office by Year

Vienna 2010

Office	Year	Licenses
Vienna	2010	700

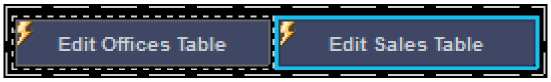
Vienna by Year

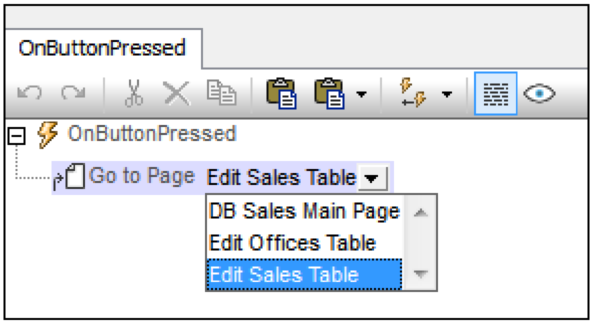


Edit Sales Table

ID	Office	Month	Year	Licenses
49	Mosco	11	2011	190
50	Tokyo	12	2010	156
51	Londo	11	2014	2020
52	Munic	5	2014	560
53	Tokyo	6	2011	700
54	Tokyo	7	2013	1200
55	Tokyo	3	2012	900
56	Londo	3	2012	1050

In the design, the **Edit** buttons (*first screenshot below*) have both been assigned the *Go to Page* action on their respective **`OnButtonPressed`** events (right-click the button and select **Control Actions for OnButtonPressed**). These *Go to Page* actions (*second screenshot below*) load the respective target pages.



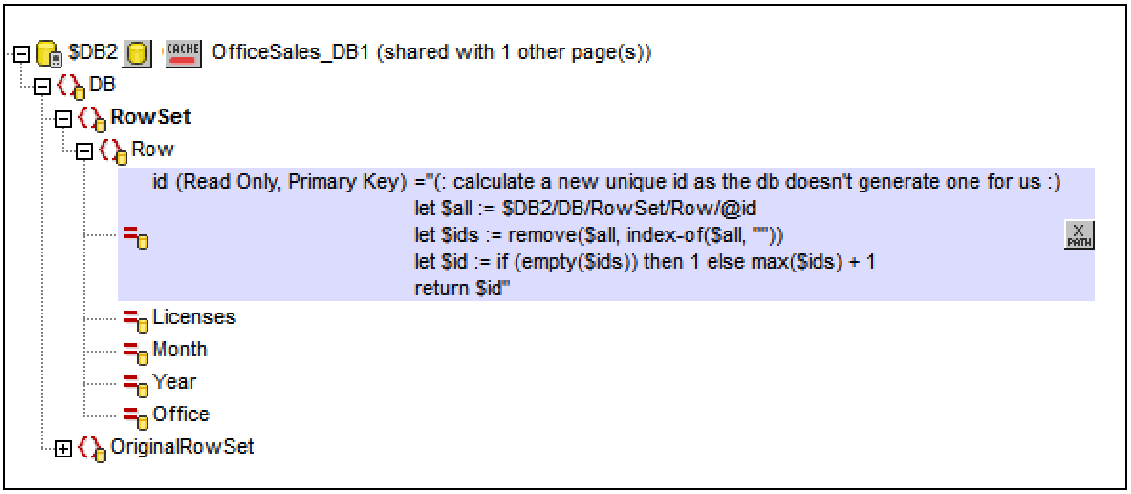


Go to Page Edit Sales Table

Creating the editable Sales table

The Sales table in the DB has the structure displayed in the data tree of **`$DB2`** (*screenshot below*). Since the **`@id`** attribute is the primary key, it cannot be changed. This means that when a new record is appended, the end user cannot enter an `@id` value via the solution. The `@id` value must be generated automatically using an XQuery expression. The XQuery expression is inserted by using the context menu command, **Ensure Exists before Page Load (XPath Value)**:

```
let $all := $DB2/DB/RowSet/Row/@id
let $ids := remove($all, index-of($all, ""))
let $id := if (empty($ids)) then 1 else max($ids) + 1
return $id
```



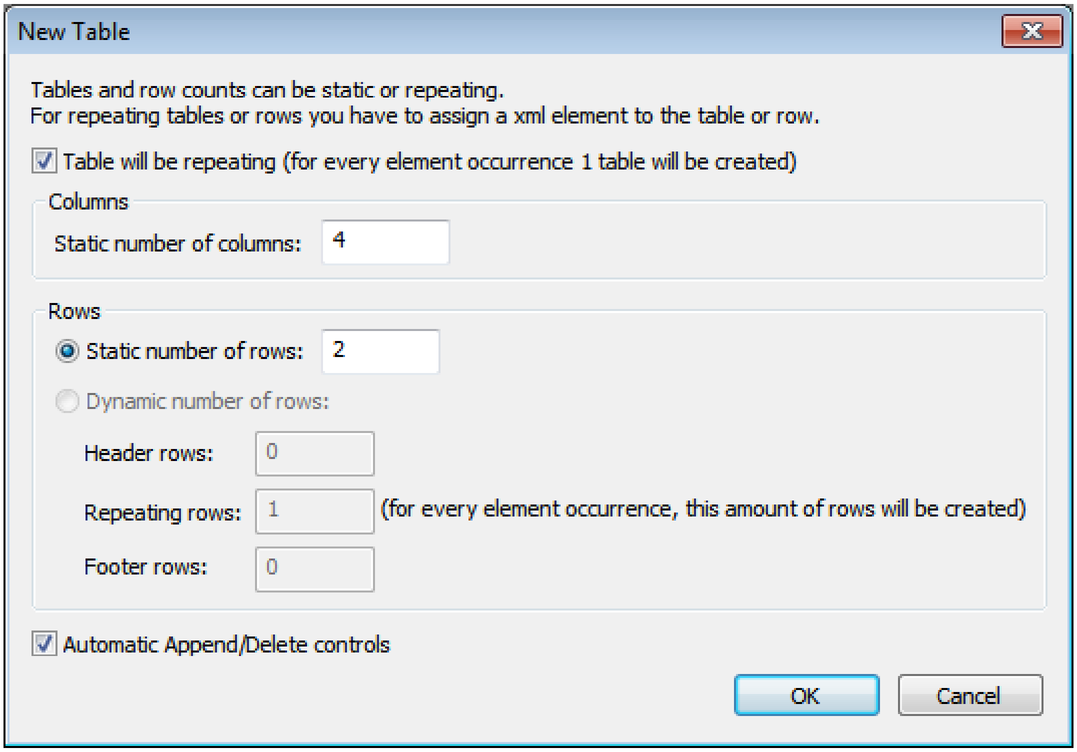
```
$DB2 OfficeSales_DB1 (shared with 1 other page(s))
  DB
    RowSet
      Row
        id (Read Only, Primary Key) ="(: calculate a new unique id as the db doesn't generate one for us :)
                                     let $all := $DB2/DB/RowSet/Row/@id
                                     let $ids := remove($all, index-of($all, ""))
                                     let $id := if (empty($ids)) then 1 else max($ids) + 1
                                     return $id"
        Licenses
        Month
        Year
        Office
    OriginalRowSet
```

In the design, we will do the following:

To do...	How
Display all (`Sales`) rows	Add a repeating table, with the `Sales` row as the repeating element
Include controls for deletion and addition of rows	When adding the table, enable the automatic inclusion of Delete/Append controls
Enable editing of editable values	Add a combo box and edit field controls that have page source links
Save changes back to DB	Add a *Save* action to the page's `OnSubmitButtonPressed` event; Also, right-click `$DB2` and toggle on **Create OriginalRowSet**
Go back to the main page	Add a *Go to Page* action to the page's `OnBackButtonPressed` event

⊟ *Add a repeating table that has Append/Delete controls*

On dragging the table control from the Controls Pane and dropping it in the design, the New table dialog appears (*screenshot below*).



New Table

Tables and row counts can be static or repeating.
For repeating tables or rows you have to assign a xml element to the table or row.

☑ Table will be repeating (for every element occurrence 1 table will be created)

Columns

Static number of columns: 4

Rows

● Static number of rows: 2

○ Dynamic number of rows:

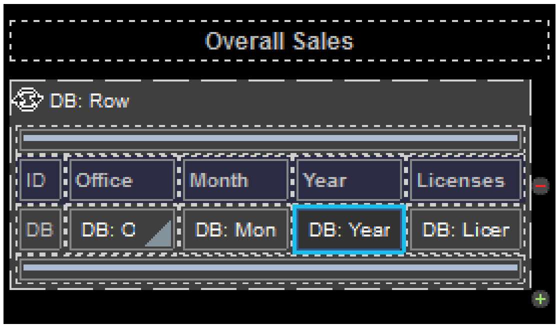
Header rows: 0

Repeating rows: 1 (for every element occurrence, this amount of rows will be created)

Footer rows: 0

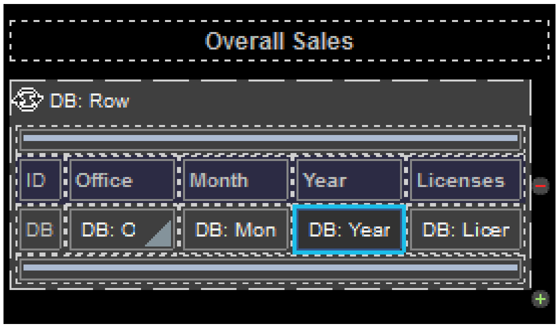
☑ Automatic Append/Delete controls

Specify that the table will be repeating, enter the number of columns (5) and rows (2), select the *Automatic Append/Delete Controls* check box, and click **OK**. Labels are added for headers to the cells of the first row. A label is added for the uneditable `@id` value to the first cell of the second row. A page source link to the `@id` node of `$DB2` is created on this label (`DB:id`).



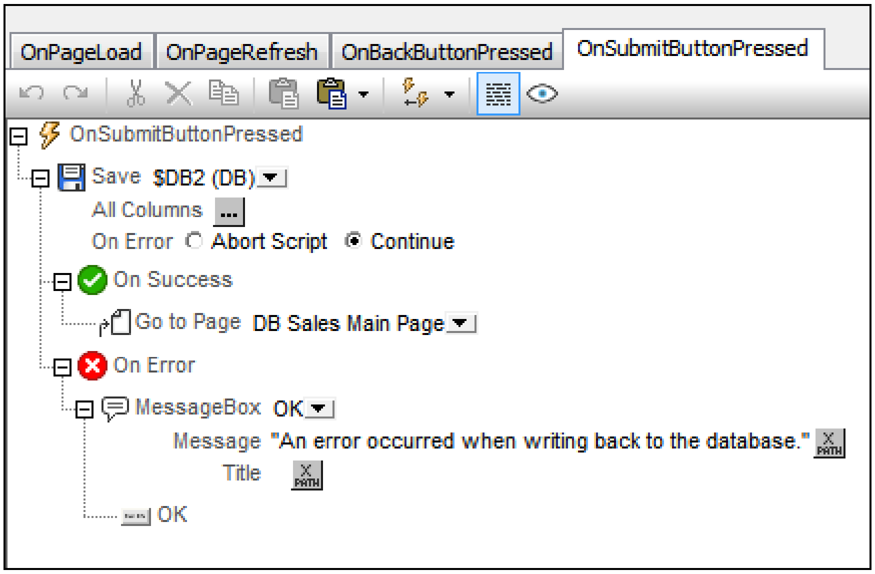
⊟ *Enable editing of the editbale nodes*

A combo box is added for the office (with a page source link to `@Office`), and edit fields are added for the month, year, and licenses cells, with page source links to the respective nodes.



⊟ *Page actions: 'Save' and 'Go to page'*

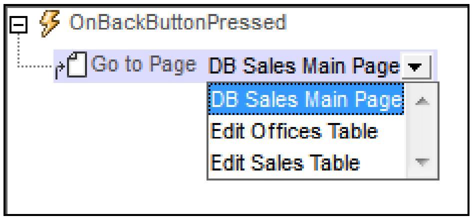
Click **Page | Page Actions** to open the Page Actions dialog (*screenshot below*).



OnSubmitButtonPressed
Save $DB2 (DB)
All Columns ...
On Error ○ Abort Script ● Continue
On Success
Go to Page DB Sales Main Page
On Error
MessageBox OK
Message "An error occurred when writing back to the database."
Title
OK

Actions are defined for the following events:

- **`OnSubmitButtonPressed`**: Saves all columns of the page to the DB (`$DB1`) and goes back to the main page.
- **`OnBackButtonPressed`**: Goes back to the main page.


OnBackButtonPressed
Go to Page DB Sales Main Page
DB Sales Main Page
Edit Offices Table
Edit Sales Table

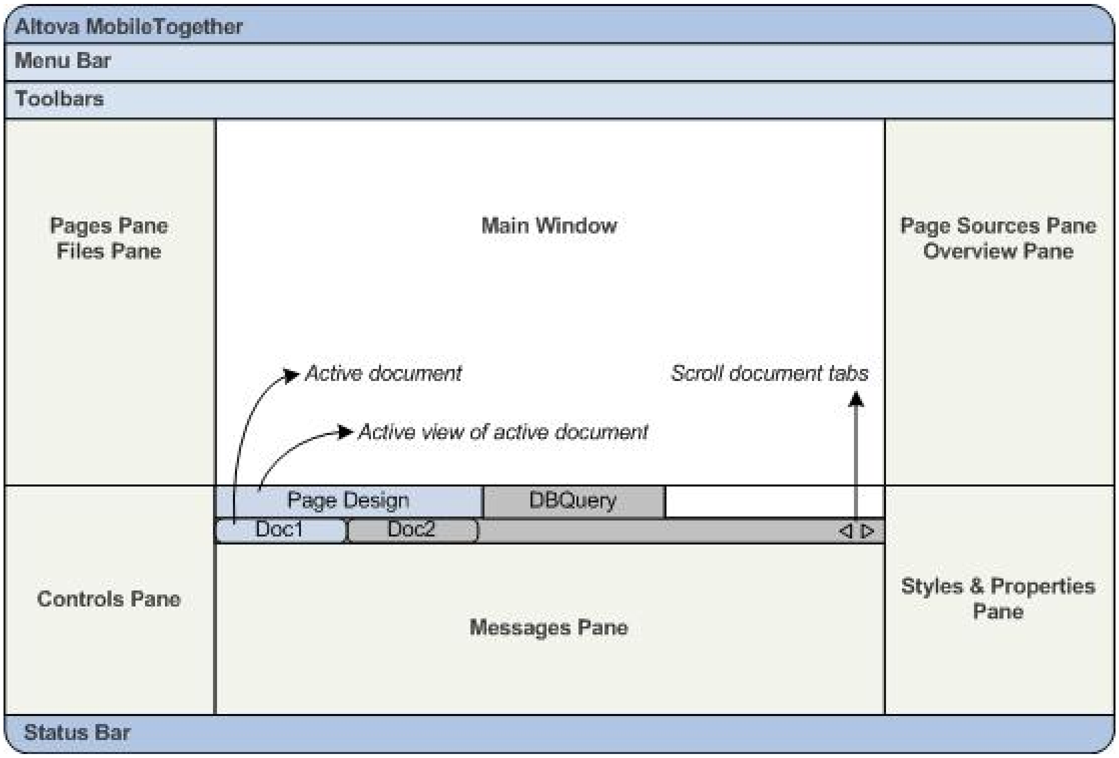
The tree of the data source must also include an `OriginalRowSet` element, which is a copy of the `RowSet` element. Original data is saved in the `OriginalRowSet` element, so that the columns of the `RowSet` element can be edited. The `OriginalRowSet` element is updated with the new value only when the data is saved back to the DB.

Chapter 4

The User Interface

4 The User Interface

The Graphical User Interface (GUI) consists of a Main Window (consisting of the Page Design and DB Query View tabs) and several panes (*see screenshot below*). By default, the panes are located around the Main Window, but can be moved around the GUI, minimized, or hidden.



The panes, which are listed below, are described in the sub-sections of this section:

- Pages Pane
- Files Pane
- Controls Pane
- Page Sources Pane
- Overview Pane
- Styles & Properties Pane
- Messages Pane

Showing/hiding panes
A pane can be displayed or hidden by toggling it, respectively, on or off in the **View** menu. A displayed pane can also be hidden by right-clicking its title bar of the displayed pane and selecting the command **Hide**.

Floating and docking the panes

An individual pane 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.

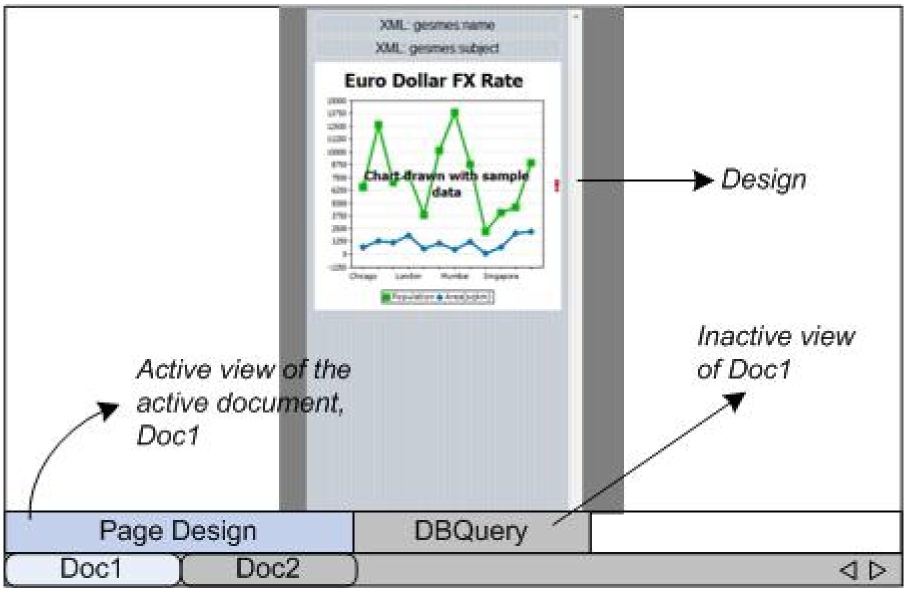
Auto-hiding panes

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

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

4.1 Main Window

The Main Window (*screenshot below*) is where you design the pages of the MobileTogether Designer project and directly query a database to preview table data. It consists of two views, only one of which can be active at a time: Page Design and DB Query.



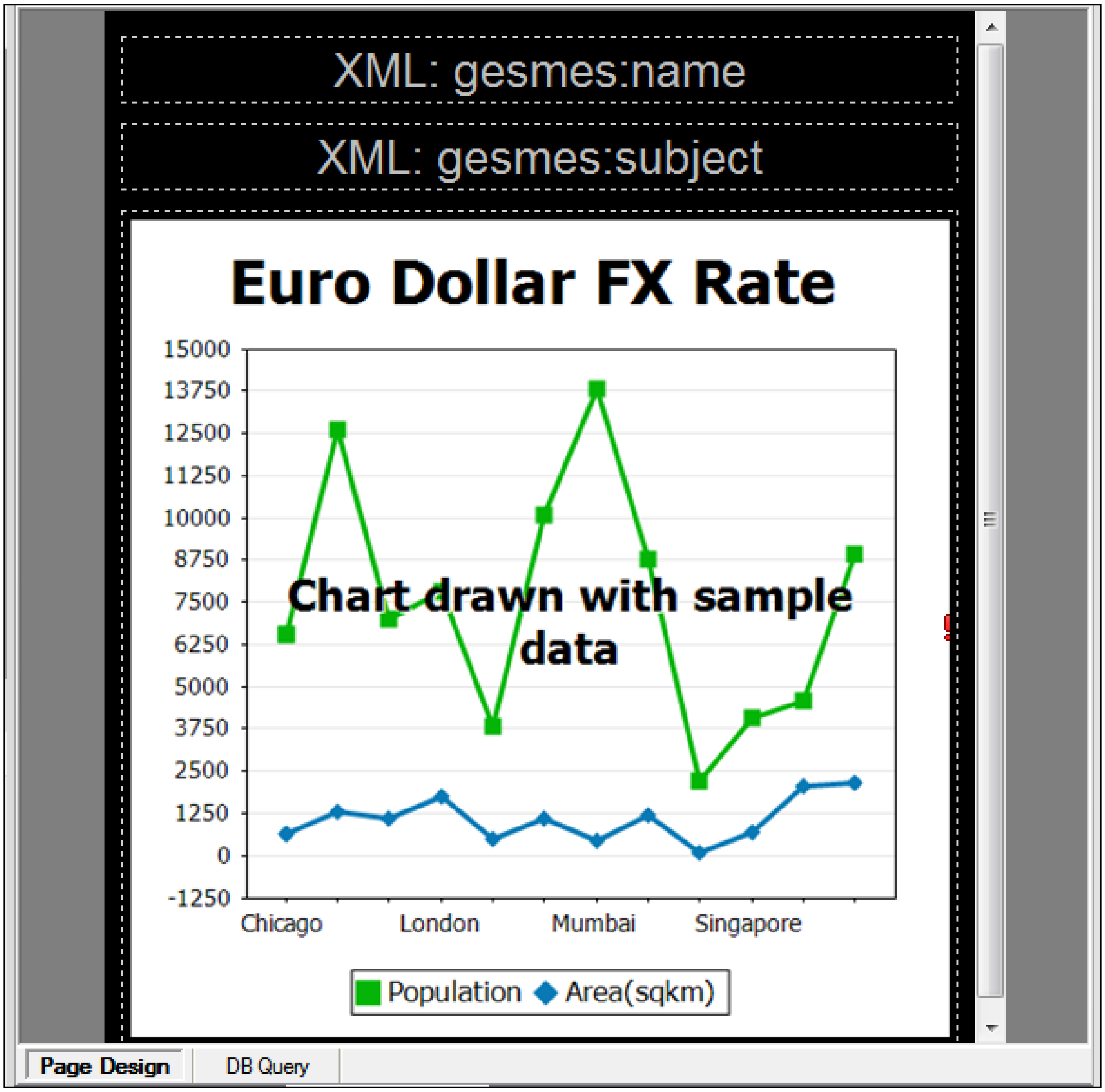
MobileTogether Design (.mtd) files in the Main Window

Note the following points:

- Any number of MobileTogether Design (`*.mtd`) files can be open simultaneously. You can switch among the open documents and edit them.
- 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.
- You can activate open files in the sequence in which they were opened by using **Ctrl+Tab** or **Ctrl+F6**.
- Right-clicking a file tab opens a context-menu that contains a selection of **File** commands, such as **Print** and **Close**.
- Placing the mouse cursor over components in the Main Window displays a popup containing further information about the function of that component.

Page Design

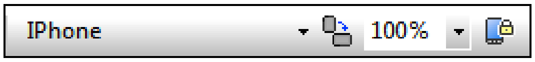
The **Page Design View** (**Page View** for short) is the view in which the page that is going to be deployed to the mobile device is designed (*see screenshot below*).



To design a page, drag-and-drop controls from the Controls Pane into the design, and then specify, in the Properties Pane, the settings of that control. Controls can be positioned anywhere on the page. As you drag a control over the page, possible drop positions are indicated with an arrow. The settings of a control can be edited at any later time by selecting the control in the design and editing its settings in the Properties Pane. Delete a control in the design by selecting it and pressing **Delete**.

Page View settings

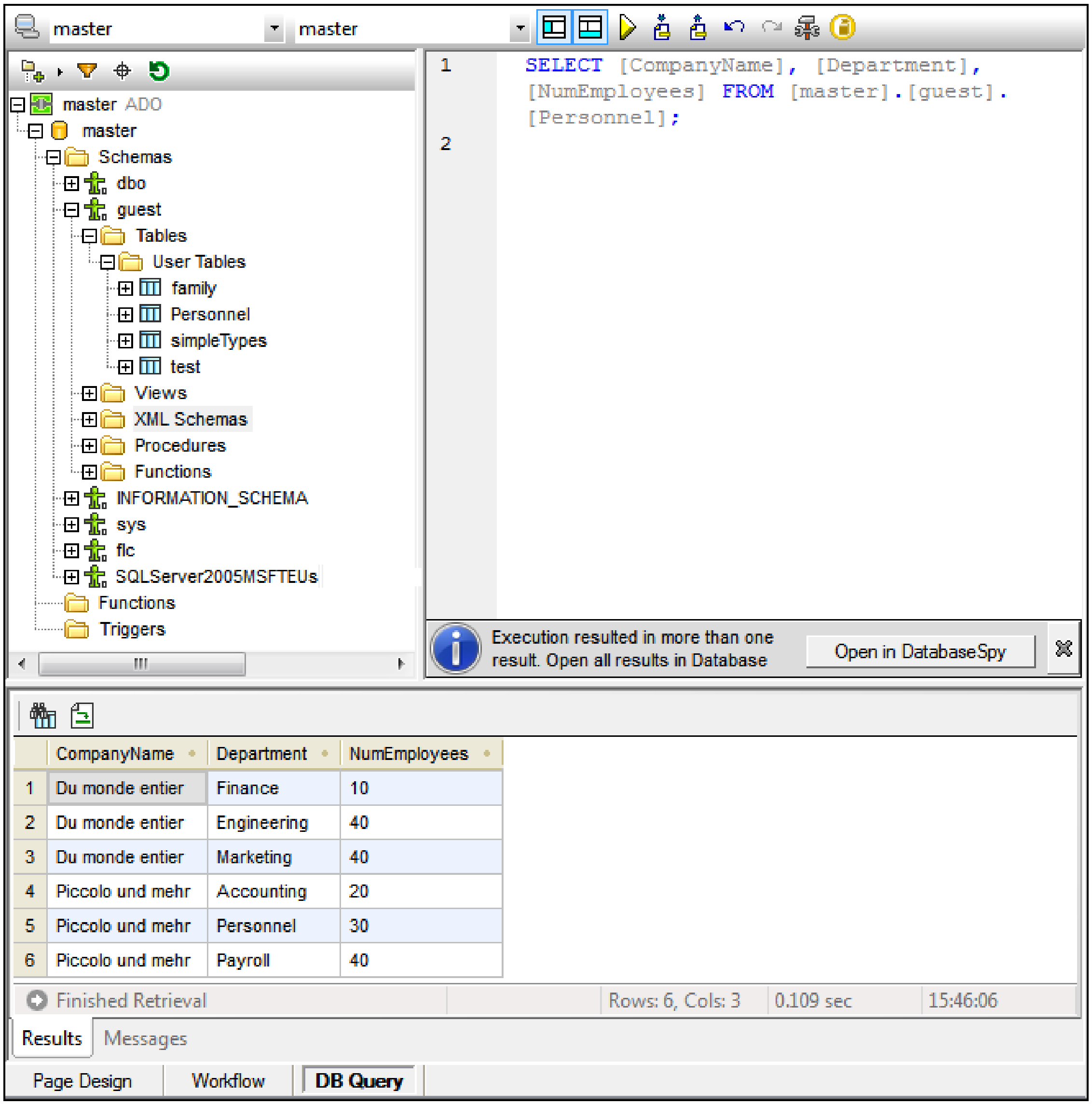
The screenshot below shows the Page View settings in the Main toolbar.



- *Device Selector:* This combo box allows you to select various mobile devices, so that the design can be previewed for specific devices.
- *Portrait/Landscape Preview Toggle:* Toggles the design preview between portrait and landscape.
- *Zoom level:* A combo box to select the zoom level in 10% steps from 10% to 100%. The zoom level can also be modified via the **View** menu.
- *Lock all page views to same device and zoom level:* The page views of all open documents will be locked to the currently selected device and zoom level.

DB Query

The **DB Query View** (*screenshot below*) enables you to directly query any major database from within the MobileTogether Designer GUI. The database could be a data source referenced in the active document or an external database. Note that each DB Query pane is associated with the currently active design. You can have connections to multiple databases for a single design. There can also be multiple designs open in MobileTogether Designer. Queries and actions defined in the DB Query View are independent of other MobileTogether Designer tabs, and are not saved as part of the **`.mtd`** design file.



```
SELECT [CompanyName], [Department],
[NumEmployees] FROM [master].[guest].
[Personnel];
```

Execution resulted in more than one result. Open all results in Database

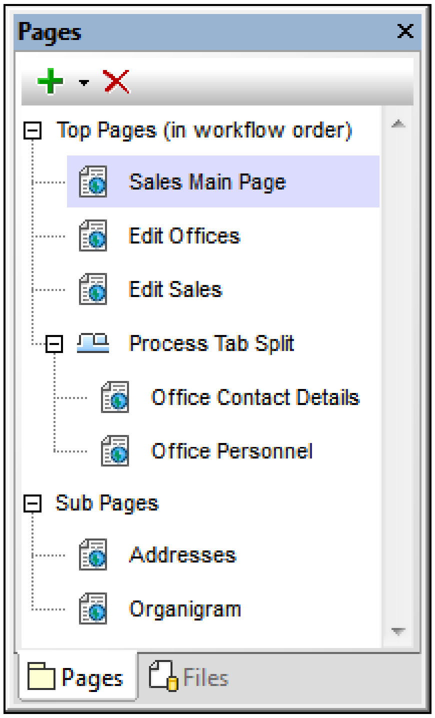
	CompanyName	Department	NumEmployees
1	Du monde entier	Finance	10
2	Du monde entier	Engineering	40
3	Du monde entier	Marketing	40
4	Piccolo und mehr	Accounting	20
5	Piccolo und mehr	Personnel	30
6	Piccolo und mehr	Payroll	40

Finished Retrieval Rows: 6, Cols: 3 0.109 sec 15:46:06

See the section, Database Query, for a detailed description.

4.2 Pages Pane

The **Pages Pane** enables you to add new pages to a project and displays a tree of all the pages in the project (*see screenshot below*). To see the default location of the Pages Pane, go to The Graphical User Interface (GUI).



To display a page, click its entry in the Pages Pane.

Top pages, sub pages, and tabbed pages
There are three categories of project page:

- *Top pages:* A top page is part of the sequence of pages that makes up a workflow. When the solution is started, it progresses from the first page to the last, in the order listed in the Pages Pane. You can change the position of a page in the list by dragging the page to a new position. Tabbed pages are also part of the workflow sequence, but sub pages are not.
- *Sub pages:* A sub page is not part of the sequence of pages that make up the workflow of the project. It is similar to a module that is called by a control in a top page (or tabbed page). For example, an `OnButtonPressed` event of a Button control in a top page could use the `GoToSubpage` action to go to a particular subpage, and then return to the top page (or go to another page).
- *Tabbed pages (or tab splits):* A tabbed page (or tab split) is a page with tabs, each of which, contains a page. For example, in the screenshot above, the tabbed page (*Process Tab Split*) is defined to have three tabs that contain, respectively, the pages *Details*, *Person*, and *Fonts*. Tabbed pages are part of the page sequence that makes up the project's workflow.

Adding, renaming, and deleting pages

Besides the methods listed in the following table, you can also use context menu commands for some of these tasks (*see further below*).

To...	Do this...
Add a page	Click the **Add Page** icon in the pane's toolbar. From the dropdown menu, select **Add Top Page**, **Add Tab Split**, or **Add Sub Page**. A new page is added to the Pages Pane with a name of *New Page X* or *Process Tab Split*, and the empty new page is displayed in the Main Window.
Rename a page	Double-click the name in the Pages Pane and edit the name.
Delete a page	Select the page you want to delete and click the **Delete** icon in the pane's toolbar, or press **Delete**.

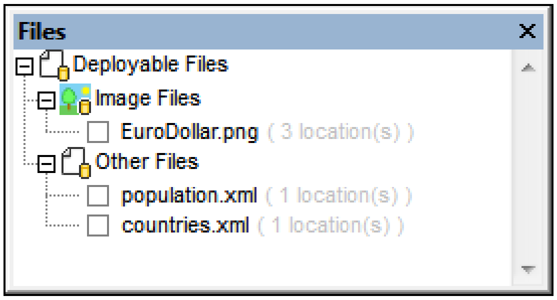
Context menu

The context menu of items in the Pages Pane enables you to insert pages before the selected item (**Insert**), or to append pages (**Add**) to the sequence of top pages and to the list of sub pages. Child pages can only be added to tabbed pages via the context menu of the tabbed page item.

To...	Do this...
Add a page before a top page, sub page, tabbed page, or child page of a tabbed page	Right-click the page and select **Insert Page**
Add a top page at the end of the *Top Pages* list	Right-click any item and select **Add Top Page**
Add a sub page at the end of the *Sub Pages* list	Right-click any item and select **Add Sub Page**
Add a child page at the end of a list of child pages of a tabbed page	Right-click the tabbed page and select **Add Page as Child**

4.3 Files Pane

The **Files Pane** (*screenshot below*) lists the files of the project that can be deployed to the server. Some files, such as the default files of data sources, are added automatically to the list of deployable files when these files are first associated with the project. But you can also add files directly to the list of deployable files via the context menu (obtained by right-clicking inside the Files Pane). These deployable files can be image files and other files, such as XML files, and they are organized in the pane under two headings: *Image Files* and *Other Files* (*see screenshot below*).


Files
Deployable Files
Image Files
□ EuroDollar.png (3 location(s))
Other Files
□ population.xml (1 location(s))
□ countries.xml (1 location(s))

Each file has a check box next to it. To deploy a file, select its check box. To not deploy a file, de-select its check box.

The following information and actions are available in the Files Pane:

- Add or remove files via commands in the pane's context menu (*see below*).
- The total number of times the file is used across all the pages of the project is displayed next to the file name.
- Check the box in front of each file name to deploy the file to MobileTogether Server when the project is deployed to the server. Uncheck the box if you do not want to deploy the file.

About deployed files
Deployed files are read-only files that are stored on the server. Deploying is ideal for default XML files, image files, and other data files that will be used to only read data. If the data file is to be written to and stored on the server, do not deploy it, but store it on the server at a location that the MTD file correctly references. (See the section Location of Project Files for details.) Files to be deployed are transferred to the server at the time when the solution is deployed. (See the section Project Deployment for details.)

+ Deploy only if the file is used as a read-only file.
+ Deploy if, for some reason, the file's URL will not be accessible to clients.
+ Deploy if you want the file in the same unchanged state when accessed by clients.
+ Deploy if file loading needs to be faster.
- Do not deploy if the files needs to be written to.
- Do not deploy if the large size of files on the server is an issue.
- Do not deploy if the design is to be sent to others; in this case, consider embedding data files in the design.

Adding and removing deployable files from within the Files Pane

To...	Do this...
Add a file	Right-click in the Files Pane and, from the context menu that appears, click **Add File**, and then browse for the file you want. The added file appears in the *Image Files* or *Other Files* list, according to what file type it is.
Remove a file	Right-click the file you want to remove and, from the context menu that appears, click **Remove File**.

Adding deployable files to the project from outside the Files Pane

Files can be deployed from from outside the Files Pane is the following ways:

- When a file is added as a page source or when an image file is added to a page design, a dialog is displayed that asks whether the file should be deployed to MobileTogether Server. If you click **Yes**, the file is added to the Files Pane with its check box selected. If you click **No**, the file is added to the Files Pane with its check box unselected. You can select/unselect a file's check box in the Files Pane (*see screenshot above*) at any later time.
- In the context menu of the root node of a data source in the Page Sources Pane.

⊟ *Also see*

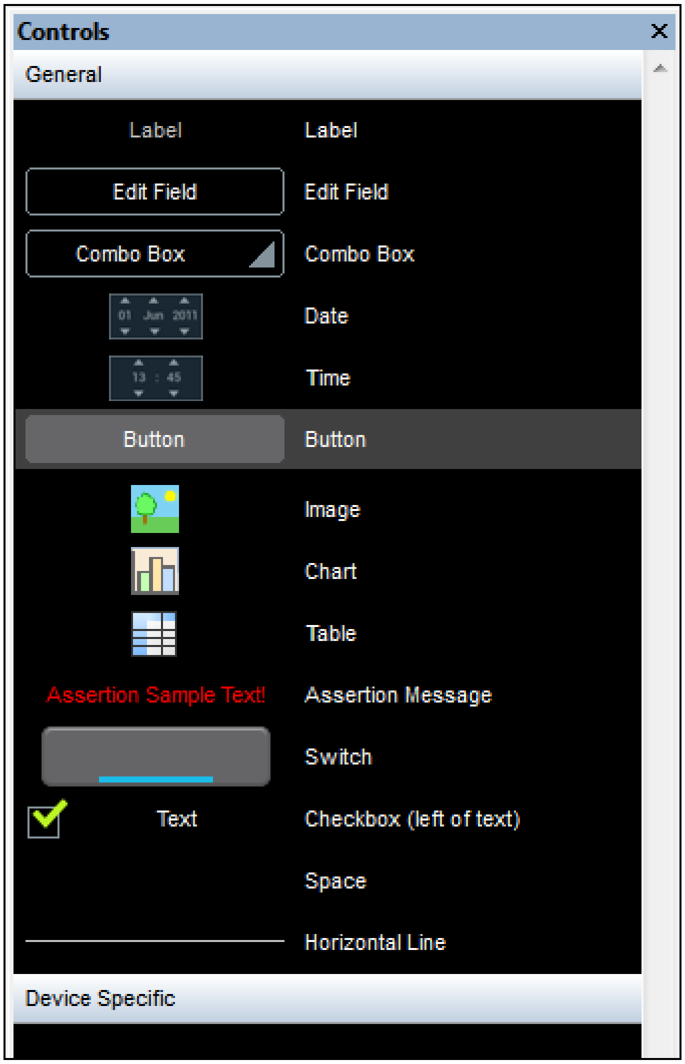
Deploy to MobileTogether Server
Location of Project Files
Deploying the Project
Data Storage on Servers.

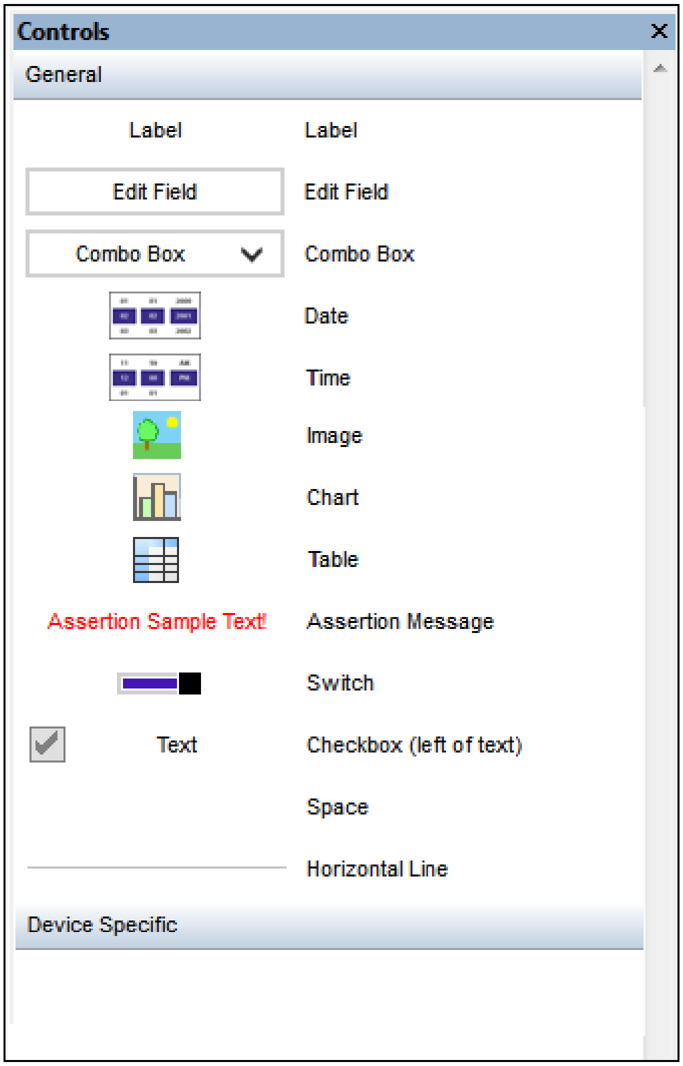
4.4 Controls Pane

The **Controls Pane** shows all the controls that can be added to a page design or to a workflow. To see the default location of the Controls Pane, go to The Graphical User Interface (GUI).

Page design controls

Page design controls are available when the Page Design tab is selected in the Main Window. The appearance of the Controls Pane and the controls corresponds to the currently selected device. For example, the Controls Panes shown in the screenshots below are for Android Galaxy 5 (*left*) and Windows applications (*right*).

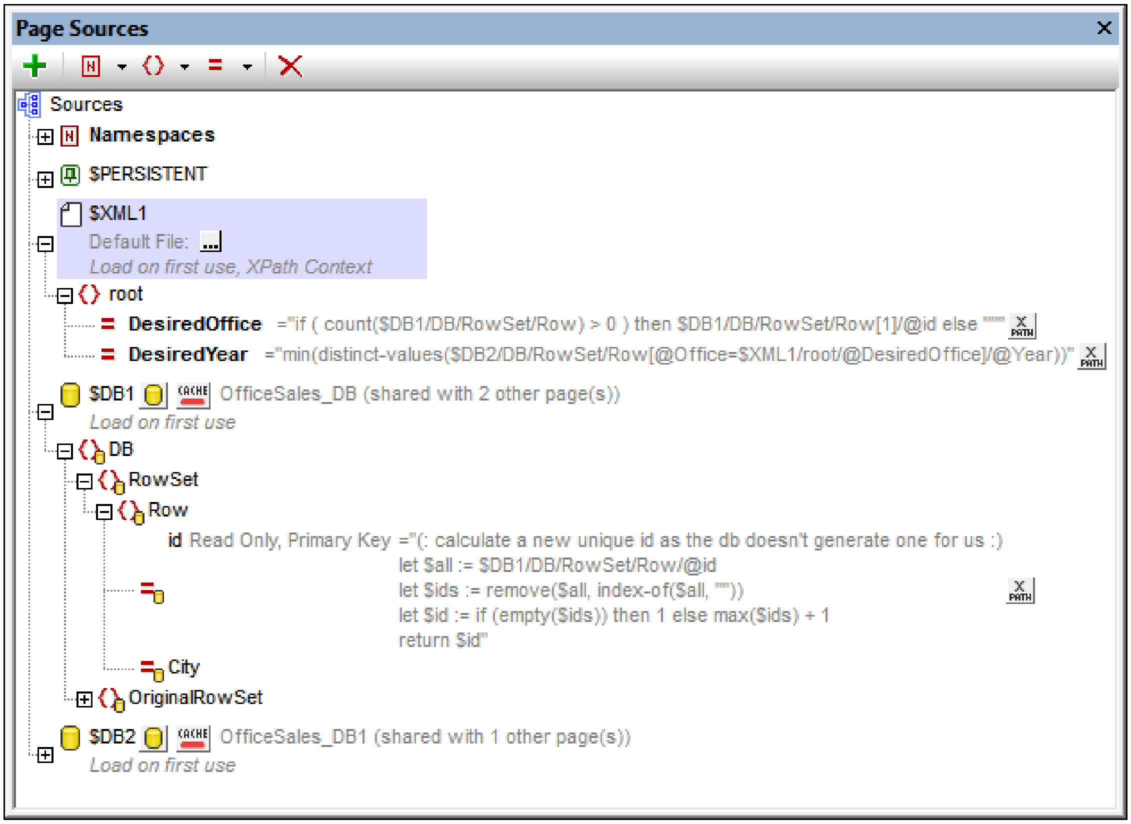




The pane's controls are organized into a *General* section and a *Device-Specific* section. To add a control to a page design, drag and drop the control onto the desired location in the page design.

4.5 Page Sources Pane

The **Page Sources Pane** (*screenshot below*) is where page data sources are managed. These data sources provide both (i) the structure of data trees used in the design, as well as (ii) the data held in the data trees.



```
Sources
  Namespaces
  $PERSISTENT
  $XML1
    Default File: ...
    Load on first use, XPath Context
    root
      DesiredOffice ="if ( count($DB1/DB/RowSet/Row) > 0 ) then $DB1/DB/RowSet/Row[1]/@id else """
      DesiredYear ="min(distinct-values($DB2/DB/RowSet/Row[@Office=$XML1/root/@DesiredOffice]/@Year))"
  $DB1  OfficeSales_DB (shared with 2 other page(s))
    Load on first use
    DB
      RowSet
        Row
          id Read Only, Primary Key ="(: calculate a new unique id as the db doesn't generate one for us :)
                                     let $all := $DB1/DB/RowSet/Row/@id
                                     let $ids := remove($all, index-of($all, ""))
                                     let $id := if (empty($ids)) then 1 else max($ids) + 1
                                     return $id"
          City
      OriginalRowSet
  $DB2  OfficeSales_DB1 (shared with 1 other page(s))
    Load on first use
```

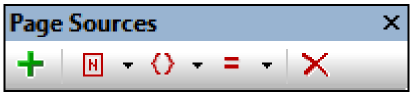
The pane provides the following main functionality:

- Displays all the data sources of the page currently selected in the Pages Pane
- Displays all the namespaces declared for the active project
- Enables data sources to be added to the page, via the toolbar's **Add Source** icon
- Enables namespaces to be added to the project, via the toolbar's **Add Namespace** icon
- Enables the default data file of a data source to be set. The data file provides the data that goes into the nodes of the data source
- Enables elements and attributes to be added to a tree, relative to the selected node
- Enables the default XPath context node to be set (*the framed node in the screenshot above*). The selected node will be the context node for all XPath expressions defined for that page
- Enables nodes to be associated with controls in the page design. This is done by dragging the node onto the control
- Enables items in the pane (namespaces, trees, elements, and attributes) to be deleted

For detailed information about how to manage and work with page sources, see the section, Page Data Sources.

Manually creating a tree structure

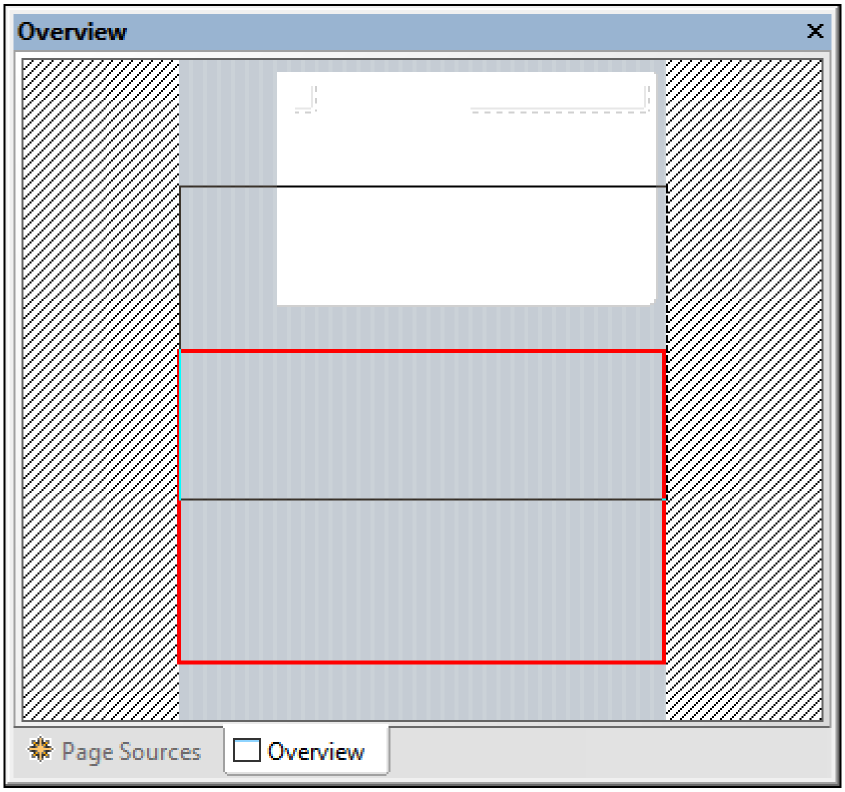
Elements and attributes can be added relative to any node in a tree structure (including the root node), and they can be deleted. Select a node in a data source, and click the appropriate toolbar command (*see toolbar screenshot below*). Temporary elements and attributes are intended to hold data used for calculations or data that for any other reason should not be saved to file. The data of temporary nodes is not saved.



Icon	Command	Does this...
+	**Add Source**	Displays the Add Page Source dialog. A root node is created for the data source that is added. Only one child element can be added to a root node.
N ▾	**Add Namespace**	Inserts or appends a namespace declaration under the *Namespace* entry. Edit the default prefix if you want, and enter a namespace.
{} ▾	**Add Element**	Inserts, appends, or adds a child element relative to the selected node.
= ▾	**Add Attribute**	Inserts, appends, or adds a child attribute relative to the selected node.
✕	**Delete**	Deletes the selected node.

4.6 Overview Pane

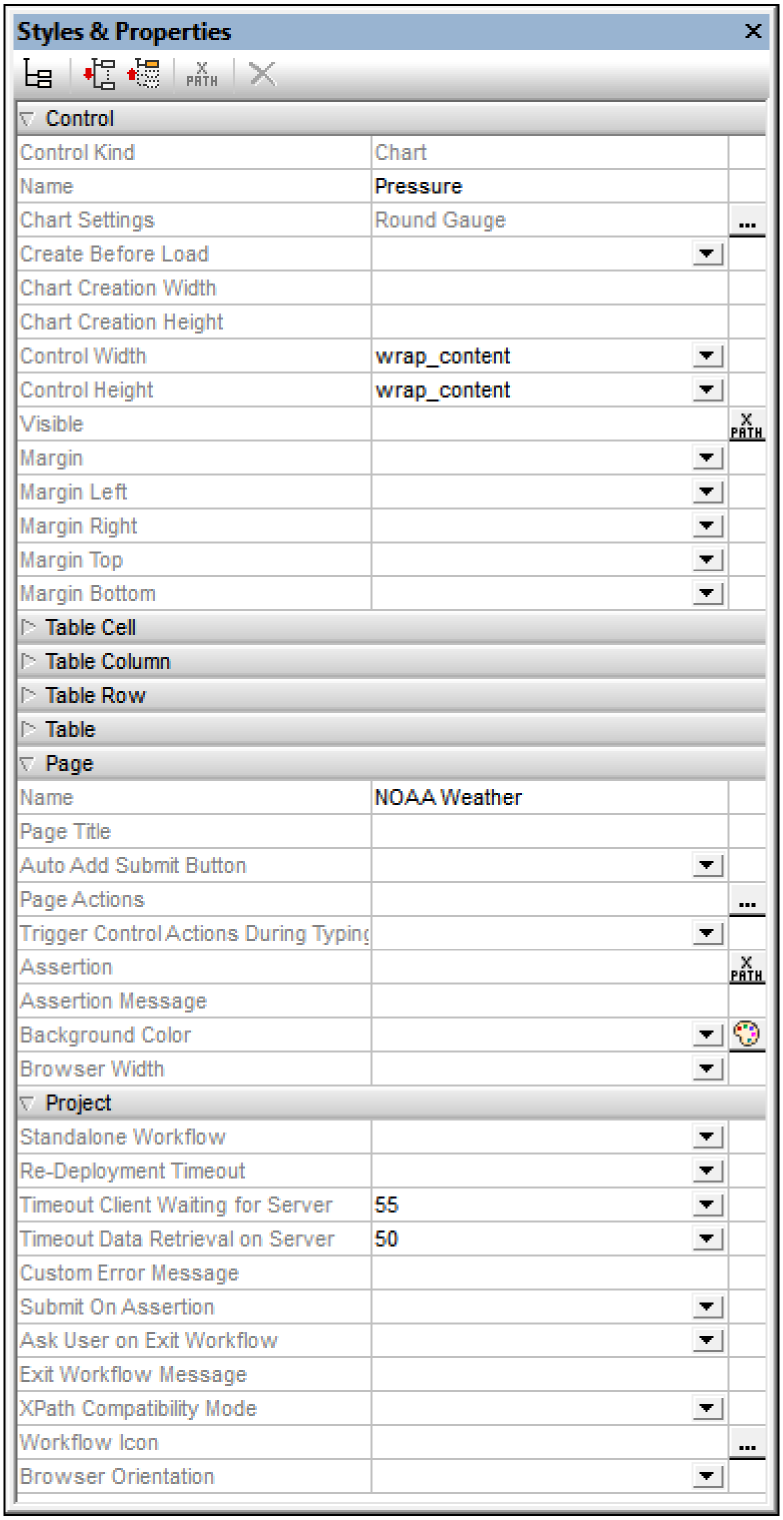
The **Overview Pane** (*screenshot below*) shows a small version of the Page Design View. To see the default location of the Overview Pane, go to The Graphical User Interface (GUI).



In the screenshot above, the extent of the design is indicated by the gray area (with the white vertical stripes) in the center. The red rectangle is the viewport. It indicates the part of the design that is currently visible in Page Design View, and its dimensions correspond to those of the displayed part of the design. From the Overview Pane shown above, it can be seen that only part of the design is in the viewport, that is, within the Main Window. To bring any part of the design into the viewport (Main Window), grab the red rectangle in the Overview Pane and move it so that the required diagram part is brought within the viewport. The black rectangle appears as a result of grabbing the red rectangle and dragging it up. On releasing the mouse, the black rectangle will turn to red and will be the new viewport, thus bringing this part of the design into view.

4.7 Styles & Properties Pane

The **Styles & Properties Pane** (*screenshot below*) displays the properties of the currently selected page controls, pages, and project, and enables these to be conveniently edited in one place. The Styles & Properties Pane is divided into the following sub-panes: (i) *Control,* for the properties of the page control currently selected in the design, (ii) *Page,* for the properties of the current page, and (iii) *Project,* for the properties of the current project. The Table control and its parts (cells, columns, and rows) are each given a separate pane (*see screenshot below*). Table and table-part sub panes are only displayed if, in the design, a part of a table is selected.

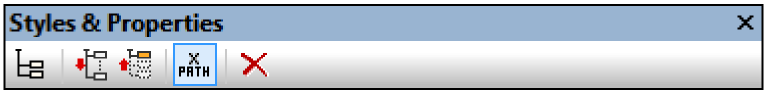


Styles & Properties

▽ Control	
Control Kind	Chart
Name	Pressure
Chart Settings	Round Gauge
Create Before Load	
Chart Creation Width	
Chart Creation Height	
Control Width	wrap_content
Control Height	wrap_content
Visible	
Margin	
Margin Left	
Margin Right	
Margin Top	
Margin Bottom	
▷ Table Cell	
▷ Table Column	
▷ Table Row	
▷ Table	
▽ Page	
Name	NOAA Weather
Page Title	
Auto Add Submit Button	
Page Actions	
Trigger Control Actions During Typing	
Assertion	
Assertion Message	
Background Color	
Browser Width	
▽ Project	
Standalone Workflow	
Re-Deployment Timeout	
Timeout Client Waiting for Server	55
Timeout Data Retrieval on Server	50
Custom Error Message	
Submit On Assertion	
Ask User on Exit Workflow	
Exit Workflow Message	
XPath Compatibility Mode	
Workflow Icon	
Browser Orientation	

Styles & Properties Pane toolbar

The commands available in the Styles & Properties toolbar (*screenshot below*) are listed in the table below:



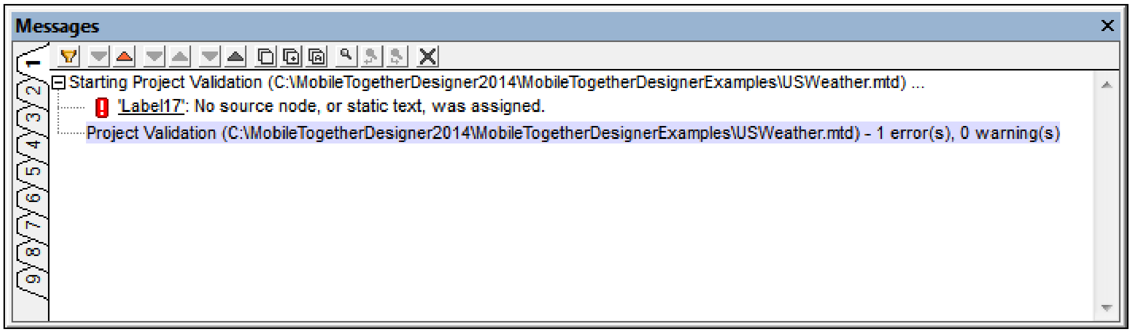
Icon	Command	Does this...
	List Non-Empty	Toggles between displaying all properties and only those that have values defined for them. Properties that have default values are considered empty.
	Expand All	Expands all sub-panes.
	Collapse All	Collapses all sub-panes.
X PATH	**Edit XPath**	Enabled when a property that requires an XPath expression is selected. It displays up the Edit XPath/XQuery Expression dialog.
✕	**Reset**	Resets the property to empty or to its default value.

Entering and editing property values
Property values are entered or edited depends on the type of entry field the property has:

Type of entry field	Do this...
Text field	Double-click, and enter or edit the property value.
Combo box	Select a value from the dropdown list.
Color palette	Pops up the color palette to enable selection of a color.
Edit XPath	Pops up the Edit XPath/XQuery Expression dialog to enable the entry of an XPath/XQuery expression. The expression provides the value of the property.
Additional dialog	Click to open the associated dialog. This could be, for example, the Actions dialog or the Open File dialog.

4.8 Messages Pane

The **Messages Pane** (*screenshot below*) displays the validation results of the currently active project. This includes all pages in the project. If an error is reported, the error message contains a link that points to the component that generated the error.



Starting Project Validation (C:\MobileTogetherDesigner2014\MobileTogetherDesignerExamples\USWeather.mtd) ...
'Label17': No source node, or static text, was assigned.
Project Validation (C:\MobileTogetherDesigner2014\MobileTogetherDesignerExamples\USWeather.mtd) - 1 error(s), 0 warning(s)

The toolbar of the Messages Pane contains commands that enable the following actions: filtering of the messages, navigation of the messages, copying of messages, searching the messages, and clearing the current tab. There are nine Messages tabs. Therefore, you can retain the results of a validation in one tab while carrying out another validation in a new tab.

Chapter 5

The Project

5 The Project

This section describes options that are relevant at the project level:

- Location of Project Files
- Deploying the Project
- Project Properties
- Namespaces
- XPath/XQuery Functions
- Global Variables
- Performance

5.1 Location of Project Files

The project file (aka design or MTD file), which has the **.mtd** extension, is deployed to the server. This file is the solution that will be accessed by the MobileTogether Client app. The project file uses other files, from which it reads data and to which it writes data. These files can be stored at the following locations:

Deployed to the server with the project file

- These data files will be read-only on the server.
- The advantage is that access to the data files is internally handled within the project.
- For the details of deployment, see: Deploying the Project, Files Pane, Deploy to MobileTogether Server.

Embedded within the project file

- This applies only to XML data files (typically the default files of data sources).
- The advantage is that the XML data will be transported together with the project file, and will be correctly accessed from within the project.
- As with deployed files, embedded XML files will be read-only.
- The disadvantage is that the size of the project file increases.
- To embed a file, right-click its root node in the Page Sources Pane, and select **Embed XML in Design File**.

A directory on the server

- Files of any type can be stored in any directory on the server. These files can be read-write.
- However, care must be taken to correctly configure (i) the file's location when it is added, and (ii) MobileTogether Server's *Server Side Solution's Working Directory* setting.
- If files are referenced by relative paths, the relative paths are resolved relative to the Working Directory.
- If files are referenced by absolute paths, the directory containing the file must be a descendant directory of the Working Directory. For example:
  If the absolute path of the referenced file is: `C:\Altova\MobileTogether\Test\First.xml`
  and the Working Directory is set to: `C:\Altova\MobileTogether`
  then the XML file will be accessed correctly.
  It will not be accessed correctly if the Working Directory is set to: `C:\Altova\MobileTogether\Files` or to `C:\Altova\MTD`
- The advantage of using directories on the server to store data files is that the data accessed by the solution will always be up-to-date.

A URL accessible over the Internet

- Files of any type can be stored at any URL that the server can access over the Internet.
- If a file is added as a data source, security authorizations can be set when specifying the properties of the data source.
- The advantages of using Internet locations are: (i) data accessed by the solution will

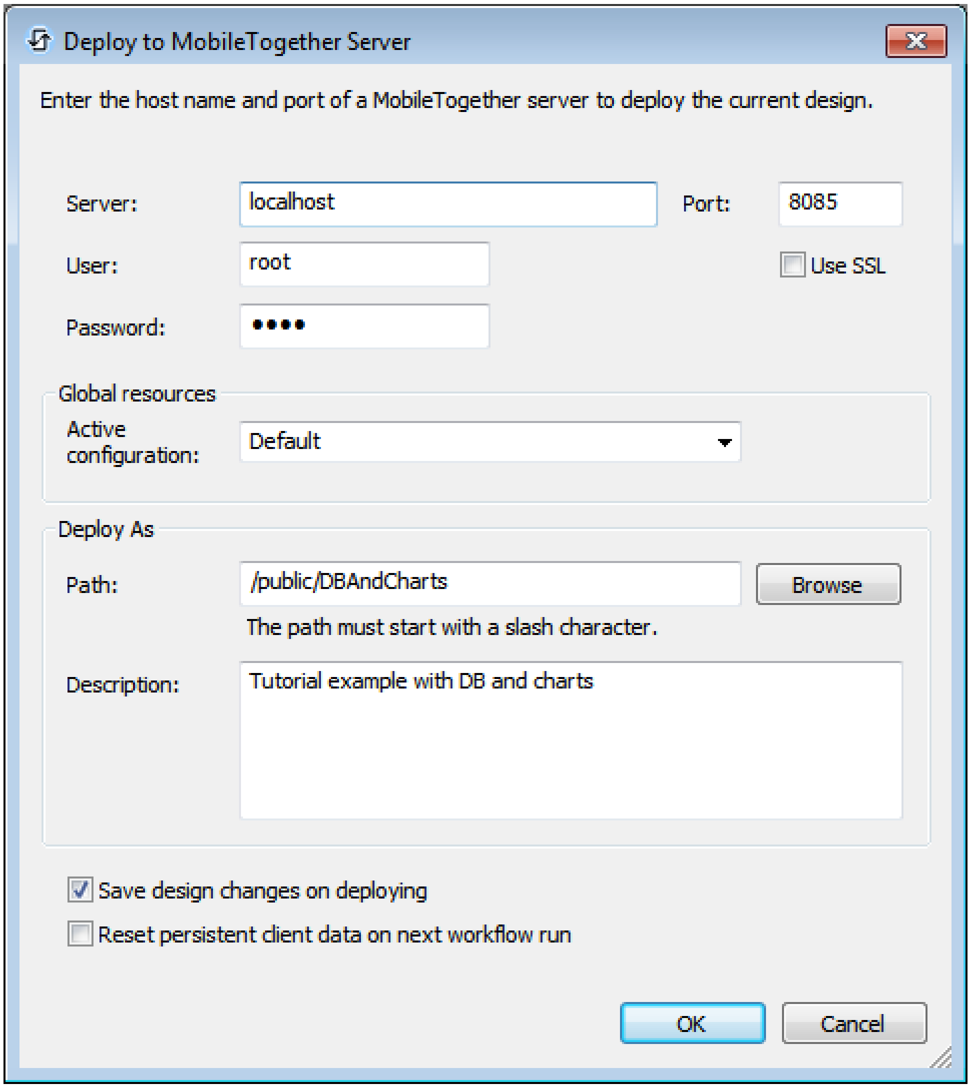
always be up-to-date, and (ii) the solution is portable.

⊟ *Also see*

Deploy to MobileTogether Server
Files Pane
Deploying the Project
Data Storage on Servers.

5.2 Deploying the Project

A project is deployed to the server with the **File | Deploy to Server** command. This command displays the Deploy to MobileTogether Server dialog (*screenshot below*), in which you specify the server connection details.



Deploy to MobileTogether Server

Enter the host name and port of a MobileTogether server to deploy the current design.

Server: localhost Port: 8085

User: root □ Use SSL

Password: ••••

Global resources

Active configuration: Default

Deploy As

Path: /public/DBAndCharts

The path must start with a slash character.

Description: Tutorial example with DB and charts

☑ Save design changes on deploying

□ Reset persistent client data on next workflow run

What is deployed?

The following files are deployed when the project is deployed with the **File | Deploy to Server** command:

- The project file (aka design or MTD file), which has the **`.mtd`** extension, is deployed to the server. This file is the solution that will be accessed by the MobileTogether Client app.
- All the deployable files in the Files Pane that have their check boxes selected. These files are typically image files and the default files of data sources.

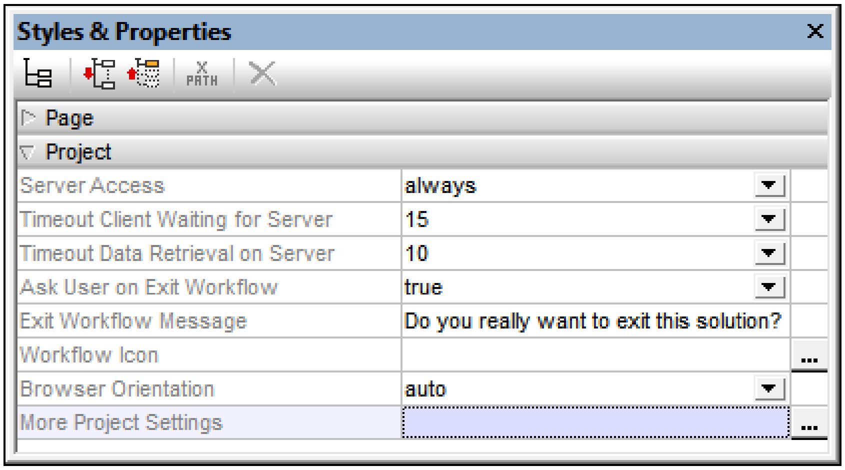
All deployed files will be stored on the server and will be correctly accessed by the design file. This is very convenient because you do not have to worry about the files being correctly accessed. Note, however, that these files are read-only. So, any files that need to be written to cannot be deployed, but must be stored manually on the server. The server and the design file must then be correctly configured to access the writable file. See the section Location of Project Files for details of how to do this.

⊟ *Also see*

Deploy to MobileTogether Server
Location of Project Files
Files Pane
Data Storage on Servers.

5.3 Project Properties

Project properties are defined in the Styles & Properties Pane and are described below. The screenshot below shows default values.



Server Access	always
Timeout Client Waiting for Server	15
Timeout Data Retrieval on Server	10
Ask User on Exit Workflow	true
Exit Workflow Message	Do you really want to exit this solution?
Workflow Icon	
Browser Orientation	auto
More Project Settings	

Server Access

This option specifies the level of server access while the solution runs.

- *Always:* Connection to the server is required in order to run the solution, and the server is continually accessed while the solution runs.
- *On Demand:* The MobileTogether Client app runs the solution on its own; it connects to the server only when it needs to exchange data with the server. To run the solution, the app uses data in the internal `$PERSISTENT` tree.
- *Never:* The MobileTogether Client app runs the solution entirely on its own and without needing any data from the Internet.

Timeout: Client Waiting for Server

The amount of time the client waits for a response from the server. The value is an integer value in seconds that can be entered or selected from the dropdown list of the combo box. The default value is 15 seconds. If the timeout period is exceeded, then an error message is displayed on the client.

Timeout: Data Retrieval on Server

The amount of time the server waits for data to be retrieved from a source external to the server (from a DB or URL, for example). The value is an integer value in seconds that can be entered or selected from the dropdown list of the combo box. The default value is 10 seconds. If the timeout period is exceeded, then an error message is displayed on the client. An exception to this is when load actions have the setting *On error* set to `Continue`.

Ask User on Exit Workflow

A boolean setting that sets whether the user is asked to confirm (workflow) solution exit. Select `true` or `false` in the combo box. Default value is `true`. If `true`, the text defined as the value of the next property, *Exit Workflow Message*, is displayed before the solution exits.

Exit Workflow Message

The text of the message that is displayed as a prompt to confirm (workflow) solution exit. The message is displayed only if the previous property, *Ask User on Exit Workflow*, is set to `true`. The default message is: *Do you really want to exit this solution?*

Workflow Icon

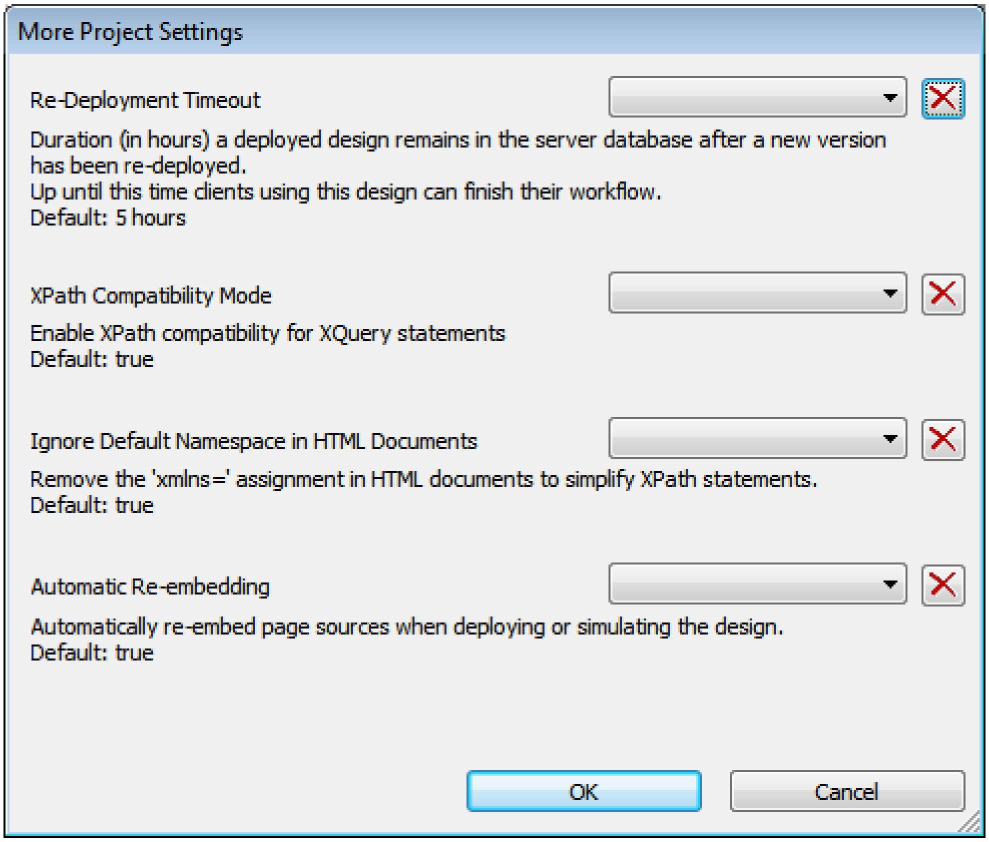
Clicking the property's **Additional Dialog** button displays a Browse dialog in which you can browse for the PNG image file to use as the icon of the project. By default, the MobileTogether icon will be used.

Browser Orientation

The combo box options are: `Auto`; `Force Portrait`; and `Force Landscape`. The `Auto` option enables the automatic switching of orientation to match the orientation of the device's viewport.

More Project Settings

Clicking the property's **Additional Dialog** button displays the More Project Settings dialog (*screenshot below*).



Re-Deployment Timeout

Duration (in hours) a deployed design remains in the server database after a new version has been re-deployed.
Up until this time clients using this design can finish their workflow.
Default: 5 hours

XPath Compatibility Mode

Enable XPath compatibility for XQuery statements
Default: true

Ignore Default Namespace in HTML Documents

Remove the 'xmlns=' assignment in HTML documents to simplify XPath statements.
Default: true

Automatic Re-embedding

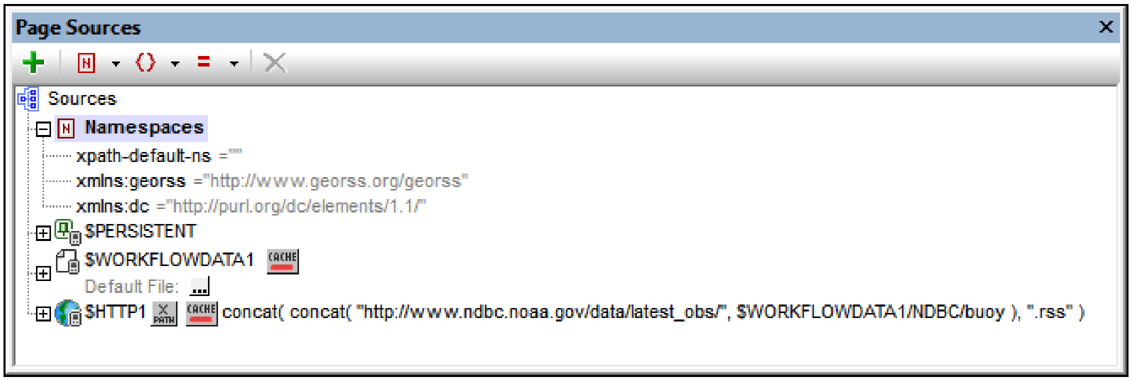
Automatically re-embed page sources when deploying or simulating the design.
Default: true

The following settings can be defined:

- *Re-deployment Timeout:* The time in hours after a new version of a solution is deployed that the superseded solution is kept on the server. This overlap time enables clients currently using the older solution to complete work. The default is 5 hours.
- *XPath Compatibility Mode:* When set to `true`, XQuery constructs that are invalid in XPath are resolved so that XQuery statements containing these constructs are compatible with XPath and can be used where XPath expressions are allowed. Currently, this concerns XQuery entity and character references, which are allowed in XQuery, but not in XPath. When *XPath compatibility mode* is set to `true`, XQuery entity and character references are read in XPath as text; they are not resolved. The default value for this setting is `true`.
- *Ignore Default Namespace in HTML Documents:* Since only one default namespace is allowed in an XML document, not ignoring the default namespace in HTML documents could create errors in reading XML data sources. The default is `true`: The HTML default namespace is ignored.
- *Automatic Re-embedding:* Embedding refers to the embedding of data sources in the project (design) file. If *Automatic Re-embedding* is enabled, then page data sources are re-embedded, ensuring that the latest data source files are embedded and that the data, therefore, is up-to-date. The default is `true`.

5.4 Namespaces

Namespaces are important for correctly identifying nodes, and for correctly locating nodes with the use of XPath expressions. The *Namespaces* item in the Page Sources Pane (*screenshot below*) contains all the namespaces that have been declared for the project, irrespective of what page is currently active in Page Design View.



```
xpath-default-ns =""
xmlns:georss ="http://www.georss.org/georss"
xmlns:dc ="http://purl.org/dc/elements/1.1/"
$PERSISTENT
$WORKFLOWDATA1
Default File:
$HTTP1 concat( concat( "http://www.ndbc.noaa.gov/data/latest_obs/", $WORKFLOWDATA1/NDBC/buoy ), ".rss" )
```

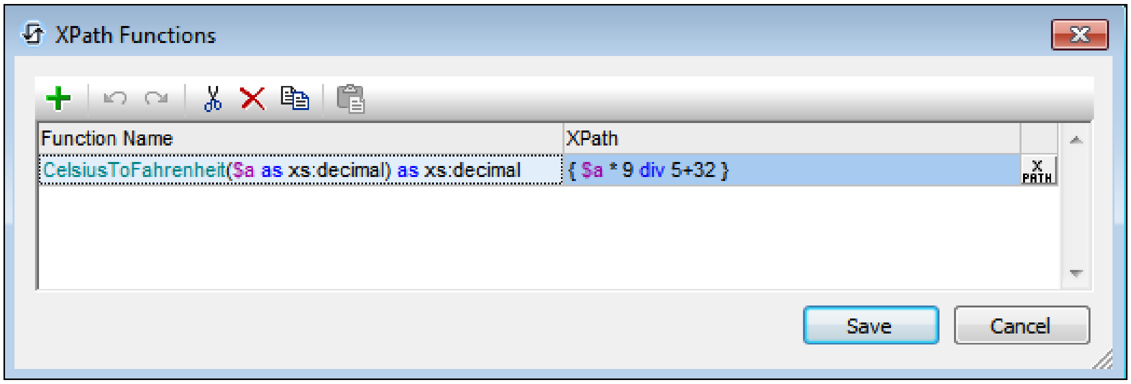
Namespaces can be declared in two ways:

- *Automatic declaration on data import:* When an external XML file is added as a page source, namespaces in the source are automatically imported into the design and declared for the scope of the entire project. They then appear under the *Namespaces* item in the Page Sources Pane (*see screenshot above*). The namespace prefixes are automatically set to match the original prefixes if such matching creates no ambiguities in the design. Prefixes assigned in the namespace declaration are used in node names, and must be used in XPath expressions that are intended to locate these nodes in the data source.
- *User-defined:* You can also add namespaces by clicking the **Namespace** icon in the toolbar of the Page Sources Pane (*screenshot above*). Being able to add your own namespaces to a project enables you to create nodes that belong to one or more user-declared namespaces. This is useful for disambiguating between nodes that have the same local name.

To delete a namespace, select it and click **Delete** in the pane's toolbar.

5.5 XPath/XQuery Functions

You can create your own custom-made XPath/XQuery functions for individual projects, which you can then use in all XPath expressions in the project. The access point for creating and managing these user-defined functions is the XPath Functions dialog, accessed with the command, **Project | XPath/XQuery Functions**. The XPath Functions dialog (*screenshot below*) lists all the user-defined XPath functions in the project. You can add and delete functions using the corresponding icons in the toolbar of the dialog. To edit the definition of a function, click the function's **Edit XPath Expression** button.

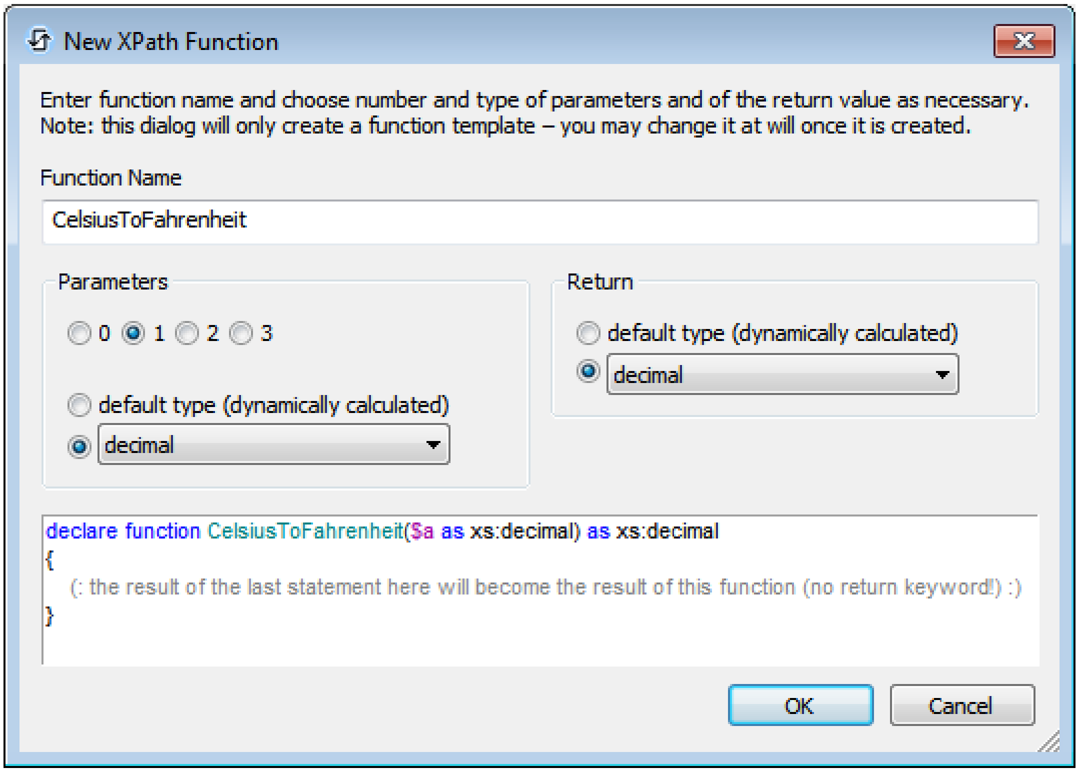


Function Name	XPath
CelsiusToFahrenheit($a as xs:decimal) as xs:decimal	{ $a * 9 div 5+32 }

Add a new user-defined XPath function

Adding a new user-defined function involves two steps: (i) declaring the function, and (ii) defining the function.

To add a new function, do the following, click **Add** in the toolbar of the dialog (*see screenshot above*). This displays the New XPath Function dialog (*screenshot below*).



Enter function name and choose number and type of parameters and of the return value as necessary.
Note: this dialog will only create a function template – you may change it at will once it is created.

Function Name: CelsiusToFahrenheit

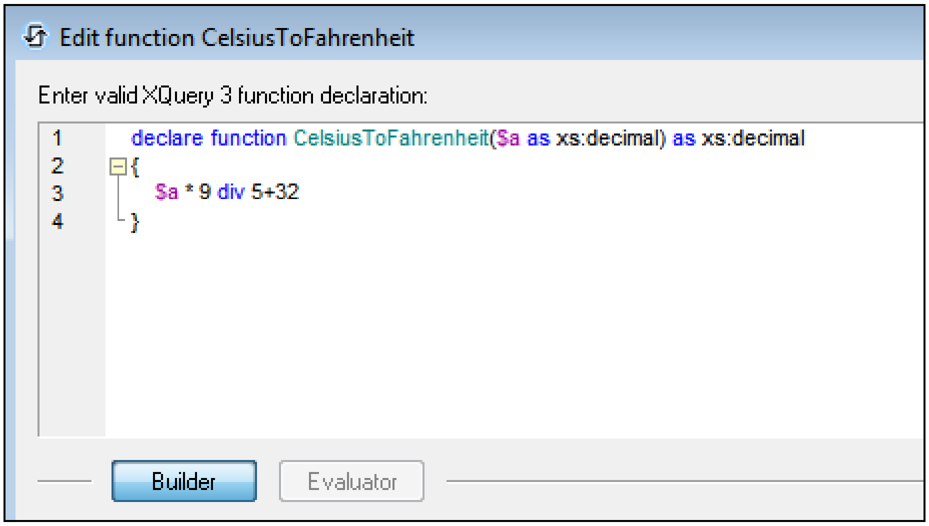
Parameters: ○ 0 ● 1 ○ 2 ○ 3

○ default type (dynamically calculated)
● decimal

Return:
○ default type (dynamically calculated)
● decimal

```
declare function CelsiusToFahrenheit($a as xs:decimal) as xs:decimal
{
   (: the result of the last statement here will become the result of this function (no return keyword!) :)
}
```

In this dialog, you can declare the name of the function, specify the number of function parameters (arguments) and their types, and specify the return type of the function. In the screenshot above we have declared a function to convert a decimal number from Celsius to Fahrenheit. The function takes one parameter, which is the input Celsius value as a decimal. It will output a decimal value, the Fahrenheit temperature. What the function does is defined in the next step. After declaring the function (*screenshot above*), click **OK**. This displays the Edit Function dialog (*screenshot below*), which contains the template of the newly declared function and in which you can now define the function.



Enter valid XQuery 3 function declaration:

```
declare function CelsiusToFahrenheit($a as xs:decimal) as xs:decimal
{
  $a * 9 div 5+32
}
```

Enter the definition of the function within the braces. In the definition shown in the screenshot above, **$a** is the input parameter. Click **OK**. when done. The function will be added to the list of user-defined functions in the XPath Functions dialog and can be used in all XPath expressions in the project.

5.6 Global Variables

Global variables are predefined for a project and can be used in XPath/XQuery expressions anywhere in the project. They are of three types (*see screenshot below*):

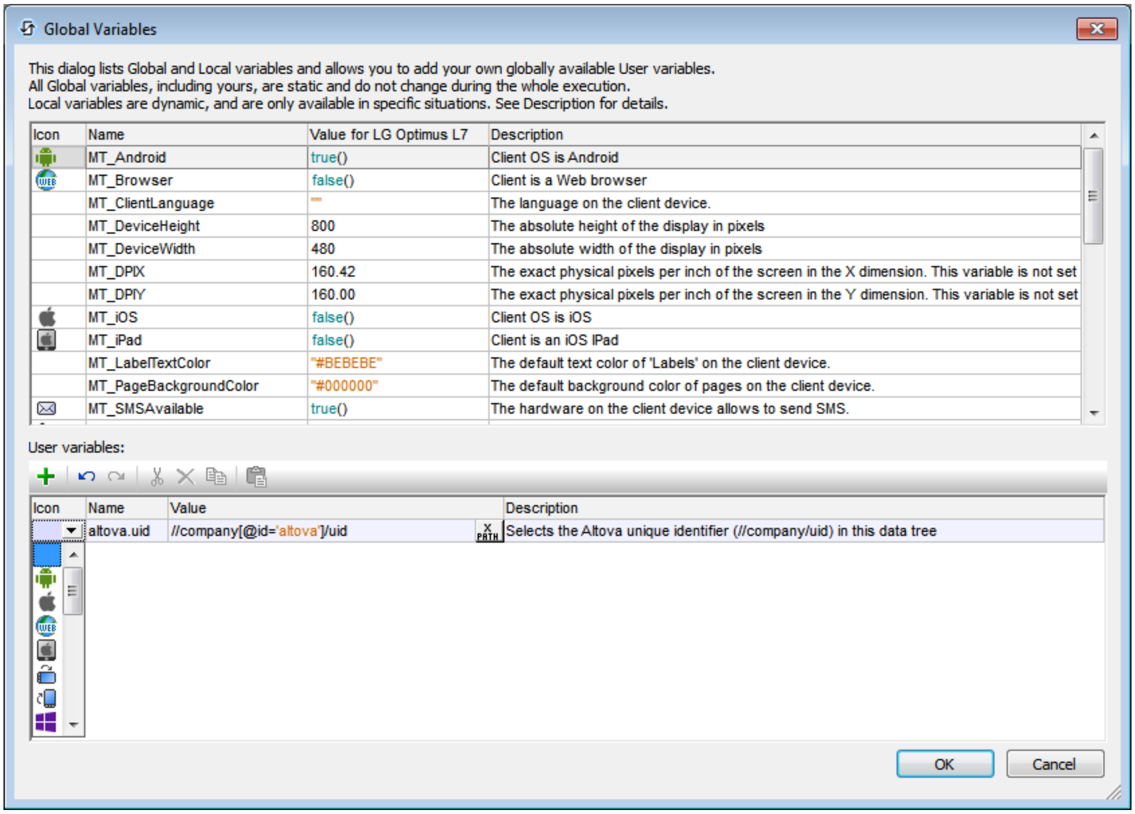
- Static-value variables (called *Global Variables* in the dialog): These variables contain information about the mobile device. Values of these variables do not change during the execution of the project. Notice that the *Value* column in the upper pane specifies the currently selected mobile device. The listed values are for that particular device. For simulations, the client device is considered to be the device selected in the Device Selector combo box. For example, the variable **`$MT_Android`** has a value of **`true`** when the mobile device being used is an Android. (Device information is sent by the device as part of standard mobile communication procedures.)
- Dynamic-value variables (called *Local Variables* in the dialog): These variables contain information related to the design page and its controls. Their values could change during execution. For example, the **`$MT_ControlNode`** variable has different values according to which node is the current node at a given time during project execution.
- *User variables*: In addition to the standard library of global variables, you can add your own global variables (called *User Variables* in the dialog) in the lower pane of the dialog. You can give a user variable any value, and this value can then be used subsequently in any XPath/XQuery expression in the project.

Note: When defining a user variable, do not use a **`$`** symbol in the name of the variable. When you use any global variable in an XPath expression, however, you must, as usual, use the **`$`** symbol. For example:

```
concat('http://www.', $company, '.com')
```

Global Variables dialog

The Global Variables dialog (**Project | Global Variables**, *screenshot below*) displays the predefined global variables and enables you to add and edit user variables.



This dialog lists Global and Local variables and allows you to add your own globally available User variables.
All Global variables, including yours, are static and do not change during the whole execution.
Local variables are dynamic, and are only available in specific situations. See Description for details.

Icon	Name	Value for LG Optimus L7	Description
	MT_Android	true()	Client OS is Android
	MT_Browser	false()	Client is a Web browser
	MT_ClientLanguage	""	The language on the client device.
	MT_DeviceHeight	800	The absolute height of the display in pixels
	MT_DeviceWidth	480	The absolute width of the display in pixels
	MT_DPIX	160.42	The exact physical pixels per inch of the screen in the X dimension. This variable is not set
	MT_DPIY	160.00	The exact physical pixels per inch of the screen in the Y dimension. This variable is not set
	MT_iOS	false()	Client OS is iOS
	MT_iPad	false()	Client is an iOS IPad
	MT_LabelTextColor	"#BEBEBE"	The default text color of 'Labels' on the client device.
	MT_PageBackgroundColor	"#000000"	The default background color of pages on the client device.
✉	MT_SMSAvailable	true()	The hardware on the client device allows to send SMS.

User variables:

Icon	Name	Value	Description
	altova.uid	//company[@id='altova']/uid	Selects the Altova unique identifier (//company/uid) in this data tree

To add a user variable, in the lower pane, do the following:

1. Click the **Append** or **Insert** icons (located in the pane's toolbar) to add a line to the list.
2. Enter the name of your new variable (in the *Name* column, and without a **$** symbol) and give the variable a description (*Description* column). *See screenshot above*.
3. Click in the *Value* field to bring up the Edit XPath/XQuery Expression dialog, and enter the XPath expression that determines the value of this variable.
4. Select an icon to help identify the new variable as belonging to a particular group.
5. Click **OK** to finish. The variable is added as a global variable, and can be used in programming contexts.

Static Global Variables

Static-value variables are called *Global Variables* in the Global Variables dialog. They are variables that contain static information about the mobile device, such as the device's type and dimensions. Values of these static (global) variables do not change during the execution of the project. They are displayed in the Global Variables dialog (**Project | Global Variables**). In the dialog, the header of the *Value* column displays the currently active mobile device. For simulations, the client device is considered to be the device selected in the Device Selector combo box. For example, the variable **`$MT_Android`** has a value of **`true()`** when the mobile device being used is an Android device. (Device information is sent by the device as part of standard mobile communication procedures.)

▼ Variables that indicate the mobile-device type

Description

These are a set of variables (*see table below*) that can be used to specify actions that are conditional on the device type. For example: **`if ($MT_iOS=true()) then 'http://www.apple.com/' else 'http://www.altova.com'`**. Information about the type of client device on which the solution runs, or is simulated to run, is sent by the device to the solution. The corresponding global variable (*see table below*) is set to **`true()`**; all the other variables in the group are set to **`false()`**. All theses variables can then be used in XPath/XQuery expressions.

MT_Android	true() | false()
MT_Browser	true() | false()
MT_iOS	true() | false()
MT_iPad	true() | false()
MT_Windows	true() | false()
MT_WindowsPhone	true() | false()

▼ Variables that indicate the device's communications capability

Description

These variables indicate whether SMS and telephony services are available on the mobile device, and can be used to make checks before initiating SMS or call actions. Information about the communications capability is received from the client device. Values can be **`true()`** or **`false()`**.

MT_SMSAvailable	true() | false()
MT_TelephonyAvailable	true() | false()

▼ Variables that contain the device's dimensions and resolution

Description

The device's absolute height and width values are held by these variables as pixel values. The resolution is in terms of dpi (pixels per inch), in the X and Y dimensions. Resolution is not set on iOS devices.

`MT_DeviceHeight`	*Length value in pixels*
`MT_DeviceWidth`	*Length value in pixels*
`MT_DPIX`	*Horizontal pixel density in pixels per inch*
`MT_DPIY`	*Vertical pixel density in pixels per inch*

Variables that contain default colors of device elements

Description

Some elements of the device's display have default colors. It is useful to know these when designing the appearance of a page. For example, a label's background color can be set conditionally according to what the default label text color on the device is: **`if ($MT_LabelTextColor = '#000000') then '#FFFFFF' else '#000000'`**. The default colors are received from the client device and are hexadecimal values, e.g: `#336699` and `#ffaaff`.

`MT_EditFieldBackgroundColor`	*Background color of edit fields; Hex values, e.g:* `#ffaaff`
`MT_EditFieldTextColor`	*Text color of edit fields; Hex values, e.g:* `#336699`
`MT_LabelBackgroundColor`	*Background color of labels; Hex values, e.g:* `#ffaaff`
`MT_LabelTextColor`	*Text color of labels; Hex values, e.g:* `#336699`
`MT_PageBackgroundColor`	*Background color of pages; Hex values, e.g:* `#ffaaff`

Miscellaneous

`MT_ClientLanguage`

The language on the client device.

`MT_DBExecuteResult`

The XML result of the last executed DB Execute action. Note that any kind of SQL statement can be used in the DB Execute action. So executing the action could obtain various kinds of result XML data, including, for example, data from the DB, boolean values, or computational results.

`MT_SimulationMode`

Indicates, using the values listed in the table below, the kind of simulation that is currently running. The empty option indicates that the solution is running in an actual end-user scenario, and not in a simulation. `$MT_SimulationMode` is useful, for example, if you want to provide conditional processing depending on what kind of simulation (or actual use) is currently running. See the section Simulation for more information.

`"designer"`	*Simulation runs directly in designer*
`"designer-server"`	*Simulation with a standalone server*
`"designer-client"`	*Simulation is a trial run on client*
`""`	*Server to client/browser, end-user run*

⊟ **`MT_UserName`**

The name with which to log in to MobileTogether Server. The value is taken from the setting made in the Server Settings tab of the Options dialog.

Dynamic Local Variables

Dynamic-value variables are called *Local Variables* in the Global Variables dialog. They contain information related to the design page and its controls. This is information that can change during the execution of the project. For example, the device orientation variables will change according to how the end user is currently holding the device (*see the description of the device orientation variables below*).

The variables that contain information about the current control (*see below*) are particularly useful because they provide information about the node that is associated with the current control (the page source link of the control). It is important to note that the control that is current changes as the execution of the project progresses. As a result, the values of the control variables change according to which control is being currently processed. The values of these variables can therefore be used for conditional processing. For example, the **`$MT_ControlNode`** variable can be used to test which node is the current node at a given time during project execution. The **`$MT_ControlValue`** variable holds the content of the node associated with the current control.

Variables that indicate device orientation

Description

The values of both `MT_Portrait` and `MT_Landscape` can be **`true()`** or **`false()`**and can change during the course of project execution. They can be used to specify page or control properties according to device orientation.

`MT_Portrait`	`true()	false()`
`MT_Landscape`	`true()	false()`

Variables that indicate the device's viewport dimensions

Description

Thees variables provide, respectively, the width (X dimension) and height (Y dimension) of the device's viewport. The values are pixel values and will necessarily be less than the device's width and height dimensions.

`MT_CanvasX`	*Length value in pixels*
`MT_CanvasY`	*Length value in pixels*

Variables that contain information about the current control

Description

These variables contain information related to the data source node that is associated with the current control (the page source link of the control). The values of these variables change during execution according to which control is being currently processed. For example, the **`$MT_ControlNode`** variable has different values according to which node is associated with the current control. These variables can therefore be used in the XPath expressions of all controls that have page source links. (Note that some controls, such as the space and horizontal line controls, do not have page source links, while other, like the chart control, will not have an XML value as the content of its associated node.)

These variables are useful for changing a control's properties based on the control's values. For example, a **`$MT_ControlValue`** variable can be used to change a label's background color to red in case an error is encountered: **`if ($MT_ControlValue = 'NaN') then '#FF0000' else '#FFFFFF'`**.

`MT_ControlKind`	*String value*
`MT_ControlName`	*String value*
`MT_ControlNode`	*XML node that is the control's page source link*
`MT_ControlValue`	*Value of the control's page-source-link node*
`MT_ControlValueBeforeChange`	*Previous value of the control's page-source-link node, before the control or node is edited*

Miscellaneous

⊟ **`MT_FirstPageLoad`**

The first page that loads.

⊟ **`MT_PageName`**

The name of the page.

⊟ **`MT_TargetNode`**

This variable identifies the target node of an Update Node(s), Insert Node(s), or Append Node(s) action. It can be used to generate update values and new node properties according to what the target node is. See the descriptions of the respective actions for examples of how the variable can be used. `$MT_TargetNode` can also be used with the DB Execute action.

⊟ **`MT_UserRoles`**

The roles of the currently logged in user. The roles are those assigned the user by the MobileTogether Server administrator, and are obtained from MobileTogether Server.

5.7 Performance

You can optimize performance of your MobileTogether solutions by being aware of how MobileTogether Server and the MobileTogether Client app work together, how MobileTogether Server can be used to do the most data-intensive processing and store large amounts of data, and how settings to optimize performance can be made in the project design in MobileTogether Designer. This section describes important optimization concepts.

- Global Resources
- Embed XML in Design File
- Data Querying with XQuery 3.0
- Data Storage on Servers
- Persistent Data Storage on Clients

Global Resources

Global Resources in MobileTogether allow you to easily specify different database server names, file locations, or other configuration-specific parameters in a way that lets you easily switch from a development to a production system—and you can do that independently for each mobile solution. Because of this, Global Resources enable you to design and test quickly, and, thus, to save time.

As you develop a mobile solution in MobileTogether Designer, the Global Resources dialog allows you to identify each file, database connection, or directory with a user-defined name that can then be used to reference that resource anywhere in your solution. With Global Resources you can set up different configurations and easily switch your mobile solution modes by selecting a different Global Resources configuration.

Once you deploy a new mobile solution to MobileTogether Server, you can upload this configuration with your solution to control the access of your solution to specific servers even after it has been deployed. Administrators can configure the mobile solution's resource configuration on the fly, so a mobile solution can switch from a testing to a production environment with one click.

For more information about how to work with Global Resources, see the section, Altova Global Resources.

Embed XML in Design File

Instead of the solution referencing external XML data sources, any XML data can be embedded directly into the design file. This option is perfect for smaller data sets that are needed on the client side, such as a list of choices for a combo-box or other static data. This data is then transmitted to the client as a part of the overall design file and is always instantly available on the client side every time you run the app. So no additional data transfers between client and server are needed.

You can embed a data source in the design file, in one of two ways:

- When you add a data source, a dialog appears that asks whether you wish to deploy the file together with your design when deploying the design file to the server. If you click **Yes**, the data source will be embedded in the design file.
- The Files Pane lists all the data files that have been added as data sources. To embed any file in the design, select that file's check box.

Data Querying with XQuery 3.0

Using the powerful XQuery 3.0 language, you can write expressions that significantly reduce the amount of data being transferred between the server and client. Database views, queries, or web service calls to external data sources will yield raw data for a mobile application. This often contains redundancies or may not be the ideal structure for the intended data display in the mobile application. XQuery's powerful FLWOR expressions allow you to easily restructure and regroup the data to ensure the most efficient data transfer from server to client and the most useful presentation in the client application.

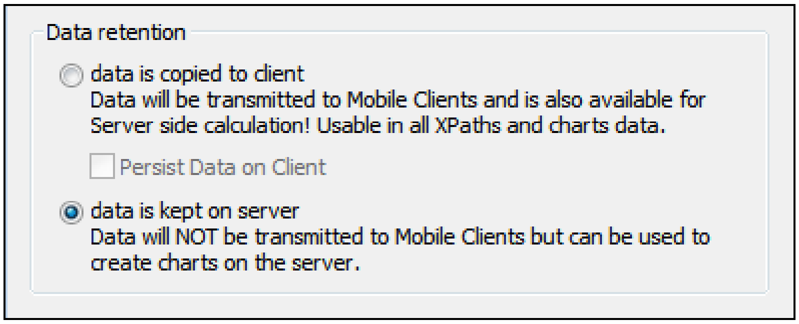
Data Storage on Servers

The speed with which data for solutions is processed can be enhanced by storing certain types of data on the server:

- Data that is used to generate charts and graphs, which are images, does not need to be sent to the client; only the image needs to be sent. So the data used to generate the charts and graphs can be kept on the server, and does not need to be transferred.

Powerful "Keep Data on Server" Setting
To reduce the amount of data transmitted over the mobile data network—which improves the performance of any mobile solution—MobileTogether lets you select exactly which data you want to transmit to the client devices and which data to keep on the server. For example, if a certain data set is only necessary to display a graph, then that data can be kept on the server. The graph image will be rendered by the server and transmitted to the client without the underlying data being transferred over the mobile network. For large data sets this produces a significant performance boost.

This setting (to keep data on the server and not send it to the client) is made for a data source at the time the data source is added. The setting is in the Data Retention pane (*screenshot below*) of the Add Page Source dialog. It is also available as a command in the context menus of root nodes in the Page Sources Pane.



Data retention

○ data is copied to client
Data will be transmitted to Mobile Clients and is also available for Server side calculation! Usable in all XPaths and charts data.

□ Persist Data on Client

● data is kept on server
Data will NOT be transmitted to Mobile Clients but can be used to create charts on the server.

Caching
Using settings in MobileTogether Designer and MobileTogether Server, you can specify caching behavior for all data sources. This boosts the speed of MobileTogether greatly because when the server receives a request from the Mobile App, it will already have the data available. There are two main reasons to create caches: (i) If a page data source generates reports slowly (for example, a large database); (ii) If a data source is not modified often. In such cases, execution of a solution would be faster if data is taken from data caches on the server. In order to keep caches up-to-date, the frequency of cache updates can be specified when the cache is created. Once a cache has been defined in MobileTogether Designer, it can be used by the data sources of different designs, providing the underlying data structure is compatible.

As pioneered in Altova MapForce and FlowForce Servers, MobileTogether contains more than just the usual caching parameters such as expiry and refresh time. You can manually determine the amount of time that passes before caching again. Also you can define how many unique combinations of multiple query parameters (either for databases or for web services) should automatically be cached. A client requesting the data will now immediately get it from the cache, whereas the server will retrieve it only if the cache time has elapsed. This is beyond simple caching as MobileTogether actually automatically executes the query to whatever interval the designer specifies. When it is a query with parameters, the designer can specify how many unique combinations of parameters should be cached, and then the server will follow those instructions.

A new cache is defined in MobileTogether Designer for a data source. Right-click a data source in the Page Sources Pane, select **Cache Settings**, and specify the properties of the cache.

If a data source is defined as having a cache, the cached data will be used when the solution is run. Caches can be used as soon as the solution has been deployed to the server.

Persistent Data Storage on Clients

For user-entered data and data that doesn't change too frequently, you can choose to store data persistently on each client device. This reduces the amount of data being transferred between the server and client, and so increases performance speed. It also provides a performance boost because it reduces the round-trip time between server and client—even between different sessions of the same user that end up being hours apart. That's because the data is saved persistently on the client device.

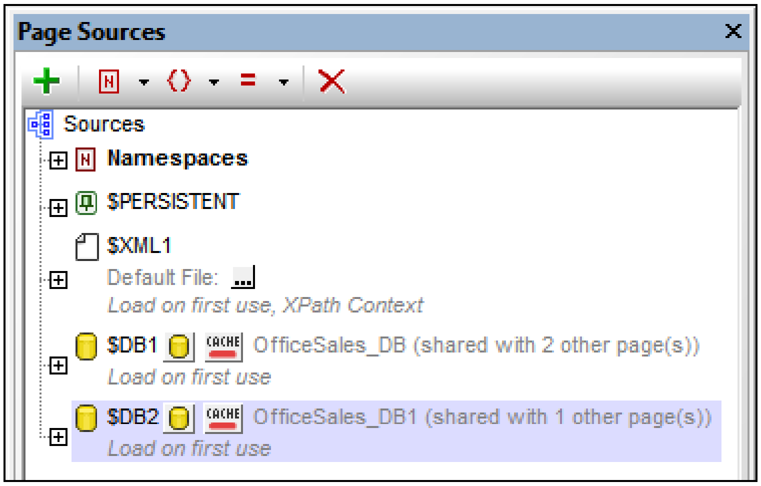
Chapter 6

Page Data Sources

6 Page Data Sources

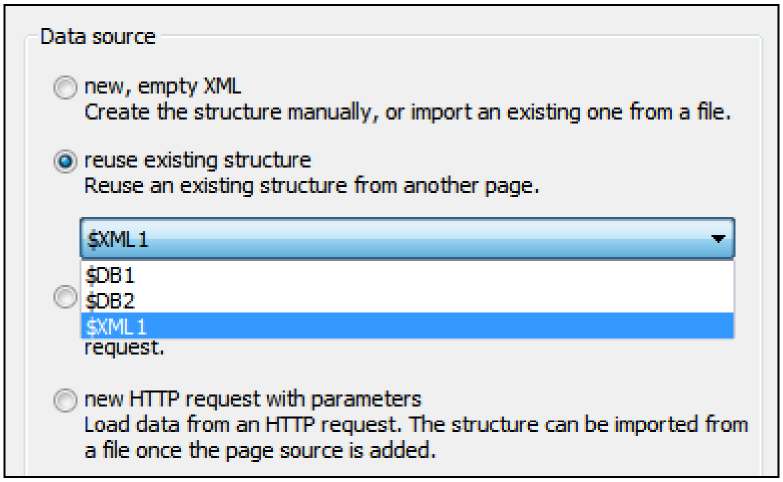
Each page can use a set of multiple data sources. The data sources of a page (*see screenshot below left*) are available to controls of that page, and that page only. However, a single data source can be reused among multiple pages.

- New data sources are added via the Add Source command of the Page Sources Pane. How to add a new source is described in the section Adding Data Sources.
- A page can reuse data sources of other pages. An option for specifying the page to reuse is available when adding a new data source (*screenshot below right*).



Page Sources

Sources
Namespaces
$PERSISTENT
$XML1
Default File: ...
Load on first use, XPath Context
$DB1 OfficeSales_DB (shared with 2 other page(s))
Load on first use
$DB2 OfficeSales_DB1 (shared with 1 other page(s))
Load on first use



Data source

○ new, empty XML
Create the structure manually, or import an existing one from a file.

● reuse existing structure
Reuse an existing structure from another page.

$XML1
$DB1
$DB2
$XML1

○ request.

○ new HTTP request with parameters
Load data from an HTTP request. The structure can be imported from a file once the page source is added.

Note the following points:

- Page data sources are added in the Page Sources Pane (*screenshot above*), one source at a time.
- Data sources can be of different types (XML documents, databases, and XQuery documents). See the section Types of Data Sources for detailed information.
- Each page can have multiple data sources of different kinds. For example, the screenshot above left shows a page that has one XML and two DB data sources.
- A data source can be editable or non-editable (read-only). This property is specified at the time the data source is added and can be modified with the root node's context menu

toggle command **Read Only Data**.

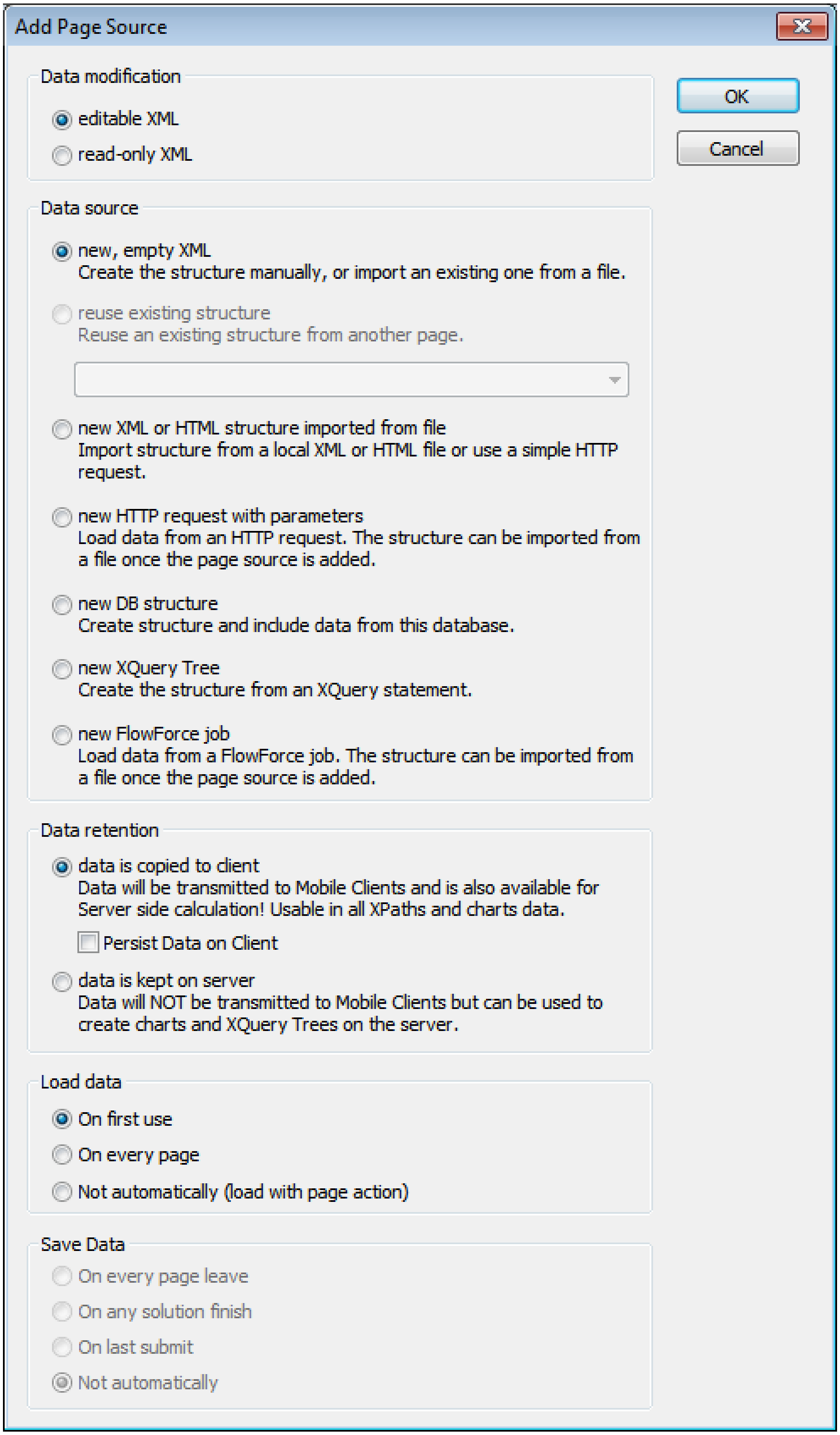
Organization of this section
This section is organized as follows:

- Adding Data Sources: shows how to add a data source
- Types of Data Sources: lists and describes the various type of data sources that can be added to a page
- Root Nodes: explains the concept of root nodes and root elements, and how these are used in the trees of data sources. This section also gives the naming convention of root nodes
- Data Source Trees: shows how the structure of a source tree and data in the tree's nodes are used in a page
- Namespaces in the Project: explains the significance of namespaces in the project, and how namespaces are used
- Context Menu: describes the context menu commands of the Page Sources Pane

6.1 Adding Data Sources

To add a data source for a page, select the page in the Pages Pane, and then do the following.

Click the **Add Source** icon in the toolbar of the Page Sources Pane to display the Add Page Source dialog (*screenshot below*). Select the type of data source to add, specify the basic properties of this data source (*listed below*), and click **OK**. The various types of data sources are described in the section, Types of Page Data Sources.



Data modification

● editable XML

○ read-only XML

Data source

● new, empty XML
Create the structure manually, or import an existing one from a file.

○ reuse existing structure
Reuse an existing structure from another page.

○ new XML or HTML structure imported from file
Import structure from a local XML or HTML file or use a simple HTTP request.

○ new HTTP request with parameters
Load data from an HTTP request. The structure can be imported from a file once the page source is added.

○ new DB structure
Create structure and include data from this database.

○ new XQuery Tree
Create the structure from an XQuery statement.

○ new FlowForce job
Load data from a FlowForce job. The structure can be imported from a file once the page source is added.

Data retention

● data is copied to client
Data will be transmitted to Mobile Clients and is also available for Server side calculation! Usable in all XPaths and charts data.

□ Persist Data on Client

○ data is kept on server
Data will NOT be transmitted to Mobile Clients but can be used to create charts and XQuery Trees on the server.

Load data

● On first use

○ On every page

○ Not automatically (load with page action)

Save Data

○ On every page leave

○ On any solution finish

○ On last submit

● Not automatically

The following data source options are available in the Add Page Source dialog (*see screenshot above*).

- *Data modification:* Select *Editable XML* or *Read-only XML*. When a data source is created as editable, data in its tree nodes can be modified. Data in read-only data sources cannot be modified. Both types of data can be used to display data. But if you want to enable end users to write to data nodes, create the data source as an editable XML.
- *Data source:* Select from a range of data source types. These are described in the section, Types of Page Data Sources.
- *Data retention:* Specifies whether data is transmitted to the client or kept on the server. When transmitted to the client, data can be used in calculations on the client. Otherwise, calculations are carried out on MobileTogether Server and only the result is passed to the client. The *Persist Data on Client* option keeps data on the client and so speeds up the loading of a solution when it is next started. If data is transferred to the client, it saves transfer time. On the other hand, if the data will not be required on the client—for example, if the data is used to generate charts (which are image files)—then keeping data on the server is probably more efficient.
- *Load data:* Selects whether data is loaded the first time the page is loaded, every time the page is loaded, or when specified by a page action. The selected option can be changed subsequently via the context menu of the source tree's root node.
- *Save data:* This option is enabled if the data source is an external file (XML, HTML, or DB). It selects whether data is saved on (i) leaving the page, (ii) exiting the entire solution, (iii) when the user clicks the last Submit option, or (iv) when expressly defined as a page or control action. *See Page Source Actions*.

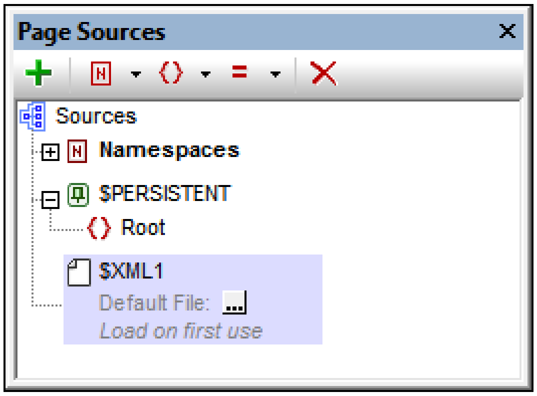
6.2 Types of Data Sources

The types of page data sources are listed below:

New empty XML

When a new empty XML is added in the Add Page Source dialog, a root node is created for the new empty XML data source (*see screenshot below*). No other node or any namespace is created. You can now construct an XML document structure under the root node by using one or both of the following methods:

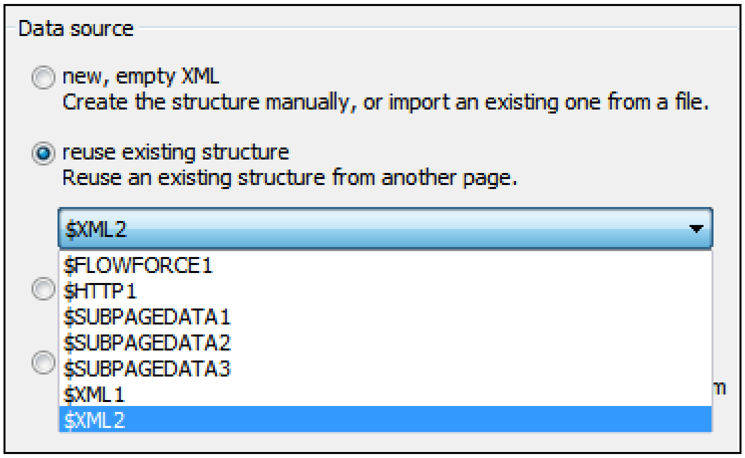
- adding element and attribute nodes to the root node (with the toolbar commands to do this). See Tree Structure for details.
- importing a structure from an XML file via the **Import** command of the root node's context menu.



How to add data to the tree (including by assigning a default file) is described in the section, Tree Data.

Reuse structure from another project page

This option is useful if you want to reuse a structure that you have already created in another page of the project. The reusable data source must be in *another page*—not the same page—of the project, and the page could be either a top page or a sub page. The *Reuse* option is enabled only if a data source exists in another page of the project.



Data source

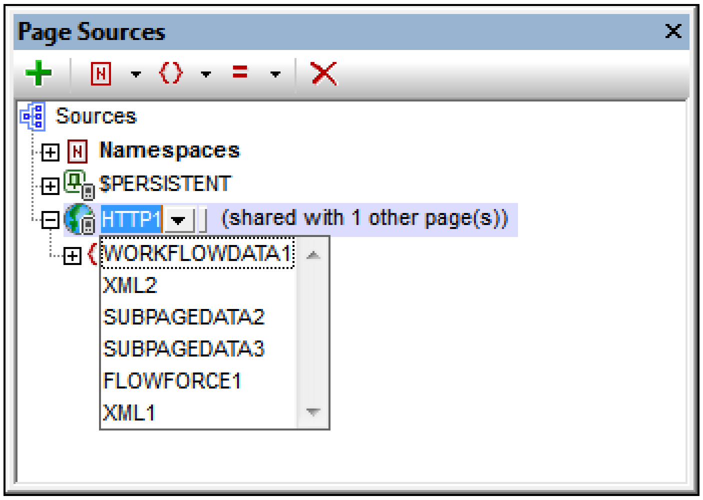
○ new, empty XML
Create the structure manually, or import an existing one from a file.

● reuse existing structure
Reuse an existing structure from another page.

$XML2

$FLOWFORCE1
$HTTP1
$SUBPAGEDATA1
$SUBPAGEDATA2
$SUBPAGEDATA3
$XML1
$XML2

The available data sources are listed by the names of their root nodes in the dropdown list of the options's combo box (*see screenshot above*). Select the data source you want to reuse and click **OK**. A new root node is created with the same name and structure as the reused data source (*see screenshot below*). The number of pages with which the data source is shared is listed (*see screenshot below*) and the name/s of the sharing pages are displayed when you place your mouse over the root node's name in the tree. You can subsequently change the data structure to that of another data source by selecting another reusable data source in the combo box next to the name of the root node (*see screenshot below*).



Page Sources

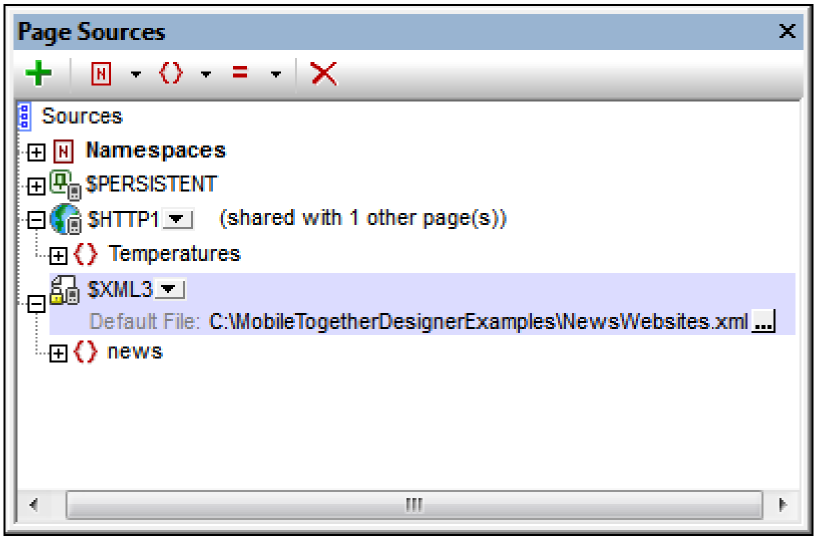
Sources
Namespaces
$PERSISTENT
HTTP1 (shared with 1 other page(s))

WORKFLOWDATA1
XML2
SUBPAGEDATA2
SUBPAGEDATA3
FLOWFORCE1
XML1

How to add data to the tree (including by assigning a default file) is described in the section, Tree Data. How to modify the tree structure is described in the section, Tree Structure.

New XML or HTML structure imported from file

Opens a dialog box in which you select an XML or HTML file that provides the data structure. The XML or HTML file can be selected by browsing for the file, using a file URL, or using a global resource. On clicking **OK**, the structure of the XML file is imported as the structure of the data source (*see screenshot below*). The XML or HTML file is also automatically defined as the default file of this data source. This means that data from the file is used as data for the nodes of the new data source.


Page Sources
Sources
Namespaces
$PERSISTENT
$HTTP1 (shared with 1 other page(s))
Temperatures
$XML3
Default File: C:\MobileTogetherDesignerExamples\NewsWebsites.xml
news

To change the file URL, double-click the URL entry or click the **Additional Dialog** icon at the right-hand side of the entry. If a reusable structure from another project page is available, a combo box next to the name of the root node enables you to select the reusable structure (*see screenshot above*). How to modify the tree structure is described in the section, Tree Structure.

Note that HTML retrieval is done using a correcting parser. As a result, if an imported HTML structure is invalid because of missing elements (according to the HTML 5 specification), then these missing elements are added to the data source tree in the Page Sources Pane. For example:

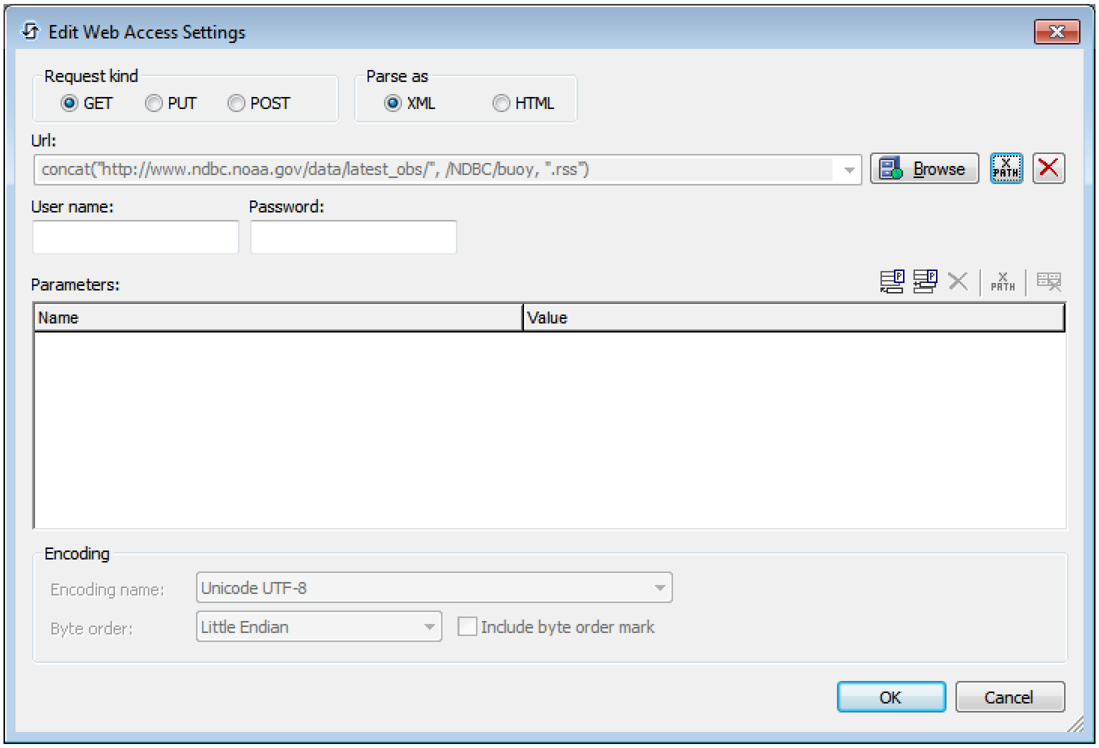
```
<table>
   <tr/>
   <tr/>
</table>
```

will be corrected to:

```
<table>
   <tbody>
      <tr/>
      <tr/>
   </tbody>
</table>
```

New HTTP request with parameters

A data source containing data can be added via an HTTP request and the structure added subsequently. Selecting this option and clicking **OK** displays the Edit Web Access dialog (*screenshot below*). Enter the request kind, the URL, user authentication information, and, optionally, parameters. In the screenshot below, for example, the `GET` request is composed using an XPath expression: It targets a `.rss` page on the `http://www.ndbc.noaa.gov` website. The name of the RSS page is taken from the `/NDBC/buoy` node.



Url: concat("http://www.ndbc.noaa.gov/data/latest_obs/", /NDBC/buoy, ".rss")

User name:

Password:

Encoding name: Unicode UTF-8

Byte order: Little Endian

□ Include byte order mark

On clicking **OK**, the data source is added with the data, but the tree structure is not created. How to create the tree structure is described in the section, Tree Structure.

Note that HTML retrieval is done using a correcting parser. As a result, if an imported HTML structure is invalid because of missing elements (according to the HTML 5 specification), then these missing elements are added to the data source tree in the Page Sources Pane. For example:

```
<table>
   <tr/>
   <tr/>
</table>
```

will be corrected to:

```
<table>
   <tbody>
      <tr/>
      <tr/>
   </tbody>
</table>
```

▼ New DB structure

This option enables you to create a structure and add data from a database. On selecting this option and clicking **OK**, the database (DB) Connection Wizard appears. After making the connection to the DB, you can select the table data to be imported for data source structure and data content. A root node is inserted, with the database name next to the name of the root node. The root node also has the DB structure below it. See the Database-And-Charts tutorial for an example of DBs.

▼ New XQuery tree

Selecting this option and clicking **OK** opens the Edit XPath/XQuery Expression dialog. Enter an XQuery statement that retrieves the required data, and click **OK**. A data source is created that has the structure of the data retrieved by the XQuery statement.

▼ New FlowForce job

Selecting this option and clicking **OK** opens the Edit FlowForce Settings dialog. Enter the FlowForce settings to connect to a FlowForce server and specify a FlowForce job. On clicking **OK**, a new data source is created that contains data retrieved by running the FlowForce job. The structure of the data source can be modified subsequently. How to do this is described in the section, Tree Structure. For information about Altova's FlowForce product, see the FlowForce page on the Altova website.

6.3 Root Nodes

Each page data source is conceptualized as a tree. The **root node** of each tree is identified by a variable name that is unique within each MobileTogether design (project). The terminology used to address and describe the basic parts of data source trees is given below.

`/ = $RootNodeName`	- The **root node** is the topmost node of a tree. - A root node is represented by a variable of the form **`$RootNodeName`**, where `RootNodeName` is a name that identifies a particular tree's root node. *See Names of Root Nodes below.*

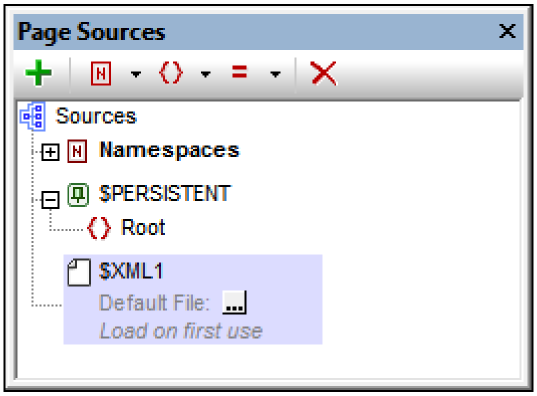
`<RootElement>`
`<Element-1/>`
`...`
`<Element-N/>`
`</RootElement>`
	- The **root element** is the outermost element of an XML document, and contains all the other elements of the document. (In XML language, the root element is also sometimes called the **document element**.)
	- The root element is considered to be a child of the root node.

Accessing tree nodes with XPath

The nodes of a tree can be accessed with XPath expressions. When using an XPath expression, be aware of what the context node is. One node across all the data sources of a page can be set as the XPath Context of the page (by right-clicking that node and selecting **Set as Page XPath Context**). This node will be the context node for all XPath expressions on the page. Exceptions are some controls, such as the charts control, that use their respective page source links as context nodes for XPath expressions used within the control. Whether a page XPath context node is set or not, a node in any tree is always accessible by starting the XPath expression with the tree's root node. For example: **`$XML1/root/element1`**.

Names of root nodes

The names of root nodes are generated automatically when page data sources are added. The screenshot below, for example, shows two root nodes, named `$PERSISTENT` and `$XML1`. The name of a root node can be changed by double-clicking it and editing it.



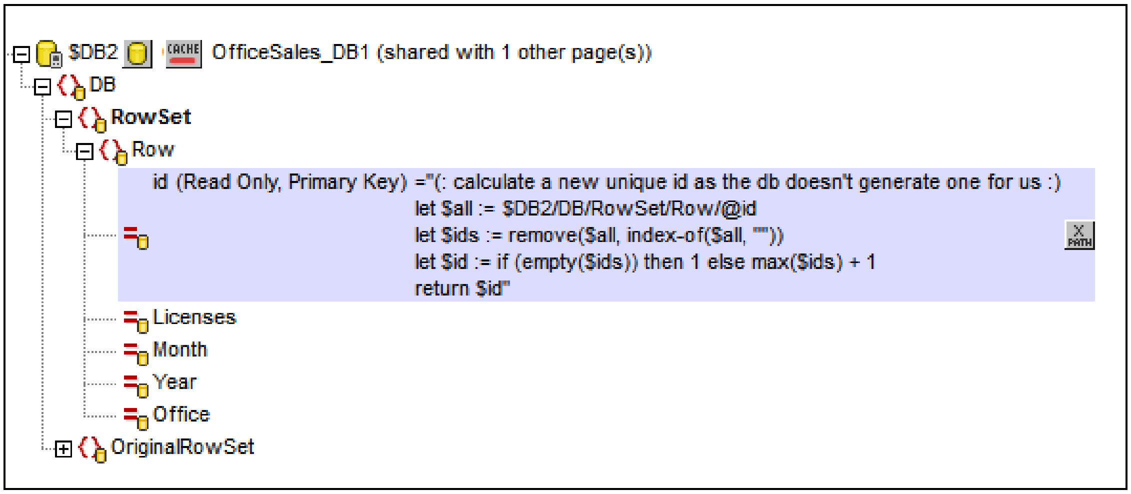
The automatically generated name of the root node depends on: (i) whether the page data source is for the client (`$PERSISTENT`) or for the design; and (ii) what the type of the page data source is: XML, HTML, HTTP-accessed, DB, XQuery, or FlowForce job

$PERSISTENT	The `$PERSISTENT` tree is the tree saved on the client. - This tree is created in every new design with an empty root element called `Root` (*see screenshot above*). A structure must be created for it, either manually, or by importing an XML structure via its context menu's **Import** command. - Nodes of the tree can be assigned data, either static or dynamic (using XPath expressions or by assigning a page source link to a control). - `$PERSISTENT` tree nodes that are assigned as page source links will be updated on the client. This means that when a solution is loaded in the client, nodes that are assigned as page source links will take their data from the `$PERSISTENT` tree, and not from other data sources.
$XML	XML documents, created manually or imported. Default data file is optional.
$HTML	HTML documents, created manually or imported. Default data file is optional.
$HTTP	Documents accessed via HTTP. The specified file provides the data.
$DB	The selected database table provides both structure and data.
$XQ	XQuery documents.
$FLOWFORCE	FlowForce jobs.

6.4 Data Source Trees

Tree structure

A page data source has an XML tree structure. The screenshot below shows the XML tree structure of a data source that is a DB table.



```
$DB2  CACHE  OfficeSales_DB1 (shared with 1 other page(s))
  DB
    RowSet
      Row
        id (Read Only, Primary Key) ="(: calculate a new unique id as the db doesn't generate one for us :)
                                     let $all := $DB2/DB/RowSet/Row/@id
                                     let $ids := remove($all, index-of($all, ""))
                                     let $id := if (empty($ids)) then 1 else max($ids) + 1
                                     return $id"
        Licenses
        Month
        Year
        Office
    OriginalRowSet
```

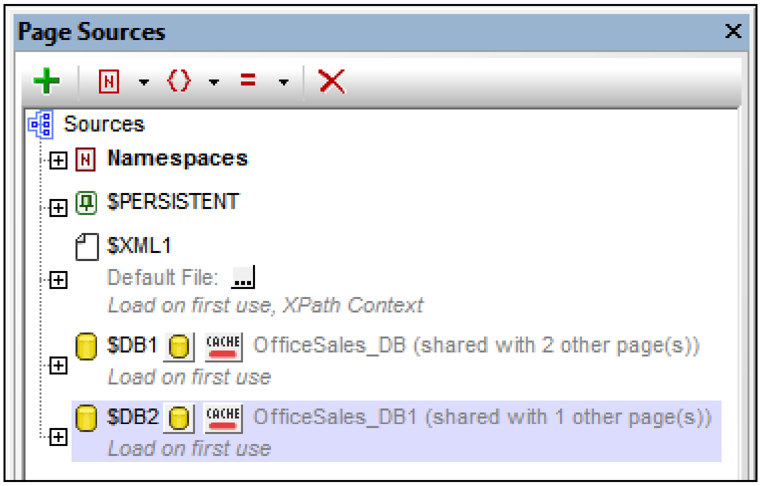
The structure of a data source is created in the following ways:

- Imported from an XML or HTML document, or an XQuery statement, when a data source is added (*see* *Adding Page Data Sources*), or subsequently
- Created manually in the Page Sources Pane by adding elements and attributes to a new empty XML tree (via the pane's toolbar icons)

Accessing tree nodes

The nodes of every data source in the page can be accessed by XPath expressions anywhere in that page.

On each page, any node in any tree can be set as the **XPath context node for that page** (by right-clicking the node and selecting **Set as Page XPath Context**). All XPath expressions on the page will then be evaluated in the context of that node. The XPath context node of a page is indicated with the text *XPath Context*. In the screenshot below, **`$XML1`** is the XPath context node of the page. All XPath expressions on this page will be evaluated relative to **`$XML1`**.


Page Sources
Sources
Namespaces
$PERSISTENT
$XML1
Default File:
Load on first use, XPath Context
$DB1
OfficeSales_DB (shared with 2 other page(s))
Load on first use
$DB2
OfficeSales_DB1 (shared with 1 other page(s))
Load on first use

Whether a page context node is set or not, any node can be addressed by starting the locator expression with the root node of the specific tree. For example, in the XQuery statement that is highlighted in the first screenshot on this page, the second line has a **`let`** expression that locates the **`@id`** node with a locator expression that starts with the root node **`$DB2`**.

A **page source link** is a tree node that is associated with a control. In the cases of some controls—such as the charts control—the control's page source link serves as the context node (XPath origin) of all XPath expressions used to define properties of the control. A page source link is added to a control by dragging the tree node onto the control.

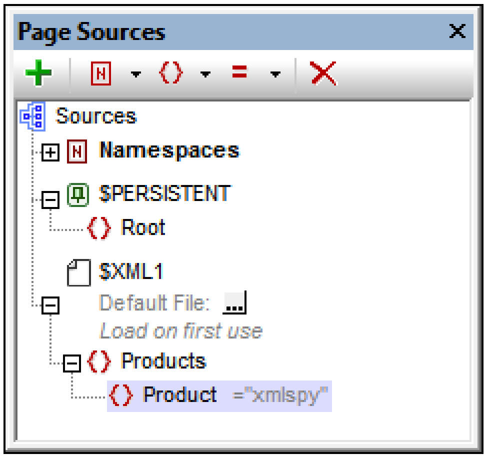
Tree data
The data that is used in a MobileTogether solution is stored in the nodes of the project's data-source trees. This data is obtained in one of the following ways:

- A file is specified as the default file of a data source. This file must have a structure that corresponds to the data source. Its data is then used as the data of the data source.
- A node can be given a fixed value (via the **Ensure exists (fixed value)** command in the node's context menu). This value overrides any value imported from a default file.
- A node is assigned an XPath expression (via the **Ensure exists (XPath value)** command in the node's context menu). The XPath expression generates the content of the node. This value overrides any value imported from a default file.
- A node is updated when an event is defined to trigger an *Update node* action, or when a node is the page source link of a control that provides editing functionality (for example, the combo box and edit field controls).

A **page source link** is a tree node that is associated with a control. In the cases of controls that provide editing functionality—such as the combo box and edit field controls—data edited by the end user can be passed to the tree node. A page source link is added to a control by dragging the tree node onto the control.

Tree Structure

When a data source is added to a page, a root node is created for that data source in the Page Sources Pane (*screenshot below*). Depending upon the type of the page data source that is added, a tree structure might or might not be automatically created. For example, when a new XML tree is created with the structure being imported from an external XML file, a structure is created automatically at the time the data source is added. On the other hand, when a data source is added via an HTTP request, for example, no tree structure is created.



Page Sources

Sources
Namespaces
$PERSISTENT
Root
$XML1
Default File: ...
Load on first use
Products
Product ="xmlspy"

After a data source has been added and a root node has been created, a tree structure can be added in the following ways:

- By importing an XML structure from an external XML file
- By manually building up the tree using commands in the toolbar of the Page Sources Pane

These methods are described below.

Importing an XML structure

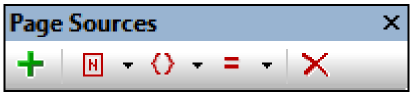
Right-click the root node in the Page Sources Pane, click **Import Structure from XML** in the root node's context menu, and browse for the XML file to use. The following outcomes are possible:

- If the root node has no descendant tree structure, the XML file's root element and its tree structure are imported. The XML file's root element is added as the root element of the data source.
- If the root node has a root element, then this root element and all its descendants are replaced by the root element of the XML file and its tree structure.

The imported tree structure can be modified manually, as described below, and data can be assigned to its nodes (described in the next section, Tree Data).

Manually creating a tree structure

Elements and attributes can be added relative to any node in a tree structure (including the root node), and they can be deleted. Select a node in a data source, and click the appropriate toolbar command (*see toolbar screenshot below*). Temporary elements and attributes are intended to hold data used for calculations or data that for any other reason should not be saved to file. The data of temporary nodes is not saved.



Icon	Command	Does this...
+	**Add Source**	Displays the Add Page Source dialog. A root node is created for the data source that is added. Only one child element can be added to a root node.
N	**Add Namespace**	Inserts or appends a namespace declaration under the *Namespace* entry. Edit the default prefix if you want, and enter a namespace.
{}	**Add Element**	Inserts, appends, or adds a child element relative to the selected node.
=	**Add Attribute**	Inserts, appends, or adds a child attribute relative to the selected node.
✕	**Delete**	Deletes the selected node.

Adding node content manually

You can manually add content to individual nodes via two commands in the context menu of the selected node:

- *Ensure Exists Before Page Load (Fixed Value):* A fixed string value is added as content of the node and displayed in the tree.
- *Ensure Exists Before Page Load (XPath Value):* An XPath expression provides the content of the node. The XPath expression and and Edit XPath icon are displayed in the tree.

The node's content is generated before the page is loaded, and the page is passed with this node content to the client.

Note that content added manually in this way overrides content assigned in other ways.

Tree Data

Editable and read-only data

Data in tree nodes can be editable or non-editable (read-only), depending on whether the tree is that of an editable data source or a read-only data source. Whether a data source is editable or read-only is defined at the time the data source is added. If you wish to change the editable/read-only definition, delete the data source and re-create it with the new definition.

Client actions can change the content of editable nodes. For example, if a combo box is associated with a node of an editable data source, the end user's combo box selection will be passed to the associated node and become its modified value. In the case of read-only data sources, the content of associated nodes is used for display purposes only. These associated nodes are know as **page source links**. A page source link is added to a control by dragging the tree node onto the control.

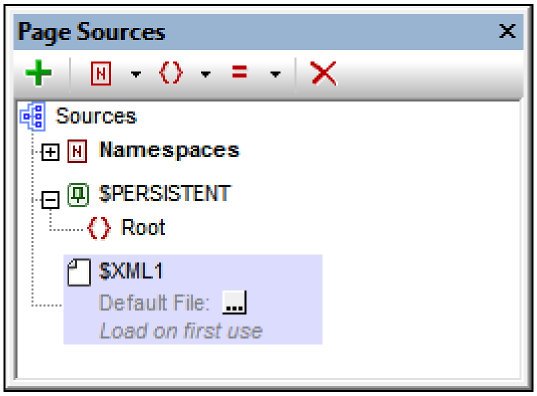
Assigning data to data sources

Data is assigned to nodes (in both editable and read-only data-source trees) in the following ways:

- *Assign a default file*: The data in the default file is passed to the nodes of the tree and becomes the content of the nodes. The structure of the default file must be the same as the data source tree.
- *Add node content manually*: The context menu of every node has commands that allow the content of the node to be specified (the **Ensure exists** commands). If the node already has content assigned by another method (such as via a default file), the manually added node content overrides the previously assigned content.

Assigning a default file

An XML data source can have a default file assigned to it. The data in the default file will be passed to the data source as its data tree. To assign a default file do the following: Just below the root node name of the data source is an entry for the default file (*see screenshot below*).

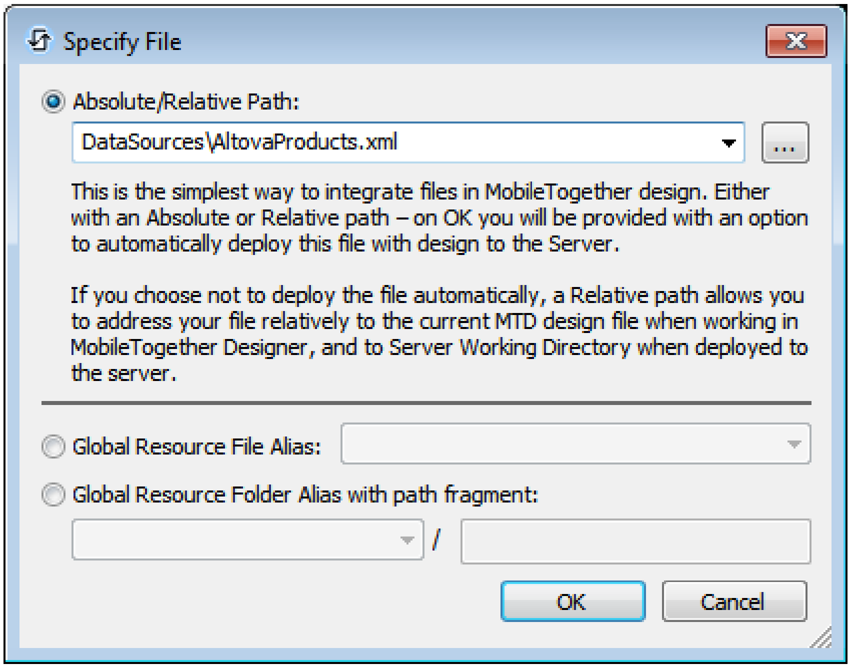


Click the **Additional Dialog** button to display the Specify File dialog (*screenshot below*), select the required file, and click **OK**. The assignment is made and the file path appears in the *Default File* entry. After a default file has been assigned, you can change the assignment by double-clicking the *Default File* entry and browsing for the new default file.

Data from the default file will be used as the data of the data source. However, in order for the data to be used, the default file must have the same structure as the data source. Note that, when a default file is assigned to a data source, its structure is not automatically imported. To import the structure of the XML file, use the context menu command **Import Structure from XML**. You can also manually create the structure of the data source to match the structure of the default file.

Specify File dialog

The Specify File dialog enables you to browse for a file or specify the file via a global resource. Select the option you want.



Specify File

● Absolute/Relative Path:

DataSources\AltovaProducts.xml

This is the simplest way to integrate files in MobileTogether design. Either with an Absolute or Relative path – on OK you will be provided with an option to automatically deploy this file with design to the Server.

If you choose not to deploy the file automatically, a Relative path allows you to address your file relatively to the current MTD design file when working in MobileTogether Designer, and to Server Working Directory when deployed to the server.

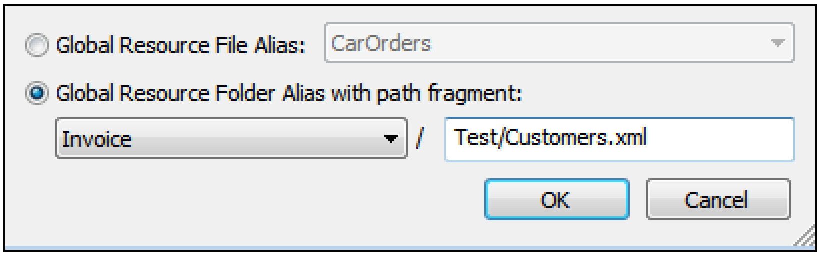
○ Global Resource File Alias:

○ Global Resource Folder Alias with path fragment:

- *Absolute/Relative Path:* You can enter an absolute or relative path or browse for a file. Note that, when the solution is deployed, the location of the default file is resolved with reference to the *Working Directory* as defined in MobileTogether Server settings. If the path in the Specify File dialog is relative, it is resolved relative to the *Working Directory*. If the path in the Specify File dialog is absolute, the default file's containing folder must be a descendant of the *Working Directory*. See the section Location of Project Files for details.

- *Global Resource File Alias:* Select a file alias from the file aliases available in the combo box. The available file aliases will be those currently defined in the Global Resources Definitions file. Each file alias maps to different file resources according to the currently active configuration in MobileTogether Designer (selected via the command **Tools | Active Configuration**). See the section Altova Global Resources for details.

- *Global Resource Folder Alias with path fragment:* Select a folder alias from the folder aliases available in the combo box (*see screenshot below*).



○ Global Resource File Alias: CarOrders

● Global Resource Folder Alias with path fragment:

Invoice / Test/Customers.xml

The available folder aliases will be those currently defined in the Global Resources Definitions file. Each folder alias maps to different folder resources according to the currently active configuration in MobileTogether Designer (selected via the command **Tools | Active Configuration**). The path fragment specifies the rest of the path to the file resource. See the section Altova Global Resources for details.

Adding node content manually

You can manually add content to individual nodes via two commands in the context menu of the selected node:

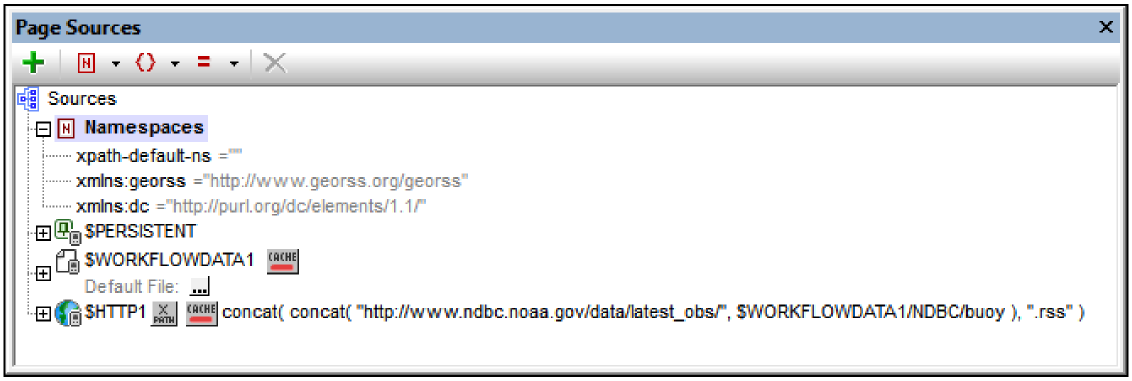
- *Ensure Exists Before Page Load (Fixed Value):* A fixed string value is added as content of the node and displayed in the tree.
- *Ensure Exists Before Page Load (XPath Value):* An XPath expression provides the content of the node. The XPath expression and and Edit XPath icon are displayed in the tree.

The node's content is generated before the page is loaded, and the page is passed with this node content to the client.

Note that content added manually in this way overrides content assigned in other ways.

6.5 Namespaces in the Project

Namespaces are important for correctly identifying nodes, and for correctly locating nodes with the use of XPath expressions. The *Namespaces* item in the Page Sources Pane (*screenshot below*) contains all the namespaces that have been declared for the project, irrespective of what page is currently active in Page Design View.



xpath-default-ns =""
xmlns:georss ="http://www.georss.org/georss"
xmlns:dc ="http://purl.org/dc/elements/1.1/"

concat(concat("http://www.ndbc.noaa.gov/data/latest_obs/", $WORKFLOWDATA1/NDBC/buoy), ".rss")

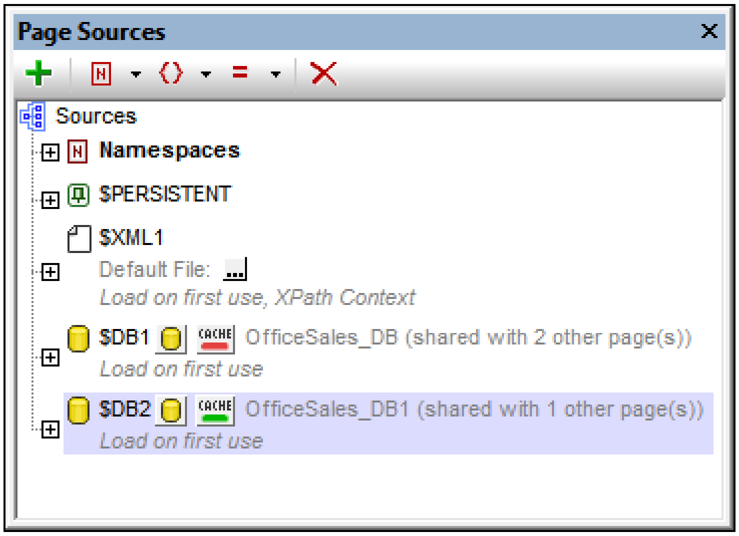
Namespaces can be declared in two ways:

- *Automatic declaration on data import:* When an external XML file is added as a page source, namespaces in the source are automatically imported into the design and declared for the scope of the entire project. They then appear under the *Namespaces* item in the Page Sources Pane (*see screenshot above*). The namespace prefixes are automatically set to match the original prefixes if such matching creates no ambiguities in the design. Prefixes assigned in the namespace declaration are used in node names, and must be used in XPath expressions that are intended to locate these nodes in the data source.
- *User-defined:* You can also add namespaces by clicking the **Namespace** icon in the toolbar of the Page Sources Pane (*screenshot above*). Being able to add your own namespaces to a project enables you to create nodes that belong to one or more user-declared namespaces. This is useful for disambiguating between nodes that have the same local name.

To delete a namespace, select it and click **Delete** in the pane's toolbar.

6.6 Caches

The data in external resources (XML files and DBs) that are linked to a data source can be cached on the server. Whether a data source has been cached on the server is indicated by the color of the Cache icon of the data source (*see screenshot below*). A red cache symbol indicates that the data source has not been cached. A green symbol indicates that the data source has been cached. If a data source is not linked to an external resource (for example via a default file), then it has no cache symbol (as for `$XML1` in the screenshot below).



Page Sources
Sources
Namespaces
$PERSISTENT
$XML1
Default File:
Load on first use, XPath Context
$DB1 CACHE OfficeSales_DB (shared with 2 other page(s))
Load on first use
$DB2 CACHE OfficeSales_DB1 (shared with 1 other page(s))
Load on first use

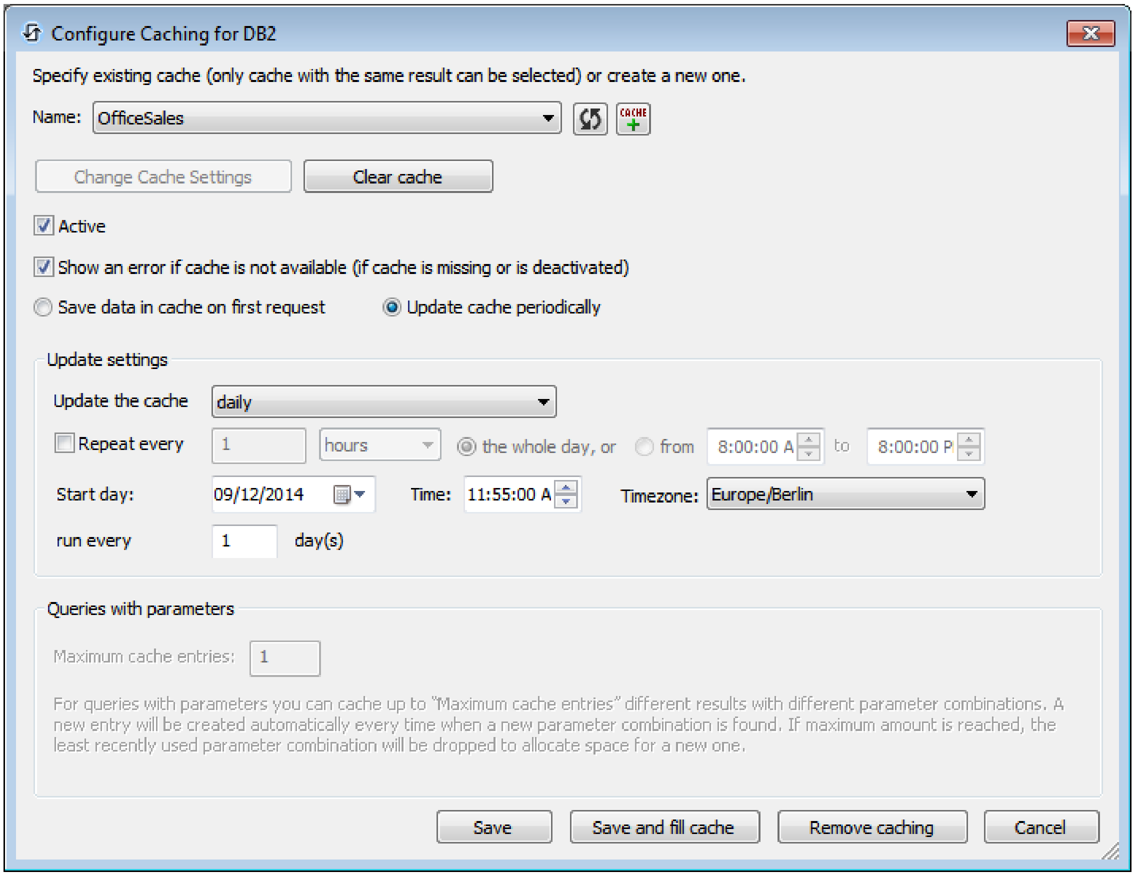
Reasons for caching

There are two main reasons to create caches: (i) If a page data source generates reports slowly (for example, a large database); (ii) If a data source is not modified often. In cases of both these types, execution of a solution would be faster if data is taken from data caches on the server.

In order to keep caches up-to-date, the frequency of cache updates can be specified when the cache is created. Once a cache has been defined in MobileTogether Designer, it can be used by the data sources of different designs, providing the underlying data structure is compatible. If a data source is defined as having a cache, the cached data will be used when the solution is run. Caches can be used as soon as the solution has been deployed to the server.

Creating caches

To create a cache for a data source, or to edit the settings of a cache that has already been created, click the **Cache** icon of the data source or select **Cache Settings** from the context menu of the root node of the data source. This displays the Configure Caching dialog (*screenshot below*).



Specify existing cache (only cache with the same result can be selected) or create a new one.

Name: OfficeSales

☑ Active

☑ Show an error if cache is not available (if cache is missing or is deactivated)

○ Save data in cache on first request ● Update cache periodically

Update settings

Update the cache: daily

□ Repeat every 1 hours ● the whole day, or ○ from 8:00:00 A to 8:00:00 P

Start day: 09/12/2014 Time: 11:55:00 A Timezone: Europe/Berlin

run every 1 day(s)

Queries with parameters

Maximum cache entries: 1

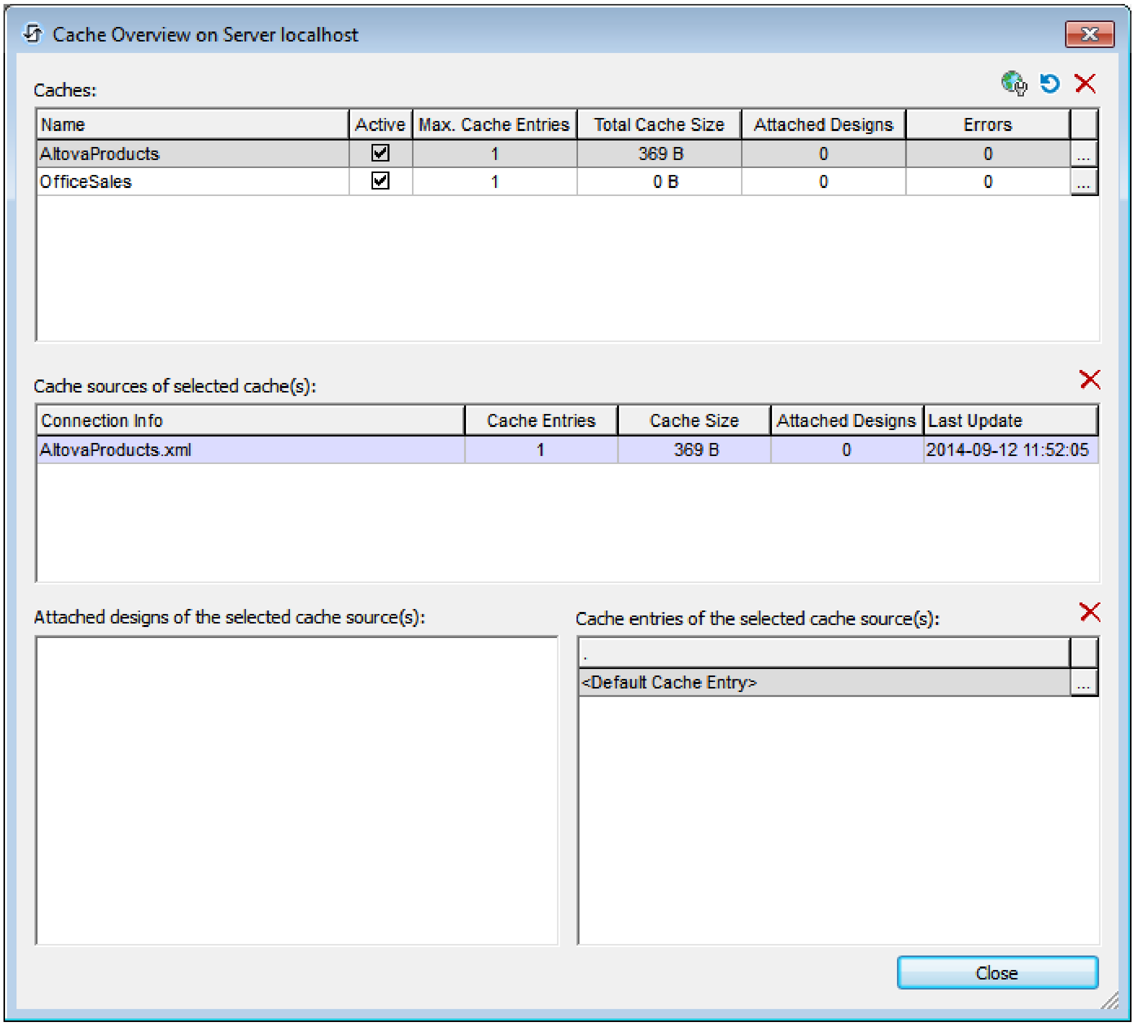
For queries with parameters you can cache up to "Maximum cache entries" different results with different parameter combinations. A new entry will be created automatically every time when a new parameter combination is found. If maximum amount is reached, the least recently used parameter combination will be dropped to allocate space for a new one.

In this dialog, you can do the following:

- Refresh the connection to the server to check for the latest caches that match the structure of this data source.
- Add a new cache for this data source.
- Click Change Cache Settings to edit the settings of a cache that already exists.
- Activate/deactivate the cache.
- Specify whether an error should be displayed if the cache is missing when the solution is run.
- Specify the frequency with which the cache should be filled.

Cache Overview

The Cache Overview dialog (*screenshot below*) is accessed with the menu command **Project | Cache Overview**.



Cache Overview on Server localhost

Caches:

Name	Active	Max. Cache Entries	Total Cache Size	Attached Designs	Errors
AltovaProducts	☑	1	369 B	0	0
OfficeSales	☑	1	0 B	0	0

Cache sources of selected cache(s):

Connection Info	Cache Entries	Cache Size	Attached Designs	Last Update
AltovaProducts.xml	1	369 B	0	2014-09-12 11:52:05

Attached designs of the selected cache source(s):

Cache entries of the selected cache source(s):

<Default Cache Entry>

The dialog provides an overview of all the caches on the server. In it, you can also do the following:

- Activate/deactivate caches.
- Delete caches.

6.7 Context Menus

Commands in the context menus of items in the Page Sources Pane are described below. They are organized into two groups:

- Context menus of root nodes
- Context menus of tree nodes.

Context menus of root nodes
The commands listed below are available in the context menus of root nodes (`$XML`, `$DB`, `$HTML`, etc). In addition to the commands that are common across all types of data sources (XML, DB, HTML, etc), some types of data sources have commands that are specific to its type (for example, commands for DB data sources). The specificity of such commands is noted where relevant.

▾ Insert, Append, Add Child

Enables the addition of elements and attributes relative to the selected node. **Insert** adds the node before the selected node. **Append** adds the node after the last node of that type. If you wish to add a node immediately after the selected node, go to the following node and use the **Insert** command.

▾ Keep data on server

To reduce the amount of data transmitted over the mobile data network—which improves the performance of any mobile solution—MobileTogether lets you specify exactly which data to transmit to client devices and which data to keep on the server. For example, if a certain data set is only necessary to display a graph, then that data can be kept on the server. The graph image will be rendered by the server and transmitted to the client without the underlying data being transferred over the mobile network. For large data sets this produces a significant performance boost.

This toggle command specifies whether data in the tree is stored on the server. Note that if it is stored on the server, then it cannot be defined as persistent. See the **Persist data on client** command below.

▾ Read only data

A toggle command which specifies that data in the tree is read only. A read-only data tree is used to provide data for display and calculations. It cannot be used to hold data that needs to be edited.

Persist data on client

A toggle command that makes a tree a persistent tree. Any number of trees can be defined as persistent. When a tree is defined as persistent, data in it is retained on the client device after the solution is exited. When the solution is opened the next time, on that client, the persistent data is displayed. If a tree is defined as persistent, then it cannot be stored on the server. See the **Keep data on server** command above.

Load data

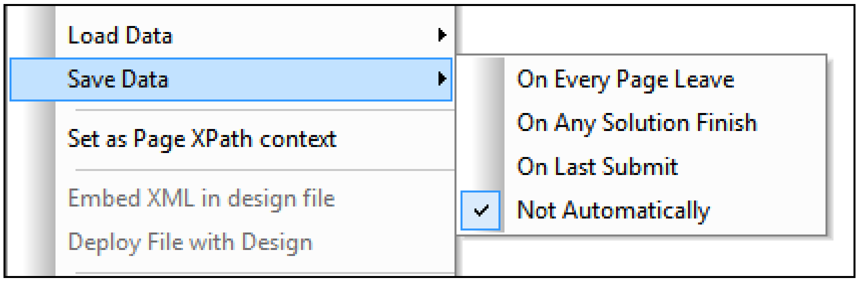
This command rolls out a sub-menu with the following mutually exclusive options (only one can be selected):

- *On First Use:* Loads the tree on entering a page where it is used. Once loaded, it will not automatically be reloaded anymore. If you share the same tree on multiple pages, then the first time one of such pages is opened (doesn't matter whether it is a Top page or Sub page), the tree will be loaded and will remain in memory.
- *On Every Page:* Reloads the tree every time you open a page containing the tree, whether the page is a Top page or Sub page. You must be careful with this option: It can slow the processing if loading takes significant time. But this will ensure that the data is retrieved afresh for every page.
- *Not Automatically:* The tree will not automatically be loaded for you. You must use the Reload, Load from File, or Load from HTTP actions to load it. Alternatively, it can be created completely from scratch with the Append Node and Insert Node actions, without having to load data from any source. Note that you can use any of these five actions independently of the Load Data setting. That is, these actions can be used to reload your tree at specific moments even if **Load Data** is set to *On First Use* or *On Every Page*.

The default is *On First Use*.

Save data

The **Save Data** command rolls out a sub-menu with the following mutually exclusive options (only one can be selected):



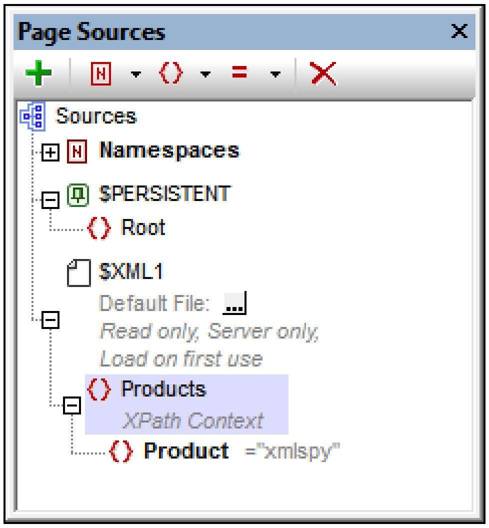
- *On Every Page Leave:* Data in the tree is saved every time a page containing that tree is exited.

- *On Any Solution Finish:* Data in the tree is saved when the solution is exited, no matter at what point or how the solution is exited.
- *On Last Submit:* Data in the tree is saved when the workflow progresses as designed, from first page to last page, and when the last **Submit** button is tapped. If this option is selected and the solution is exited before the last **Submit** button is tapped, then data in the tree will not be saved.
- *Not Automatically:* The tree will not automatically be saved. You must use the Save, Save to File, or Save to HTTP actions to save data.

The default is *Not Automatically*.

Set as page XPath context

Sets the selected node as the XPath context node of the page. An annotation to the effect is displayed below the node (*see screenshot below*). The XPath context of the page is the context node for all XPath expressions on the page.



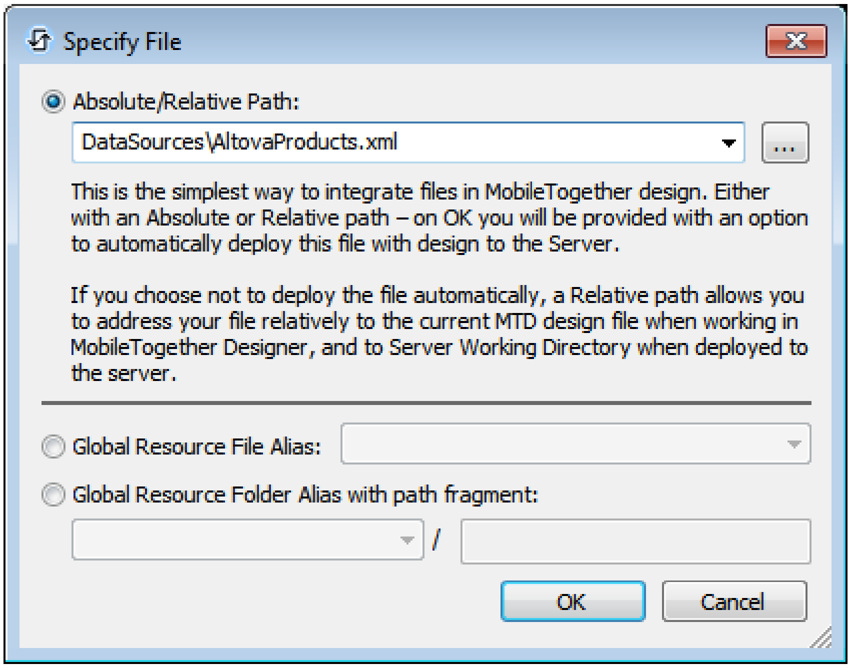
This command can be toggled on and off. So, you can enable a node as the XPath context node of the page, or you can toggle off a node's setting as the XPath context node of the page. If a node is set as the XPath context node when another node already has this setting, then the setting is toggled off for the previously assigned node and toggled on for the newly assigned node.

Choose default file

Displays the Specify File dialog (*screenshot below*), in which you specify the file to use as the default file. Data from the default file will be used as the data of the data source. However, in order for the data to be used, the default file must have the same structure as the data source. Note that, when a default file is assigned to a data source, its structure is not automatically imported. To import the structure of the XML file, use the context menu command **Import**

Structure from XML; *see below*. You can also manually create the structure of the data source to match the structure of the default file.

The Specify File dialog enables you to browse for a file or specify the file via a global resource. Select the option you want.



● Absolute/Relative Path:

DataSources\AltovaProducts.xml

This is the simplest way to integrate files in MobileTogether design. Either with an Absolute or Relative path – on OK you will be provided with an option to automatically deploy this file with design to the Server.

If you choose not to deploy the file automatically, a Relative path allows you to address your file relatively to the current MTD design file when working in MobileTogether Designer, and to Server Working Directory when deployed to the server.

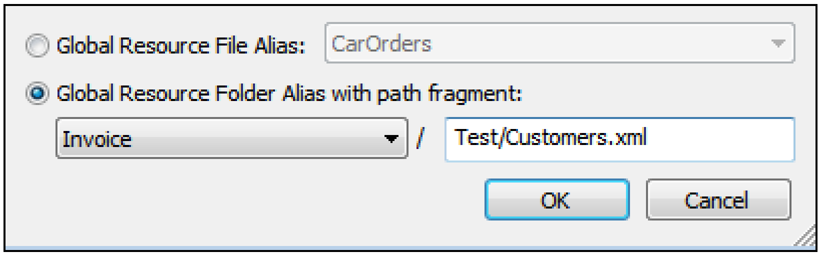
○ Global Resource File Alias:

○ Global Resource Folder Alias with path fragment:

- *Absolute/Relative Path:* You can enter an absolute or relative path or browse for a file. Note that, when the solution is deployed, the location of the default file is resolved with reference to the *Working Directory* as defined in MobileTogether Server settings. If the path in the Specify File dialog is relative, it is resolved relative to the *Working Directory*. If the path in the Specify File dialog is absolute, the default file's containing folder must be a descendant of the *Working Directory*. See the section Location of Project Files for details.

- *Global Resource File Alias:* Select a file alias from the file aliases available in the combo box. The available file aliases will be those currently defined in the Global Resources Definitions file. Each file alias maps to different file resources according to the currently active configuration in MobileTogether Designer (selected via the command **Tools | Active Configuration**). See the section Altova Global Resources for details.

- *Global Resource Folder Alias with path fragment:* Select a folder alias from the folder aliases available in the combo box (*see screenshot below*).



○ Global Resource File Alias: CarOrders

● Global Resource Folder Alias with path fragment:

Invoice / Test/Customers.xml

The available folder aliases will be those currently defined in the Global Resources Definitions file. Each folder alias maps to different folder resources according to the currently active configuration in MobileTogether Designer (selected via the command **Tools | Active Configuration**). The path fragment specifies the rest of the path to the file resource. See the section Altova Global Resources for details.

Embed XML in design file

This command is enabled when the root node of an XML data source is selected that has a default file assigned to it. Selecting it embeds the XML data source in the design (or project) file. After a data source is embedded, the property Embedded is added to the annotation of the root node. See the sections Location of Project Files and Embed XML in Design File for more information about (i) the advantages and disadvantages of embedding, and (ii) alternatives to embedding.

Deploy file with design

This toggle command is enabled when a data source is associated with a deployable file, typically a default file. The deployable file will already be listed in the Files Pane.

- Toggling the command on selects the check box of the file in the Files Pane, thus deploying it.
- Toggling the command off de-selects the file in the Files Pane; the file will not be deployed.

Note that when a file is first added to the design, you are prompted about whether you wish to deploy the file or not.

Reload structure

Reloads the structure of the selected data source. The command is enabled only if the structure is based on an external resource, such as an XML file or DB. In the case of XML files, this means that if there is a default file, then the command is enabled.

▾ Import structure from XML

Opens a Browse dialog in which you can select the XML or HTML file from which to import the XML structure of the data source tree. If the tree already contains a structure, you will be prompted about whether one or multiple nodes of the existing structure should be retained or not. If you choose to retain the existing structure and the new structure cannot be merged into the existing structure, then the new structure is imported as a sibling of the existing structure.

Note: When a structure is imported from an XML file, the file is set as the default file and the file's data is also imported.

▾ Export structure to XML

Opens a Browse dialog in which you can select an XML file to which to export the XML structure of the data source tree. You can choose an existing XML file or create a new on. If you choose an existing file, the file's existing data will be overwritten by the exported structure.

▾ Choose DB data source [$DB only]

This command is enabled for database type (`$DB`) root nodes. It opens MobileTogether Designer's Database Connection Wizard, with which you can connect to a DB data source. After connecting to the DB, you can select the table to add as the data source. If the new table data cannot be merged into the existing structure, then the new table structure is created as a sibling of the existing structure.

▾ Choose DB tables and views [$DB only]

This command is enabled for database type (`$DB`) root nodes. It opens MobileTogether Designer's Database Object Selector window, in which you can select the DB tables and views to import as a data source.

▾ Create OriginalRowSet [$DB only]

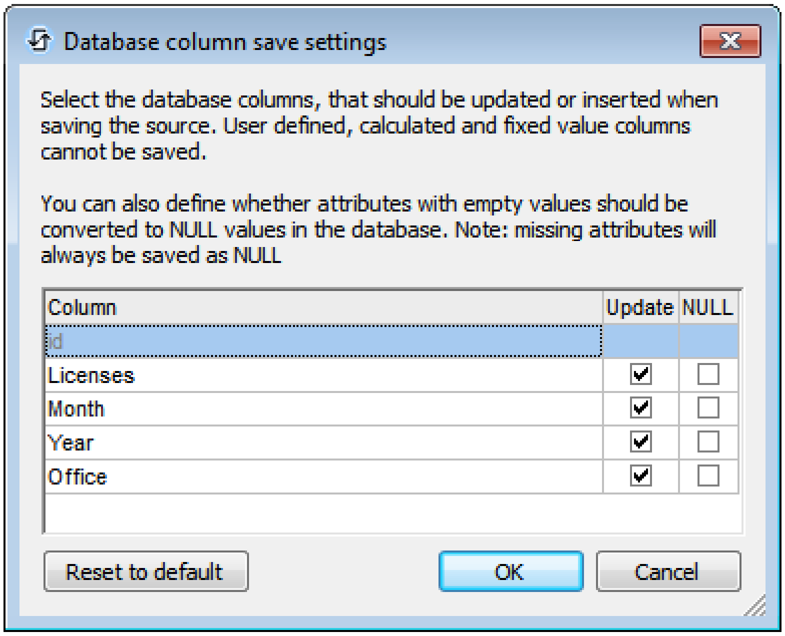
In order for data to be edited and saved, the tree of the data source must also include an `OriginalRowSet` element, which is a copy of the `RowSet` element. The original data is saved in the `OriginalRowSet` element, while edited data is saved in the `RowSet` element. When the data source is saved, the difference between the two trees, `OriginalRowSet` and `RowSet`, is calculated, and the data source is modified accordingly. If the modification is successful, then the modified data from `RowSet` is copied to `OriginalRowSet` so that `OriginalRowSet` is again current.

To create an `OriginalRowSet` for a data source, right-click the root node of the data source and toggle on the command **Create OriginalRowSet**.

The **Create OriginalRowSet** command is enabled for database type (`$DB`) root nodes. It is a toggle command that creates/removes an `OriginalRowSet` data structure that contains the original data of the data source. User modified data is saved in the main structure created from the data source. When modified data is saved back to the DB, the `OriginalRowSet` structure is modified to contain the data newly saved to the DB. Till the time modified data is saved to the DB, the original DB data is retained in the `OriginalRowSet` structure. This ensures that the original DB data is still available in the tree.

Filter columns [$DB only]

This command is enabled for database type (`$DB`) root nodes. It opens the Database Column Save Settings dialog, in which you can specify which columns should be saved to the DB data source.



Select the database columns, that should be updated or inserted when saving the source. User defined, calculated and fixed value columns cannot be saved.

You can also define whether attributes with empty values should be converted to NULL values in the database. Note: missing attributes will always be saved as NULL

Column	Update	NULL
id		
Licenses	☑	□
Month	☑	□
Year	☑	□
Office	☑	□

The dialog displays the columns of the data source. Columns that cannot be updated (because they are user-defined, fixed-value, or calculated-value) will not have an *Update* option. Deselect the columns you do not want to update. Attributes with empty values can be converted to `NULL` values in the DB by checking that column's *NULL* check box. Note that missing attributes will always be saved as `NULL`. If you wish to reset the Save settings so that all columns are updtaed, click **Reset to default**.

Cache settings

These two commands are available on root nodes and roll out sub-menus with load and save options.

- *Loading options:* Data can be loaded when the page source is first loaded, every time when the page source is loaded, or not loaded at all automatically. In the last case, loading can be specified via a page action. Default is *On First Use*, that is, the data source is loaded when it is used for the first time.
- *Saving options:* Data can be saved automatically when the page is left, when a solution finishes, when the last submit action is carried out, or not at all automatically. In the last case, saving can be specified via a page action. Default is *Not Automatically*.

Context menus of tree nodes

The commands listed below are available in the context menus of **tree nodes** (all elements and attributes except the root node). In addition to the commands that are common across all types of data sources (XML, DB, HTML, etc), some types of data sources have commands that are specific to its type (for example, commands for DB data sources). The specificity of such commands is noted where relevant.Tree nodes are element or attributes. All tree nodes of all kinds of

▼ Insert, Append, Add Child

Enables the addition of elements and attributes relative to the selected node. **Insert** adds the node before the selected node. **Append** adds the node after the last node of that type. If you wish to add a node immediately after the selected node, go to the following node and use the **Insert** command.

▼ Ensure exists on load (fixed value)

This context menu item is available for tree nodes. A fixed value can be provided for the selected node when the page is loaded. Click the command and enter the value. This command is a toggle command. So, clicking it when a fixed value is already assigned will remove the value.

▼ Ensure exists on load (XPath value)

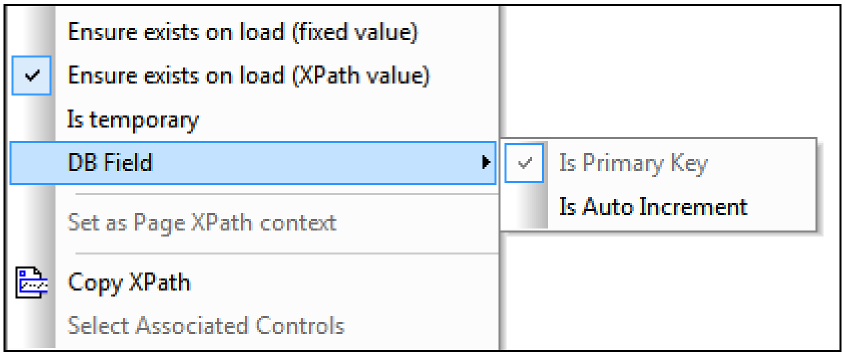
This context menu item is available for tree nodes. An XPath-generated value can be provided for the selected node when the page is loaded. On clicking the command, the Edit XPath/ XQuery Expression Dialog is displayed. Enter the XPath expression to generate the value of the node. This command is a toggle command. So, clicking it when an XPath value is already assigned will remove the value.

Is temporary

Sets the selected node as a temporary node. Data in temporary nodes is not saved when the tree is saved. Since temporary nodes are outside the framework of valid workflow data, they are intended for use in calculations and for storage of any data that is not wanted as part of the final data.

DB field

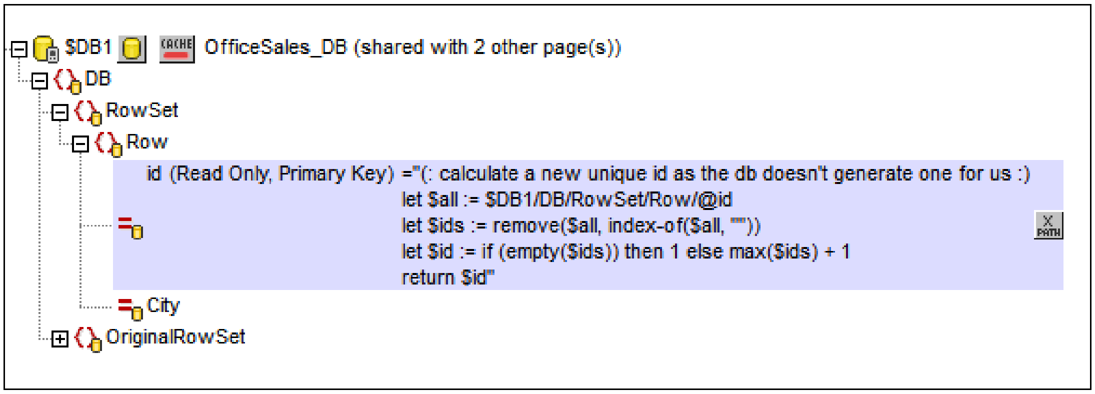
This context menu item is available for DB nodes, and has a sub-menu with two commands:


Ensure exists on load (fixed value)
✓ Ensure exists on load (XPath value)
Is temporary
DB Field
✓ Is Primary Key
Is Auto Increment
Set as Page XPath context
Copy XPath
Select Associated Controls

- *Is Primary Key:* Sets the selected node as the primary key column if a primary key has not already been auto-detected during retrieval from the DB.
- *Is Auto Increment:* Sets the selected node to auto increment. The node then becomes read-only.

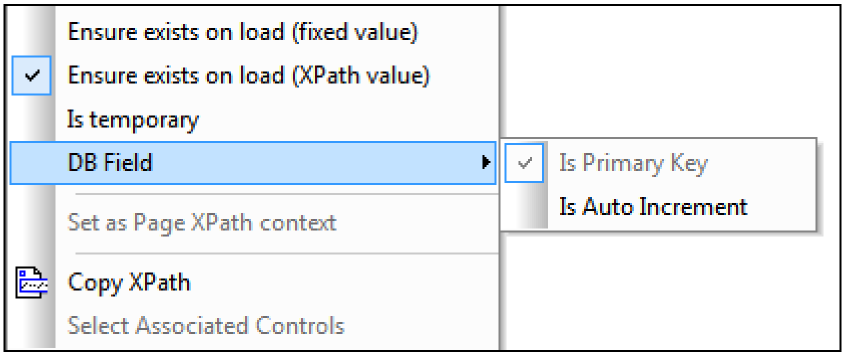
Primary keys in MobileTogether Designer

Primary keys in DBs typically are auto-incrementing. When this is the case and a new row is added to a table, the primary key column of the added row is automatically incremented. In MobileTogether Designer, when a table is retrieved the primary key and auto-increment information is automatically retrieved and displayed in the Page Sources Pane (*see screenshot below*).



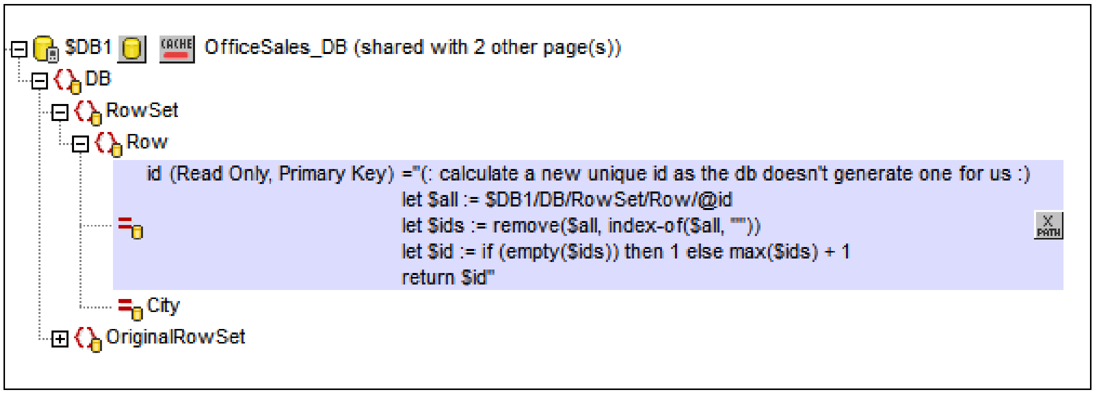
```
id (Read Only, Primary Key) ="(: calculate a new unique id as the db doesn't generate one for us :)
                              let $all := $DB1/DB/RowSet/Row/@id
                              let $ids := remove($all, index-of($all, ""))
                              let $id := if (empty($ids)) then 1 else max($ids) + 1
                              return $id"
```

If auto-retrieval of this information was not successful, the context menu of tree nodes contains toggle commands that enable you to correctly annotate nodes (*see screenshot below*).


Ensure exists on load (fixed value)
Ensure exists on load (XPath value)
Is temporary
DB Field
Is Primary Key
Is Auto Increment
Set as Page XPath context
Copy XPath
Select Associated Controls

If the primary key column is not auto-incrementing, new primary key values for appended rows must be automatically generated using an XQuery expression. This is because primary key columns cannot be edited. The XQuery expression is inserted by using the primary-key node's context menu command, **Ensure Exists before Page Load (XPath Value)**. In the example below, a new value is generated for the primary key `@id` by using the following XQuery expression:

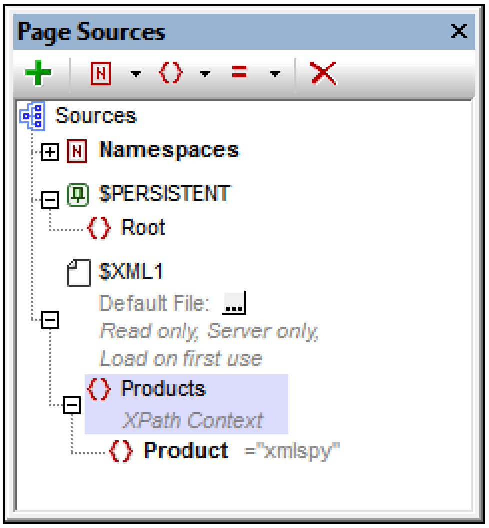
```
let $all := $DB1/DB/RowSet/Row/@id
let $ids := remove($all, index-of($all, ""))
let $id := if (empty($ids)) then 1 else max($ids) + 1
return $id
```



```
id (Read Only, Primary Key) ="(: calculate a new unique id as the db doesn't generate one for us :)
                                let $all := $DB1/DB/RowSet/Row/@id
                                let $ids := remove($all, index-of($all, ""))
                                let $id := if (empty($ids)) then 1 else max($ids) + 1
                                return $id"
```

Set as page XPath context

Sets the selected node as the XPath context node of the page. An annotation to the effect is displayed below the node (*see screenshot below*). The XPath context of the page is the context node for all XPath expressions on the page.



This command can be toggled on and off. So, you can enable a node as the XPath context node of the page, or you can toggle off a node's setting as the XPath context node of the page. If a node is set as the XPath context node when another node already has this setting, then the setting is toggled off for the previously assigned node and toggled on for the newly assigned node.

Copy XPath

Copies the XPath locator expression of the node to the clipboard. The locator expression starts at the root node. For example: `$XML1Products/Product` is the locator expression of the `Product` node.

Select associated controls

Selects the controls in the design diagram that are associated with the selected node. Such associations are typically page source links between the node and page controls.

Chapter 7

Page Design

7 Page Design

A page design consists of page controls, such as combo boxes, edit fields, labels, and charts. Page controls are dragged from the Controls Pane into the design and dropped at the required location in the page design. Controls can display and process data from data sources. These data sources are displayed in the form of trees in the Page Sources Pane. Data is associated with a control, typically, by dragging a data node from the tree onto the control. These data associations are called the page source links of the control.

The data in page source links can be used in the following ways:

- For display in a control, either directly or after being processed
- For triggering control events and actions, or page events and actions

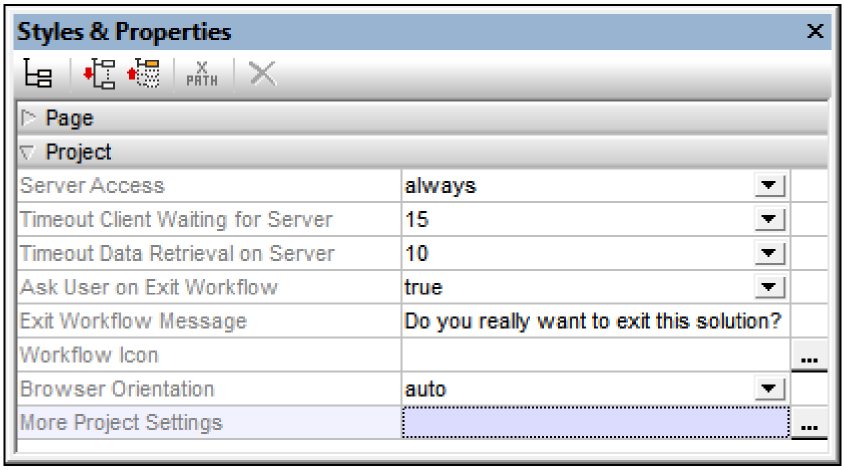
---

In this section
This section is organized as follows:

- Page Properties, which explains the page properties set in the Styles & Properties Pane
- Edit XPath/XQuery Expression Dialog, which describes the dialog for editing XPath/XQuery expressions
- Controls, which describes the various controls available for inclusion in page designs
- Events, which describes page events and control events
- Actions, and describes a set of common actions that can be defined for both page and control events
- Charts, which explains how charts can be configured in the design

7.1 Page Properties

Project properties are defined in the Styles & Properties Pane and are described below. The screenshot below shows default values.



Server Access	always
Timeout Client Waiting for Server	15
Timeout Data Retrieval on Server	10
Ask User on Exit Workflow	true
Exit Workflow Message	Do you really want to exit this solution?
Workflow Icon	
Browser Orientation	auto
More Project Settings	

Server Access

This option specifies the level of server access while the solution runs.

- *Always:* Connection to the server is required in order to run the solution, and the server is continually accessed while the solution runs.
- *On Demand:* The MobileTogether Client app runs the solution on its own; it connects to the server only when it needs to exchange data with the server. To run the solution, the app uses data in the internal `$PERSISTENT` tree.
- *Never:* The MobileTogether Client app runs the solution entirely on its own and without needing any data from the Internet.

Timeout: Client Waiting for Server

The amount of time the client waits for a response from the server. The value is an integer value in seconds that can be entered or selected from the dropdown list of the combo box. The default value is 15 seconds. If the timeout period is exceeded, then an error message is displayed on the client.

Timeout: Data Retrieval on Server

The amount of time the server waits for data to be retrieved from a source external to the server (from a DB or URL, for example). The value is an integer value in seconds that can be entered or selected from the dropdown list of the combo box. The default value is 10 seconds. If the timeout period is exceeded, then an error message is displayed on the client. An exception to this is when load actions have the setting *On error* set to `Continue`.

Ask User on Exit Workflow

A boolean setting that sets whether the user is asked to confirm (workflow) solution exit. Select `true` or `false` in the combo box. Default value is `true`. If `true`, the text defined as the value of the next property, *Exit Workflow Message*, is displayed before the solution exits.

Exit Workflow Message

The text of the message that is displayed as a prompt to confirm (workflow) solution exit. The message is displayed only if the previous property, *Ask User on Exit Workflow*, is set to `true`. The default message is: *Do you really want to exit this solution?*

Workflow Icon

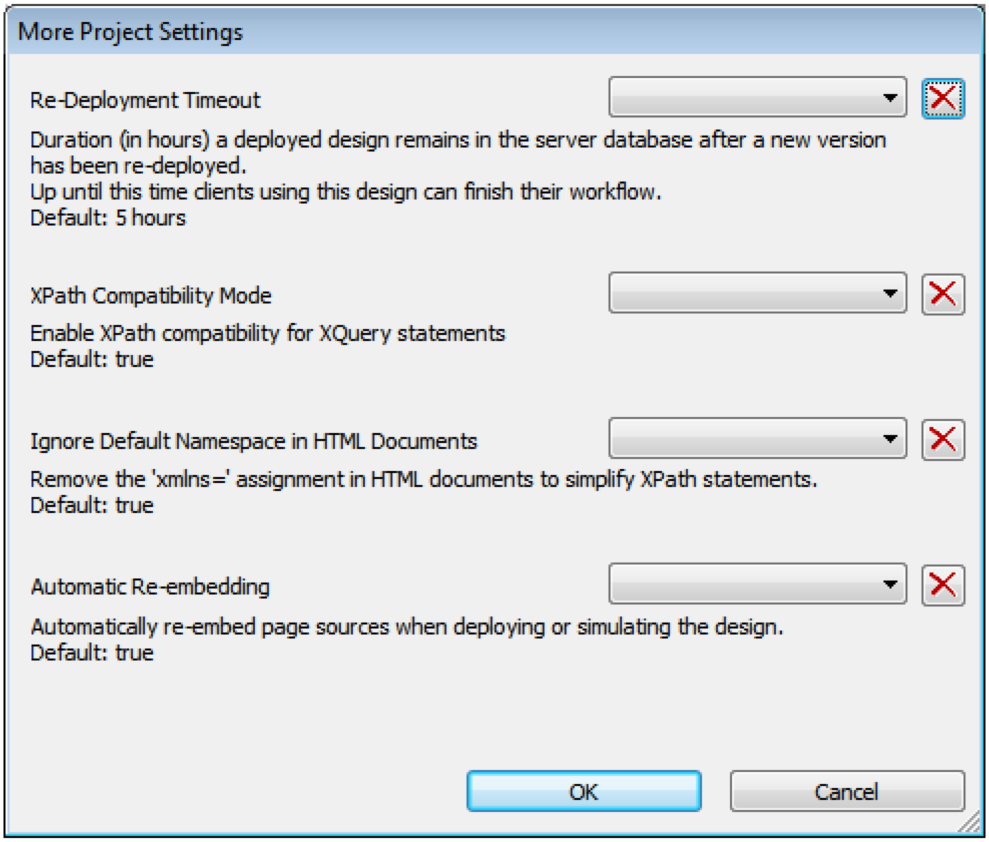
Clicking the property's **Additional Dialog** button displays a Browse dialog in which you can browse for the PNG image file to use as the icon of the project. By default, the MobileTogether icon will be used.

Browser Orientation

The combo box options are: `Auto`; `Force Portrait`; and `Force Landscape`. The `Auto` option enables the automatic switching of orientation to match the orientation of the device's viewport.

More Project Settings

Clicking the property's **Additional Dialog** button displays the More Project Settings dialog (*screenshot below*).



More Project Settings

Re-Deployment Timeout

Duration (in hours) a deployed design remains in the server database after a new version has been re-deployed.
Up until this time clients using this design can finish their workflow.
Default: 5 hours

XPath Compatibility Mode

Enable XPath compatibility for XQuery statements
Default: true

Ignore Default Namespace in HTML Documents

Remove the 'xmlns=' assignment in HTML documents to simplify XPath statements.
Default: true

Automatic Re-embedding

Automatically re-embed page sources when deploying or simulating the design.
Default: true

The following settings can be defined:

- *Re-deployment timeout:* The time in hours after a new version of a solution is deployed that the superseded solution is kept on the server. This overlap time enables clients currently using the older solution to complete work. The default is 5 hours.
- *XPath compatibility mode:* When set to `true`, XQuery constructs that are invalid in XPath are resolved in such a way that XQuery statements containing these constructs are compatible with XPath and can be used where XPath expressions are allowed. Currently, this concerns XQuery entity and character references, which are allowed in XQuery, but not in XPath. When *XPath compatibility mode* is set to `true`, XQuery entity and character references are read in XPath as text; they are not resolved to the specified character or entity. The default value for this setting is `true`. Use `False` if you wish to resolve XQuery character and entity references to their correct string literals.
- *Ignore default namespace in HTML documents:* Since only one default namespace is allowed in an XML document, not ignoring the default namespace in HTML documents could create errors in reading XML data sources. The default is `true`: The HTML default namespace is ignored.
- *Automatic re-embedding:* Re-embedding page data sources enables the data to be up-to-date. The default is `true`.

7.2 Edit XPath/XQuery Expression Dialog

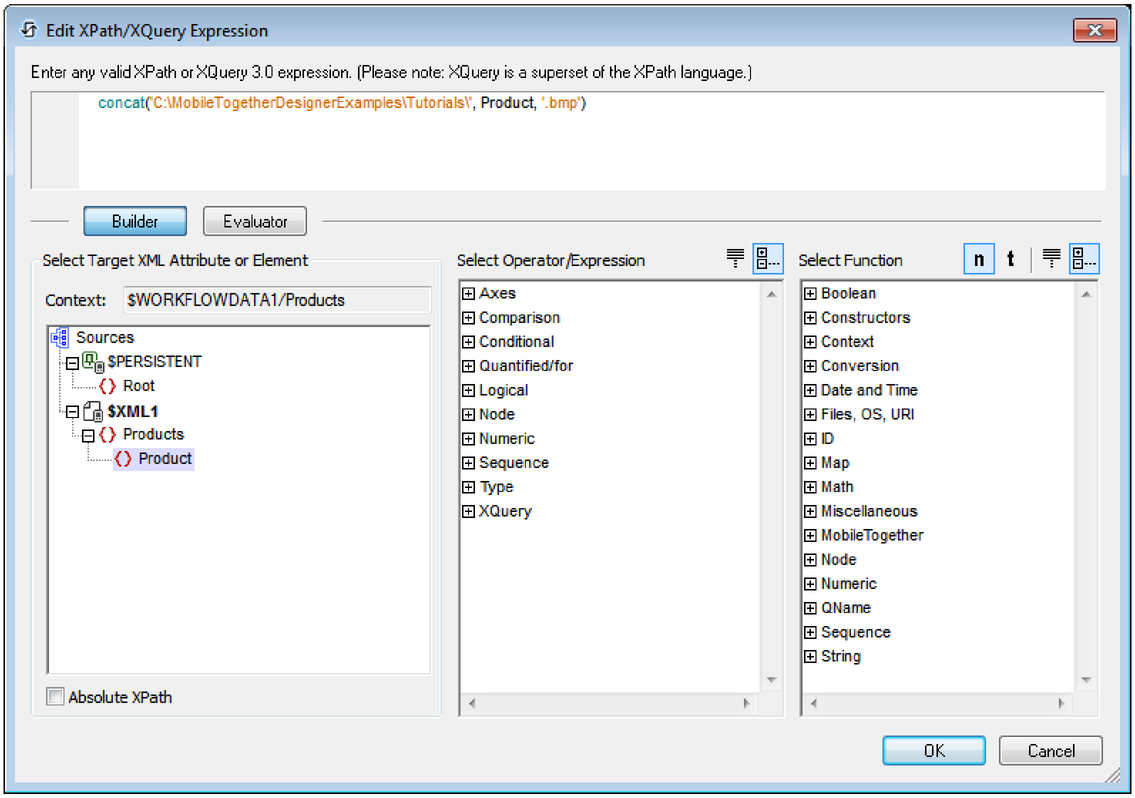
The **Edit XPath Expression** dialog (*screenshot below*) is used to create and edit XPath expressions for a range of StyleVision features. The dialog provides a tree of XML data sources and a library of XPath/XQuery 3.0 operators and functions, and thus supports the building of valid XPath 3.0 and XQuery 3.0 expressions.

The Edit XPath Expression dialog has two modes: (i) Builder mode, for creating XPath expressions, and (ii) Evaluator mode for checking the result of the XPath expression being currently edited. You can switch between the two modes by clicking the respective buttons (**Builder** and **Evaluator**).

Click **OK** when you have completed editing the XPath expression.

Builder mode

In Builder mode, you can build XPath expressions quickly and correctly by either (i) entering the XPath expression in the Expression text box via the keyboard, or (ii) using the entry helpers of Builder mode to insert nodes, operators, and functions by double-clicking them in their respective lists (*see screenshot below*). When an expression that has been entered in the Expression text box contains errors, the expression is underlined in red, thus alerting you to the problem. Builder mode is described in detail in the section, XPath Expression Builder.



Edit XPath/XQuery Expression

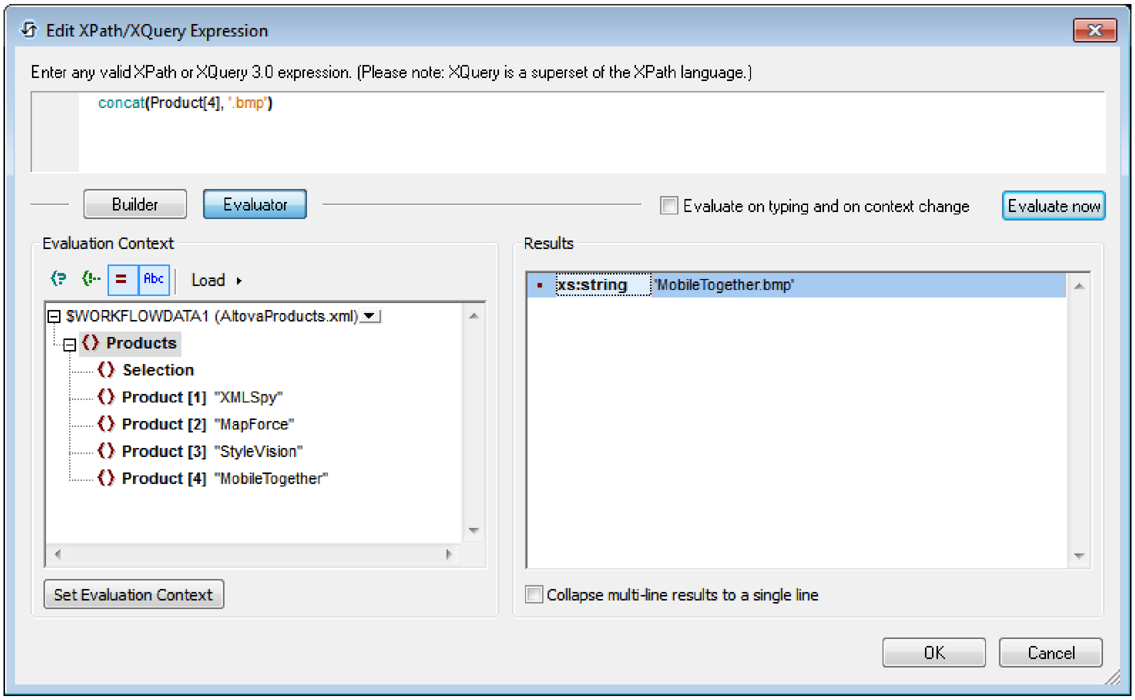
Enter any valid XPath or XQuery 3.0 expression. (Please note: XQuery is a superset of the XPath language.)

```
concat('C:\MobileTogetherDesignerExamples\Tutorials\', Product, '.bmp')
```

Context: $WORKFLOWDATA1/Products

Evaluator mode

In Evaluator mode, you can see, in the *Results* pane on the right-hand side of the dialog (*see screenshot below*), the results of evaluating the currently entered XPath expression. The *Evaluation Context* pane shows the structure and contents of the currently assigned Working XML document. The Edit XPath Expression dialog's Evaluator mode is described in detail in the section, XPath Expression Evaluator.


Edit XPath/XQuery Expression
Enter any valid XPath or XQuery 3.0 expression. (Please note: XQuery is a superset of the XPath language.)
Builder
Evaluator
☐ Evaluate on typing and on context change
Evaluate now
Evaluation Context
Load
Results
Set Evaluation Context
☐ Collapse multi-line results to a single line
OK
Cancel

```
concat(Product[4], '.bmp')
```

```
$WORKFLOWDATA1 (AltovaProducts.xml)
  Products
    Selection
    Product [1] "XMLSpy"
    Product [2] "MapForce"
    Product [3] "StyleVision"
    Product [4] "MobileTogether"
```

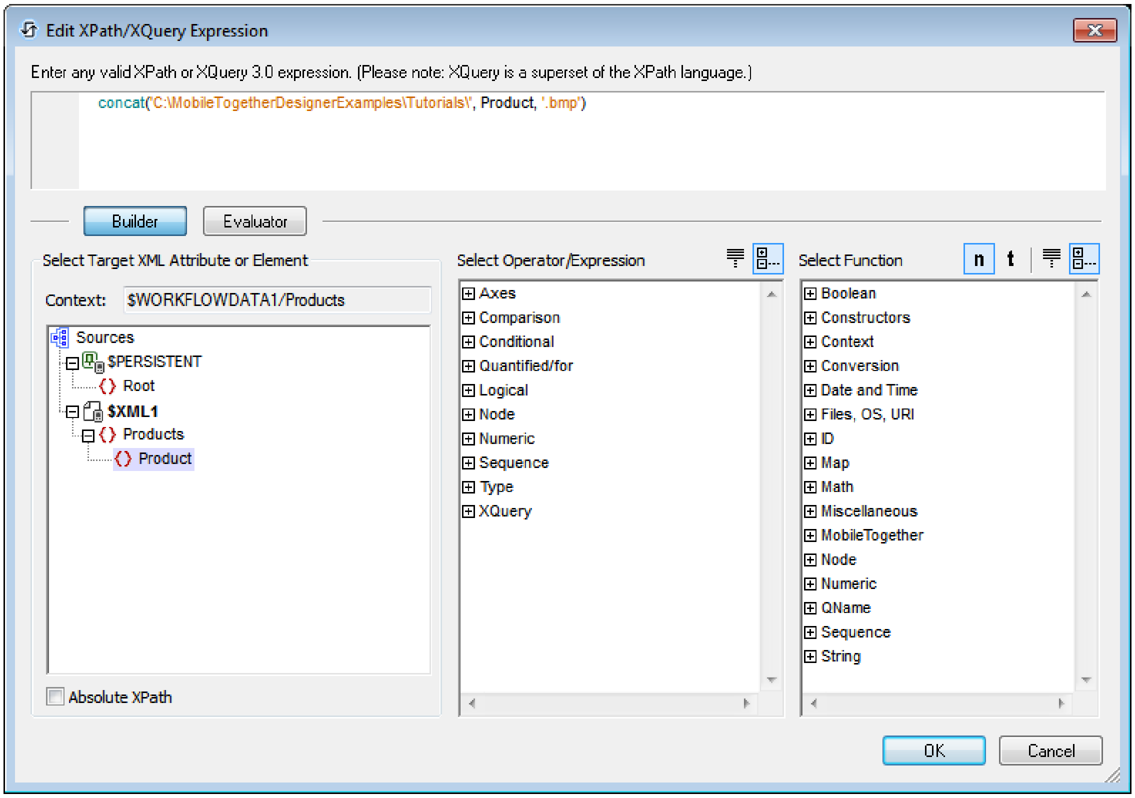
```
xs:string 'MobileTogether.bmp'
```

XPath/XQuery Expression Builder

When the **Builder** button in the Edit XPath Expression dialog is clicked (*see screenshot below*), entry helper panes to help you build an XPath expression become visible. Double-click an entry in any of these entry helpers to enter it at the current cursor point in the XPath expression.

There are three entry helper panes:

- A schema tree for entering element and attribute nodes in the XPath expression. If the *Absolute XPath* check box is unchecked, then the location path to the selected node is entered relative to the context node (the node in the design within which the XPath expression is being built). An absolute XPath expression starts at the document root. Absolute paths are used for the selected node if the *Absolute XPath* check box is checked.
- An entry helper pane for: (i) axes (`ancestor::`, `parent::`, etc), (ii) operators (for example `eq` and `div`), and (iii) expressions (`for # in  # return #`, etc). This pane displays the axes, operators, and expressions either listed alphabetically or grouped by functional category. Select the option you want by clicking the appropriate icon above the pane.
- An entry helper with the functions of the active XPath version either listed alphabetically or grouped by functional category. Select the option you want by clicking the appropriate icon above the pane. The **n** and **t** buttons above the pane display the arguments of functions, respectively as names and datatypes.



Enter any valid XPath or XQuery 3.0 expression. (Please note: XQuery is a superset of the XPath language.)

```
concat('C:\MobileTogetherDesignerExamples\Tutorials\', Product, '.bmp')
```

Context: $WORKFLOWDATA1/Products

☐ Absolute XPath

Building XPath expressions

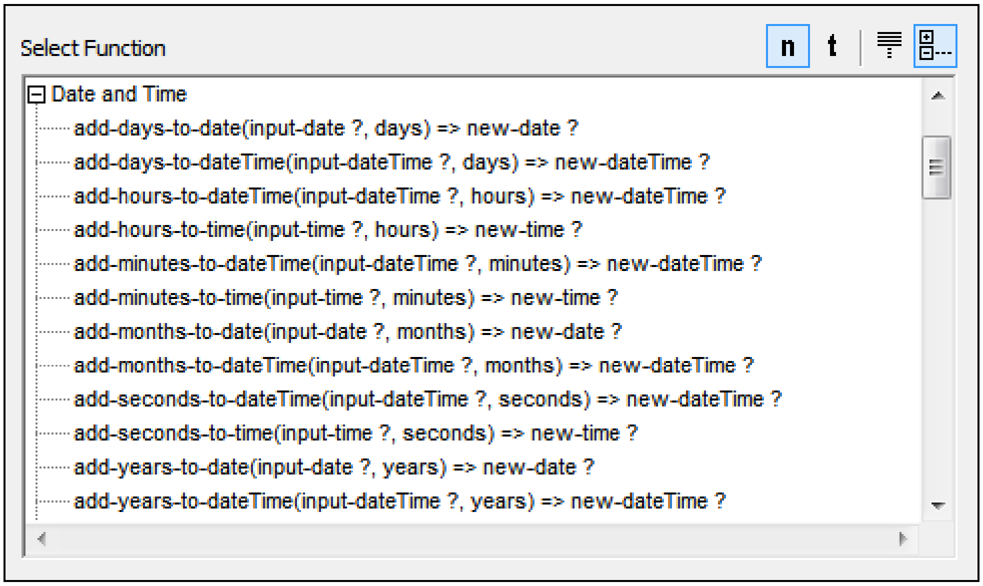
The Edit XPath Expression dialog helps you to build XPath expressions in the following ways.

- **Context node and schema tree**
  The *Selection* text box in the *Select Schema Attribute or Element* pane immediately shows you the context node. Place the cursor over the text box to see the full path of the context node. In the schema tree below the *Selection* text box, you can see where the context node occurs and quickly build the XPath expression by referring to the schema tree.

- **Inserting a node from the target XML tree**
  In the *Select Target XML Attribute or Element* pane, the structure of the target XML document is displayed. Double-click a node in the schema tree to insert that node in the XPath expression. If the *Absolute XPath* check box is not checked, the selected node will be inserted with a location path expression that is relative to the context node. For example, in the screenshot above, the `Product` element, which is a child of the `Products` element (the context node), has been inserted with a location path that is relative to the context node (that is, as `Product`). If the Absolute XPath check box were checked, the `Product` node would have been inserted as `$XML1/Products/Product`.

- **Inserting XPath axes, operators and expressions**
  The *Select Operator/Expression* pane lists the XPath axes (`ancestor::`, `parent::`, etc), operators (for example, `eq` and `div`), and expressions (`for # in  # return #`, etc) for the XPath version selected as the XSLT version for the SPS. The display can be toggled between an alphabetical listing and a hierarchical listing (which groups the items according to functionality). To insert an axis or operator in the XPath expression, double-click the required item.

- **Inserting XPath functions**
  The *Select Function* pane lists XPath functions alphabetically or grouped according to functionality (click the respective icon at the top of the pane to switch between the two arrangements). Each function is listed with its signature. If a function has more than one signature, that function is listed as many times as the number of signatures. Arguments in a signature are separated by commas, and each argument can have an occurrence indicator (**?** indicates a sequence of zero or one items of the specified type; **`*`** indicates a sequence of zero or more items of the specified type). The arguments can be displayed as names or as datatypes; select the **n** or **t** button above the pane to toggle between the two display options. Each function also specifies the return type of that function. For example: `=> date ?` indicates that the expected return datatype is a sequence of none or one `date` item. Placing the mouse over a function pops up a brief description of the function.



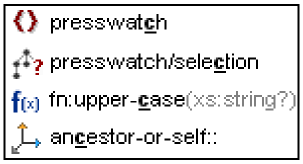
Select Function

Date and Time
- add-days-to-date(input-date ?, days) => new-date ?
- add-days-to-dateTime(input-dateTime ?, days) => new-dateTime ?
- add-hours-to-dateTime(input-dateTime ?, hours) => new-dateTime ?
- add-hours-to-time(input-time ?, hours) => new-time ?
- add-minutes-to-dateTime(input-dateTime ?, minutes) => new-dateTime ?
- add-minutes-to-time(input-time ?, minutes) => new-time ?
- add-months-to-date(input-date ?, months) => new-date ?
- add-months-to-dateTime(input-dateTime ?, months) => new-dateTime ?
- add-seconds-to-dateTime(input-dateTime ?, seconds) => new-dateTime ?
- add-seconds-to-time(input-time ?, seconds) => new-time ?
- add-years-to-date(input-date ?, years) => new-date ?
- add-years-to-dateTime(input-dateTime ?, years) => new-dateTime ?

To insert a function in the XPath expression, double-click the required function.

Intelligent editing during direct text entry

If you type an expression directly in the *Expression* text box, options that are available at that point are displayed in a popup (*see screenshot below*).

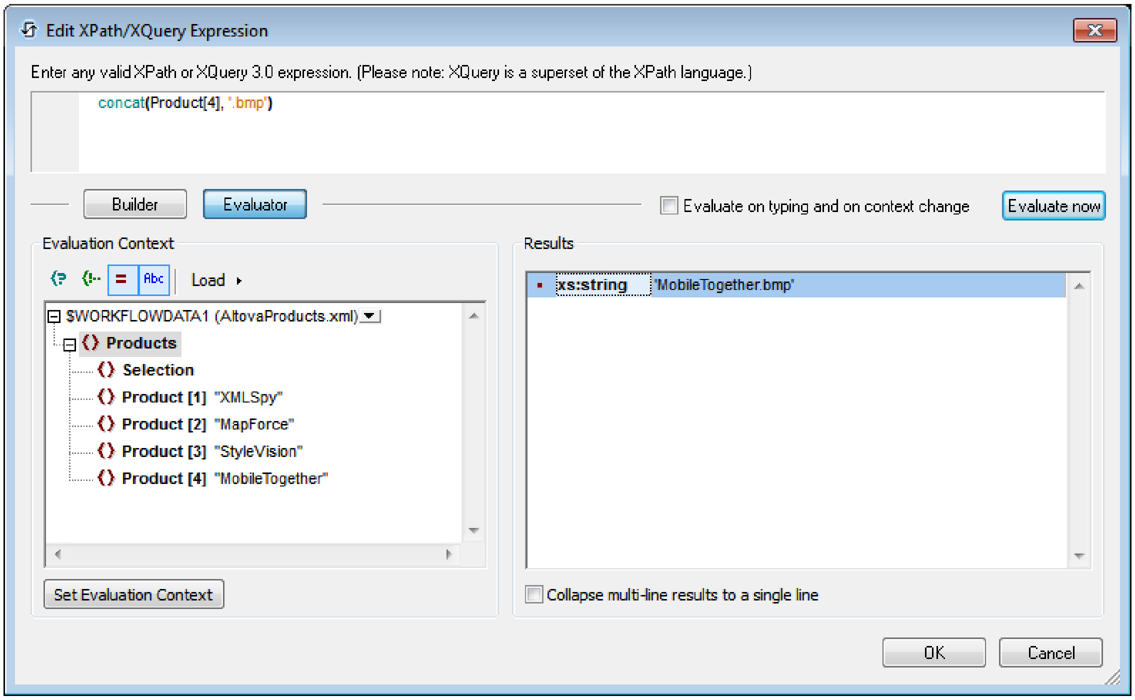


presswatch
presswatch/selection
fn:upper-case(xs:string?)
ancestor-or-self::

These include elements (such as `presswatch` in the screenshot above), descendant nodes (`presswatch/selection` in the screenshot above), XPath functions (`fn:upper-case` above) and XPath axes (`ancestor-or-self` above). The list of available options becomes more restricted as the expression is entered in the *Expression* text box. Go up and down the list of options using the **Up** and **Down** keys, and press **Enter** if you wish to select an option and enter it in the expression.

XPath/XQuery Expression Evaluator

Clicking the **Evaluator** button in the Edit XPath Expression dialog switches the dialog to Evaluator mode (*see screenshot below*). The dialog in this mode has two panes: the *Evaluation Context* pane and the *Results* pane.



Edit XPath/XQuery Expression

Enter any valid XPath or XQuery 3.0 expression. (Please note: XQuery is a superset of the XPath language.)

```
concat(Product[4], '.bmp')
```

Builder
Evaluator
□ Evaluate on typing and on context change
Evaluate now
Evaluation Context
Load
$WORKFLOWDATA1 (AltovaProducts.xml)
Products
Selection
Product [1] "XMLSpy"
Product [2] "MapForce"
Product [3] "StyleVision"
Product [4] "MobileTogether"
Set Evaluation Context
Results
xs:string 'MobileTogether.bmp'
□ Collapse multi-line results to a single line
OK
Cancel

The XPath expression and its evaluation

The XPath expression in the *XPath Expression* text box can be edited, and the expression can be evaluated. The results of the evaluation are displayed in the *Results* pane. In the screenshot above, for example, the result of evaluating the XPath expression `concat(Product[4], '.bmp')` is displayed as the string `MobileTogether.bmp` (because `MobileTogether` is the content of the fourth `Product` element).

Using Builder mode and switching to Evaluator mode for the results

If you wish to use entry helpers to build the XPath expression, you can switch to Builder mode (by clicking the **Builder** button), build the expression in Builder mode, then switch to Evaluator mode to see the results of the evaluation.

When is the XPath expression evaluated?

Evaluation is carried out in two mutually exclusive situations:

- *Evaluate on typing:* If this check box is selected, the XPath expression is evaluated: (i) with every keystroke used to edit the expression, and (ii) when the mode is switched from Builder mode to Evaluator mode.
- *Evaluate now:* This button is enabled when the *Evaluate on Typing* option is not checked. Click it to evaluate the expression.

The Evaluation Context pane
The *Evaluation Context* pane shows the structure and contents of the data sources. Nodes in document trees can be expanded or collapsed by clicking the respective icons of individual nodes. You can load an XML file structure by clicking **Load** and browsing for the file you want.

The icons above the pane display or hide the following XML syntactic constructs: (i) processing instructions, (ii) comments, (iii) attributes, (iv) text nodes. You can therefore see the entire XML document structure, together with the text contents of nodes, but you can also hide certain constructs if you wish to reduce clutter in the pane.

Changing the context node for evaluation purposes
You can change the context node of the XPath expression by clicking the node in the document tree that you want as the new context node and then selecting **Set Evaluation Context**. If the Evaluate on Typing option is checked, then the result will appear immediately in the Results pane.

This feature is useful for checking results with different context nodes. Note, however, that the actual context node for the expression will be the context node within which the current design component is being created. At runtime, the actual context node will be used, not the context node used in the Evaluator.

MobileTogether XPath Extension Functions

The following MobileTogether-specific extension XPath/XQuery functions are available

Function	Description (as shown in the tooltip)	Usage	Explanation
`mt-reload-dateTime`	Returns time the page-source was reloaded	`mt-reload-dateTime($XML1)`	If not loaded yet then an empty sequence is returned.
`mt-cache-update-dateTime`	Returns time the page-source cache was updated	`mt-cache-update-dateTime($XML1)`	If not cached then an empty sequence is returned.
`mt-db-new-rows`	Returns new Row elements	`mt-db-new-rows($DB1) -> returns a list of Row elements that are listed under the RowSet element but not under OriginalRowSet`	These function can be used to figure out modifications that were performed to the data that was read from the database. Note that this will only work if OriginalRowSet was enabled on the page source!
`mt-db-modified-rows`	Returns modified Row elements	`mt-db-modified-rows($DB1) -> returns a list of Row elements that were modified (their attributes)`	
`mt-db-deleted-original-rows`	Returns original Row elements	`mt-db-deleted-original-rows($DB1) -> returns all OriginalRow elements for which no Row element exists`	
`mt-db-new-fields`	Returns new Field attributes: for new rows – all; for modified rows – only new; for deleted original rows – none	`mt-db-new-fields($DB1/DB/RowSet/Row[1])`	If called for a new Row (no corresponding OriginalRow element), return all fields. If called for an existing but modified Row, return only the new ones (the ones not listed under OriginalRow) If called with an OriginalRow (where the Row was deleted), return an empty list.
`mt-db-modified-fields`	Returns modified Field attributes: for new and	`mt-db-modified-fields($DB1/DB/RowSet/Row[3])`	If called for a new Row or on an OriginalRow (where the Row was deleted), return all fields.

	deleted original rows – all; for modified rows – only modified		If called for a modified Row, return only the modified (the ones with a value different from the one stored in OriginalRow).
`mt-db-deleted-original-fields`	Returns Field attributes from original Row element: for new rows – none; for modified rows – only deleted; for deleted original rows – all	`mt-db-deleted-original-fields($DB1/DB/RowSet/`	If called for a new Row, return empty list. If called for a modified Row, return the fields from the corresponding OriginalRow element that are not listed under its Row element. If called for a OriginalRow (where the corresponding Row element was deleted), return all fields.
`mt-db-any-changed-rows`	Returns true if the db-source-variable has new, modified or deleted rows. Returns false if unmodified.	`mt-db-any-changed-rows($DB1)`	Tests if there was any modification.
`mt-db-any-changed-fields`	Returns true if the row element contains new, modified or deleted columns. Returns false if unmodified.	`mt-db-any-changed-fields($DB1/DB/RowSet/Row[3])`	Tests if there was any modification in the Row element.
`mt-external-error-code`	Returns the error code of the last DB, Load or Save action	`mt-external-error-code()`	Returns the native error code of the OS or database, like 404 when you can't find the website
`mt-external-error-text`	Returns the error text of the last DB, Load or Save action	`mt-external-error-text()`	Returns the error text that was provided along the returned error code
`mt-has-serveraccess`	Returns true if server access is possible, false otherwise	`mt-has-serveraccess()`	Checks if we are online and can establish a connection to our MT server.
`mt-refresh-userroles`	Loads available user roles from server	`mt-refresh-userroles()`	Updates the user roles from the server which can be queried via the global variable.

FAQ about XPath/XQuery

▾ ***I have an XPath expression inside the repeating row of a table. Should I use an absolute or relative XPath expression to target a child attribute?***

If your XPath expression is inside the repeating row of a table, then the element corresponding to the row of the table is the context node, say, `Row`.

- If you use an absolute path, for example `$XML/Row/@id`, then the XPath expression will return a sequence of the `@id` values of **all** `Row` elements. If you are using an operation that expects one atomic value, thie operation will generate an error.
- If you use a relative path, for example `@id`, then, since for every repeating row, you have a context `$XML/Row`, the XPath expression will correctly return the single atomic value of the one `@id` attribute of the current `Row` element.

7.3 Controls

A page design (*screenshot below*) consists of page controls—such as combo boxes, tables, and images—that are laid out and formatted exactly as the end user will see the page. Controls are dragged into the design from the Controls Pane. When a control is selected in the design, its formatting and processing properties are displayed in the Styles & Properties Pane and can be edited there. Some controls can have a data source node associated with it. The association is done by dragging a data source node from the Page Sources Pane onto the control.

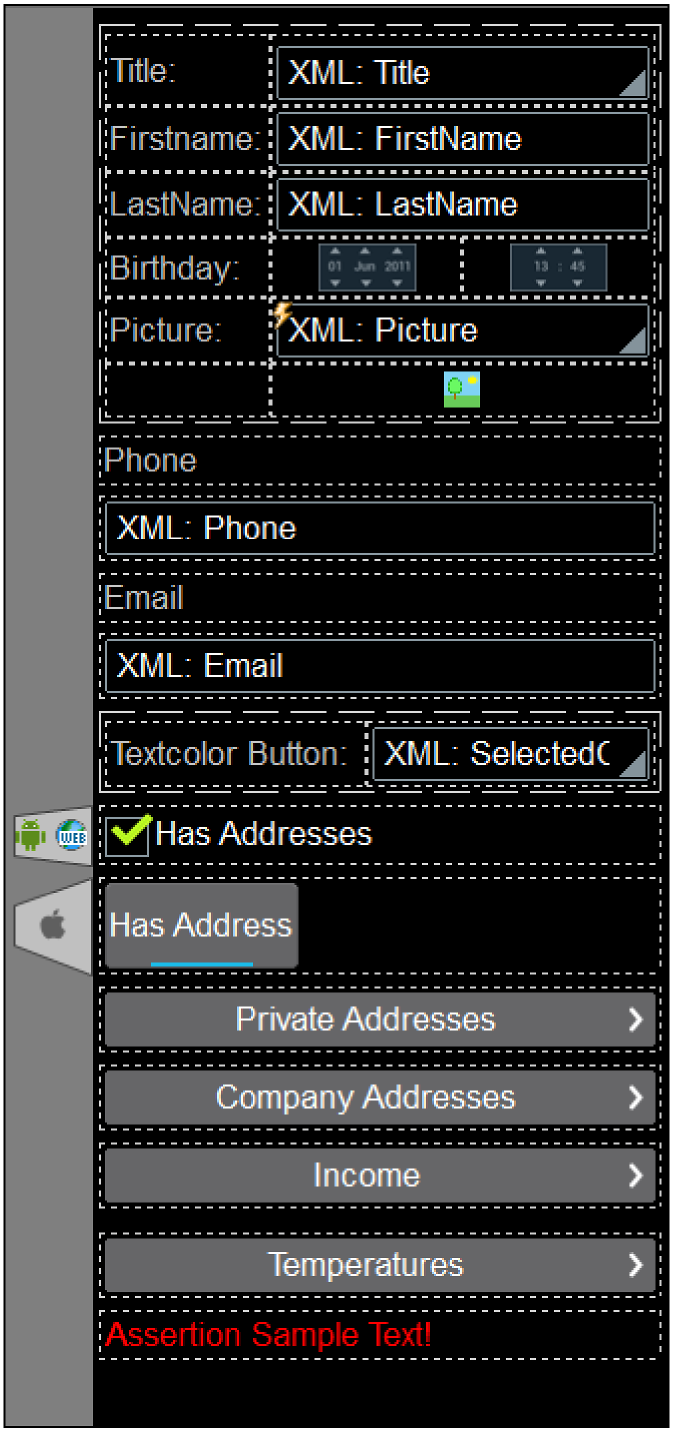
⊟ **Notes**

- When the control is associated with a data source node (page source link), placing the mouse over the control (in Page Design View) displays the associated node in a popup.
- To remove a data-source node association, right-click the control (in Page Design View) and click **Delete Page Source Link**.
- To reset a style or property (in the Styles & Properties Pane), select the property and click **Reset** in the pane's toolbar.
- To edit the XPath expression of a style or property (in the Styles & Properties Pane), select the style or property, and click **Edit XPath** in the pane's toolbar.

List of controls

Given below is a list of available controls, arranged alphabetically.

- Assertion Message
- Button
- Chart
- Checkbox
- Combo Box
- Date
- DateTime (iOS)
- Edit Field
- Horizontal Line
- Image
- Label
- Space
- Switch
- Table
- Time



Context menu
Each control in the page design has a context menu. The following control-related commands are common to most context menus.

▾ **Delete Page Source Link**

⊟ *Description*

This command is enabled for those controls that can be associated with a data source node and for which an association exists. **Delete Page Source Link** deletes the association between control and data source. Note that there is no other way to delete the control's association with a node.

Control Actions

⊟ *Description*

Displays the Control Actions dialog, in which you can set actions for various control events. For a description of available actions for control events and how to set control actions, go to the section, Page Design | Events and Actions | Control Events and Actions.

Enter Text

⊟ *Description*

Enabled for controls in which text can be entered. Rolls out a sub-menu with the following options:

- *Directly:* To enter static text directly as the text of the control
- *XPath:* Displays the Edit XPath/XQuery Expression dialog, in which you can enter the XPath expression that selects the text of the control
- *XML Node:* Displays a hint that a data source node can be dragged from the Page Sources Pane onto the control. The dragged-and-dropped node is associated with the control, and the node's content is entered as the text of the control

Localization

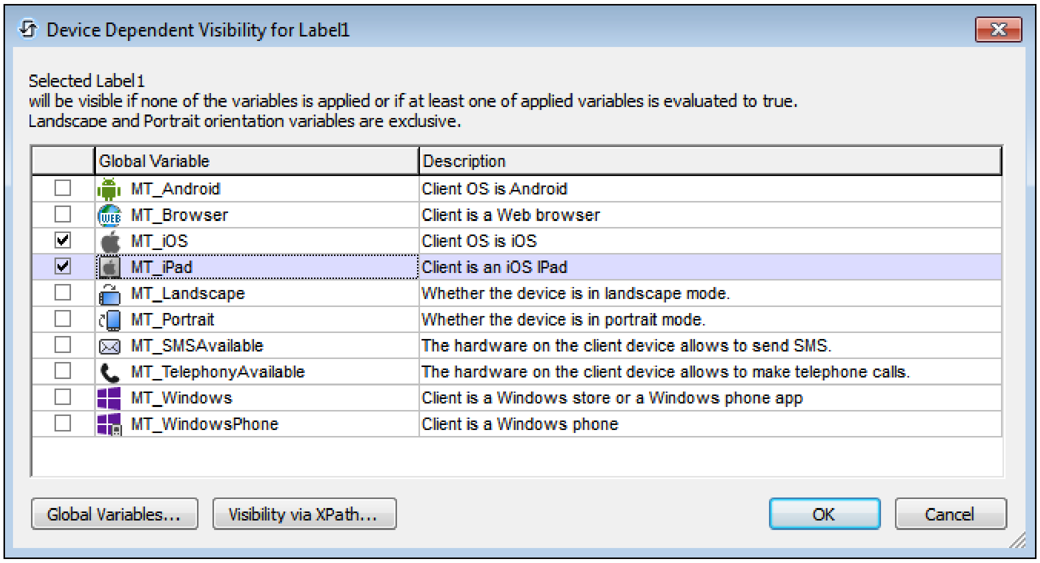
⊟ *Description*

Displays the Localization dialog, in which you can define the localization (translation) of strings that appear in various controls of the project. This command has the same effect as the **Project | Localization** command. For a description of localization, go to the description of the **Project | Localization** command.

Device Dependent Visibility

⊟ *Description*

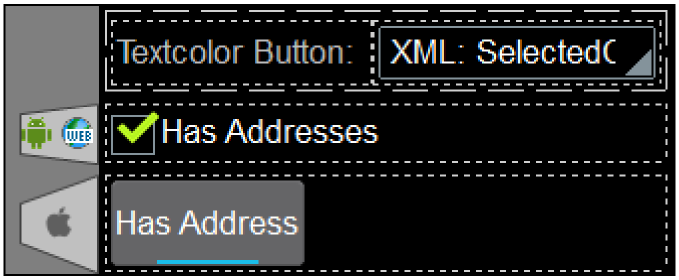
Displays the Device Dependent Visibility dialog (*screenshot below*). The dialog contains a list of client-device types. Select the client-device type for which you want the control to be visible, and click **OK**.



Selected Label1
will be visible if none of the variables is applied or if at least one of applied variables is evaluated to true.
Landscape and Portrait orientation variables are exclusive.

	Global Variable	Description
□	MT_Android	Client OS is Android
□	MT_Browser	Client is a Web browser
☑	MT_iOS	Client OS is iOS
☑	MT_iPad	Client is an iOS IPad
□	MT_Landscape	Whether the device is in landscape mode.
□	MT_Portrait	Whether the device is in portrait mode.
□	MT_SMSAvailable	The hardware on the client device allows to send SMS.
□	MT_TelephonyAvailable	The hardware on the client device allows to make telephone calls.
□	MT_Windows	Client is a Windows store or a Windows phone app
□	MT_WindowsPhone	Client is a Windows phone

If the client-device type has an icon, this icon appears in the design, to the left of the control to which it applies (*see screenshot below*).



Page Actions

Description

Displays the Page Actions dialog, in which you can set actions for various page events. This command has the same effect as the **Page | Page Actions** command. For a description of available actions for page events and how to set page actions, go to the description of the **Page | Page Actions** command.

Actions Overview

Description

Displays the Actions Overview dialog of the currently active page. This command has the same effect as the **Page | Actions Overview** command. For more information, go to the description of the **Page | Actions Overview** command.

Assertion Message

The Assertion Message control displays the assertion message of an invalid assertion. If there are multiple invalid assertions on a page, then the assertion message of the first invalid assertion (in the sequence in which assertions appear on the page) is displayed. As a consequence, the Assertion Message should be inserted only once on a page.

Assertions and assertion messages work as follows:

- The `Assertion` property of a control sets a condition to be met in order for the assertion to be valid. The assertion's condition is specified with an XPath expression.
- If the assertion is invalid, then the text of the control's `Assertion Message` property is displayed by the Assertion Message control.

For example: The XPath expression **`LastName != ""`** in the `Assertion` property of a control asserts that the node `LastName` must not be empty. If this node is empty, then the control's assertion message is displayed in the page at the point where the Assertion Message control is inserted.

Assertion Message properties

The control's properties are available in the Styles & Properties Pane, and are listed below in alphabetical order.

▼ **`Background Color`**

Sets the background color of the control. You can do one of the following to select the color:

- Click the color palette to select a background color
- Select a color from the dropdown list of the combo box
- Double-click in the value field to enter a color code (for example, `#FF0000`), or click the **XPath** toolbar button and enter an XPath expression to generate the required text

▼ **`Bold Text`**

Select `true` or `false` from the dropdown list of the combo box. Default is `false`.

▼ **`Control Width`**

Sets the width of the control. Select `fill_parent` or `wrap_content` from the dropdown list of the combo box. Default is `fill-parent`.

▼ **`Italic Text`**

Select `true` or `false` from the dropdown list of the combo box. Default is `false`.

▼ **`Margin`**

Sets the control's margin offsets relative to the surrounding objects or to the borders of the containing object. Select a value in pixels from the dropdown list of the combo box, or double-click in the value field to enter a pixel length.

▼ `Margin Bottom`

Sets the control's bottom margin offset relative to the object below it or to the border of the containing object. Select a value in pixels from the dropdown list of the combo box, or double-click in the value field to enter a pixel length.

▼ `Margin Left`

Sets the control's left margin offset relative to the object to the left or to the border of the containing object. Select a value in pixels from the dropdown list of the combo box, or double-click in the value field to enter a pixel length.

▼ `Margin Right`

Sets the control's right margin offset relative to the object to the right or to the border of the containing object. Select a value in pixels from the dropdown list of the combo box, or double-click in the value field to enter a pixel length.

▼ `Margin Top`

Sets the control's top margin offset relative to the object above it or to the border of the containing object. Select a value in pixels from the dropdown list of the combo box, or double-click in the value field to enter a pixel length.

▼ `Multiline`

Sets multiline input/display on or off (`true/false`). The default is `false`.

▼ `Name`

The name is used to reference the control from elsewhere in the page or project. Double-click inside the value field to edit.

▼ `Text Color`

Sets the color of the control's text. You can do one of the following to select the color:

- Click the color palette to select a text color
- Select a color from the dropdown list of the combo box
- Double-click in the value field to enter a color code (for example, `#FF0000`), or click the **XPath** toolbar button and enter an XPath expression to generate the required text

▼ `Text Size`

Select a size from the dropdown list of the combo box, or double-click in the value field to enter a text size.

▼ `Underline Text`

Select `true` or `false` from the dropdown list of the combo box. Default is `false`.

Button

Buttons can used to execute an action when the button is pressed. The name of the button can be static text (entered as the value of the `Text` property; *see below*) or a dynamic value obtained from a data source node (by dragging the node onto the button). The `OnButtonPressed` event is associated with the button control. To define an action for this event, click the **Additional Dialog** button of the `Control Action` property. This displays the Control Actions dialog, in which you can specify the required action.

⊟ **Notes**

- When the control is associated with a data source node (page source link), placing the mouse over the control (in Page Design View) displays the associated node in a popup.
- To remove a data-source node association, right-click the control (in Page Design View) and click **Delete Page Source Link**.
- To reset a style or property (in the Styles & Properties Pane), select the property and click **Reset** in the pane's toolbar.
- To edit the XPath expression of a style or property (in the Styles & Properties Pane), select the style or property, and click **Edit XPath** in the pane's toolbar.

Button properties

The control's properties are available in the Styles & Properties Pane, and are listed below in alphabetical order.

▼ **`Background Color`**

Sets the background color of the control. You can do one of the following to select the color:

- Click the color palette to select a background color
- Select a color from the dropdown list of the combo box
- Double-click in the value field to enter a color code (for example, `#FF0000`), or click the **XPath** toolbar button and enter an XPath expression to generate the required text

▼ **`Bold Text`**

Select `true` or `false` from the dropdown list of the combo box. Default is `false`.

▼ **`Control Action`**

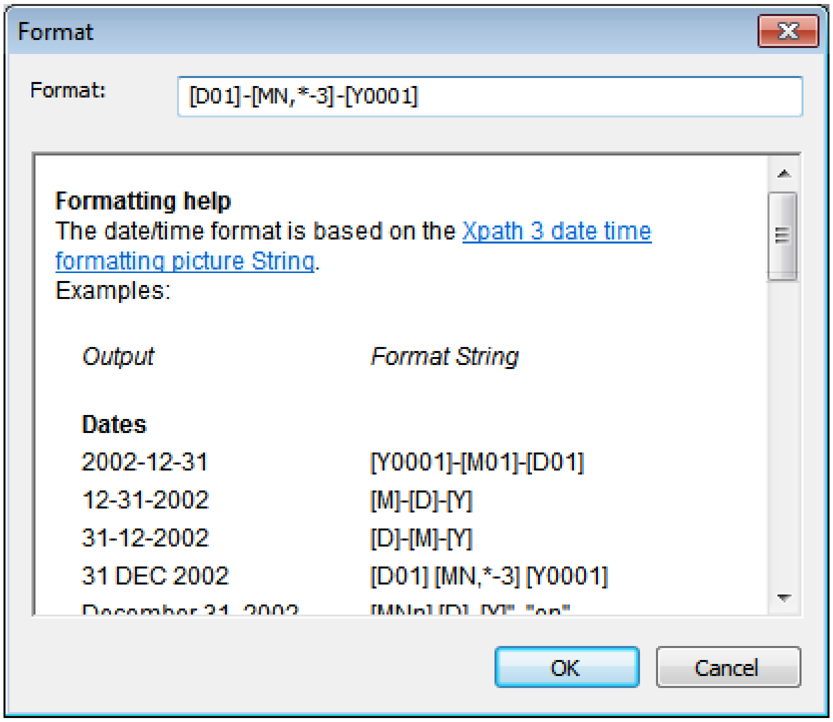
Click the **Additional Dialog** button to pop up the control's Actions dialog.

▼ **`Control Width`**

Sets the width of the control. Select `fill_parent` or `wrap_content` from the dropdown list of the combo box. Default is `fill-parent`.

▼ **`Date/Time Format String`**

Click the **Additional Dialog** button and enter a date, time, or date-time format in the Format dialog that appears (*screenshot below*).



Format: [D01]-[MN,*-3]-[Y0001]

Formatting help
The date/time format is based on the Xpath 3 date time formatting picture String.
Examples:

Output	*Format String*
Dates	
2002-12-31	[Y0001]-[M01]-[D01]
12-31-2002	[M]-[D]-[Y]
31-12-2002	[D]-[M]-[Y]
31 DEC 2002	[D01] [MN,*-3] [Y0001]
December 31, 2002	[MNn] [D], [Y]", "en"

The formatting will be applied to the control's content if the content has the correct lexical form of `xs:date` (for the Date control), `xs:time` (for the Time control), or `xs:dateTime` (for the Date, Time, and DateTime controls). Basic examples are:

- *xs:date:* `2014-12-31`
- *xs:time:* `23:59:59`
- *xs:dateTime:* `2014-12-31T23:59:59`

`Enabled/Editable`

The control is either enabled (and editable) or it is disabled. The value of the property must be an XPath expression that evaluates to boolean `true` or `false`. If the expression evaluates to `true`, the control is enabled/editable. If the expression evaluates to `false`, the control is disabled. The default value is `true`. Typically the XPath expression of the `Enabled/Editable` property of a control would check the presence or value of a node. If the expression evaluates to `true`, the control will be enabled/editable.

`Hide Keyboard On Click  [Android only]`

Hides the keyboard when the control is clicked. Default is `false`.

`Italic Text`

Select `true` or `false` from the dropdown list of the combo box. Default is `false`.

`Margin`

Sets the control's margin offsets relative to the surrounding objects or to the borders of the containing object. Select a value in pixels from the dropdown list of the combo box, or double-click in the value field to enter a pixel length.

Margin Bottom

Sets the control's bottom margin offset relative to the object below it or to the border of the containing object. Select a value in pixels from the dropdown list of the combo box, or double-click in the value field to enter a pixel length.

Margin Left

Sets the control's left margin offset relative to the object to the left or to the border of the containing object. Select a value in pixels from the dropdown list of the combo box, or double-click in the value field to enter a pixel length.

Margin Right

Sets the control's right margin offset relative to the object to the right or to the border of the containing object. Select a value in pixels from the dropdown list of the combo box, or double-click in the value field to enter a pixel length.

Margin Top

Sets the control's top margin offset relative to the object above it or to the border of the containing object. Select a value in pixels from the dropdown list of the combo box, or double-click in the value field to enter a pixel length.

Multiline

Sets multiline input/display on or off (`true/false`). The default is `false`.

Name

The name is used to reference the control from elsewhere in the page or project. Double-click inside the value field to edit.

Number Format String

Click the **Additional Dialog** button and enter a number format in the Format dialog that pops up. The formatting will be applied to label content if the label content is numeric.

Text

The Text property takes as its value either of the following:

- The text to be displayed in the control
- An XPath expression that retrieves data from a node in a data source.

Double-click inside the value field to edit, or click the **XPath** toolbar button and enter an XPath expression to generate the required text.

Text Color

Sets the color of the control's text. You can do one of the following to select the color:

- Click the color palette to select a text color

- Select a color from the dropdown list of the combo box
- Double-click in the value field to enter a color code (for example, `#FF0000`), or click the **XPath** toolbar button and enter an XPath expression to generate the required text

▼ **`Text Size`**

Select a size from the dropdown list of the combo box, or double-click in the value field to enter a text size.

▼ **`Underline Text`**

Select `true` or `false` from the dropdown list of the combo box. Default is `false`.

▼ **`Visible`**

An XPath expression that should evaluate to `true()` or `false()`. If the expression evaluates to `false()`, and only if it evaluates to `false()`, is the control not visible. If the expression evaluates to `true()` or returns some other value, then the control is visible. The default is `true()`. Double-click inside the value field, or click the **XPath** button, to enter or edit an XPath expression.

Chart

The Chart control enables data from a source data file to be displayed in the form of a chart. The available chart types are: pie charts, bar charts, line graphs, area charts, candlestick charts, and gauge charts. Data for the X-Axis, Y-Axis, and other chart components is selected with XPath expressions. The context node for these XPath expressions is set by dragging it from the source data tree and dropping it onto the chart control in the design.

The chart's display settings within the page are defined in the Styles & Properties Pane. The settings for chart type, data selection, and appearance are defined in the Chart Configuration dialog. This dialog is accessed by clicking the **Additional Dialog** button of the `Chart Settings` property, or by double-clicking the chart in the design.

For detailed information about how to configure charts, see the Charts section.

⊟ **Notes**

- When the control is associated with a data source node (page source link), placing the mouse over the control (in Page Design View) displays the associated node in a popup.
- To remove a data-source node association, right-click the control (in Page Design View) and click **Delete Page Source Link**.
- To reset a style or property (in the Styles & Properties Pane), select the property and click **Reset** in the pane's toolbar.
- To edit the XPath expression of a style or property (in the Styles & Properties Pane), select the style or property, and click **Edit XPath** in the pane's toolbar.

Chart properties

The control's properties are available in the Styles & Properties Pane, and are listed below in alphabetical order.

▼ **`Chart Creation Height`**

Sets the height, in pixels, of the chart to be generated. Click the **Edit XPath** icon and, in the dialog that appears, enter an expression that returns a numeric value. This value will be the height, in pixels, of the chart to be generated.

▼ **`Chart Creation Width`**

Sets the width, in pixels, of the chart to be generated. Click the **Edit XPath** icon and, in the dialog that appears, enter an expression that returns a numeric value. This value will be the width, in pixels, of the chart to be generated.

▼ **`Chart Settings`**

Click the **Additional Dialog** button to display the Chart Configuration dialog. The settings you make in this dialog will apply to the chart that is currently selected in the design. For a description of how to configure charts, see the section, Charts.

▼ **`Create Before Load`**

Click the color palette to select a background color, or select a color from the dropdown list of the combo box, or double-click in the value field to enter a color code.

- **`Control Width`**

  Sets the width of the control. Select `fill_parent` or `wrap_content` from the dropdown list of the combo box. Default is `fill-parent`.

- **`Margin`**

  Sets the control's margin offsets relative to the surrounding objects or to the borders of the containing object. Select a value in pixels from the dropdown list of the combo box, or double-click in the value field to enter a pixel length.

- **`Margin Bottom`**

  Sets the control's bottom margin offset relative to the object below it or to the border of the containing object. Select a value in pixels from the dropdown list of the combo box, or double-click in the value field to enter a pixel length.

- **`Margin Left`**

  Sets the control's left margin offset relative to the object to the left or to the border of the containing object. Select a value in pixels from the dropdown list of the combo box, or double-click in the value field to enter a pixel length.

- **`Margin Right`**

  Sets the control's right margin offset relative to the object to the right or to the border of the containing object. Select a value in pixels from the dropdown list of the combo box, or double-click in the value field to enter a pixel length.

- **`Margin Top`**

  Sets the control's top margin offset relative to the object above it or to the border of the containing object. Select a value in pixels from the dropdown list of the combo box, or double-click in the value field to enter a pixel length.

- **`Name`**

  The name is used to reference the control from elsewhere in the page or project. Double-click inside the value field to edit.

- **`Visible`**

  An XPath expression that should evaluate to `true()` or `false()`. If the expression evaluates to `false()`, and only if it evaluates to `false()`, is the control not visible. If the expression evaluates to `true()` or returns some other value, then the control is visible. The default is `true()`. Double-click inside the value field, or click the **XPath** button, to enter or edit an XPath expression.

Check Box

Check boxes have the `OnFinishEditing` event, which is triggered when the end-user makes a check box selection. To define an action for this event, click the **Additional Dialog** button of the `Control Action` property. This displays the Control Actions dialog, in which you can specify the required action.

You can use a check box control which has the check box to the left or to the right of the accompanying text. Two key properties of the check box control are:

- The text that accompanies the check box. This can be static text (entered as the value of the `Text` property; *see below*) or a dynamic value obtained from a data source node (by dragging the node onto the check box).
- The values that are to be respectively assigned the checked and unchecked states of the check box. Theses are assigned with the Checked Values property (*see below*).

⊟ **Notes**

- When the control is associated with a data source node (page source link), placing the mouse over the control (in Page Design View) displays the associated node in a popup.
- To remove a data-source node association, right-click the control (in Page Design View) and click **Delete Page Source Link**.
- To reset a style or property (in the Styles & Properties Pane), select the property and click **Reset** in the pane's toolbar.
- To edit the XPath expression of a style or property (in the Styles & Properties Pane), select the style or property, and click **Edit XPath** in the pane's toolbar.

Check box properties
The control's properties are available in the Styles & Properties Pane, and are listed below in alphabetical order.

▼ **`Background Color`**

Sets the background color of the control. You can do one of the following to select the color:

- Click the color palette to select a background color
- Select a color from the dropdown list of the combo box
- Double-click in the value field to enter a color code (for example, `#FF0000`), or click the **XPath** toolbar button and enter an XPath expression to generate the required text

▼ **`Bold Text`**

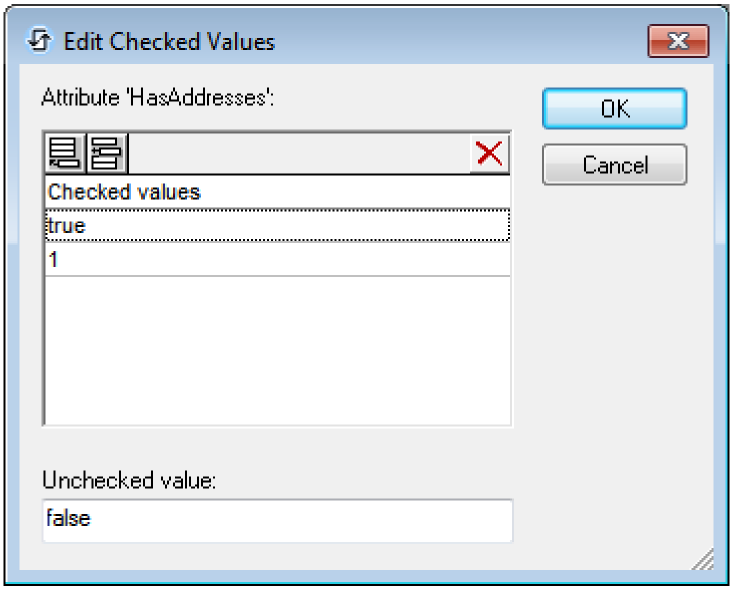
Select `true` or `false` from the dropdown list of the combo box. Default is `false`.

▼ **`CheckBox Position`**

Sets the position of the check box relative to the control's text: either to the left or right of the text. The default value is *Left of text*.

Checked Values (true/false)

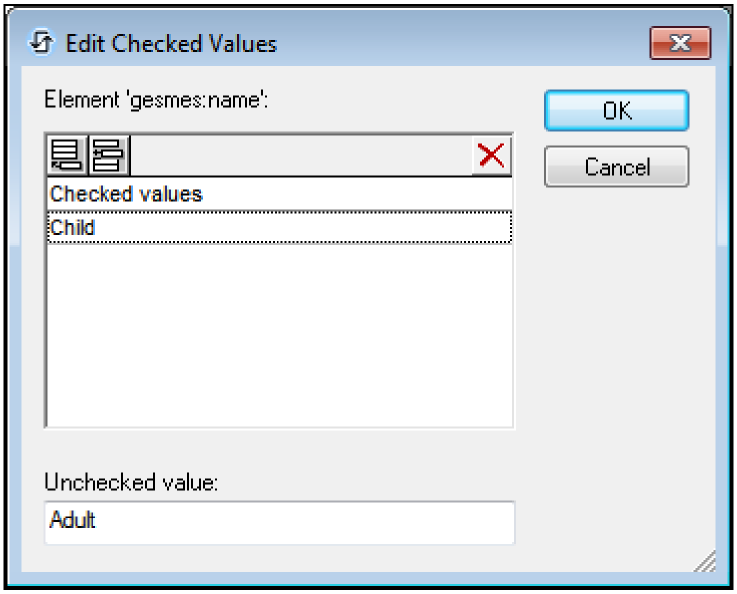
Provides an XML data value for the selected/unselected state of the control. Click the **Additional Dialog** button to display the Edit Checked Values dialog (*screenshots below*). Enter a value for the checked (selected) and unchecked (unselected) state. The value corresponding to the control state (selected/unselected) chosen by the end-user will be entered as data in the XML node associated with the control.



Attribute 'HasAddresses':

Checked values
true
1

Unchecked value:
false



Element 'gesmes:name':

Checked values
Child

Unchecked value:
Adult

Control Action

Click the **Additional Dialog** button to pop up the control's Actions dialog.

Control Width

Sets the width of the control. Select `fill_parent` or `wrap_content` from the dropdown list of the combo box. Default is `fill-parent`.

▾ **`Enabled/Editable`**

The control is either enabled (and editable) or it is disabled. The value of the property must be an XPath expression that evaluates to boolean `true` or `false`. If the expression evaluates to `true`, the control is enabled/editable. If the expression evaluates to `false`, the control is disabled. The default value is `true`. Typically the XPath expression of the `Enabled/Editable` property of a control would check the presence or value of a node. If the expression evaluates to `true`, the control will be enabled/editable.

▾ **`Hide Keyboard On Click  [Android only]`**

Hides the keyboard when the control is clicked. Default is `false`.

▾ **`Italic Text`**

Select `true` or `false` from the dropdown list of the combo box. Default is `false`.

▾ **`Margin`**

Sets the control's margin offsets relative to the surrounding objects or to the borders of the containing object. Select a value in pixels from the dropdown list of the combo box, or double-click in the value field to enter a pixel length.

▾ **`Margin Bottom`**

Sets the control's bottom margin offset relative to the object below it or to the border of the containing object. Select a value in pixels from the dropdown list of the combo box, or double-click in the value field to enter a pixel length.

▾ **`Margin Left`**

Sets the control's left margin offset relative to the object to the left or to the border of the containing object. Select a value in pixels from the dropdown list of the combo box, or double-click in the value field to enter a pixel length.

▾ **`Margin Right`**

Sets the control's right margin offset relative to the object to the right or to the border of the containing object. Select a value in pixels from the dropdown list of the combo box, or double-click in the value field to enter a pixel length.

▾ **`Margin Top`**

Sets the control's top margin offset relative to the object above it or to the border of the containing object. Select a value in pixels from the dropdown list of the combo box, or double-click in the value field to enter a pixel length.

▾ **`Multiline`**

Sets multiline input/display on or off (`true/false`). The default is `false`.

▾ **`Name`**

The name is used to reference the control from elsewhere in the page or project. Double-click inside the value field to edit.

▼ **`Text`**

The Text property takes as its value either of the following:

- The text to be displayed in the control
- An XPath expression that retrieves data from a node in a data source.

Double-click inside the value field to edit, or click the **XPath** toolbar button and enter an XPath expression to generate the required text.

▼ **`Text Color`**

Sets the color of the control's text. You can do one of the following to select the color:

- Click the color palette to select a text color
- Select a color from the dropdown list of the combo box
- Double-click in the value field to enter a color code (for example, `#FF0000`), or click the **XPath** toolbar button and enter an XPath expression to generate the required text

▼ **`Text Size`**

Select a size from the dropdown list of the combo box, or double-click in the value field to enter a text size.

▼ **`Underline Text`**

Select `true` or `false` from the dropdown list of the combo box. Default is `false`.

▼ **`Visible`**

An XPath expression that should evaluate to `true()` or `false()`. If the expression evaluates to `false()`, and only if it evaluates to `false()`, is the control not visible. If the expression evaluates to `true()` or returns some other value, then the control is visible. The default is `true()`. Double-click inside the value field, or click the **XPath** button, to enter or edit an XPath expression.

Combo Box

The combo box control provides a mechanism consisting of two parts:

- *Associated XML node*. A node from the Page Sources Pane is dropped on the combo box to make the association with it. The associated node will receive the combo box value selected by the end-user. In order to use the end-user-selected value elsewhere in the page, the node's content is referenced by either dragging the node onto controls or selecting the node in XPath expressions.
- *Entry text and XML values*: The text of combo box entries and their corresponding values are specified with the `Combo Box Entry Values` property (*see below for details*).

⊟ **Notes**

- When the control is associated with a data source node (page source link), placing the mouse over the control (in Page Design View) displays the associated node in a popup.
- To remove a data-source node association, right-click the control (in Page Design View) and click **Delete Page Source Link**.
- To reset a style or property (in the Styles & Properties Pane), select the property and click **Reset** in the pane's toolbar.
- To edit the XPath expression of a style or property (in the Styles & Properties Pane), select the style or property, and click **Edit XPath** in the pane's toolbar.

Combo box properties

The control's properties are available in the Styles & Properties Pane, and are listed below in alphabetical order.

▼ **`Background Color`**

Sets the background color of the control. You can do one of the following to select the color:

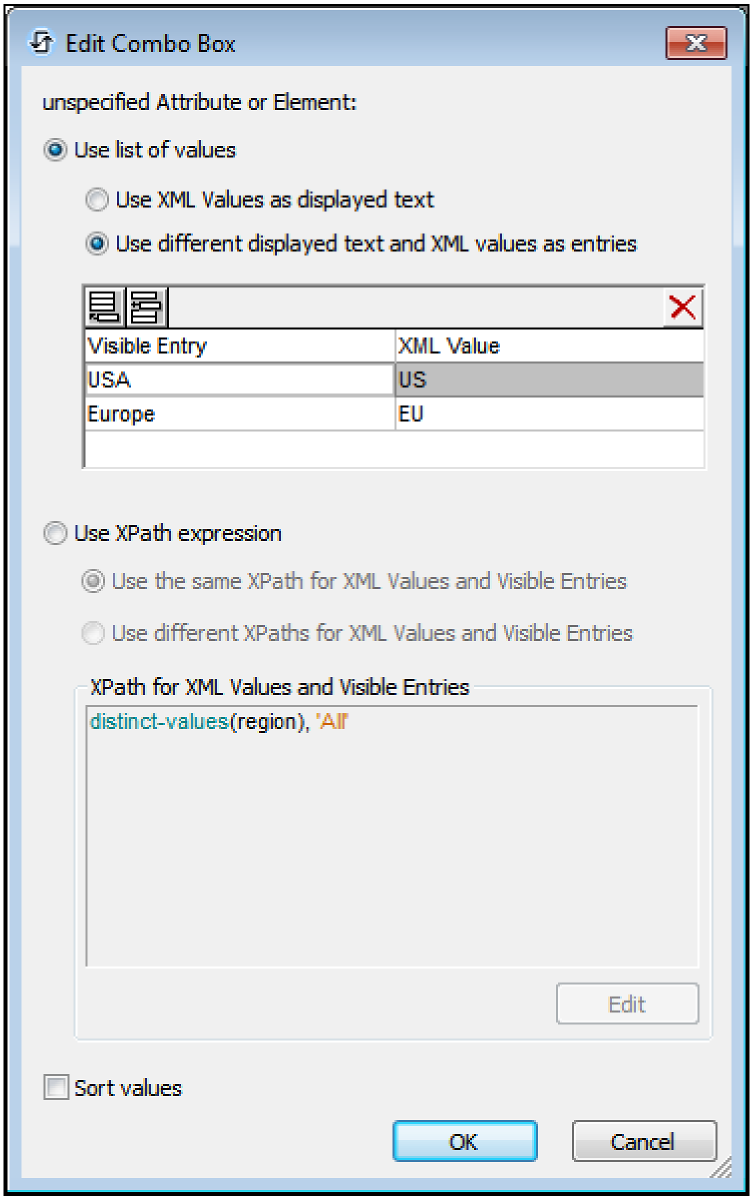
- Click the color palette to select a background color
- Select a color from the dropdown list of the combo box
- Double-click in the value field to enter a color code (for example, `#FF0000`), or click the **XPath** toolbar button and enter an XPath expression to generate the required text

▼ **`Bold Text`**

Select `true` or `false` from the dropdown list of the combo box. Default is `false`.

▼ **`Combo Box Entry Values`**

Provides XML data values for the items of the dropdown list of the combo box. Click the **Additional Dialog** button to display the Edit Combo Box dialog (*screenshot below*). Alternatively, you can double-click the combo box to display the Edit Combo Box dialog.



Edit Combo Box

unspecified Attribute or Element:

● Use list of values

○ Use XML Values as displayed text

● Use different displayed text and XML values as entries

Visible Entry	XML Value
USA	US
Europe	EU

○ Use XPath expression

● Use the same XPath for XML Values and Visible Entries

○ Use different XPaths for XML Values and Visible Entries

XPath for XML Values and Visible Entries

```
distinct-values(region), 'All'
```

□ Sort values

To define the entries and values for the combo box, do the following:

1. Select the method with which you wish to define the entries and values by clicking the appropriate radio button to select values: (i) list of values, or (ii) XPath expressions.
2. If you select *Use List of Values*, you can specify the text of entries that appear in the dropdown list of the combo box and the XML values that these entries map to. You can choose between using (i) the entry text as the XML value, and (ii) text-to-value mappings. You could also use an XPath expression to create the visible entries and XML values. The items in the sequence returned by an XPath expression are used for visible entries and/or XML values. You can specify: (i) that the same XPath expression be used for visible entries and for XML values, or (ii) that different XPath expressions be used for visible entries and for XML values. In the latter case, a one-to-one index mapping between the items of the two sequences determines the correspondence of visible entry to XML value. If the number of items in the two

sequences are not equal, an error is reported.

3. If you wish to have the items that appear in the drop-down list of the combo box sorted, check the *Sort Values* check box.
4. Click **OK** to finish.

Note

- Using an XPath expression to select the items of the combo box drop-down list enables you to create combo boxes with dynamic entries from an XML data source.
- If the items in the drop-down list of the combo box are obtained from from an XML data source, they will appear, by default, in document order.

▼ `Control Action`

Click the **Additional Dialog** button to pop up the control's Actions dialog.

▼ `Control Width`

Sets the width of the control. Select `fill_parent` or `wrap_content` from the dropdown list of the combo box. Default is `fill-parent`.

▼ `Enabled/Editable`

The control is either enabled (and editable) or it is disabled. The value of the property must be an XPath expression that evaluates to boolean `true` or `false`. If the expression evaluates to `true`, the control is enabled/editable. If the expression evaluates to `false`, the control is disabled. The default value is `true`. Typically the XPath expression of the `Enabled/Editable` property of a control would check the presence or value of a node. If the expression evaluates to `true`, the control will be enabled/editable.

▼ `Hide Keyboard On Click  [Android only]`

Hides the keyboard when the control is clicked. Default is `false`.

▼ `Italic Text`

Select `true` or `false` from the dropdown list of the combo box. Default is `false`.

▼ `Margin`

Sets the control's margin offsets relative to the surrounding objects or to the borders of the containing object. Select a value in pixels from the dropdown list of the combo box, or double-click in the value field to enter a pixel length.

▼ `Margin Bottom`

Sets the control's bottom margin offset relative to the object below it or to the border of the containing object. Select a value in pixels from the dropdown list of the combo box, or double-click in the value field to enter a pixel length.

▼ `Margin Left`

Sets the control's left margin offset relative to the object to the left or to the border of the containing object. Select a value in pixels from the dropdown list of the combo box, or double-click in the value field to enter a pixel length.

▼ `Margin Right`

Sets the control's right margin offset relative to the object to the right or to the border of the containing object. Select a value in pixels from the dropdown list of the combo box, or double-click in the value field to enter a pixel length.

▼ `Margin Top`

Sets the control's top margin offset relative to the object above it or to the border of the containing object. Select a value in pixels from the dropdown list of the combo box, or double-click in the value field to enter a pixel length.

▼ `Name`

The name is used to reference the control from elsewhere in the page or project. Double-click inside the value field to edit.

▼ `Text Color`

Sets the color of the control's text. You can do one of the following to select the color:

- Click the color palette to select a text color
- Select a color from the dropdown list of the combo box
- Double-click in the value field to enter a color code (for example, `#FF0000`), or click the **XPath** toolbar button and enter an XPath expression to generate the required text

▼ `Text Size`

Select a size from the dropdown list of the combo box, or double-click in the value field to enter a text size.

▼ `Underline Text`

Select `true` or `false` from the dropdown list of the combo box. Default is `false`.

▼ `Visible`

An XPath expression that should evaluate to `true()` or `false()`. If the expression evaluates to `false()`, and only if it evaluates to `false()`, is the control not visible. If the expression evaluates to `true()` or returns some other value, then the control is visible. The default is `true()`. Double-click inside the value field, or click the **XPath** button, to enter or edit an XPath expression.

Date

Date controls are used to format dates obtained from a node in a data source. This is useful if you wish to format dates differently for different regions, for example: `12-31-2014` (US format) and `31-12-2014` (EU format). The formatting is defined in the `Date/Time Format String` property (*see below for details*). Note that the source node content must be in the correct lexical format as defined by the XSD specification: `YYYY-MM-DD`.

⊟ **Notes**

- When the control is associated with a data source node (page source link), placing the mouse over the control (in Page Design View) displays the associated node in a popup.
- To remove a data-source node association, right-click the control (in Page Design View) and click **Delete Page Source Link**.
- To reset a style or property (in the Styles & Properties Pane), select the property and click **Reset** in the pane's toolbar.
- To edit the XPath expression of a style or property (in the Styles & Properties Pane), select the style or property, and click **Edit XPath** in the pane's toolbar.

Date properties

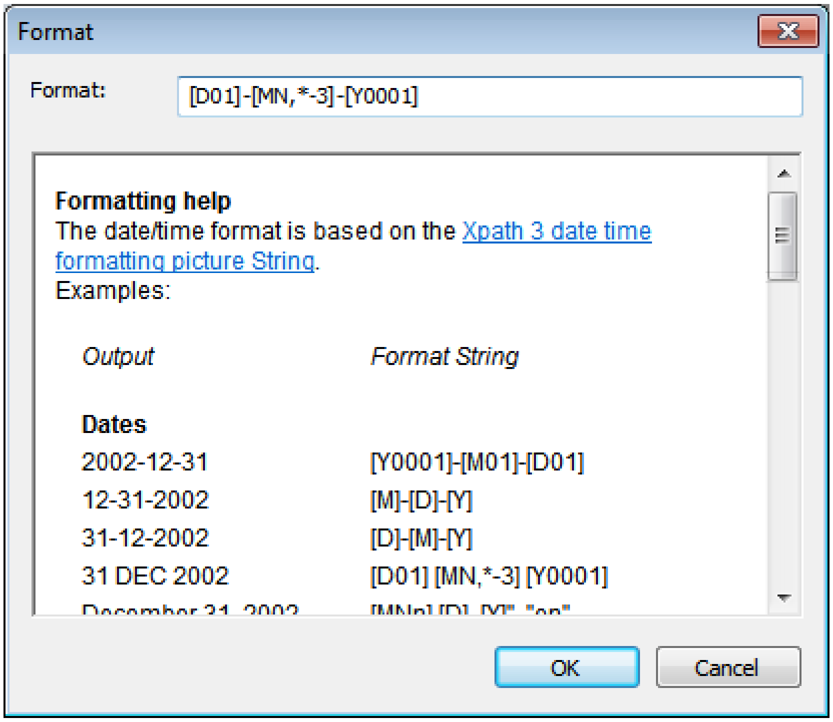
The control's properties are available in the Styles & Properties Pane, and are listed below in alphabetical order.

▼ **`Name`**

The name is used to reference the control from elsewhere in the page or project. Double-click inside the value field to edit.

▼ **`Date/Time Format String`**

Click the **Additional Dialog** button and enter a date, time, or date-time format in the Format dialog that appears (*screenshot below*).



Format: [D01]-[MN,*-3]-[Y0001]

Formatting help
The date/time format is based on the Xpath 3 date time formatting picture String.
Examples:

Output	*Format String*
Dates	
2002-12-31	[Y0001]-[M01]-[D01]
12-31-2002	[M]-[D]-[Y]
31-12-2002	[D]-[M]-[Y]
31 DEC 2002	[D01] [MN,*-3] [Y0001]
December 31, 2002	[MNn] [D], [Y]", "en"

The formatting will be applied to the control's content if the content has the correct lexical form of `xs:date` (for the Date control), `xs:time` (for the Time control), or `xs:dateTime` (for the Date, Time, and DateTime controls). Basic examples are:

- *xs:date:* `2014-12-31`
- *xs:time:* `23:59:59`
- *xs:dateTime:* `2014-12-31T23:59:59`

Time Handling

Specifies how to handle the `Time` part of a value. The three available options are:

- *Cut time*, which removes the `Time` part
- *Set time to zero*, which keeps the `Time` part, but changes it to `00:00:00`
- *Keep time*, which keeps the `Time` part

Assertion

Sets a condition to be met in order for the control's assertion to be valid. If the assertion is invalid, then the text of the `Assertion Message` property (*see below*) is displayed by the Assertion Message control. The assertion's condition is specified with an XPath expression. For example: The XPath expression **`LastName != ""`** asserts that the node `LastName` must not be empty. If this node is empty, then the control's assertion message is displayed in the page at the point where the Assertion Message control is inserted. If there are multiple invalid assertions on a page, then the assertion message of the first invalid assertion (in the sequence in which assertions appear on the page) is displayed.

Assertion Message

Sets the assertion message to be displayed if the control's assertion is not valid. Double-click inside the value field of the property to edit the assertion message, or click the **XPath** toolbar button and enter an XPath expression to generate the required text. The text of the assertion message property is displayed by the Assertion Message control. The assertion's condition is specified with an XPath expression. For example: The XPath expression **`LastName != ""`** asserts that the node `LastName` must not be empty. If this node is empty, then the control's assertion message is displayed in the page at the point where the Assertion Message control is inserted. If there are multiple invalid assertions on a page, then the assertion message of the first invalid assertion (in the sequence in which assertions appear on the page) is displayed.

▼ Text Color

Sets the color of the control's text. You can do one of the following to select the color:

- Click the color palette to select a text color
- Select a color from the dropdown list of the combo box
- Double-click in the value field to enter a color code (for example, `#FF0000`), or click the **XPath** toolbar button and enter an XPath expression to generate the required text

▼ Text Size

Select a size from the dropdown list of the combo box, or double-click in the value field to enter a text size.

▼ Bold Text

Select `true` or `false` from the dropdown list of the combo box. Default is `false`.

▼ Italic Text

Select `true` or `false` from the dropdown list of the combo box. Default is `false`.

▼ Underline Text

Select `true` or `false` from the dropdown list of the combo box. Default is `false`.

▼ Background Color

Sets the background color of the control. You can do one of the following to select the color:

- Click the color palette to select a background color
- Select a color from the dropdown list of the combo box
- Double-click in the value field to enter a color code (for example, `#FF0000`), or click the **XPath** toolbar button and enter an XPath expression to generate the required text

▼ Hint

Provides a textual hint to the end-user. For example, the hint could be about an action that the end-user has to carry out by using the control. Double-click inside the value field of the property to edit the textual hint, or click the **XPath** toolbar button and enter an XPath expression to generate the required text..

▼ Control Action

Click the **Additional Dialog** button to pop up the control's Actions dialog.

▼ `Control Width`

Sets the width of the control. Select `fill_parent` or `wrap_content` from the dropdown list of the combo box. Default is `fill-parent`.

▼ `Visible`

An XPath expression that should evaluate to `true()` or `false()`. If the expression evaluates to `false()`, and only if it evaluates to `false()`, is the control not visible. If the expression evaluates to `true()` or returns some other value, then the control is visible. The default is `true()`. Double-click inside the value field, or click the **XPath** button, to enter or edit an XPath expression.

▼ `Enabled/Editable`

The control is either enabled (and editable) or it is disabled. The value of the property must be an XPath expression that evaluates to boolean `true` or `false`. If the expression evaluates to `true`, the control is enabled/editable. If the expression evaluates to `false`, the control is disabled. The default value is `true`. Typically the XPath expression of the `Enabled/Editable` property of a control would check the presence or value of a node. If the expression evaluates to `true`, the control will be enabled/editable.

▼ `Margin`

Sets the control's margin offsets relative to the surrounding objects or to the borders of the containing object. Select a value in pixels from the dropdown list of the combo box, or double-click in the value field to enter a pixel length.

▼ `Margin Left`

Sets the control's left margin offset relative to the object to the left or to the border of the containing object. Select a value in pixels from the dropdown list of the combo box, or double-click in the value field to enter a pixel length.

▼ `Margin Right`

Sets the control's right margin offset relative to the object to the right or to the border of the containing object. Select a value in pixels from the dropdown list of the combo box, or double-click in the value field to enter a pixel length.

▼ `Margin Top`

Sets the control's top margin offset relative to the object above it or to the border of the containing object. Select a value in pixels from the dropdown list of the combo box, or double-click in the value field to enter a pixel length.

▼ `Margin Bottom`

Sets the control's bottom margin offset relative to the object below it or to the border of the containing object. Select a value in pixels from the dropdown list of the combo box, or double-click in the value field to enter a pixel length.

DateTime (iOS)

DateTime controls are available only on iOS client devices and are used to format dateTimes obtained from a node in a data source. This is useful if you wish to format dateTimes differently for different regions, for example: `12-31-2014 12:00:00` (US format) and `31-12-2014 12:00:00` (EU format). The formatting is defined in the `Date/Time Format String` property (*see below for details*). Note that the source node content must be in the correct lexical format as defined by the XSD specification: `YYYY-MM-DDTHH:MM:SS`. Optional timezone and millisecond parts are allowed.

⊟ **Notes**

- When the control is associated with a data source node (page source link), placing the mouse over the control (in Page Design View) displays the associated node in a popup.
- To remove a data-source node association, right-click the control (in Page Design View) and click **Delete Page Source Link**.
- To reset a style or property (in the Styles & Properties Pane), select the property and click **Reset** in the pane's toolbar.
- To edit the XPath expression of a style or property (in the Styles & Properties Pane), select the style or property, and click **Edit XPath** in the pane's toolbar.

DateTime properties

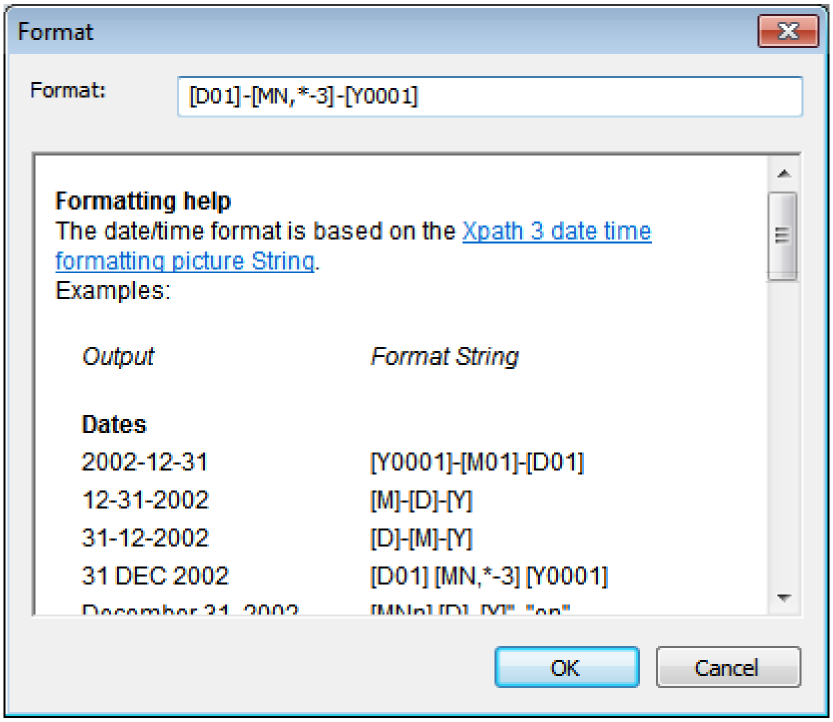
The control's properties are available in the Styles & Properties Pane, and are listed below in alphabetical order.

▼ **`Name`**

The name is used to reference the control from elsewhere in the page or project. Double-click inside the value field to edit.

▼ **`Date/Time Format String`**

Click the **Additional Dialog** button and enter a date, time, or date-time format in the Format dialog that appears (*screenshot below*).



Format: [D01]-[MN,*-3]-[Y0001]

Formatting help
The date/time format is based on the Xpath 3 date time formatting picture String.
Examples:

Output	*Format String*
Dates	
2002-12-31	[Y0001]-[M01]-[D01]
12-31-2002	[M]-[D]-[Y]
31-12-2002	[D]-[M]-[Y]
31 DEC 2002	[D01] [MN,*-3] [Y0001]
December 31, 2002	[MNn] [D], [Y]", "en"

The formatting will be applied to the control's content if the content has the correct lexical form of `xs:date` (for the Date control), `xs:time` (for the Time control), or `xs:dateTime` (for the Date, Time, and DateTime controls). Basic examples are:

- *xs:date:* `2014-12-31`
- *xs:time:* `23:59:59`
- *xs:dateTime:* `2014-12-31T23:59:59`

`Assertion`

Sets a condition to be met in order for the control's assertion to be valid. If the assertion is invalid, then the text of the `Assertion Message` property (*see below*) is displayed by the Assertion Message control. The assertion's condition is specified with an XPath expression. For example: The XPath expression **`LastName != ""`** asserts that the node `LastName` must not be empty. If this node is empty, then the control's assertion message is displayed in the page at the point where the Assertion Message control is inserted. If there are multiple invalid assertions on a page, then the assertion message of the first invalid assertion (in the sequence in which assertions appear on the page) is displayed.

`Assertion Message`

Sets the assertion message to be displayed if the control's assertion is not valid. Double-click inside the value field of the property to edit the assertion message, or click the **XPath** toolbar button and enter an XPath expression to generate the required text. The text of the assertion message property is displayed by the Assertion Message control. The assertion's condition is specified with an XPath expression. For example: The XPath expression **`LastName != ""`** asserts that the node `LastName` must not be empty. If this node is empty, then the control's assertion message is displayed in the page at the point where the Assertion Message control is inserted. If there are multiple invalid assertions on a page, then the assertion message of the first invalid assertion (in the sequence in which assertions appear on the page) is displayed.

`Text Color`

Sets the color of the control's text. You can do one of the following to select the color:

- Click the color palette to select a text color
- Select a color from the dropdown list of the combo box
- Double-click in the value field to enter a color code (for example, `#FF0000`), or click the **XPath** toolbar button and enter an XPath expression to generate the required text

`Text Size`

Select a size from the dropdown list of the combo box, or double-click in the value field to enter a text size.

`Bold Text`

Select `true` or `false` from the dropdown list of the combo box. Default is `false`.

`Italic Text`

Select `true` or `false` from the dropdown list of the combo box. Default is `false`.

`Underline Text`

Select `true` or `false` from the dropdown list of the combo box. Default is `false`.

`Background Color`

Sets the background color of the control. You can do one of the following to select the color:

- Click the color palette to select a background color
- Select a color from the dropdown list of the combo box
- Double-click in the value field to enter a color code (for example, `#FF0000`), or click the **XPath** toolbar button and enter an XPath expression to generate the required text

`Hint`

Provides a textual hint to the end-user. For example, the hint could be about an action that the end-user has to carry out by using the control. Double-click inside the value field of the property to edit the textual hint, or click the **XPath** toolbar button and enter an XPath expression to generate the required text..

`Control Action`

Click the **Additional Dialog** button to pop up the control's Actions dialog.

`Control Width`

Sets the width of the control. Select `fill_parent` or `wrap_content` from the dropdown list of the combo box. Default is `fill-parent`.

▼ **`Visible`**

An XPath expression that should evaluate to `true()` or `false()`. If the expression evaluates to `false()`, and only if it evaluates to `false()`, is the control not visible. If the expression evaluates to `true()` or returns some other value, then the control is visible. The default is `true()`. Double-click inside the value field, or click the **XPath** button, to enter or edit an XPath expression.

▼ **`Enabled/Editable`**

The control is either enabled (and editable) or it is disabled. The value of the property must be an XPath expression that evaluates to boolean `true` or `false`. If the expression evaluates to `true`, the control is enabled/editable. If the expression evaluates to `false`, the control is disabled. The default value is `true`. Typically the XPath expression of the `Enabled/Editable` property of a control would check the presence or value of a node. If the expression evaluates to `true`, the control will be enabled/editable.

▼ **`Margin`**

Sets the control's margin offsets relative to the surrounding objects or to the borders of the containing object. Select a value in pixels from the dropdown list of the combo box, or double-click in the value field to enter a pixel length.

▼ **`Margin Left`**

Sets the control's left margin offset relative to the object to the left or to the border of the containing object. Select a value in pixels from the dropdown list of the combo box, or double-click in the value field to enter a pixel length.

▼ **`Margin Right`**

Sets the control's right margin offset relative to the object to the right or to the border of the containing object. Select a value in pixels from the dropdown list of the combo box, or double-click in the value field to enter a pixel length.

▼ **`Margin Top`**

Sets the control's top margin offset relative to the object above it or to the border of the containing object. Select a value in pixels from the dropdown list of the combo box, or double-click in the value field to enter a pixel length.

▼ **`Margin Bottom`**

Sets the control's bottom margin offset relative to the object below it or to the border of the containing object. Select a value in pixels from the dropdown list of the combo box, or double-click in the value field to enter a pixel length.

Edit Field

Edit Fields, typically, are associated with a node in a data source, but can have static text or an XPath expression associated with it. If associated with a node, the content of the node can be edited by the end-user and the modified content can be saved to file and/or be used as input for other controls. It is possible to specify an XPath expression (via the `Text` property) to generate text for the control. The XPath expression could select some node or generate some static content. However, if static text is required, the Label control would be more suitable.

⊟ **Notes**

- When the control is associated with a data source node (page source link), placing the mouse over the control (in Page Design View) displays the associated node in a popup.
- To remove a data-source node association, right-click the control (in Page Design View) and click **Delete Page Source Link**.
- To reset a style or property (in the Styles & Properties Pane), select the property and click **Reset** in the pane's toolbar.
- To edit the XPath expression of a style or property (in the Styles & Properties Pane), select the style or property, and click **Edit XPath** in the pane's toolbar.

Edit Field properties

The control's properties are available in the Styles & Properties Pane, and are listed below in alphabetical order.

▼ **`Assertion`**

Sets a condition to be met in order for the control's assertion to be valid. If the assertion is invalid, then the text of the `Assertion Message` property (*see below*) is displayed by the Assertion Message control. The assertion's condition is specified with an XPath expression. For example: The XPath expression **`LastName != ""`** asserts that the node `LastName` must not be empty. If this node is empty, then the control's assertion message is displayed in the page at the point where the Assertion Message control is inserted. If there are multiple invalid assertions on a page, then the assertion message of the first invalid assertion (in the sequence in which assertions appear on the page) is displayed.

▼ **`Assertion Message`**

Sets the assertion message to be displayed if the control's assertion is not valid. Double-click inside the value field of the property to edit the assertion message, or click the **XPath** toolbar button and enter an XPath expression to generate the required text. The text of the assertion message property is displayed by the Assertion Message control. The assertion's condition is specified with an XPath expression. For example: The XPath expression **`LastName != ""`** asserts that the node `LastName` must not be empty. If this node is empty, then the control's assertion message is displayed in the page at the point where the Assertion Message control is inserted. If there are multiple invalid assertions on a page, then the assertion message of the first invalid assertion (in the sequence in which assertions appear on the page) is displayed.

Background Color

Sets the background color of the control. You can do one of the following to select the color:

- Click the color palette to select a background color
- Select a color from the dropdown list of the combo box
- Double-click in the value field to enter a color code (for example, `#FF0000`), or click the **XPath** toolbar button and enter an XPath expression to generate the required text

Bold Text

Select `true` or `false` from the dropdown list of the combo box. Default is `false`.

Control Action

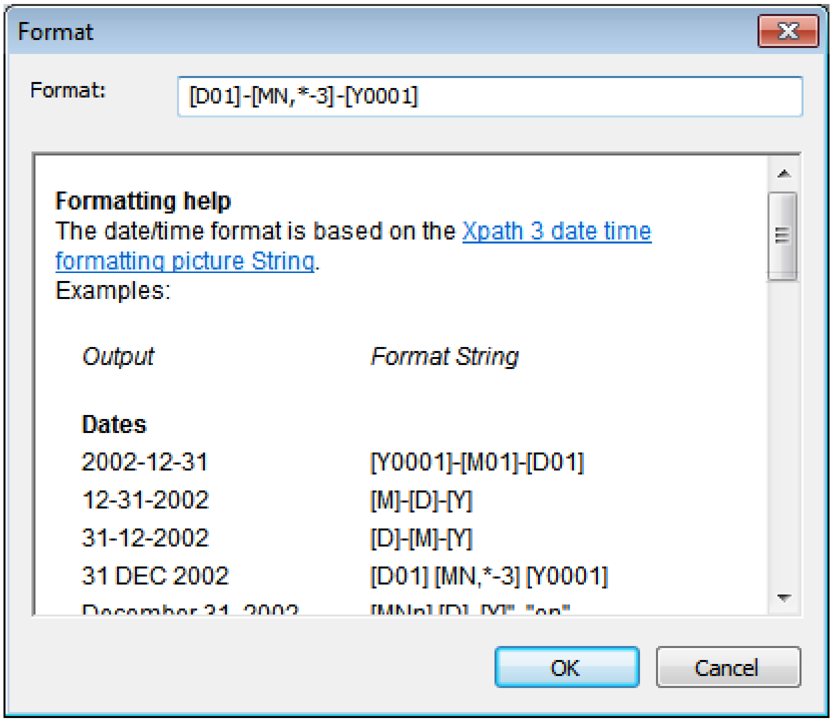
Click the **Additional Dialog** button to pop up the control's Actions dialog.

Control Width

Sets the width of the control. Select `fill_parent` or `wrap_content` from the dropdown list of the combo box. Default is `fill-parent`.

Date/Time Format String

Click the **Additional Dialog** button and enter a date, time, or date-time format in the Format dialog that appears (*screenshot below*).



Format: [D01]-[MN,*-3]-[Y0001]

Formatting help
The date/time format is based on the Xpath 3 date time formatting picture String.
Examples:

Output	*Format String*
Dates	
2002-12-31	[Y0001]-[M01]-[D01]
12-31-2002	[M]-[D]-[Y]
31-12-2002	[D]-[M]-[Y]
31 DEC 2002	[D01] [MN,*-3] [Y0001]
December 31, 2002	[MNn] [D], [Y]", "en"

The formatting will be applied to the control's content if the content has the correct lexical form of `xs:date` (for the Date control), `xs:time` (for the Time control), or `xs:dateTime` (for the Date, Time, and DateTime controls). Basic examples are:

- *xs:date:* `2014-12-31`
- *xs:time:* `23:59:59`
- *xs:dateTime:* `2014-12-31T23:59:59`

Enabled/Editable

The control is either enabled (and editable) or it is disabled. The value of the property must be an XPath expression that evaluates to boolean `true` or `false`. If the expression evaluates to `true`, the control is enabled/editable. If the expression evaluates to `false`, the control is disabled. The default value is `true`. Typically the XPath expression of the `Enabled/Editable` property of a control would check the presence or value of a node. If the expression evaluates to `true`, the control will be enabled/editable.

Hint

Provides a textual hint to the end-user. For example, the hint could be about an action that the end-user has to carry out by using the control. Double-click inside the value field of the property to edit the textual hint, or click the **XPath** toolbar button and enter an XPath expression to generate the required text..

Italic Text

Select `true` or `false` from the dropdown list of the combo box. Default is `false`.

Keyboard

Sets the end-user keyboard type to activate for data entry. Select one of the following options from the dropdown list of the combo box:

- Text
- Number
- Password
- Email
- URI
- Phone

Margin

Sets the control's margin offsets relative to the surrounding objects or to the borders of the containing object. Select a value in pixels from the dropdown list of the combo box, or double-click in the value field to enter a pixel length.

Margin Bottom

Sets the control's bottom margin offset relative to the object below it or to the border of the containing object. Select a value in pixels from the dropdown list of the combo box, or double-click in the value field to enter a pixel length.

Margin Left

Sets the control's left margin offset relative to the object to the left or to the border of the containing object. Select a value in pixels from the dropdown list of the combo box, or double-click in the value field to enter a pixel length.

Margin Right

Sets the control's right margin offset relative to the object to the right or to the border of the containing object. Select a value in pixels from the dropdown list of the combo box, or double-click in the value field to enter a pixel length.

▼ **`Margin Top`**

Sets the control's top margin offset relative to the object above it or to the border of the containing object. Select a value in pixels from the dropdown list of the combo box, or double-click in the value field to enter a pixel length.

▼ **`Multiline`**

Sets multiline input/display on or off (`true/false`). The default is `false`.

▼ **`Name`**

The name is used to reference the control from elsewhere in the page or project. Double-click inside the value field to edit.

▼ **`Number Format String`**

Click the **Additional Dialog** button and enter a number format in the Format dialog that pops up. The formatting will be applied to label content if the label content is numeric.

▼ **`Text`**

The Text property takes as its value either of the following:

- The text to be displayed in the control
- An XPath expression that retrieves data from a node in a data source.

Double-click inside the value field to edit, or click the **XPath** toolbar button and enter an XPath expression to generate the required text.

▼ **`Text Color`**

Sets the color of the control's text. You can do one of the following to select the color:

- Click the color palette to select a text color
- Select a color from the dropdown list of the combo box
- Double-click in the value field to enter a color code (for example, `#FF0000`), or click the **XPath** toolbar button and enter an XPath expression to generate the required text

▼ **`Text Color Hint`**

Sets a text color hint for the control. You can do one of the following to select the color:

- Click the color palette to select a background color
- Select a color from the dropdown list of the combo box
- Double-click in the value field to enter a color code (for example, `#FF0000`), or click the **XPath** toolbar button and enter an XPath expression to generate the required text.

▼ **`Text Size`**

Select a size from the dropdown list of the combo box, or double-click in the value field to enter a text size.

▼ **`Trigger Control Actions During Typing`**

Control actions can be triggered as soon as typing starts. For example, the page can be scrolled to bottom when typing starts, or a node can be updated while typing. Property values are `true` or `false`. Default is `true`. For HTML, the value is always `false`.

▼ **`Underline Text`**

Select `true` or `false` from the dropdown list of the combo box. Default is `false`.

▼ **`Visible`**

An XPath expression that should evaluate to `true()` or `false()`. If the expression evaluates to `false()`, and only if it evaluates to `false()`, is the control not visible. If the expression evaluates to `true()` or returns some other value, then the control is visible. The default is `true()`. Double-click inside the value field, or click the **XPath** button, to enter or edit an XPath expression.

Horizontal Line

The Horizontal Line control enables you to add lines that you can style for color and width.

Horizontal Line properties

The control's properties are available in the Styles & Properties Pane, and are listed below in alphabetical order.

Name

The name is used to reference the control from elsewhere in the page or project. Double-click inside the value field to edit.

Line Width

Sets the width (thickness) in pixels of the selected horizontal line. Select a width value (in pixels) from the dropdown list of the combo box, or double-click in the value field to enter a numeric value. The numeric value is understood to be a pixel value. For this reason, no unit should be specified.

Line Color

Specifies the color of the selected horizontal line. You can do one of the following to select the color:

- Click the color palette to select a line color
- Select a color from the dropdown list of the combo box
- Double-click in the value field to enter a color code (for example, `#FF0000`), or click the **XPath** toolbar button and enter an XPath expression to generate the required text

Visible

An XPath expression that should evaluate to `true()` or `false()`. If the expression evaluates to `false()`, and only if it evaluates to `false()`, is the control not visible. If the expression evaluates to `true()` or returns some other value, then the control is visible. The default is `true()`. Double-click inside the value field, or click the **XPath** button, to enter or edit an XPath expression.

Image

The Image control inserts an image in the design. The control's `Image URL` property (*see below*) takes the location of the image via the Specify File dialog. To specify that the image URL is taken from a data source node, drop that node onto the image control. Other properties are listed below.

⊟ **Notes**

- When the control is associated with a data source node (page source link), placing the mouse over the control (in Page Design View) displays the associated node in a popup.
- To remove a data-source node association, right-click the control (in Page Design View) and click **Delete Page Source Link**.
- To reset a style or property (in the Styles & Properties Pane), select the property and click **Reset** in the pane's toolbar.
- To edit the XPath expression of a style or property (in the Styles & Properties Pane), select the style or property, and click **Edit XPath** in the pane's toolbar.

Image properties

The control's properties are available in the Styles & Properties Pane, and are listed below in alphabetical order.

▼ **`Name`**

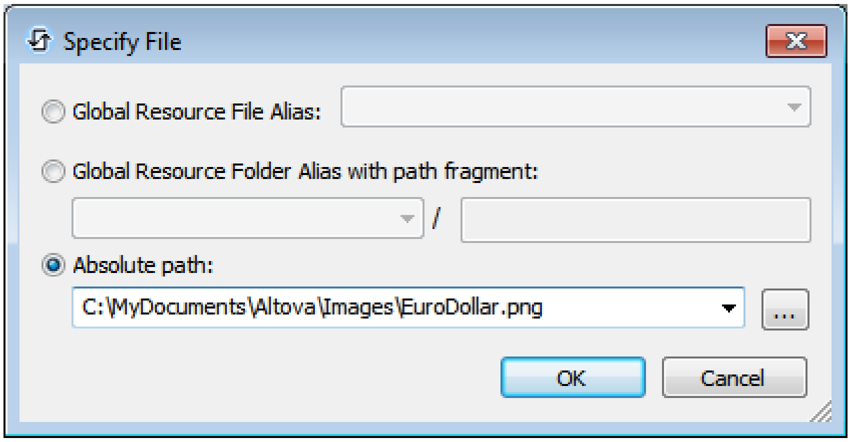
The name is used to reference the control from elsewhere in the page or project. Double-click inside the value field to edit.

▼ **`Image URL`**

The image URL can be supplied by an XPath expression, or it can be supplied directly as a file URL.

To supply the URL via an XPath expression, select the `Image URL` property and click the XPath icon of the Styles & Properties Pane. In the Edit XPath/XQuery Expression dialog that appears, enter the XPath expression, and click **OK** when done.

To directly enter the URL of the image file, click the **Additional Dialog** button to display the Specify File dialog (*screenshot below*). You can also double-click the Image control to display this dialog.



Specify File

○ Global Resource File Alias:

○ Global Resource Folder Alias with path fragment: /

● Absolute path: C:\MyDocuments\Altova\Images\EuroDollar.png

In the dialog, you can select a Global Resource file or folder alias from the respective combo boxes, or browse for the image file. Click **OK** to finish.

Username

Sets a user name for user access to content. Double-click in the property's value field to edit.

Password

Sets a password for user access to content. Double-click in the property's value field to edit.

Create Before Load

Click the color palette to select a background color, or select a color from the dropdown list of the combo box, or double-click in the value field to enter a color code.

Control Width

Sets the width of the control. Select `fill_parent` or `wrap_content` from the dropdown list of the combo box. Default is `fill-parent`.

Visible

An XPath expression that should evaluate to `true()` or `false()`. If the expression evaluates to `false()`, and only if it evaluates to `false()`, is the control not visible. If the expression evaluates to `true()` or returns some other value, then the control is visible. The default is `true()`. Double-click inside the value field, or click the **XPath** button, to enter or edit an XPath expression.

Margin

Sets the control's margin offsets relative to the surrounding objects or to the borders of the containing object. Select a value in pixels from the dropdown list of the combo box, or double-click in the value field to enter a pixel length.

Margin Left

Sets the control's left margin offset relative to the object to the left or to the border of the containing object. Select a value in pixels from the dropdown list of the combo box, or double-click in the value field to enter a pixel length.

▼ **`Margin Right`**

Sets the control's right margin offset relative to the object to the right or to the border of the containing object. Select a value in pixels from the dropdown list of the combo box, or double-click in the value field to enter a pixel length.

▼ **`Margin Top`**

Sets the control's top margin offset relative to the object above it or to the border of the containing object. Select a value in pixels from the dropdown list of the combo box, or double-click in the value field to enter a pixel length.

▼ **`Margin Bottom`**

Sets the control's bottom margin offset relative to the object below it or to the border of the containing object. Select a value in pixels from the dropdown list of the combo box, or double-click in the value field to enter a pixel length.

Label

Labels are used to display information. The information displayed could be static, entered by the designer as the `Text` property of the label. Or it could be dynamic, obtained at runtime from one of the page data sources. To specify dynamic content for the label, either drag the required data node from the Page Sources Pane onto the label, or enter, as the `Text` property of the label, an XPath expression that retrieves data from a node in a data source.

Label properties
The control's properties are available in the Styles & Properties Pane, and are listed below in alphabetical order.

▼ **`Name`**

The name is used to reference the control from elsewhere in the page or project. Double-click inside the value field to edit.

▼ **`Text`**

The Text property takes as its value either of the following:

- The text to be displayed in the control
- An XPath expression that retrieves data from a node in a data source.

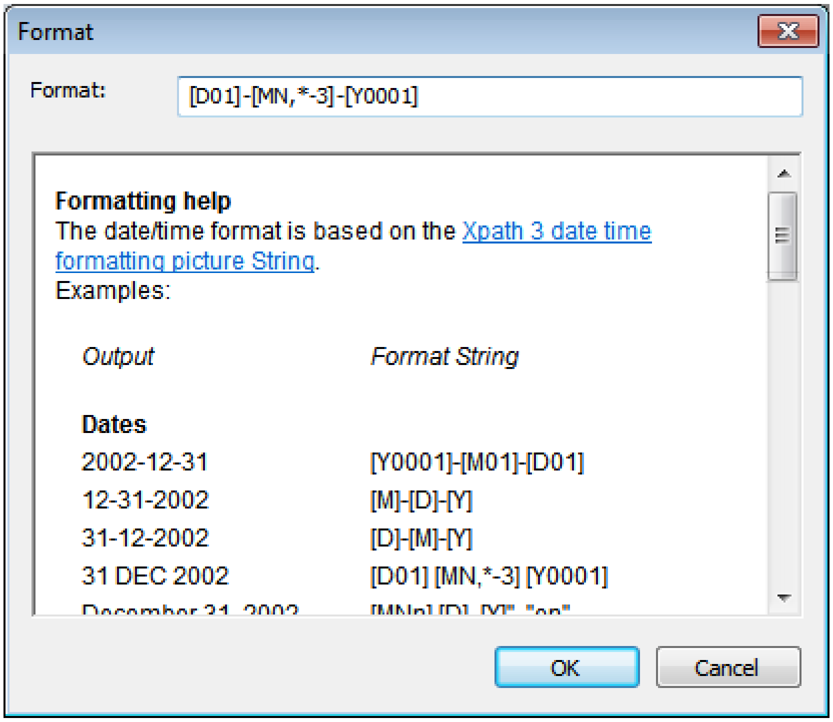
Double-click inside the value field to edit, or click the **XPath** toolbar button and enter an XPath expression to generate the required text.

▼ **`Number Format String`**

Click the **Additional Dialog** button and enter a number format in the Format dialog that pops up. The formatting will be applied to label content if the label content is numeric.

▼ **`Date/Time Format String`**

Click the **Additional Dialog** button and enter a date, time, or date-time format in the Format dialog that appears (*screenshot below*).



Format: [D01]-[MN,*-3]-[Y0001]

Formatting help
The date/time format is based on the Xpath 3 date time formatting picture String.
Examples:

Output	*Format String*
Dates	
2002-12-31	[Y0001]-[M01]-[D01]
12-31-2002	[M]-[D]-[Y]
31-12-2002	[D]-[M]-[Y]
31 DEC 2002	[D01] [MN,*-3] [Y0001]
December 31, 2002	[MNn] [D], [Y]", "en"

The formatting will be applied to the control's content if the content has the correct lexical form of `xs:date` (for the Date control), `xs:time` (for the Time control), or `xs:dateTime` (for the Date, Time, and DateTime controls). Basic examples are:

- *xs:date:* `2014-12-31`
- *xs:time:* `23:59:59`
- *xs:dateTime:* `2014-12-31T23:59:59`

Multiline

Sets multiline input/display on or off (`true/false`). The default is `false`.

Hide Keyboard On Click [Android only]

Hides the keyboard when the control is clicked. Default is `false`.

Control Action

Click the **Additional Dialog** button to pop up the control's Actions dialog.

Text Color

Sets the color of the control's text. You can do one of the following to select the color:

- Click the color palette to select a text color
- Select a color from the dropdown list of the combo box
- Double-click in the value field to enter a color code (for example, `#FF0000`), or click the **XPath** toolbar button and enter an XPath expression to generate the required text

▼ `Text Size`

Select a size from the dropdown list of the combo box, or double-click in the value field to enter a text size.

▼ `Bold Text`

Select `true` or `false` from the dropdown list of the combo box. Default is `false`.

▼ `Italic Text`

Select `true` or `false` from the dropdown list of the combo box. Default is `false`.

▼ `Underline Text`

Select `true` or `false` from the dropdown list of the combo box. Default is `false`.

▼ `Background Color`

Sets the background color of the control. You can do one of the following to select the color:

- Click the color palette to select a background color
- Select a color from the dropdown list of the combo box
- Double-click in the value field to enter a color code (for example, `#FF0000`), or click the **XPath** toolbar button and enter an XPath expression to generate the required text

▼ `Control Width`

Sets the width of the control. Select `fill_parent` or `wrap_content` from the dropdown list of the combo box. Default is `fill-parent`.

▼ `Visible`

An XPath expression that should evaluate to `true()` or `false()`. If the expression evaluates to `false()`, and only if it evaluates to `false()`, is the control not visible. If the expression evaluates to `true()` or returns some other value, then the control is visible. The default is `true()`. Double-click inside the value field, or click the **XPath** button, to enter or edit an XPath expression.

▼ `Enabled/Editable`

The control is either enabled (and editable) or it is disabled. The value of the property must be an XPath expression that evaluates to boolean `true` or `false`. If the expression evaluates to `true`, the control is enabled/editable. If the expression evaluates to `false`, the control is disabled. The default value is `true`. Typically the XPath expression of the `Enabled/Editable` property of a control would check the presence or value of a node. If the expression evaluates to `true`, the control will be enabled/editable.

▼ `Margin`

Sets the control's margin offsets relative to the surrounding objects or to the borders of the containing object. Select a value in pixels from the dropdown list of the combo box, or double-click in the value field to enter a pixel length.

▼ `Margin Left`

Sets the control's left margin offset relative to the object to the left or to the border of the containing object. Select a value in pixels from the dropdown list of the combo box, or double-click in the value field to enter a pixel length.

▼ **`Margin Right`**

Sets the control's right margin offset relative to the object to the right or to the border of the containing object. Select a value in pixels from the dropdown list of the combo box, or double-click in the value field to enter a pixel length.

▼ **`Margin Top`**

Sets the control's top margin offset relative to the object above it or to the border of the containing object. Select a value in pixels from the dropdown list of the combo box, or double-click in the value field to enter a pixel length.

▼ **`Margin Bottom`**

Sets the control's bottom margin offset relative to the object below it or to the border of the containing object. Select a value in pixels from the dropdown list of the combo box, or double-click in the value field to enter a pixel length.

Space

The Space control enables you to add vertical space to the design. This is useful if you wish to have precise control over the space between design components.

Space properties

The control's properties are available in the Styles & Properties Pane, and are listed below in alphabetical order.

▼ **`Name`**

The name is used to reference the control from elsewhere in the page or project. Double-click inside the value field to edit.

▼ **`Space Height`**

Sets the height in pixels of the selected space object. Select a height value (in pixels) from the dropdown list of the combo box, or double-click in the value field to enter a numeric value. The numeric value is understood to be a pixel value. For this reason, no unit should be specified.

▼ **`Visible`**

An XPath expression that should evaluate to `true()` or `false()`. If the expression evaluates to `false()`, and only if it evaluates to `false()`, is the control not visible. If the expression evaluates to `true()` or returns some other value, then the control is visible. The default is `true()`. Double-click inside the value field, or click the **XPath** button, to enter or edit an XPath expression.

Switch

A switch is associated with a data source node. The end-user can choose between two states: *selected* or *unselected*. Depending on which state is chosen, a corresponding XML value is entered as content of the associated node. The values corresponding to the respective states (*selected* or *unselected*) are defined in the `Checked Values (true/false)` property (*see property description below*). A typical use case would be: first, let the end-user's switch selection determine the content of a node; second, let the content of this node determine a page action, such as, whether a control is enabled/editable.

⊟ **Notes**

- When the control is associated with a data source node (page source link), placing the mouse over the control (in Page Design View) displays the associated node in a popup.
- To remove a data-source node association, right-click the control (in Page Design View) and click **Delete Page Source Link**.
- To reset a style or property (in the Styles & Properties Pane), select the property and click **Reset** in the pane's toolbar.
- To edit the XPath expression of a style or property (in the Styles & Properties Pane), select the style or property, and click **Edit XPath** in the pane's toolbar.

Switch properties

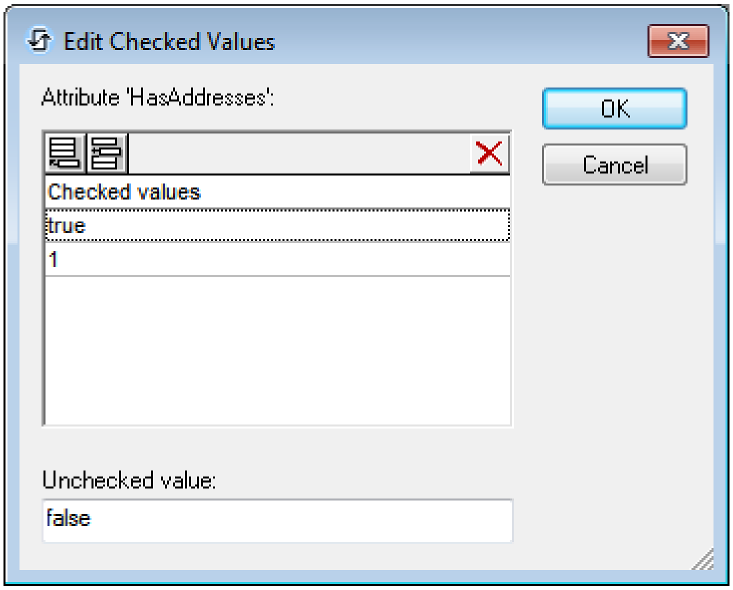
The control's properties are available in the Styles & Properties Pane, and are listed below in alphabetical order.

▼ **`Name`**

The name is used to reference the control from elsewhere in the page or project. Double-click inside the value field to edit.

▼ **`Checked Values (true/false)`**

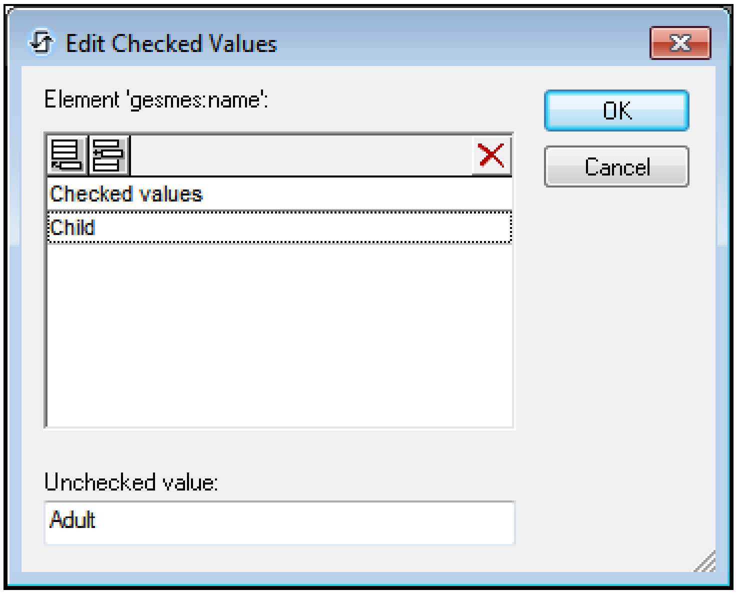
Provides an XML data value for the selected/unselected state of the control. Click the **Additional Dialog** button to display the Edit Checked Values dialog (*screenshots below*). Enter a value for the checked (selected) and unchecked (unselected) state. The value corresponding to the control state (selected/unselected) chosen by the end-user will be entered as data in the XML node associated with the control.



Attribute 'HasAddresses':

Checked values
true
1

Unchecked value:
false



Element 'gesmes:name':

Checked values
Child

Unchecked value:
Adult

- **`Toggle On Text`**

  Sets the text that appears in the control when the control is toggled on. Double-click inside the value field to edit.

- **`Toggle Off Text`**

  Sets the text that appears in the control when the control is toggled off. Double-click inside the value field to edit.

- **`Hide Keyboard On Click [Android only]`**

  Hides the keyboard when the control is clicked. Default is `false`.

- **`Control Action`**

Click the **Additional Dialog** button to pop up the control's Actions dialog.

`Text Color`

Sets the color of the control's text. You can do one of the following to select the color:

- Click the color palette to select a text color
- Select a color from the dropdown list of the combo box
- Double-click in the value field to enter a color code (for example, `#FF0000`), or click the **XPath** toolbar button and enter an XPath expression to generate the required text

`Text Size`

Select a size from the dropdown list of the combo box, or double-click in the value field to enter a text size.

`Bold Text`

Select `true` or `false` from the dropdown list of the combo box. Default is `false`.

`Italic Text`

Select `true` or `false` from the dropdown list of the combo box. Default is `false`.

`Underline Text`

Select `true` or `false` from the dropdown list of the combo box. Default is `false`.

`Background Color`

Sets the background color of the control. You can do one of the following to select the color:

- Click the color palette to select a background color
- Select a color from the dropdown list of the combo box
- Double-click in the value field to enter a color code (for example, `#FF0000`), or click the **XPath** toolbar button and enter an XPath expression to generate the required text

`Control Width`

Sets the width of the control. Select `fill_parent` or `wrap_content` from the dropdown list of the combo box. Default is `fill-parent`.

`Visible`

An XPath expression that should evaluate to `true()` or `false()`. If the expression evaluates to `false()`, and only if it evaluates to `false()`, is the control not visible. If the expression evaluates to `true()` or returns some other value, then the control is visible. The default is `true()`. Double-click inside the value field, or click the **XPath** button, to enter or edit an XPath expression.

`Enabled/Editable`

The control is either enabled (and editable) or it is disabled. The value of the property must be an XPath expression that evaluates to boolean `true` or `false`. If the expression evaluates to `true`, the control is enabled/editable. If the expression evaluates to `false`, the control is disabled. The default value is `true`. Typically the XPath expression of the `Enabled/Editable` property of a control would check the presence or value of a node. If the expression evaluates to `true`, the control will be enabled/editable.

`Margin`

Sets the control's margin offsets relative to the surrounding objects or to the borders of the containing object. Select a value in pixels from the dropdown list of the combo box, or double-click in the value field to enter a pixel length.

`Margin Left`

Sets the control's left margin offset relative to the object to the left or to the border of the containing object. Select a value in pixels from the dropdown list of the combo box, or double-click in the value field to enter a pixel length.

`Margin Right`

Sets the control's right margin offset relative to the object to the right or to the border of the containing object. Select a value in pixels from the dropdown list of the combo box, or double-click in the value field to enter a pixel length.

`Margin Top`

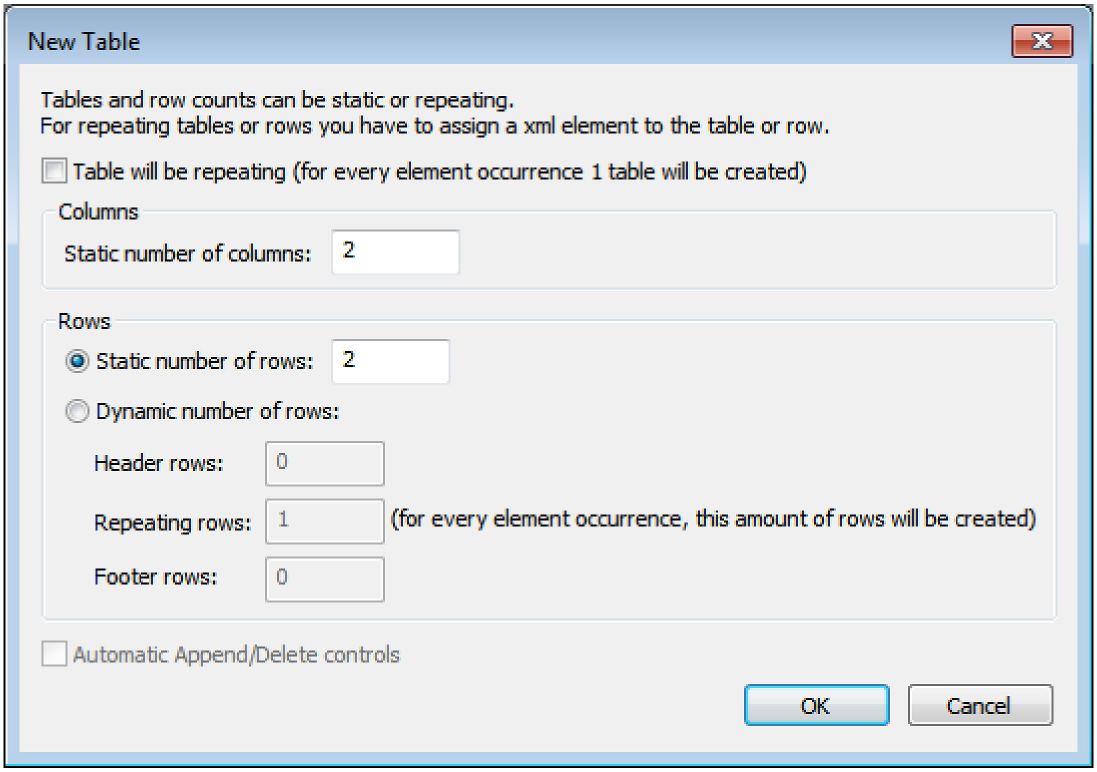
Sets the control's top margin offset relative to the object above it or to the border of the containing object. Select a value in pixels from the dropdown list of the combo box, or double-click in the value field to enter a pixel length.

`Margin Bottom`

Sets the control's bottom margin offset relative to the object below it or to the border of the containing object. Select a value in pixels from the dropdown list of the combo box, or double-click in the value field to enter a pixel length.

Table

The Table control inserts a static table or one or more dynamic tables in the design. On dropping the control in the design, the New Table dialog (*screenshots below*) appears. If you specify a static number of columns and rows (*screenshot below left*), a static table is created. A dynamic table or multiple tables (*screenshot below right*) has its structure determined by the element (of the data source) that is selected to repeat.



New Table

Tables and row counts can be static or repeating.
For repeating tables or rows you have to assign a xml element to the table or row.

□ Table will be repeating (for every element occurrence 1 table will be created)

Columns

Static number of columns: 2

Rows

● Static number of rows: 2

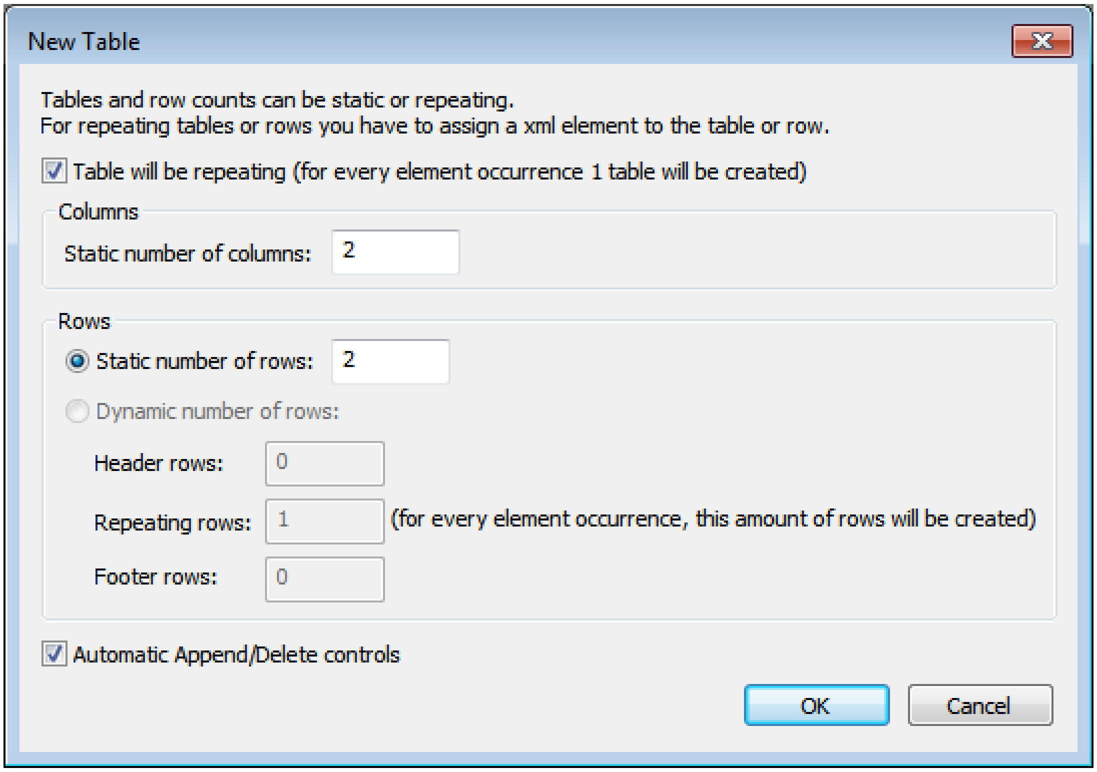
○ Dynamic number of rows:

Header rows: 0

Repeating rows: 1 (for every element occurrence, this amount of rows will be created)

Footer rows: 0

□ Automatic Append/Delete controls


New Table

Tables and row counts can be static or repeating.
For repeating tables or rows you have to assign a xml element to the table or row.

☑ Table will be repeating (for every element occurrence 1 table will be created)

Columns

Static number of columns: 2

Rows

● Static number of rows: 2

○ Dynamic number of rows:

Header rows: 0

Repeating rows: 1 (for every element occurrence, this amount of rows will be created)

Footer rows: 0

☑ Automatic Append/Delete controls

The structure of a dynamic table is built up in one of the following ways:

- Each instance of the element that repeats is created as a new table. The number of columns and of rows are specified as static numbers (*see screenshot above*).
- Each instance of the element that repeats is created as a user-specified number of rows. The number of columns is fixed (static).

Dynamic tables can also have Append/Delete controls automatically added. If added, each instance of a repeating element will have a Delete control next to it, enabling the instance to be deleted. Depending on whether each instance of the the repeating element is created as a table or row, the Add control enables a new table or row (and hence an instance of the corresponding element) to be added.

The cells of the table (both static and dynamic) can contain the following:

- A nested table (static or dynamic)
- A page control
- A node from a data source

Table formatting properties are available in the Styles & Properties Pane and in the context menu of the table.

Table properties
The control's properties are available in the Styles & Properties Pane, and are listed below in

alphabetical order.

Table

▼ **`Visible`**

An XPath expression that should evaluate to `true()` or `false()`. If the expression evaluates to `false()`, and only if it evaluates to `false()`, is the control not visible. If the expression evaluates to `true()` or returns some other value, then the control is visible. The default is `true()`. Double-click inside the value field, or click the **XPath** button, to enter or edit an XPath expression.

▼ **`Background Color`**

Sets the background color of the control. You can do one of the following to select the color:

- Click the color palette to select a background color
- Select a color from the dropdown list of the combo box
- Double-click in the value field to enter a color code (for example, `#FF0000`), or click the **XPath** toolbar button and enter an XPath expression to generate the required text

▼ **`Repeating`**

Click the color palette to select a background color, or select a color from the dropdown list of the combo box, or double-click in the value field to enter a color code.

▼ **`Margin`**

Sets the control's margin offsets relative to the surrounding objects or to the borders of the containing object. Select a value in pixels from the dropdown list of the combo box, or double-click in the value field to enter a pixel length.

▼ **`Margin Left`**

Sets the control's left margin offset relative to the object to the left or to the border of the containing object. Select a value in pixels from the dropdown list of the combo box, or double-click in the value field to enter a pixel length.

▼ **`Margin Right`**

Sets the control's right margin offset relative to the object to the right or to the border of the containing object. Select a value in pixels from the dropdown list of the combo box, or double-click in the value field to enter a pixel length.

▼ **`Margin Top`**

Sets the control's top margin offset relative to the object above it or to the border of the containing object. Select a value in pixels from the dropdown list of the combo box, or double-click in the value field to enter a pixel length.

▼ **`Margin Bottom`**

Sets the control's bottom margin offset relative to the object below it or to the border of the containing object. Select a value in pixels from the dropdown list of the combo box, or double-click in the value field to enter a pixel length.

Table Row

▼ **`Visible`**

An XPath expression that should evaluate to `true()` or `false()`. If the expression evaluates to `false()`, and only if it evaluates to `false()`, is the control not visible. If the expression evaluates to `true()` or returns some other value, then the control is visible. The default is `true()`. Double-click inside the value field, or click the **XPath** button, to enter or edit an XPath expression.

Table Column

▼ **`Width`**

Sets the width of the object. Select `fill_parent` or `wrap_content` or a percentage value from the dropdown list of the combo box. Percentage values are percentages of the parent object.

Table Cell

▼ **`Background Color`**

Sets the background color of the control. You can do one of the following to select the color:

- Click the color palette to select a background color
- Select a color from the dropdown list of the combo box
- Double-click in the value field to enter a color code (for example, `#FF0000`), or click the **XPath** toolbar button and enter an XPath expression to generate the required text

Time

Time controls are used to format times obtained from a node in a data source. This is useful if you wish to format times in a special way. The formatting is defined in the `Date/Time Format String` property (*see below for details*). Note that the source node content must be in the correct lexical format as defined by the XSD specification: `HH:MM:SS`. Optional timezone and millisecond parts are allowed.

⊟ **Notes**

- When the control is associated with a data source node (page source link), placing the mouse over the control (in Page Design View) displays the associated node in a popup.
- To remove a data-source node association, right-click the control (in Page Design View) and click **Delete Page Source Link**.
- To reset a style or property (in the Styles & Properties Pane), select the property and click **Reset** in the pane's toolbar.
- To edit the XPath expression of a style or property (in the Styles & Properties Pane), select the style or property, and click **Edit XPath** in the pane's toolbar.

Time properties

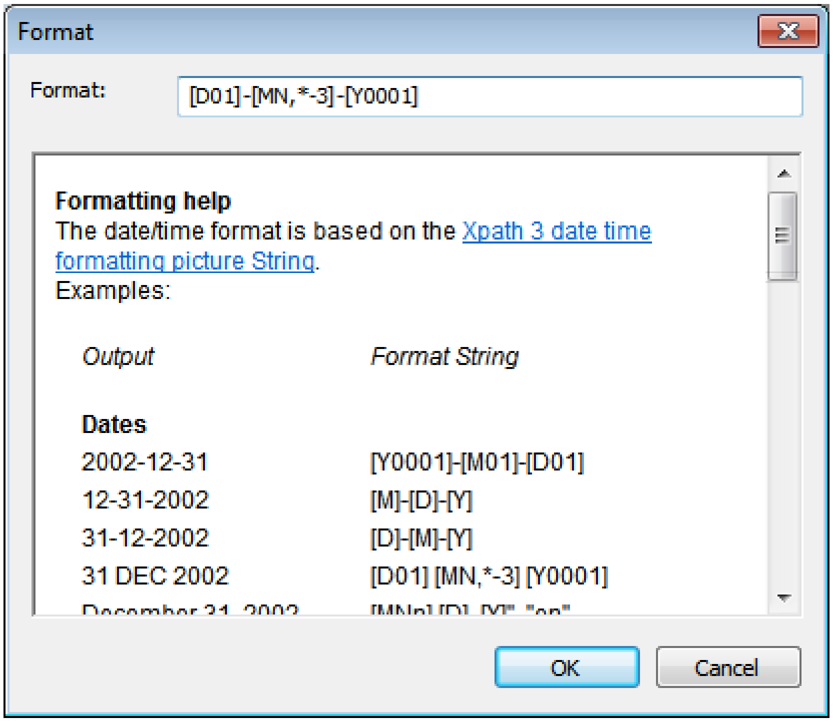
The control's properties are available in the Styles & Properties Pane, and are listed below in alphabetical order.

▼ **`Name`**

The name is used to reference the control from elsewhere in the page or project. Double-click inside the value field to edit.

▼ **`Date/Time Format String`**

Click the **Additional Dialog** button and enter a date, time, or date-time format in the Format dialog that appears (*screenshot below*).



Format: [D01]-[MN,*-3]-[Y0001]

Formatting help
The date/time format is based on the Xpath 3 date time formatting picture String.
Examples:

Output	*Format String*
Dates	
2002-12-31	[Y0001]-[M01]-[D01]
12-31-2002	[M]-[D]-[Y]
31-12-2002	[D]-[M]-[Y]
31 DEC 2002	[D01] [MN,*-3] [Y0001]
December 31, 2002	[MNn] [D], [Y]", "en"

The formatting will be applied to the control's content if the content has the correct lexical form of `xs:date` (for the Date control), `xs:time` (for the Time control), or `xs:dateTime` (for the Date, Time, and DateTime controls). Basic examples are:

- *xs:date:* `2014-12-31`
- *xs:time:* `23:59:59`
- *xs:dateTime:* `2014-12-31T23:59:59`

Assertion

Sets a condition to be met in order for the control's assertion to be valid. If the assertion is invalid, then the text of the `Assertion Message` property (*see below*) is displayed by the Assertion Message control. The assertion's condition is specified with an XPath expression. For example: The XPath expression **`LastName != ""`** asserts that the node `LastName` must not be empty. If this node is empty, then the control's assertion message is displayed in the page at the point where the Assertion Message control is inserted. If there are multiple invalid assertions on a page, then the assertion message of the first invalid assertion (in the sequence in which assertions appear on the page) is displayed.

Assertion Message

Sets the assertion message to be displayed if the control's assertion is not valid. Double-click inside the value field of the property to edit the assertion message, or click the **XPath** toolbar button and enter an XPath expression to generate the required text. The text of the assertion message property is displayed by the Assertion Message control. The assertion's condition is specified with an XPath expression. For example: The XPath expression **`LastName != ""`** asserts that the node `LastName` must not be empty. If this node is empty, then the control's assertion message is displayed in the page at the point where the Assertion Message control is inserted. If there are multiple invalid assertions on a page, then the assertion message of the first invalid assertion (in the sequence in which assertions appear on the page) is displayed.

▼ **`Text Color`**

Sets the color of the control's text. You can do one of the following to select the color:

- Click the color palette to select a text color
- Select a color from the dropdown list of the combo box
- Double-click in the value field to enter a color code (for example, `#FF0000`), or click the **XPath** toolbar button and enter an XPath expression to generate the required text

▼ **`Text Size`**

Select a size from the dropdown list of the combo box, or double-click in the value field to enter a text size.

▼ **`Bold Text`**

Select `true` or `false` from the dropdown list of the combo box. Default is `false`.

▼ **`Italic Text`**

Select `true` or `false` from the dropdown list of the combo box. Default is `false`.

▼ **`Underline Text`**

Select `true` or `false` from the dropdown list of the combo box. Default is `false`.

▼ **`Background Color`**

Sets the background color of the control. You can do one of the following to select the color:

- Click the color palette to select a background color
- Select a color from the dropdown list of the combo box
- Double-click in the value field to enter a color code (for example, `#FF0000`), or click the **XPath** toolbar button and enter an XPath expression to generate the required text

▼ **`Hint`**

Provides a textual hint to the end-user. For example, the hint could be about an action that the end-user has to carry out by using the control. Double-click inside the value field of the property to edit the textual hint, or click the **XPath** toolbar button and enter an XPath expression to generate the required text..

▼ **`Control Action`**

Click the **Additional Dialog** button to pop up the control's Actions dialog.

▼ **`Control Width`**

Sets the width of the control. Select `fill_parent` or `wrap_content` from the dropdown list of the combo box. Default is `fill-parent`.

▼ **`Visible`**

An XPath expression that should evaluate to `true()` or `false()`. If the expression evaluates to `false()`, and only if it evaluates to `false()`, is the control not visible. If the expression evaluates to `true()` or returns some other value, then the control is visible. The default is `true()`. Double-click inside the value field, or click the **XPath** button, to enter or edit an XPath expression.

▼ **`Enabled/Editable`**

The control is either enabled (and editable) or it is disabled. The value of the property must be an XPath expression that evaluates to boolean `true` or `false`. If the expression evaluates to `true`, the control is enabled/editable. If the expression evaluates to `false`, the control is disabled. The default value is `true`. Typically the XPath expression of the `Enabled/Editable` property of a control would check the presence or value of a node. If the expression evaluates to `true`, the control will be enabled/editable.

▼ **`Margin`**

Sets the control's margin offsets relative to the surrounding objects or to the borders of the containing object. Select a value in pixels from the dropdown list of the combo box, or double-click in the value field to enter a pixel length.

▼ **`Margin Left`**

Sets the control's left margin offset relative to the object to the left or to the border of the containing object. Select a value in pixels from the dropdown list of the combo box, or double-click in the value field to enter a pixel length.

▼ **`Margin Right`**

Sets the control's right margin offset relative to the object to the right or to the border of the containing object. Select a value in pixels from the dropdown list of the combo box, or double-click in the value field to enter a pixel length.

▼ **`Margin Top`**

Sets the control's top margin offset relative to the object above it or to the border of the containing object. Select a value in pixels from the dropdown list of the combo box, or double-click in the value field to enter a pixel length.

▼ **`Margin Bottom`**

Sets the control's bottom margin offset relative to the object below it or to the border of the containing object. Select a value in pixels from the dropdown list of the combo box, or double-click in the value field to enter a pixel length.

7.4 Events

Events can occur at the page level and the control level. For each event, a range of actions is available. For example, when a page is reloaded, a message can be displayed, or when a button control is pressed, an XML tree node can be updated. This section describes how to work with events and actions, and is organized into the following sub-sections:

- Page Events: describes what page events are and how actions for them can be defined
- Control Events: describes what control events are and how actions for them can be defined

Page Events

Each page in a MobileTogether design has certain predefined **events** associated with it:

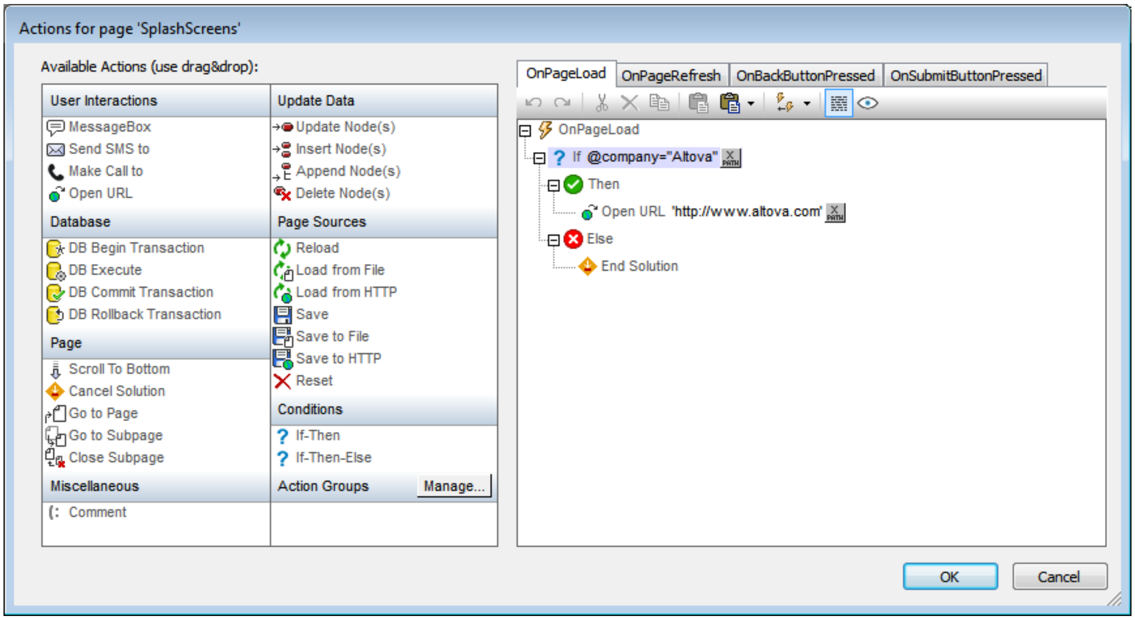
- `OnPageLoad`
- `OnPageRefresh`
- `OnBackButtonPressed`
- `OnSubmitButtonPressed`

These events are called **page events** (as opposed to control events), and each can have an **action (a page action)** defined for it. When a page event is triggered, the page action that is defined for it is carried out. The `OnPageLoad` and `OnPageRefresh` events are available for all pages. Since the **Back** button is only enabled on solution pages that are not the first of a workflow sequence, the `OnBackButtonPressed` event is available on pages that are not the first page of a workflow. The **Submit** button appears when a page provides editing functionality, so such pages can have an `OnSubmitButtonPressed` event.

Defining the action to perform when a page event is triggered

The actions to be carried out when a page event is triggered are defined in the Actions dialog (*screenshot below*). This dialog is accessed in one of the following ways:

- Click **Page | Page Actions**
- Right-click anywhere in the page, and select **Page Actions**
- In the Styles & Properties Pane, go to the *Page* section. and click the **Additional Dialog** button of the `Page Actions` property
- Click **Page | Actions Overview** to display the Actions Overview dialog. Then click the **Additional Dialog** button of the page event you want to define



To define what page action is carried out when a page event is triggered, drag the desired page action from the left-hand pane into the event tab in the right-hand pane. For more information about the Page Actions dialog and on the different types of available page actions, see the section Actions.

Control Events

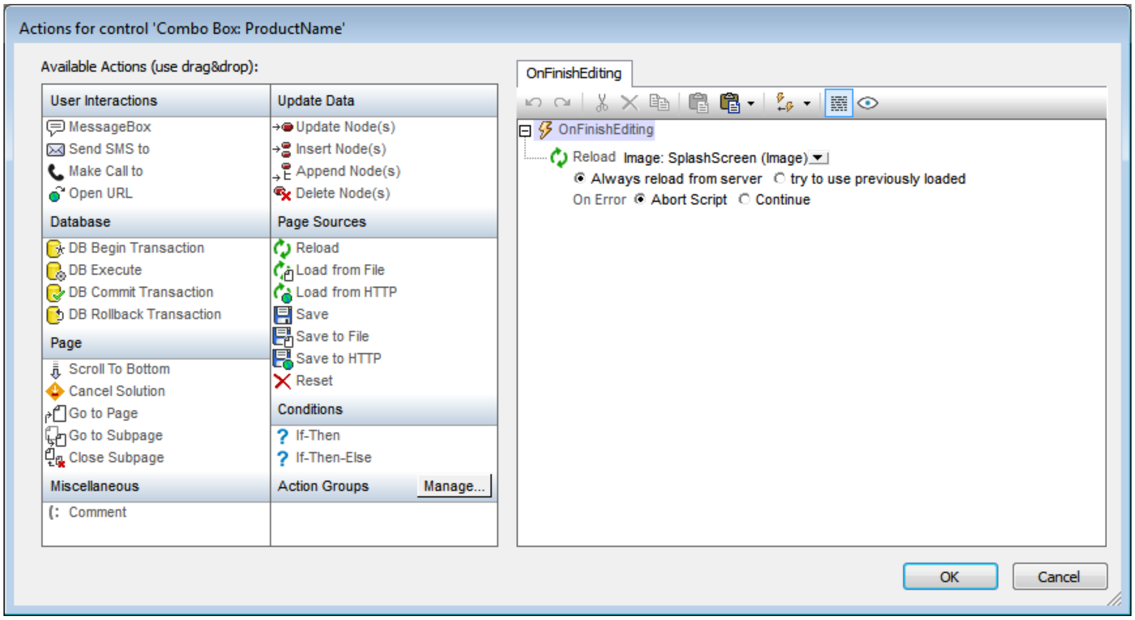
Most of the controls used in page designs have predefined **events** associated with them (*see table below*). For example a button control has an `OnButtonPressed` event associated with it. These events are called **control events**, and each can have an **action (a control action)** defined for it. When a control event is triggered while a MobileTogether solution is being executed, the control action that has been defined for the event is carried out.

The table below lists controls and their events.

Control	Event
Assertion Message	– none –
Button	`OnButtonPressed`
Chart	`OnImageClicked`
Check box	`OnFinishEditing`
Combo box	`OnFinishEditing`
Date	`OnFinishEditing`
DateTime (iOS)	`OnFinishEditing`
Edit field	`OnTyping`
Horizontal line	– none –
Image	`OnImageClicked`
Label	`OnLabelClicked`
Space	– none –
Switch	`OnFinishEditing`
Table	– none –
Time	`OnFinishEditing`

Defining the action to perform when a control event is triggered
The actions to be carried out when a control event is triggered are defined in the Actions dialog (*see screenshot below*). This dialog is accessed in one of the following ways:

- Right-click the control, and select **Control Actions for ...**
- In the Styles & Properties Pane, click the **Additional Dialog** button of the `Control Action` property
- Click **Page | Actions Overview** to display the Actions Overview dialog. Then click the **Additional Dialog** button of the control event you want to define.



OnFinishEditing

Reload Image: SplashScreen (Image)

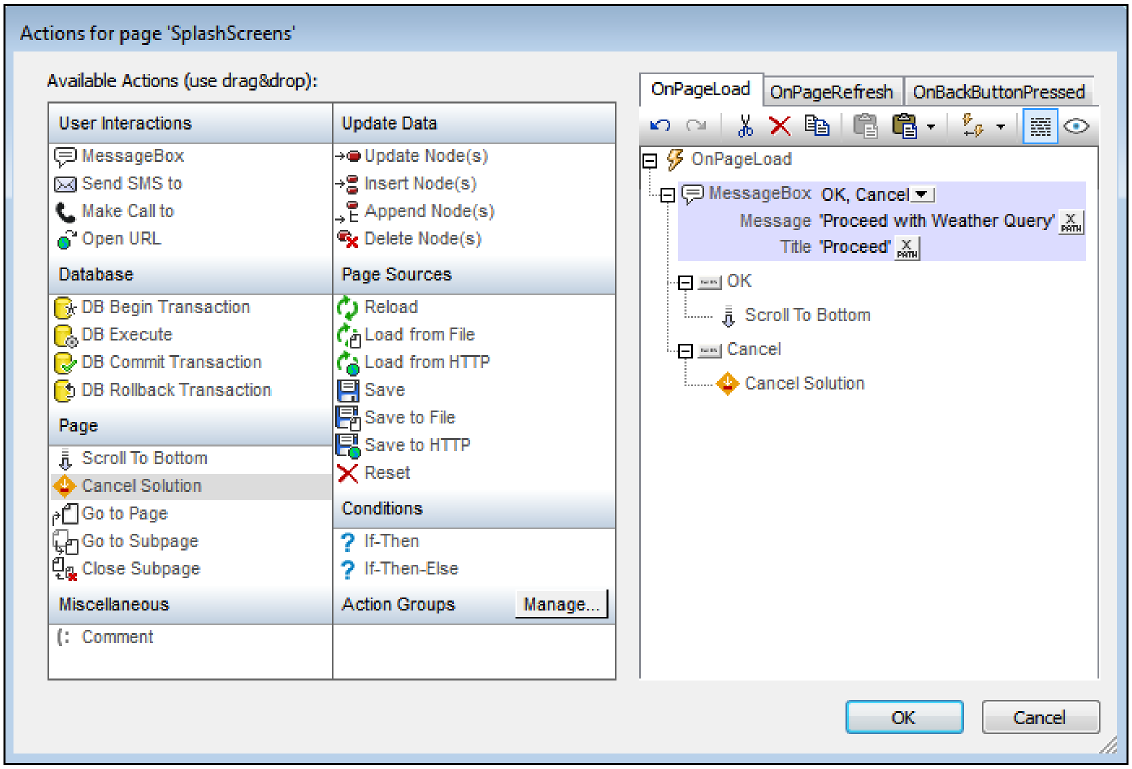
● Always reload from server ○ try to use previously loaded

On Error ● Abort Script ○ Continue

To define what control action is carried out when a control event is triggered, drag the desired control action from the left-hand pane into the event tab in the right-hand pane. For more information about the Actions dialog and on the different types of available control actions, see the section Actions.

7.5 Actions

Actions can be defined for control events and page events. Actions for both kinds of events are defined in the Actions dialog (*see screenshot below*). The left-hand pane contains the available actions grouped according to functionality. The right-hand pane contains the page or control events. Each event is shown in a single tab, and the event's actions are defined in its tab.



OnPageLoad
MessageBox OK, Cancel
Message 'Proceed with Weather Query'
Title 'Proceed'
OK
Scroll To Bottom
Cancel
Cancel Solution

To define a certain action for the selected event, drag the action from the left-hand pane into the event tab. Alternatively, you can add an action via the **Add Action** icon of the Event Pane toolbar. If an action requires further definitions—such as selecting an entry from a combo box, or entering an XPath expression (*see screenshot above*)—then complete these steps.

Add Action icon

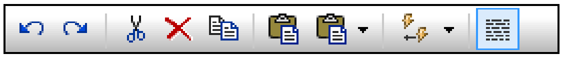
You can also define multiple actions and add child actions. In the screenshot above, for example, the `OK` option of the Message Box action triggers the Scroll To Bottom action, while the `Cancel` option triggers the End Solution action. Click **OK** when you have finished defining the control or page action/s. If there are multiple actions on the same level, then they are performed in the order in which they are defined.

Disabling an action

You can temporarily disable any individual action that is defined for an event. Disabled actions are simply passed over and processing continues as if the actions are not defined. To disable an action, right-click it in the definition of the event's actions and select **Disable Action**. This is a toggle command, so clicking it once more will enable the action again.

Event pane toolbar

The Event Pane toolbar (*screenshot below*) offers the following commands, starting from the left:

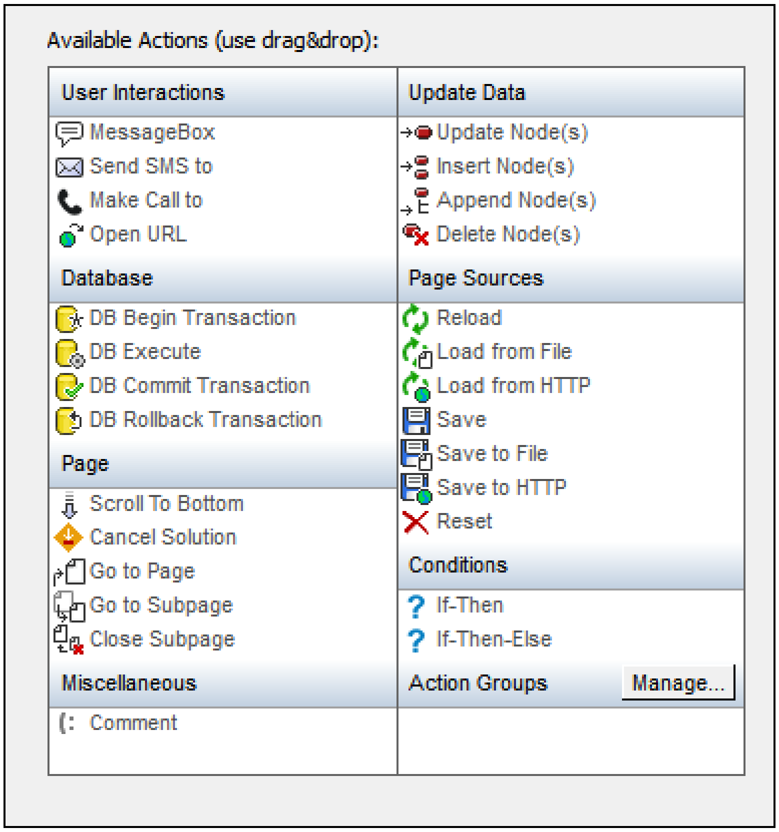


- *Undo, Redo:* Undoes and redoes your Event Pane edits
- *Cut, Delete:* Deletes the selected item in the Event Pane. *Cut* additionally puts the selected item on the clipboard
- *Copy:* Copies the selected item to the clipboard
- *Paste, Append Clipboard Contents:* Pastes the clipboard contents relative to the selected location
- *Add Action:* Drops down a list of available actions that can be placed before, after, or as a child of the selected item in the Events Pane
- *Show Statements Multiline:* Toggles on/off multiline display of statements, such as XPath expressions

User Interactions

The following actions are available in the User Interactions group of the Actions dialog (*screenshot below*):

- Message Box
- Send SMS To
- Make Call To
- Open URL



Available Actions (use drag&drop):

User Interactions
MessageBox
Send SMS to
Make Call to
Open URL

Database
DB Begin Transaction
DB Execute
DB Commit Transaction
DB Rollback Transaction

Page
Scroll To Bottom
Cancel Solution
Go to Page
Go to Subpage
Close Subpage

Miscellaneous
Comment

Update Data
Update Node(s)
Insert Node(s)
Append Node(s)
Delete Node(s)

Page Sources
Reload
Load from File
Load from HTTP
Save
Save to File
Save to HTTP
Reset

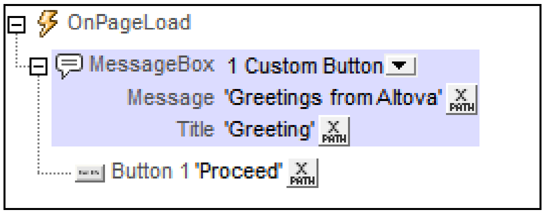
Conditions
If-Then
If-Then-Else

Action Groups

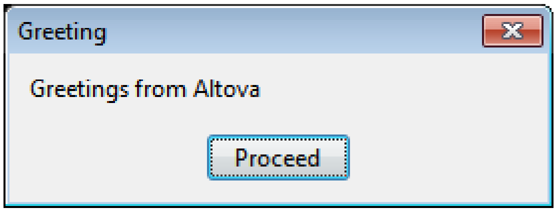
The actions in this group are available for page events and control events. The fastest way to access the Actions dialog (*screenshot above*) is to right-click the page or control and select the page/control actions command. *See also Page Events and Control Events.*

Message Box

Defines a message box that is displayed when the event is triggered. The combo box enables you to select the buttons that appear in the message box. Predefined buttons are `OK`, `Yes`, `No`, `Cancel`. You can also define from one to three custom buttons. The text of custom buttons is specified in an XPath expression (for example, in the screenshot below left it is: **`"Proceed"`**).



OnPageLoad
MessageBox 1 Custom Button
Message 'Greetings from Altova'
Title 'Greeting'
Button 1 'Proceed'



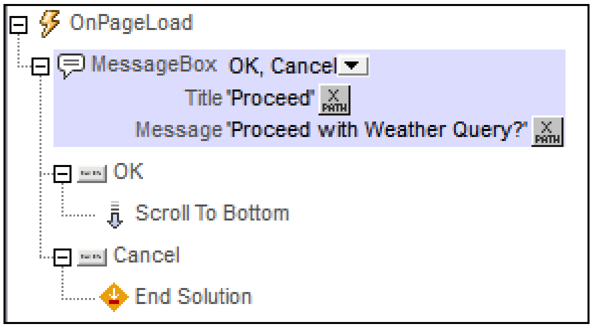
Greeting

Greetings from Altova

The screenshots above show a message box (*right*) and its definition (*left*). The definition is for a message box with a custom button. Notice how the title, message, and button text are defined.

Message boxes with multiple user options

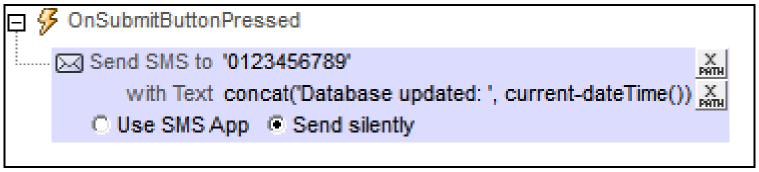
A message box can offer the user multiple options. This is done in the form of multiple buttons, each of which is linked to its own action. For example, in the screenshot below, two buttons, `OK` and `Cancel`, are defined by selecting them in the combo box. The buttons are automatically displayed in a hierarchical tree as child objects of MessageBox. Each button can then have actions assigned to them (*see screenshot*). These actions are carried out when the button is pressed by the user. For example, if the `Cancel` button defined in the screenshot below is clicked, then the End Solution action is carried out.



OnPageLoad
MessageBox OK, Cancel
Title 'Proceed'
Message 'Proceed with Weather Query?'
OK
Scroll To Bottom
Cancel
End Solution

Send SMS To

Sends an SMS to a specified number with a specified text. The receiver's number and the SMS text are specified using XPath expressions (*see screenshot below*). Numbers should not be enclosed in quotes. Numbers of multiple recipients must be separated by a semi-colon. A static global variable named **`$MT_SMSAvailable`** can be used to test whether SMS services are available on the client device. Variable values can be **`true()`** or **`false()`**.

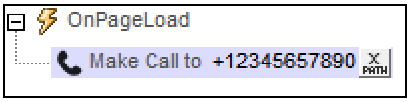


OnSubmitButtonPressed
Send SMS to '0123456789'
with Text concat('Database updated: ', current-dateTime())
○ Use SMS App ● Send silently

You can choose whether to send the message via the SMS application on the mobile device or silently.

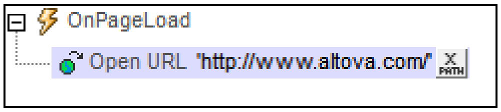
Make Call To

Makes a call to the number specified in the XPath expression of the definition (*see screenshot below*). Numbers should not be enclosed in quotes. A static global variable named **`$MT_TelephonyAvailable`** can be used to test whether telephony services are available on the client device. Variable values can be **`true()`** or **`false()`**.



Open URL

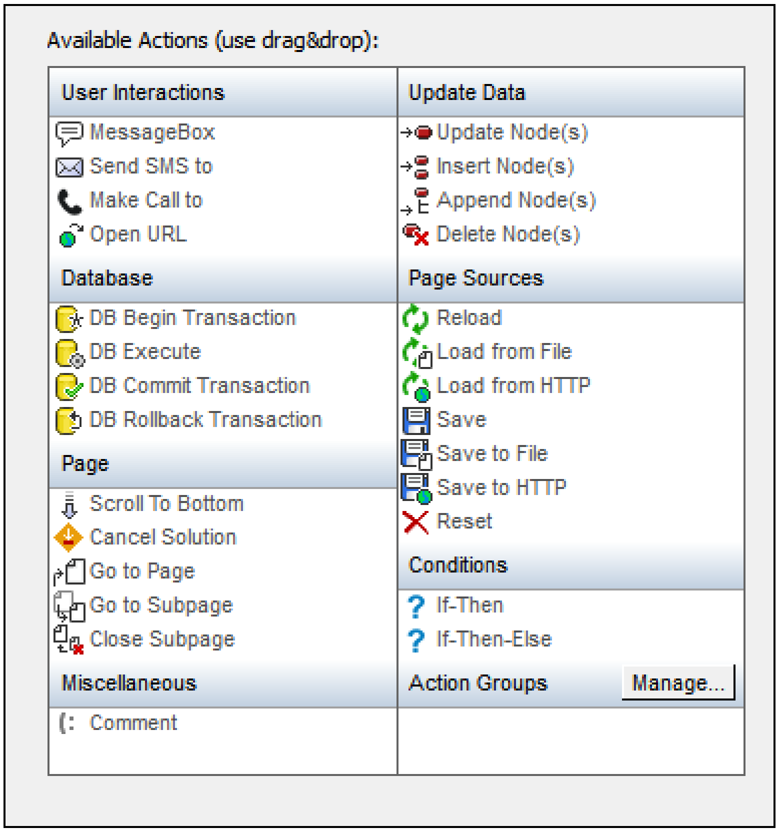
Opens the URL specified in the XPath expression of the Open URL action's definition (*see screenshot below*).



Database

The following actions are available in the Database group of the Actions dialog (*screenshot below*):

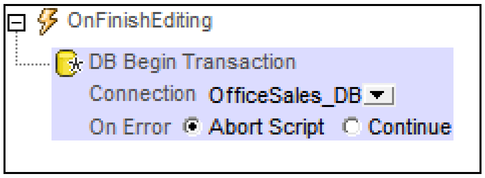
- DB Begin Transaction
- DB Execute
- DB Commit Transaction
- DB Rollback Transaction



The actions in this group are available for page events and control events. The fastest way to access the Actions dialog (*screenshot above*) is to right-click the page or control and select the page/control actions command. *See also Page Events and Control Events*.

DB Begin Transaction

When the event is triggered the DB Begin Transaction action begins a transaction with the data source selected in the *Connection* combo box. This combo box lists all the data sources of the project, and also offers the option of setting up an additional database connection specifically for use with the DB Begin Transaction action.



Connection OfficeSales_DB

On Error ● Abort Script ○ Continue

About DB Transactions

For each DB access that needs a transaction, one is automatically created and closed afterwards. This might not be desirable for some setups. For example, when you have two DB page sources that you want to update atomically together: If one table is saved successfully, then the other one is committed. To accommodate this kind of situation, transactions can be created on a connection basis.

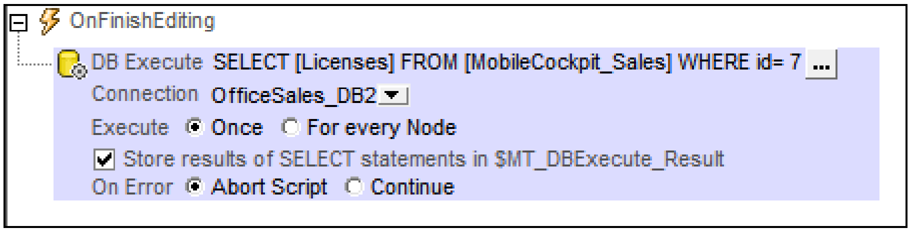
If you begin a transaction, all DB operations belonging to the same DB connection will use this transaction.

Committing a transaction makes changes visible to the world outside your transaction. Changes can be rolled back. In this case, even if you have done a Save on your page source, the changes won't be visible after a rollback! Note that any transaction that is not closed (committed or rolled back) when the end of the action tree has been reached will be rolled back automatically! A warning to this effect will be displayed in the Messages window.

It is important to bear in mind that, while the behavior above refers to explicit transaction actions, this behavior also applies to all DB operations that use the same connection as the transaction.

DB Execute

When the event is triggered, the DB Execute action executes the action's SQL statement on the data source selected in the *Connection* combo box. This combo box lists all the data sources of the project, and also offers the option of setting up an additional database connection specifically for use with the DB Execute action. If the *Store results in `$MT_DBExecute_Result`* check box is selected, then the results are stored in the `$MT_DBExecute_Result` variable. This variable can then be used in XPath expressions elsewhere on the page to provide the result of the DB Executew action.



DB Execute SELECT [Licenses] FROM [MobileCockpit_Sales] WHERE id= 7

Connection OfficeSales_DB2

Execute ● Once ○ For every Node

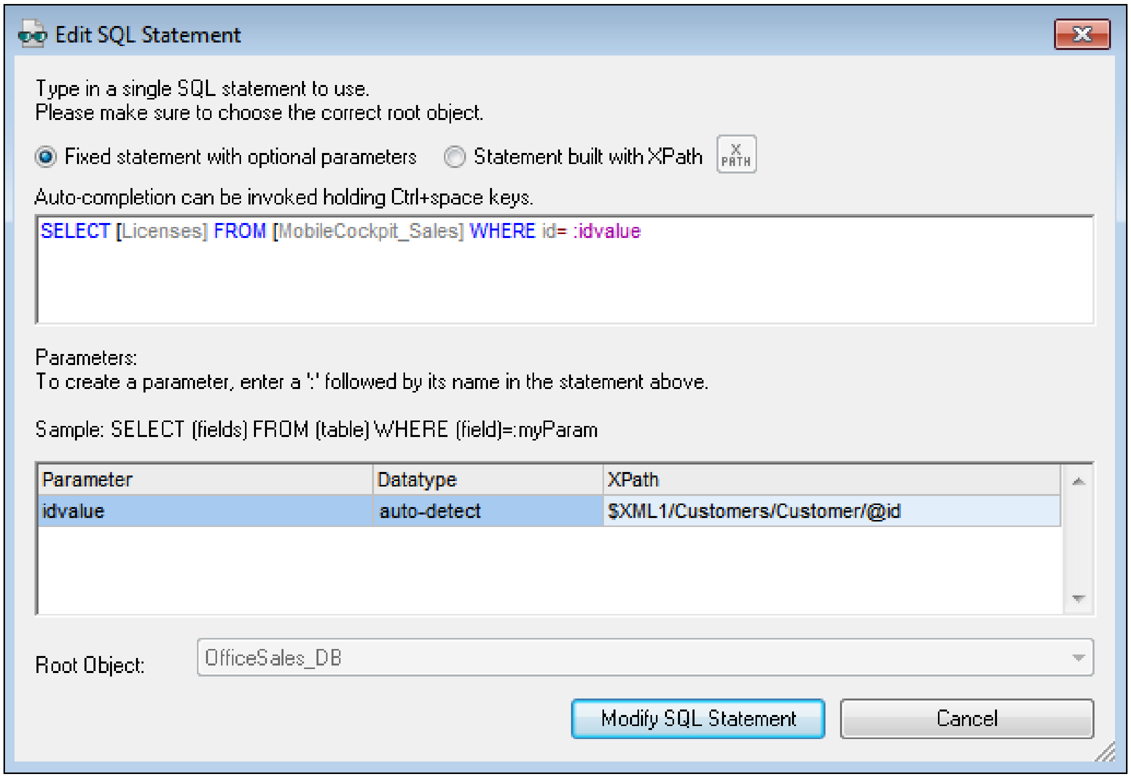
☑ Store results of SELECT statements in $MT_DBExecute_Result

On Error ● Abort Script ○ Continue

Note: Multiple `SELECT` statements can be used in the DB Execute action. In this case, the result can have multiple `RowSet` elements in the returned `DB/RowSet/Row` structure.

SQL statements with parameters

To enter or edit edit the SQL statement, click the **Additional Dialog** button. This displays the Edit SQL Statement dialog (*screenshot below*). The Root Object at the bottom of the dialog is selected automatically and is based on the selection in the Connection combo box. The *Root Object* field cannot be edited. Before proceeding, make sure that this is the root object you want.



Edit SQL Statement

Type in a single SQL statement to use.
Please make sure to choose the correct root object.

● Fixed statement with optional parameters ○ Statement built with XPath

Auto-completion can be invoked holding Ctrl+space keys.

```
SELECT [Licenses] FROM [MobileCockpit_Sales] WHERE id= :idvalue
```

Parameters:
To create a parameter, enter a ':' followed by its name in the statement above.

Sample: SELECT [fields] FROM [table] WHERE [field]=:myParam

Parameter	Datatype	XPath
idvalue	auto-detect	$XML1/Customers/Customer/@id

Root Object: OfficeSales_DB

To enter an SQL statement, select *Fixed statement with optional parameters*, and enter the SQL statement. The use of parameters in the SQL statement provides more flexibility. For example, in the screenshot above, instead of entering a fixed value for the `WHERE` clause, a parameter name `idvalue` is used to supply the value of a node in an XML data source. In the first line below, a fixed value is used; in the second line, the parameter `idvalue` is used.

```
WHERE id= 7
WHERE id= :idvalue
```

To use a parameter, write the parameter name prefixed by a colon (`:`)in the SQL statement where you want to use it. As soon as you enter the first character after the colon, an entry is created for the parameter in the Parameters pane. Next, in the Parameters pane, enter an XPath expression to provide the value of the parameter. You can enter as many parameters as you like.

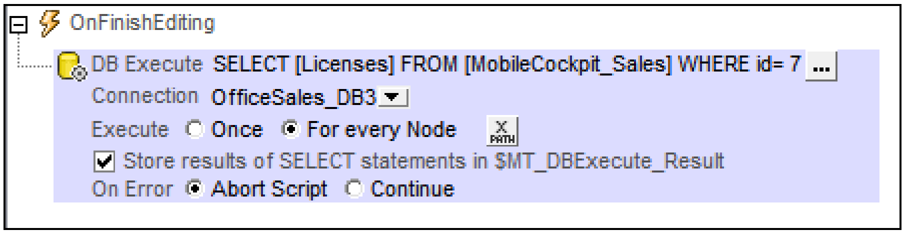
SQL statements built with XPath

You can also build the SQL statement using an XPath expression. This is useful if you wish to use an SQL statement using the specific syntax of a particular kind of DB. The result generated by the XPath expression is used as the SQL statement submitted to the DB.

To build an SQL statement using XPath, select *Statement built with XPath*. In the Edit XPath/ XQuery Expression dialog that appears, enter the XPath expression and click **OK**.

Execute once or for every node

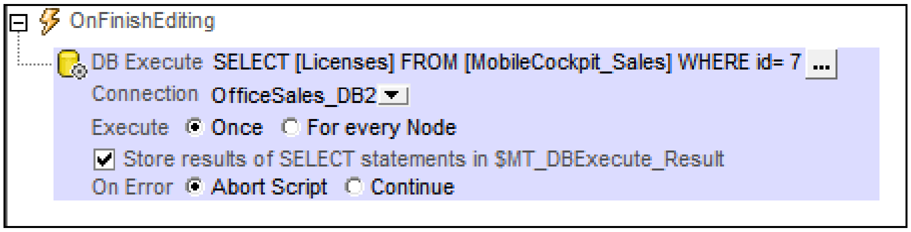
The SQL statement can be executed once on the data source, or it can be executed on all the nodes of a custom defined nodeset. If you select the latter option, you need to enter an XPath expression that generates the nodeset. The SQL statement will then be executed for each node in this nodeset. Additionally, you can query the value of the current node of the nodeset by using the variable `$MT_TargetNode`. This variable can be used, for example, in the definition of a parameter used in the SQL statement (*see "SQL statements with parameters" above*).



⊟ OnFinishEditing
DB Execute SELECT [Licenses] FROM [MobileCockpit_Sales] WHERE id= 7 ...
Connection OfficeSales_DB3
Execute ○ Once ● For every Node
☑ Store results of SELECT statements in $MT_DBExecute_Result
On Error ● Abort Script ○ Continue

The `$MT_DBExecute_Result` variable

The nodeset or other value returned by (the SQL statement of) the DB Execute action is stored in MobileTogether Designer's built-in variable **`$MT_DBExecute_Result`**. This variable stores the result of the last DB Execute action of the project, and can used in XPath expressions in other locations in the project.

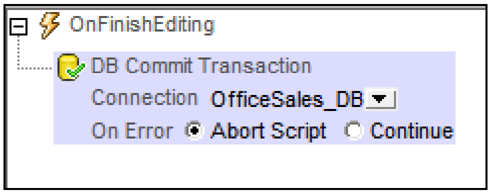


⊟ OnFinishEditing
DB Execute SELECT [Licenses] FROM [MobileCockpit_Sales] WHERE id= 7 ...
Connection OfficeSales_DB2
Execute ● Once ○ For every Node
☑ Store results of SELECT statements in $MT_DBExecute_Result
On Error ● Abort Script ○ Continue

Note that a DB Execute action can return nodesets of the kind `DB/RowSet/Row`, where `Row` elements repeat and there are multiple `RowSet` elements. Multiple `RowSet` elements are generated when multiple `SELECT` statements are used in the DB Execute action.

DB Commit Transaction

When the event is triggered the DB Commit Transaction action commits a transaction to the data source selected in the *Connection* combo box. This combo box lists all the data sources of the project, and also offers the option of setting up an additional database connection specifically for use with the DB Begin Transaction action.


OnFinishEditing
DB Commit Transaction
Connection OfficeSales_DB
On Error ● Abort Script ○ Continue

About DB Transactions
For each DB access that needs a transaction, one is automatically created and closed afterwards. This might not be desirable for some setups. For example, when you have two DB page sources that you want to update atomically together: If one table is saved successfully, then the other one is committed. To accommodate this kind of situation, transactions can be created on a connection basis.

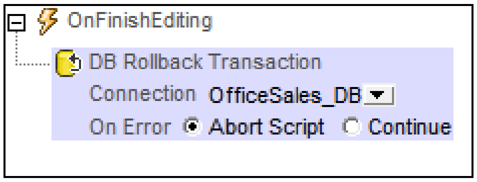
If you begin a transaction, all DB operations belonging to the same DB connection will use this transaction.

Committing a transaction makes changes visible to the world outside your transaction. Changes can be rolled back. In this case, even if you have done a Save on your page source, the changes won't be visible after a rollback! Note that any transaction that is not closed (committed or rolled back) when the end of the action tree has been reached will be rolled back automatically! A warning to this effect will be displayed in the Messages window.

It is important to bear in mind that, while the behavior above refers to explicit transaction actions, this behavior also applies to all DB operations that use the same connection as the transaction.

DB Rollback Transaction

When the event is triggered the DB Rollback Transaction action rolls back a transaction on the data source selected in the *Connection* combo box. This combo box lists all the data sources of the project, and also offers the option of setting up an additional database connection.



Connection OfficeSales_DB

On Error ● Abort Script ○ Continue

About DB Transactions
For each DB access that needs a transaction, one is automatically created and closed afterwards. This might not be desirable for some setups. For example, when you have two DB page sources that you want to update atomically together: If one table is saved successfully, then the other one is committed. To accommodate this kind of situation, transactions can be created on a connection basis.

If you begin a transaction, all DB operations belonging to the same DB connection will use this transaction.

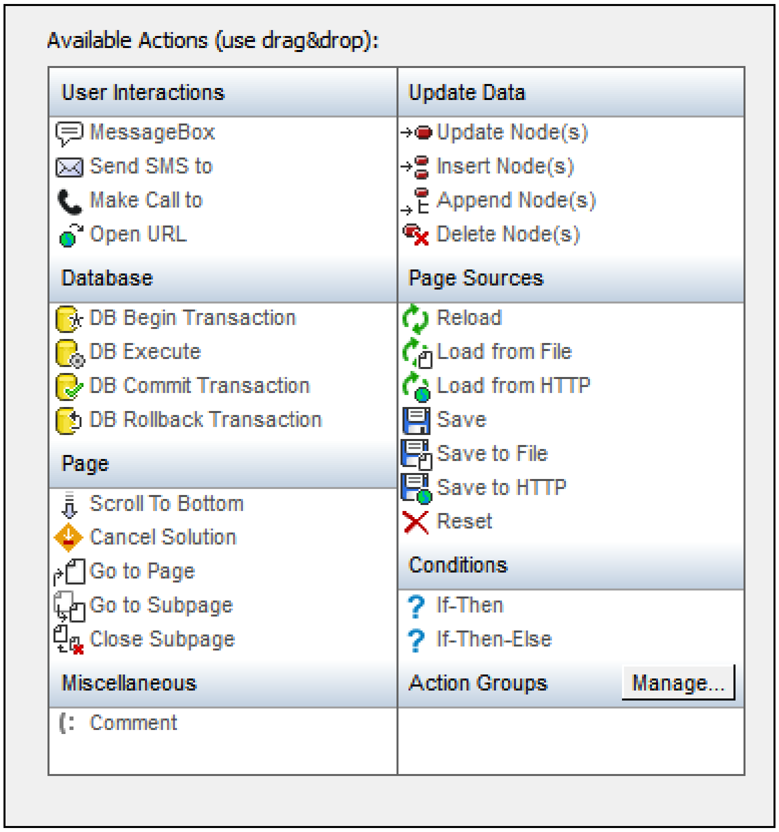
Committing a transaction makes changes visible to the world outside your transaction. Changes can be rolled back. In this case, even if you have done a Save on your page source, the changes won't be visible after a rollback! Note that any transaction that is not closed (committed or rolled back) when the end of the action tree has been reached will be rolled back automatically! A warning to this effect will be displayed in the Messages window.

It is important to bear in mind that, while the behavior above refers to explicit transaction actions, this behavior also applies to all DB operations that use the same connection as the transaction.

Page

The following actions are available in the Page group of the Actions dialog:

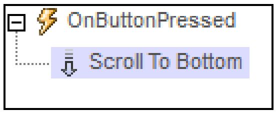
- Scroll to Bottom
- Cancel Solution
- Go to Page
- Go to Subpage
- Close Subpage



The actions in this group are available for page events and control events. The fastest way to access the Actions dialog (*screenshot above*) is to right-click the page or control and select the page/control actions command. *See also Page Events and Control Events.*

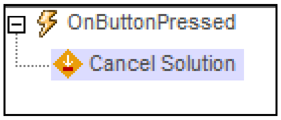
Scroll to Bottom

Scrolls to the bottom of the active page when the event is triggered.



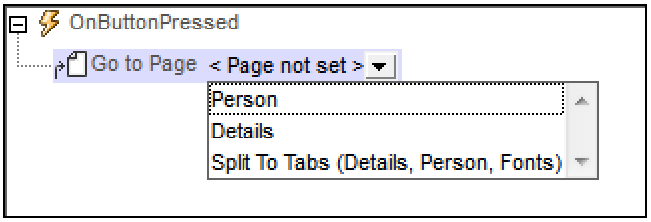
Cancel Solution

When the event is triggered, the Cancel Solution action displays a prompt asking whether the end user really wishes to terminate the solution or not. If the end user clicks **OK**, the solution is terminated.



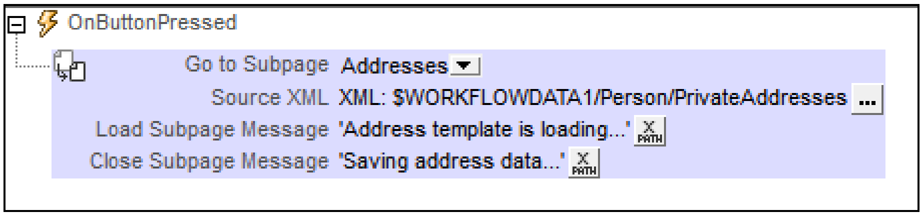
Go to Page

The Go to Page action goes to the specified page when the event is triggered. The page to go to is selected in the combo box of the event (*see screenshot below*). If no other page exists in the project, no page will be available for selection in the combo box.



Go to Subpage

Goes to the specified sub-page when the event is triggered. The subpage to go to is selected from the *Go to Subpage* combo box (*see screenshot below*), which lists all the subpages in the project. In the screenshot below, an `OnButtonPressed` event has been set to go to the *Addresses* subpage.



OnButtonPressed

Go to Subpage: Addresses

Source XML: XML: $WORKFLOWDATA1/Person/PrivateAddresses

Load Subpage Message: 'Address template is loading...'

Close Subpage Message: 'Saving address data...'

The Go to Subpage action has the following settings:

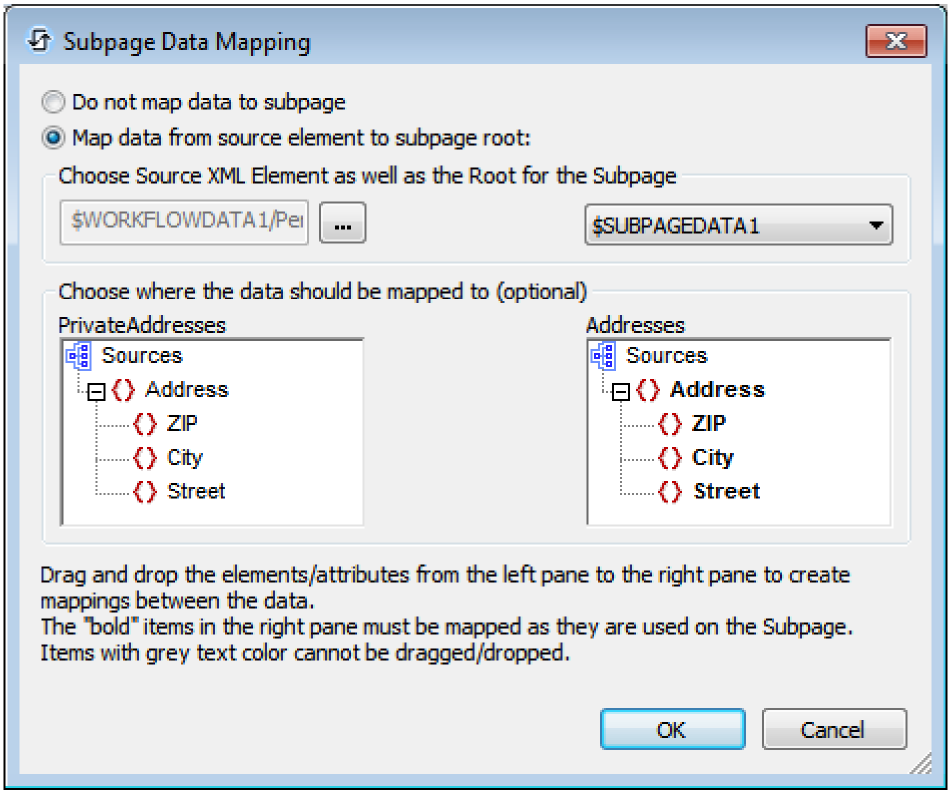
- *Source XML:* Specifies whether and how subpage data should correspond to data sources in the originating page. This definition is entered in the Subpage Data Mapping dialog, which is accessed by clicking the **Additional Dialog** button of the *Source XML* field. *See the section "Source XML of subpages" below for a detailed description of this dialog*.
- *Load Subpage Message:* In the event that the subpage is transmitted via a server request, this message is shown in a progress dialog while the subpage loads. *See screenshot above*.
- *Close Subpage Message:* In the event that the subpage is transmitted via a server request, this message is shown in a progress dialog while the subpage closes. *See screenshot above*.

Source XML of subpages

Every subpage must have at least one data source in order for the subpage to be used effectively. Data in subpage sources can be mapped to data in top page sources, or the data in subpage sources can be left unmapped. If mapped, then subpage data is transferred to the mapped nodes of the top page when the subpage is closed. The behavior of the source XML of subpages is defined in the Subpage Data Mapping dialog (*screenshot below left*). To access this dialog, click the **Additional Dialog** button of the *Source XML* field of the Go to Subpage action definition (*see description above*).

The following subpage data handling possibilities are available:

- Subpage data is not mapped. In this case, subpage data sources are handled independently of the subpage-data-mapping mechanism.
- A subpage source is mapped to a top page structure that is exactly the same; even the names of elements and attributes are the same.
- A subpage source is mapped to a top page source that has a different structure and/or different attribute/element names.



Subpage Data Mapping

○ Do not map data to subpage

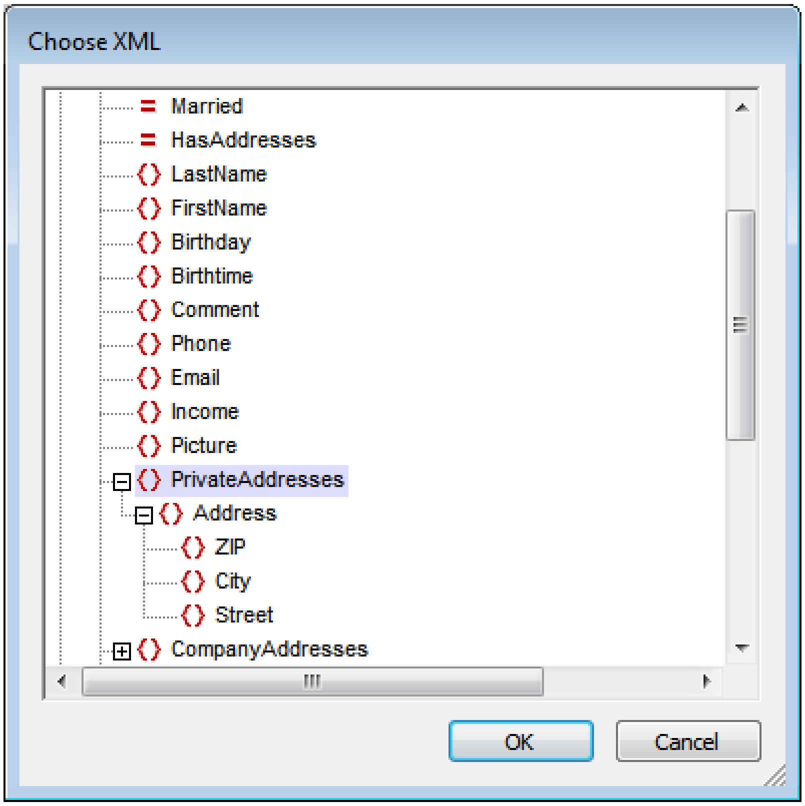
● Map data from source element to subpage root:

Choose Source XML Element as well as the Root for the Subpage

$WORKFLOWDATA1/Pe $SUBPAGEDATA1

Choose where the data should be mapped to (optional)

Drag and drop the elements/attributes from the left pane to the right pane to create mappings between the data.
The "bold" items in the right pane must be mapped as they are used on the Subpage.
Items with grey text color cannot be dragged/dropped.

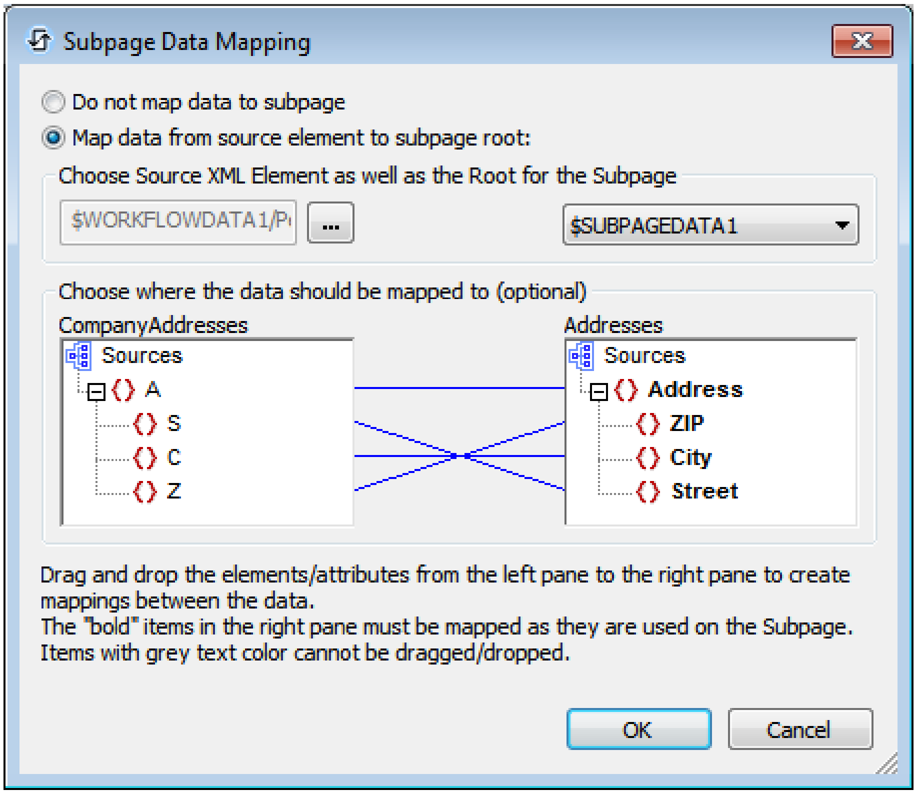


Choose XML

In the Subpage Data Mapping dialog (*screenshot above left*), the subpage data source is displayed in the right-hand source tree pane, while the top page data source is displayed in the left-hand source tree pane .

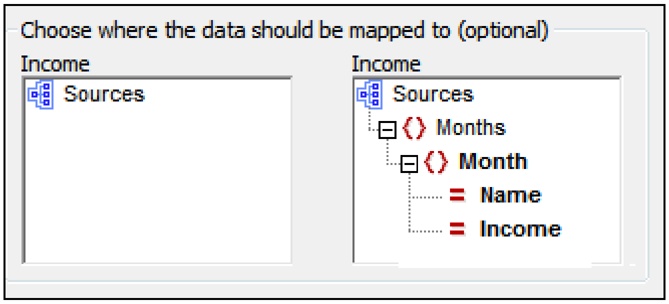
- Select the subpage data source from the combo box above the (right-hand) subpage pane. This combo box lists all the data sources of the subpage. Note that the subpage name is listed above the subpage pane.
- To select the top page data source to be mapped, first click the **Additional Dialog** button above the (left-hand) top page pane. This displays the Choose XML dialog (*screenshot above right*), which contains all the data sources of the top page. Select the node that you want to map to the subpage data. In the screenshot above, the `PrivateAddresses` node has been selected in the Choose XML dialog. As a result, the descendants of `PrivateAddresses` are displayed in the top page source tree pane and *PrivateAddresses*, the mapped node, is listed above the pane.

In the case described by the screenshot above, the mapping is automatically done between the two data sources since both XML data structures are identical, right down to the names of elements. In the screenshot below a mapping between elements is constructed by dragging a node from the left-hand pane onto a node in the right-hand pane. This is because the two data structures are not identical (the first element, `S`, must correspond to the thitd element, `Street`).


Subpage Data Mapping
○ Do not map data to subpage
● Map data from source element to subpage root:
Choose Source XML Element as well as the Root for the Subpage
$WORKFLOWDATA1/P
...
$SUBPAGEDATA1
Choose where the data should be mapped to (optional)
CompanyAddresses
Sources
A
S
C
Z
Addresses
Sources
Address
ZIP
City
Street
OK
Cancel

Drag and drop the elements/attributes from the left pane to the right pane to create mappings between the data.
The "bold" items in the right pane must be mapped as they are used on the Subpage.
Items with grey text color cannot be dragged/dropped.

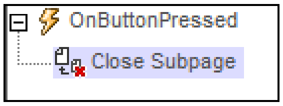
In the case described by the screenshot below, the content of the subpage data node `Income/Months/Month` is mapped to the top page data node `Income` (indicated by the title of the pane).



Choose where the data should be mapped to (optional)

Close Subpage

Closes the active subpage when the event is triggered.

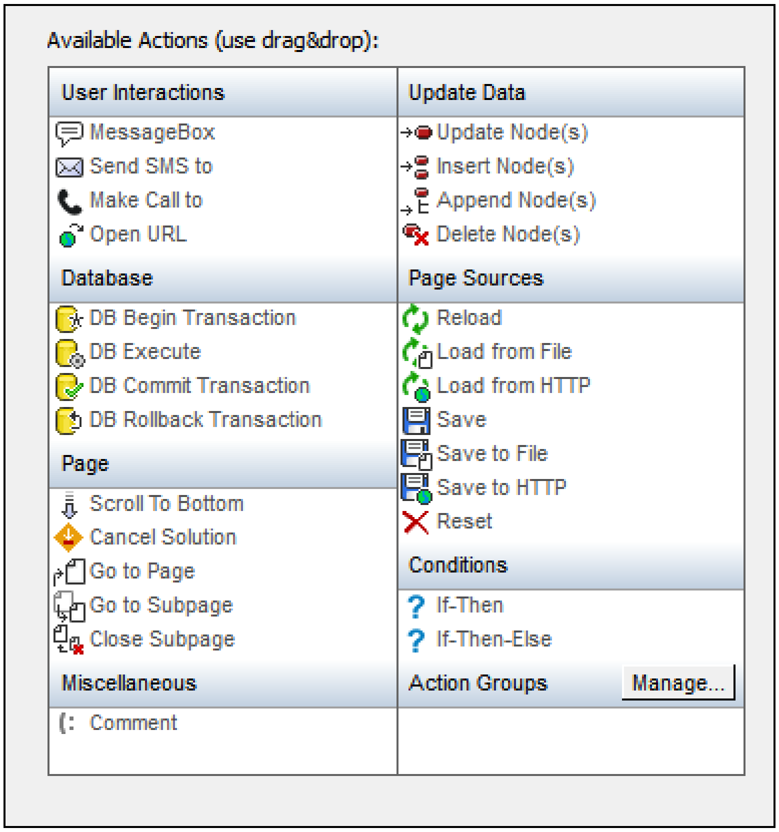

OnButtonPressed

Close Subpage

Miscellaneous

The following action is available in the Miscellaneous group of the Actions dialog (*screenshot below*):

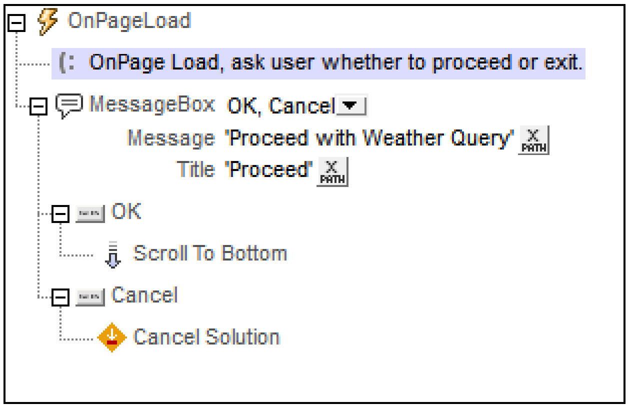
- Comment



The actions in this group are available for page events and control events. The fastest way to access the Actions dialog (*screenshot above*) is to right-click the page or control and select the page/control actions command. *See also Page Events and Control Events.*

Comment

Comments can be added to the definition of an event's actions (*see screenshot below*). This is useful for inserting explanations of the different actions in the definition of the event's actions.

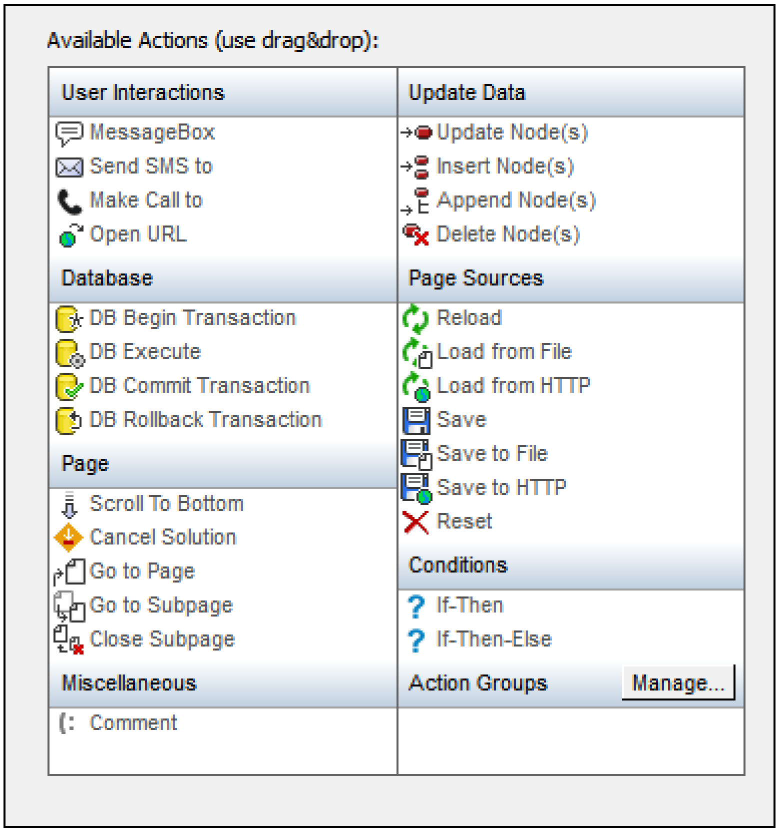


```
OnPageLoad
    (: OnPage Load, ask user whether to proceed or exit.
    MessageBox  OK, Cancel
        Message 'Proceed with Weather Query'
        Title 'Proceed'
        OK
            Scroll To Bottom
        Cancel
            Cancel Solution
```

Update Data

The following actions are available in the Update Data group of the Actions dialog (*screenshot below*):

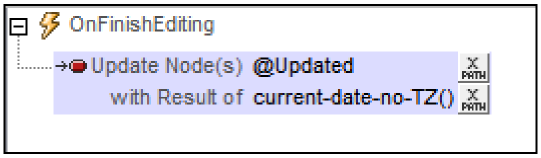
- Update Node(s)
- Insert Node(s)
- Append Node(s)
- Delete Node(s)



The actions in this group are available for page events and control events. The fastest way to access the Actions dialog (*screenshot above*) is to right-click the page or control and select the page/control actions command. *See also Page Events and Control Events.*

Update Node(s)

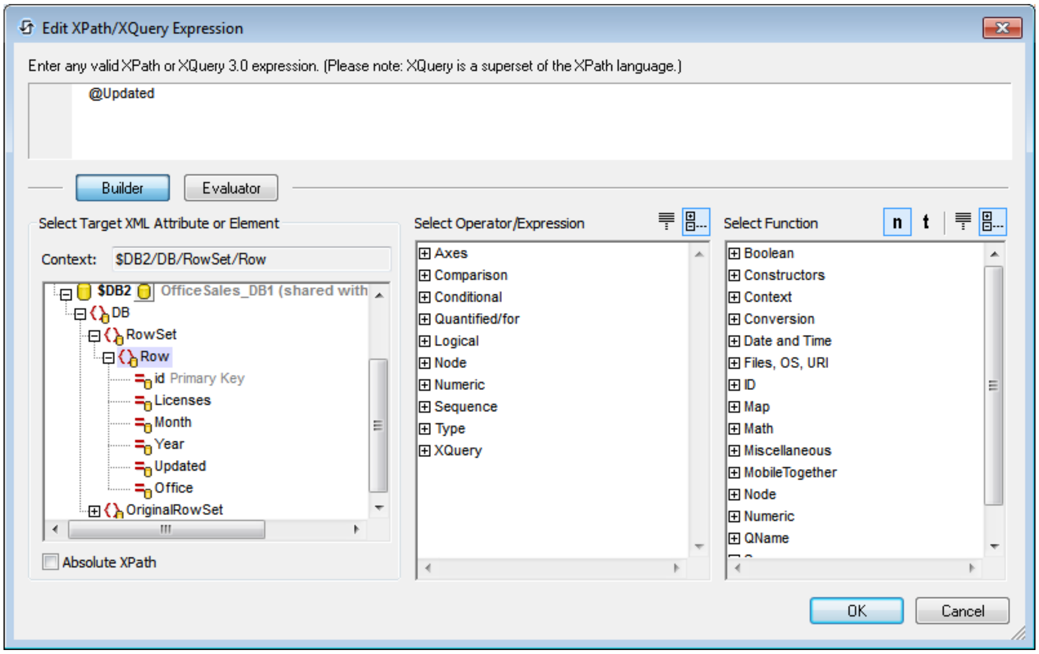
When the event is triggered, the Update Node(s) action updates one or more specified nodes with the specified value. Both the update node and update value are are specified with XPath expressions. In the screenshot below, the `@Updated` attribute node of the XPath context element is updated with the current date (the result of evaluating the XPath function `current-date-no-TZ()`).



OnFinishEditing
Update Node(s) @Updated
with Result of current-date-no-TZ()

There are a few points to note when specifying the update node/s and update value:

- In the screenshot below, notice that the context node (indicated in the *Context* field) is the `Row` element of the DB. The context node is the node within which the control (for which the action is being defined) is located (or with which the control is associated).
- The `@Updated` attribute that will be updated is therefore the `@Updated` attribute of that particular `Row` element. What this means is that when a `Row` element's control event is triggered, then the `@Updated` attribute of **only that** `Row` element is updated, in this case with the current date.



Edit XPath/XQuery Expression

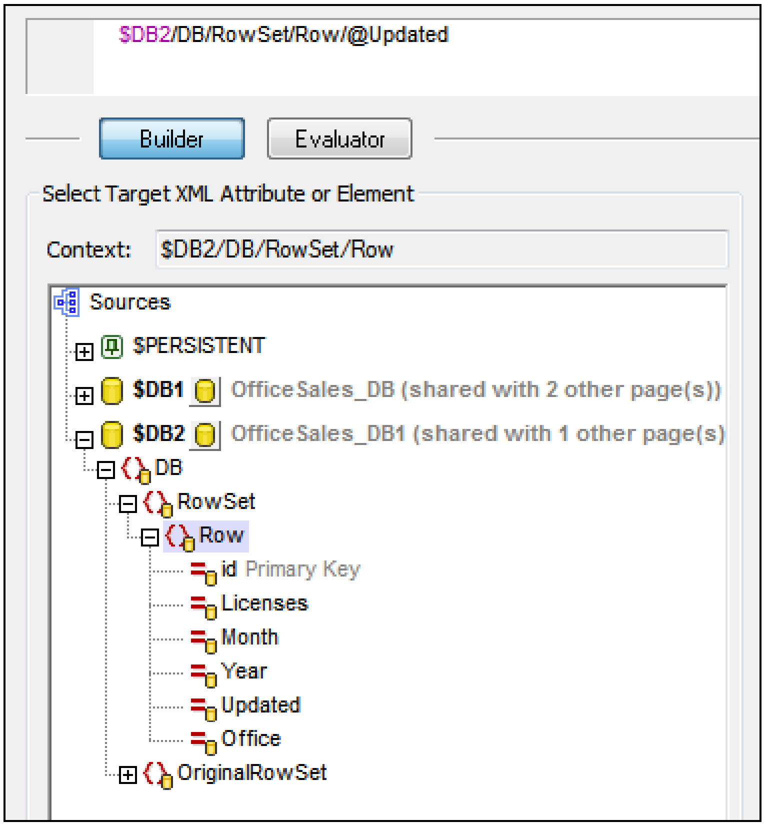
Enter any valid XPath or XQuery 3.0 expression. (Please note: XQuery is a superset of the XPath language.)

@Updated

Select Target XML Attribute or Element

Context: $DB2/DB/RowSet/Row

- If the XPath expression were changed so that it addressed the `@Updated` node starting from the root node (like this, and as in the screenshot below: `$DB2/DB/RowSet/Row/@Updated`), then the `@Updated` node of **all** `Row` elements would be updated.



$DB2/DB/RowSet/Row/@Updated

Select Target XML Attribute or Element

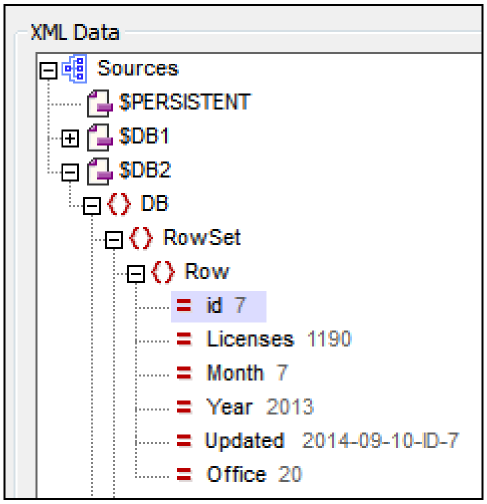
Context: $DB2/DB/RowSet/Row

```
Sources
  $PERSISTENT
  $DB1  OfficeSales_DB (shared with 2 other page(s))
  $DB2  OfficeSales_DB1 (shared with 1 other page(s)
    DB
      RowSet
        Row
          id Primary Key
          Licenses
          Month
          Year
          Updated
          Office
      OriginalRowSet
```

- Notice from the screenshot above, that you have access via XPath expressions to all the nodes of all data sources.
- After a target node for the update is defined, you can reference this node with the variable **$MT_TargetNode**. For example:

```
Update Node : @Updated
Update Value: concat(current-date-no-TZ(), '-ID-', $MT_TargetNode/../@id)
```

would give an update value that would have the `@id` attribute value of the current `Row` element suffixed to the current date, as in the screenshot below.



XML Data

```
Sources
  $PERSISTENT
  $DB1
  $DB2
    DB
      RowSet
        Row
          id 7
          Licenses 1190
          Month 7
          Year 2013
          Updated 2014-09-10-ID-7
          Office 20
```

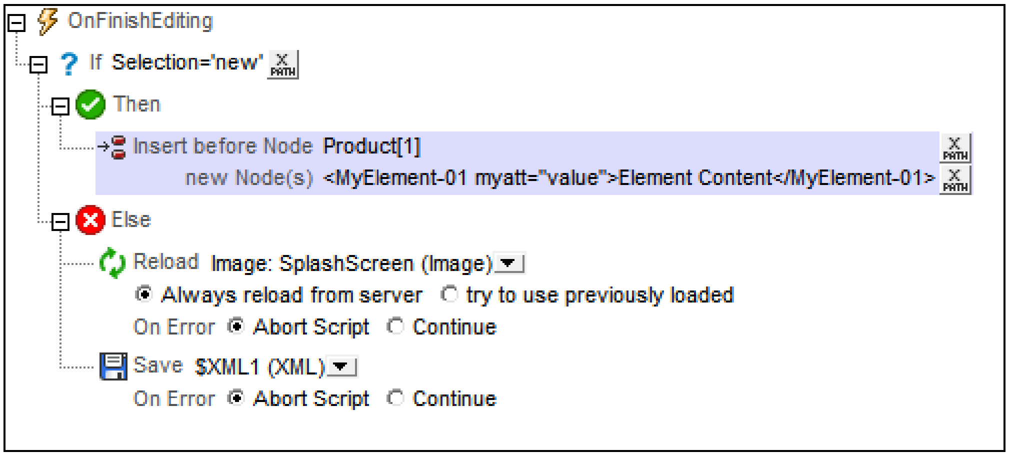
- Now, if the update nodes were the `@Updated` attributes of all `Row` elements, and the XPath expression for the update value were the same as in the previous example:

  ```
  Update Node : $DB2/DB/RowSet/Row/@Updated
  Update Value: concat(current-date-no-TZ(), '-ID-', $MT_TargetNode/../@id)
  ```

  then the `@Updated` of each `Row` element would have a value that would have the `@id` attribute value of its own `Row` element suffixed to the current date.

Insert Node(s)

When the event is triggered, the Insert Node(s) action inserts one or more new nodes before the node/s selected by the XPath expression for the *Insert before Node* setting. The inserted node/s can be a single node, sequence of nodes, or an entire tree fragment. These inserted nodes are constructed using XQuery's XML constructor syntax. All seven XML node kinds can be constructed using this XQuery syntax: elements, attributes, text, document, comment, processing-instruction, and namespace.



OnFinishEditing
If Selection='new'
Then
Insert before Node Product[1]
new Node(s) <MyElement-01 myatt="value">Element Content</MyElement-01>
Else
Reload Image: SplashScreen (Image)
● Always reload from server ○ try to use previously loaded
On Error ● Abort Script ○ Continue
Save $XML1 (XML)
On Error ● Abort Script ○ Continue

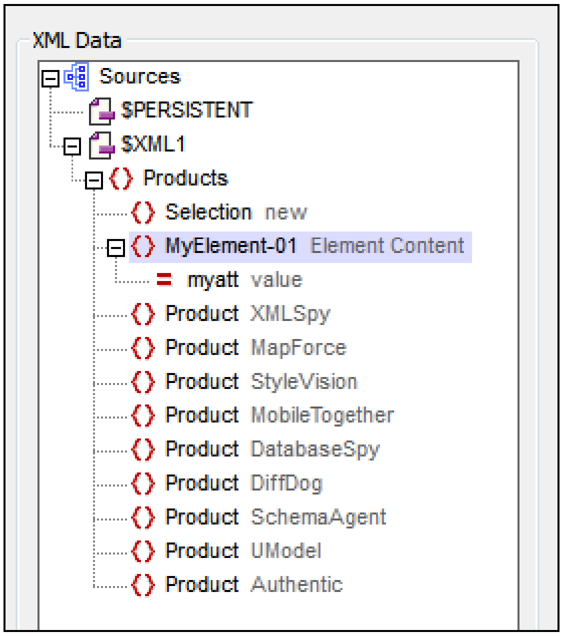
Note: The difference between Insert Node(s) and Append Node(s) is that Insert Node(s) adds the node/s before the selected node/s, whereas Append Node(s) adds the node/s as (first or last) child nodes of the selected node/s.

Location of inserted node/s
The new node/s are inserted before the node/s returned by the XPath expression for this setting (*Insert before Node*). In the screenshot above, the new node/s are inserted before the first `Product` element (selected with the XPath expression `Product[1]`). The context node is the parent of `Product`, a node named `Products`. If the predicate `[1]` is not used, all the `Product` children of `Products` will be returned by the XPath expression, and the new node/s will be inserted before each `Product` element.

New nodes
New nodes can be entered as direct XML constructors as in the screenshot above:

```
<MyElement-01 myatt="value">Element Content</MyElement-01>
```

This inserts the element `MyElement-01` before the first `Product` element, as in the screenshot below.



XML Data

```
Sources
    $PERSISTENT
    $XML1
        Products
            Selection  new
            MyElement-01  Element Content
                myatt  value
            Product  XMLSpy
            Product  MapForce
            Product  StyleVision
            Product  MobileTogether
            Product  DatabaseSpy
            Product  DiffDog
            Product  SchemaAgent
            Product  UModel
            Product  Authentic
```

You can also use an XPath locator expression to insert a node (and all its descendants) from a data source on the page. For example:

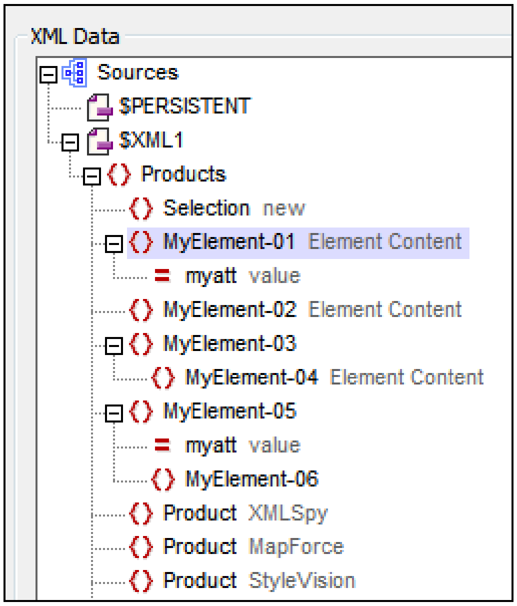
```
$XML2/Row
```

XQuery's computed node constructors can also be used, for example:

```
element MyElement-01 {xs:string("Element Content")}
attribute myatt{"value"}
```

The XPath expression below produces the output shown in the screenshot below, inserted above the first `Product` element.

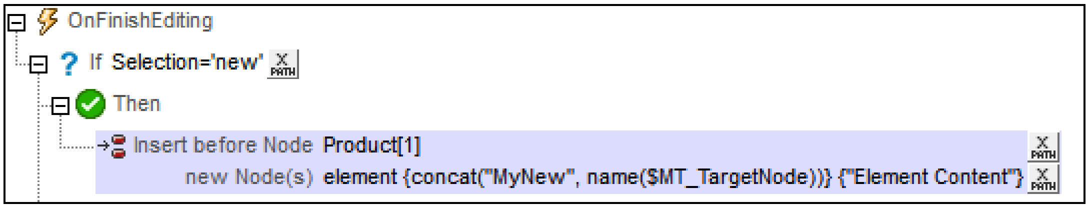
```
<MyElement-01 myatt="value">Element Content</MyElement-01>,
element MyElement-02 {"Element Content"},
element MyElement-03 {element MyElement-04 {"Element Content"}},
element MyElement-05{attribute myatt{"value"}, element MyElement-06{}}
```



XML Data
- Sources
  - $PERSISTENT
  - $XML1
    - Products
      - Selection new
      - MyElement-01 Element Content
        - myatt value
      - MyElement-02 Element Content
      - MyElement-03
        - MyElement-04 Element Content
      - MyElement-05
        - myatt value
        - MyElement-06
      - Product XMLSpy
      - Product MapForce
      - Product StyleVision

The `$MT_TargetNode` variable

The node in the Insert Node(s) definition that is targeted as the node before which to insert child node/s, is automatically saved in MobileTogether Designer's built-in variable **`$MT_TargetNode`**. This variable can then be used in the second XPath expression of the definition, as has been done in the screenshot below.

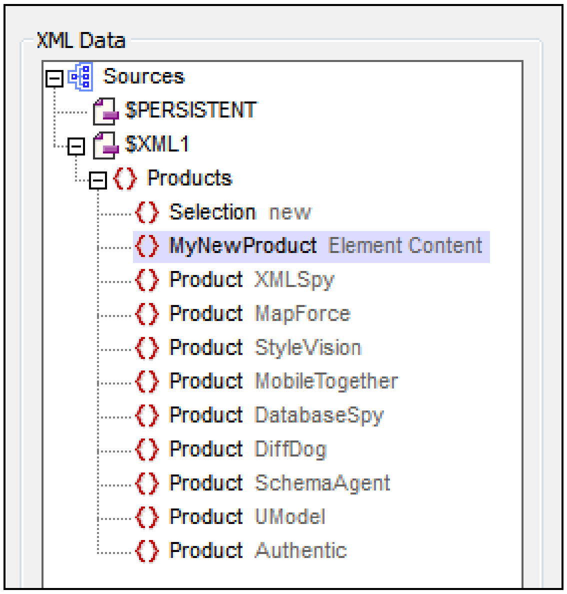


```
OnFinishEditing
  If Selection='new'
    Then
      Insert before Node Product[1]
          new Node(s) element {concat("MyNew", name($MT_TargetNode))} {"Element Content"}
```

The second XPath expression that creates the new node using the same name as the target node (`$MT_TargetNode`) as part of the new node's name.

```
  element {concat("MyNew", name($MT_TargetNode))} {"Element Content"}
```

The result of the Insert Node(s) action defined above is shown in the screenshot below.

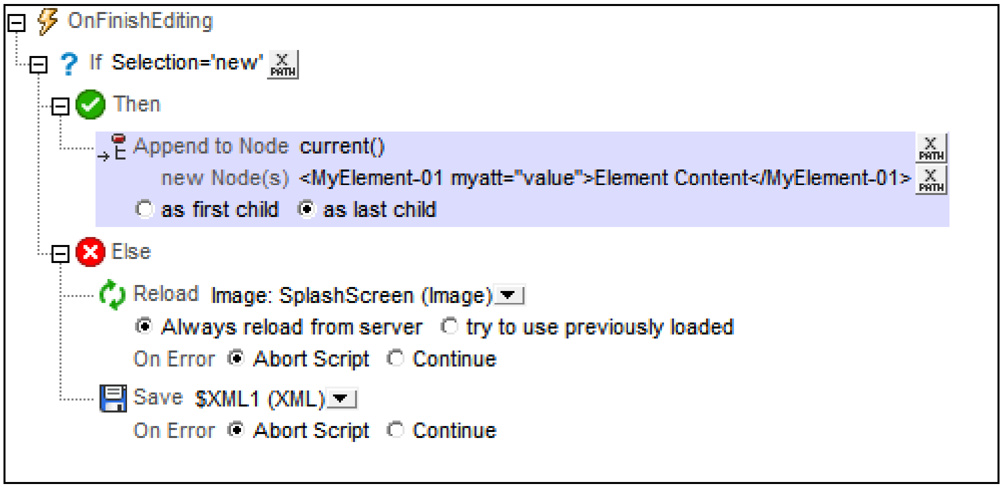


XML Data

- Sources
  - $PERSISTENT
  - $XML1
    - Products
      - Selection new
      - MyNewProduct Element Content
      - Product XMLSpy
      - Product MapForce
      - Product StyleVision
      - Product MobileTogether
      - Product DatabaseSpy
      - Product DiffDog
      - Product SchemaAgent
      - Product UModel
      - Product Authentic

Append Node(s)

When the event is triggered, the Append Node(s) action appends one or more new nodes as a first or last child (set of nodes) of the the node/s selected by the XPath expression for the *Append to Node* setting. The appended node/s can be a single node, sequence of nodes, or an entire tree fragment. These appended nodes are constructed using XQuery's XML constructor syntax. All seven XML node kinds can be constructed using this XQuery syntax: elements, attributes, text, document, comment, processing-instruction, and namespace.



OnFinishEditing
If Selection='new'
Then
Append to Node current()
new Node(s) <MyElement-01 myatt="value">Element Content</MyElement-01>
○ as first child ● as last child
Else
Reload Image: SplashScreen (Image)
● Always reload from server ○ try to use previously loaded
On Error ● Abort Script ○ Continue
Save $XML1 (XML)
On Error ● Abort Script ○ Continue

Note: The difference between Insert Node(s) and Append Node(s) is that Insert Node(s) adds the node/s before the selected node/s, whereas Append Node(s) adds the node/s as (first or last) child nodes of the selected node/s.

Location of appended node/s

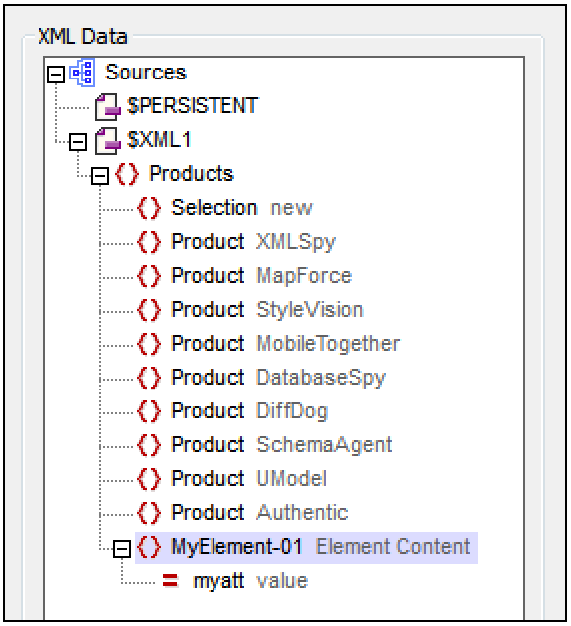
The new node/s are appended as the first or last child node/s of the node/s returned by the XPath expression for this setting (*Insert before Node*). In the screenshot above, the new node/s are appended as the last child nodes of the context node, the `Products` element (selected with the XPath expression `current()`). To select whether the new node/s should be appended as the first or last child nodes/s, select the appropriate radio button in the action's definition.

New nodes

New nodes can be entered as direct XML constructors as in the screenshot above:

```
<MyElement-01 myatt="value">Element Content</MyElement-01>
```

This append the element `MyElement-01` after the last `Product` element, as in the screenshot below.



```
Sources
  $PERSISTENT
  $XML1
    Products
      Selection new
      Product XMLSpy
      Product MapForce
      Product StyleVision
      Product MobileTogether
      Product DatabaseSpy
      Product DiffDog
      Product SchemaAgent
      Product UModel
      Product Authentic
      MyElement-01 Element Content
        myatt value
```

You can also use an XPath locator expression to append a node (and all its descendants) from a data source on the page. For example:

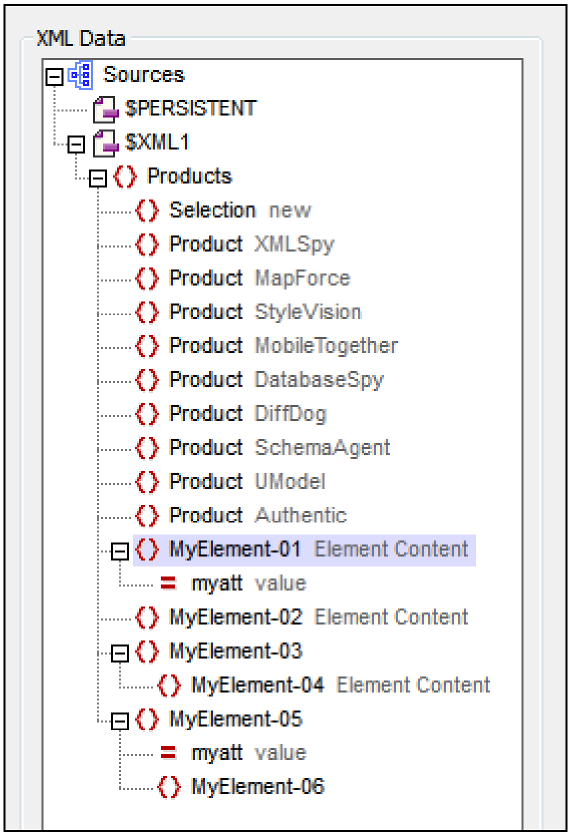
```
$XML2/Row
```

XQuery's computed node constructors can also be used, for example:

```
element MyElement-01 {xs:string("Element Content")}
attribute myatt{"value"}
```

The following XPath expression produces the output shown in the screenshot below, appended as the last child nodes of the `Products` element, that is, after the last current child node (the last `Product` node).

```
<MyElement-01 myatt="value">Element Content</MyElement-01>,
element MyElement-02 {"Element Content"},
element MyElement-03 {element MyElement-04 {"Element Content"}},
element MyElement-05{attribute myatt{"value"}, element MyElement-06{}}
```

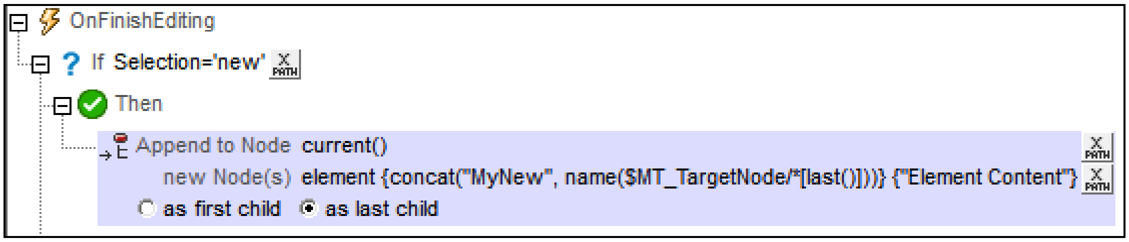


XML Data

- Sources
  - $PERSISTENT
  - $XML1
    - Products
      - Selection new
      - Product XMLSpy
      - Product MapForce
      - Product StyleVision
      - Product MobileTogether
      - Product DatabaseSpy
      - Product DiffDog
      - Product SchemaAgent
      - Product UModel
      - Product Authentic
      - MyElement-01 Element Content
        - myatt value
      - MyElement-02 Element Content
      - MyElement-03
        - MyElement-04 Element Content
      - MyElement-05
        - myatt value
        - MyElement-06

The `$MT_TargetNode` variable

The node in the Append Node(s) definition that is targeted as the node in which to append child node/s, is automatically saved in MobileTogether Designer's built-in variable **`$MT_TargetNode`**. This variable can then be used in the second XPath expression of the definition, as has been done in the screenshot below.

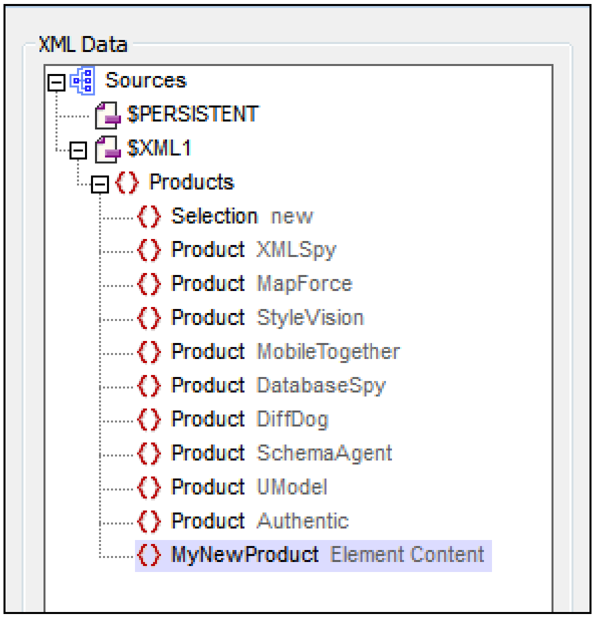


- OnFinishEditing
  - If Selection='new'
    - Then
      - Append to Node current()
        new Node(s) element {concat("MyNew", name($MT_TargetNode/*[last()]))} {"Element Content"}
        ○ as first child ● as last child

The second XPath expression uses the target node (`$MT_TargetNode`) to find the target node's last child element and then uses the name of that child element to build the name of the new element.

```
element {concat("MyNew", name($MT_TargetNode/*[last()]))} {"Element Content"}
```

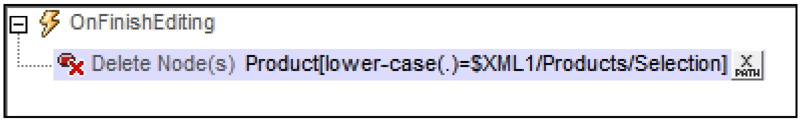
The result of the Append Node(s) action defined above is shown in the screenshot below.



XML Data
- Sources
  - $PERSISTENT
  - $XML1
    - Products
      - Selection new
      - Product XMLSpy
      - Product MapForce
      - Product StyleVision
      - Product MobileTogether
      - Product DatabaseSpy
      - Product DiffDog
      - Product SchemaAgent
      - Product UModel
      - Product Authentic
      - MyNewProduct Element Content

Delete Node(s)

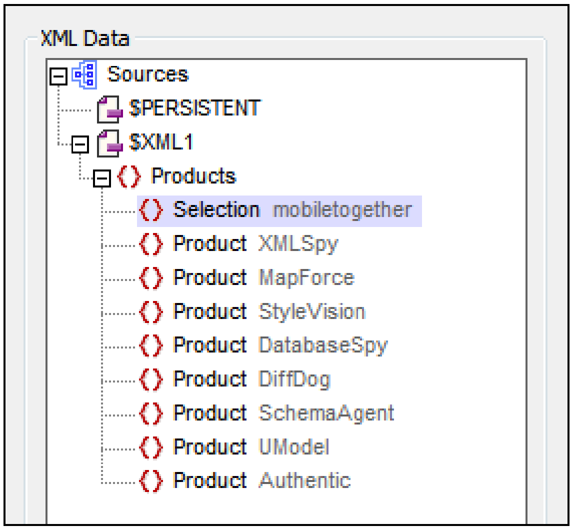
When the event is triggered, the Delete Node(s) action deletes the node/s selected in the action's XPath expression.



OnFinishEditing

Delete Node(s) Product[lower-case(.)=$XML1/Products/Selection]

The definition in the screenshot above is for an `OnFinishedEditing` event of a combo box that updates the `$XML1Products/Selection` element. The context node is `Products`. The action deletes the child `Product` element that has content, which when converted to lowercase matches the (lowercase) content of the `Selection` element. In the screenshot below, `mobiletogether` has been selected in the combo box and becomes the value of the Selection element. The `Product` element containing the text `"MobileTogether"` has been deleted.



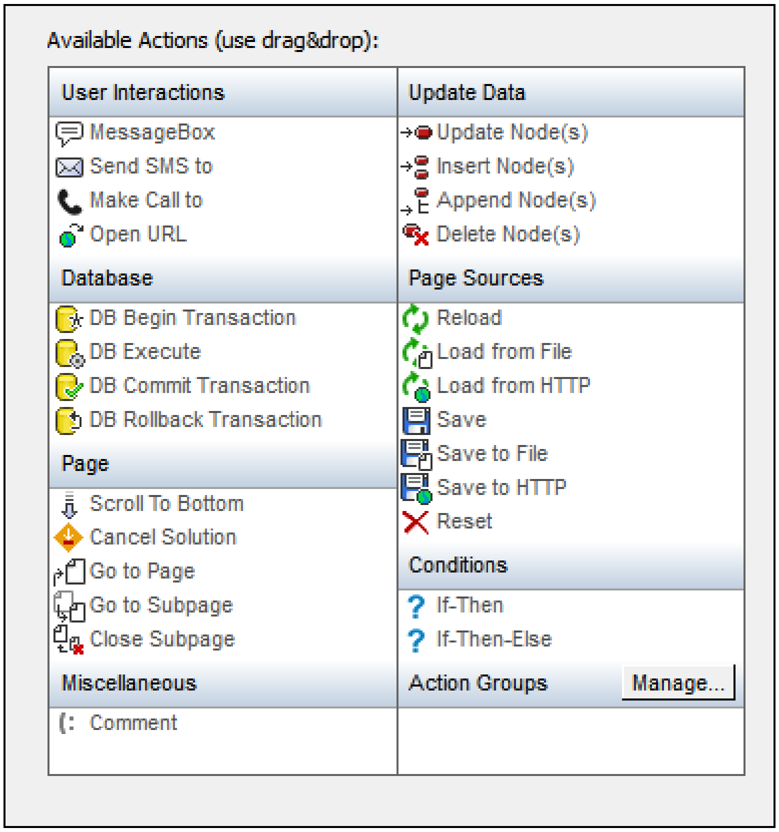
XML Data

```
Sources
  $PERSISTENT
  $XML1
    Products
      Selection  mobiletogether
      Product  XMLSpy
      Product  MapForce
      Product  StyleVision
      Product  DatabaseSpy
      Product  DiffDog
      Product  SchemaAgent
      Product  UModel
      Product  Authentic
```

Page Sources

The following actions are available in the Page Sources group of the Actions dialog (*screenshot below*):

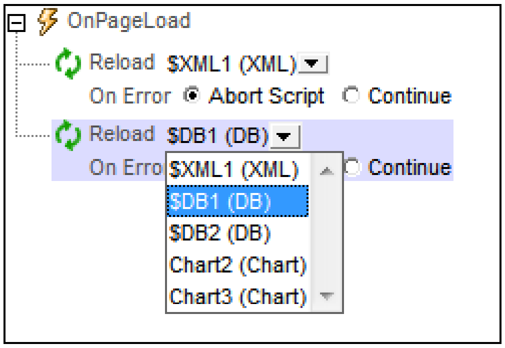
- Reload
- Load from File
- Load from HTTP
- Save
- Save to File
- Save to HTTP
- Reset



The actions in this group are available for page events and control events. The fastest way to access the Actions dialog (*screenshot above*) is to right-click the page or control and select the page/control actions command. *See also Page Events and Control Events.*

Reload

Reloads the external resource specified in the action's combo box (*see screenshot below*). External resources referenced by the project are available as entries in the combo box, and include XML files, charts, and images.



OnPageLoad

Reload $XML1 (XML)

On Error ● Abort Script ○ Continue

Reload $DB1 (DB)

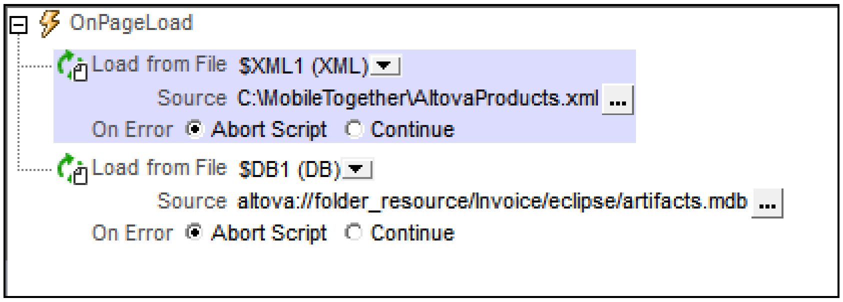
On Erro ○ Continue

Note the following:

- Notice that the screenshot above contains entries for chart controls. If selected for update, the chart image will be updated when the event is triggered. Image controls can also have their linked images updated with the Update action.
- Updates can also be triggered by control events. This means, for example, that when a combo box is edited, the OnFinishEditing event of the combo box can be set to update the image linked to an image control. See the Quick Start (Part 1) tutorial for an example of this.
- To reload multiple resources when the event is triggered, add multiple `Reload` actions, as has been done in the screenshot above.

Load from File

For each `LoadFromFile` action, you can select one data source from the the action's combo box (*see screenshot below*). You can then specify a file from which to load data for that data source when the event is triggered.



OnPageLoad

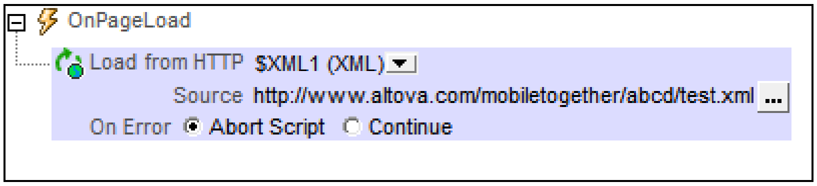
Load from File $XML1 (XML)
Source C:\MobileTogether\AltovaProducts.xml
On Error ● Abort Script ○ Continue

Load from File $DB1 (DB)
Source altova://folder_resource/Invoice/eclipse/artifacts.mdb
On Error ● Abort Script ○ Continue

To load data for multiple data sources when the event is triggered, add multiple `LoadFromFile` actions, as has been done in the screenshot above.

Load from HTTP

For each `LoadFromHTTP` action, you can select one data source from the available page sources and specify an HTTP source from which to load data for that data source when the event is triggered. To load data for multiple data sources, add multiple `LoadFromHTTP` actions.



OnPageLoad

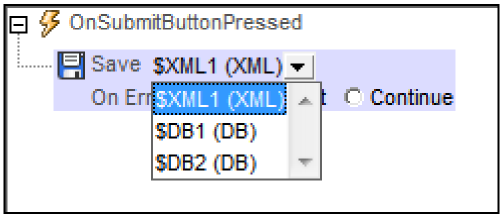
Load from HTTP $XML1 (XML)

Source http://www.altova.com/mobiletogether/abcd/test.xml

On Error ● Abort Script ○ Continue

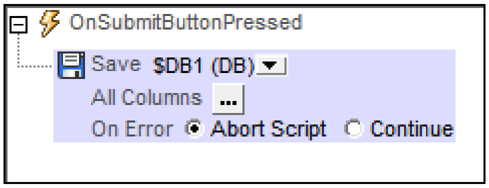
Save

Saves data that is currently in the data source tree that is selected in the action's combo box to the default file of that data source. The data source must be an editable XML file, DB, or FlowForce project. To save data for multiple data sources, add multiple `Save` actions.



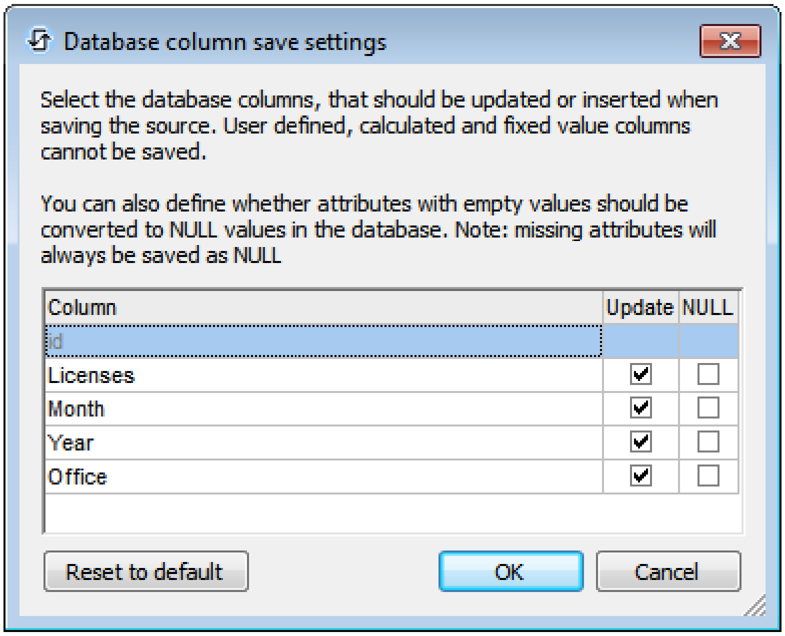
OnSubmitButtonPressed
Save $XML1 (XML) ▾
On Err $XML1 (XML) t ○ Continue
$DB1 (DB)
$DB2 (DB)

If the data source that is being saved is a DB, then, by default, all editable columns are selected for being updated (*see screenshot below*).



OnSubmitButtonPressed
Save $DB1 (DB) ▾
All Columns ...
On Error ● Abort Script ○ Continue

To select specific columns to update, click ... . This displays the Database Column Save Settings dialog (*screenshot below*).



Database column save settings

Select the database columns, that should be updated or inserted when saving the source. User defined, calculated and fixed value columns cannot be saved.

You can also define whether attributes with empty values should be converted to NULL values in the database. Note: missing attributes will always be saved as NULL

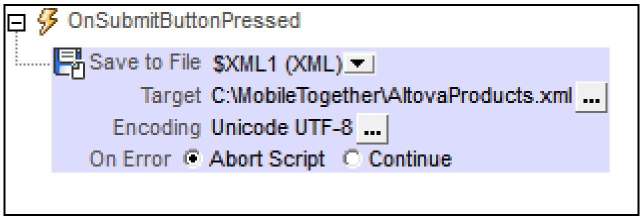
Column	Update	NULL
id		
Licenses	☑	□
Month	☑	□
Year	☑	□
Office	☑	□

Reset to default OK Cancel

The dialog displays the columns of the data source. Columns that cannot be updated (because they are user-defined, fixed-value, or calculated-value) will not have an *Update* option. Deselect the columns you do not want to update. Attributes with empty values can be converted to `NULL` values in the DB by checking that column's *NULL* check box. Note that missing attributes will always be saved as `NULL`. If you wish to reset the Save settings so that all columns are updtaed, click **Reset to default**.

Save to File

Saves the data source that is selected in the action's combo box to the XML file specified in the *Target* field (*see screenshot below*). The encoding of the XML file is specified in the *Encoding* field. To save data for multiple data sources, add multiple `SaveToFile` actions.



OnSubmitButtonPressed

Save to File $XML1 (XML)

Target C:\MobileTogether\AltovaProducts.xml

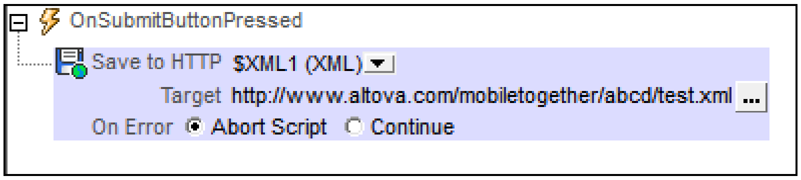
Encoding Unicode UTF-8

On Error ● Abort Script ○ Continue

Save to HTTP

Saves the data source that is selected in the action's combo box to an XML or HTML file at a target HTTP location that is specified in the *Target* field of action's definition (*see screenshot below*).

Saves the data source that is selected in the action's combo box to the XML file specified in the *Target* field (*see screenshot below*). The encoding of the XML file is specified in the *Encoding* field. To save data for multiple data sources, add multiple `SaveToFile` actions.



OnSubmitButtonPressed

Save to HTTP $XML1 (XML)

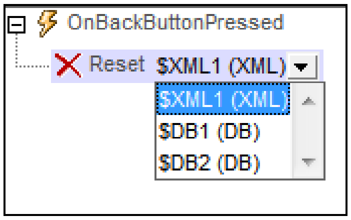
Target http://www.altova.com/mobiletogether/abcd/test.xml ...

On Error ● Abort Script ○ Continue

To enter access details of the HTTP location, click ... . This displays the Edit Web Access Settings dialog, in which you can enter the file's URL and security settings.

Reset

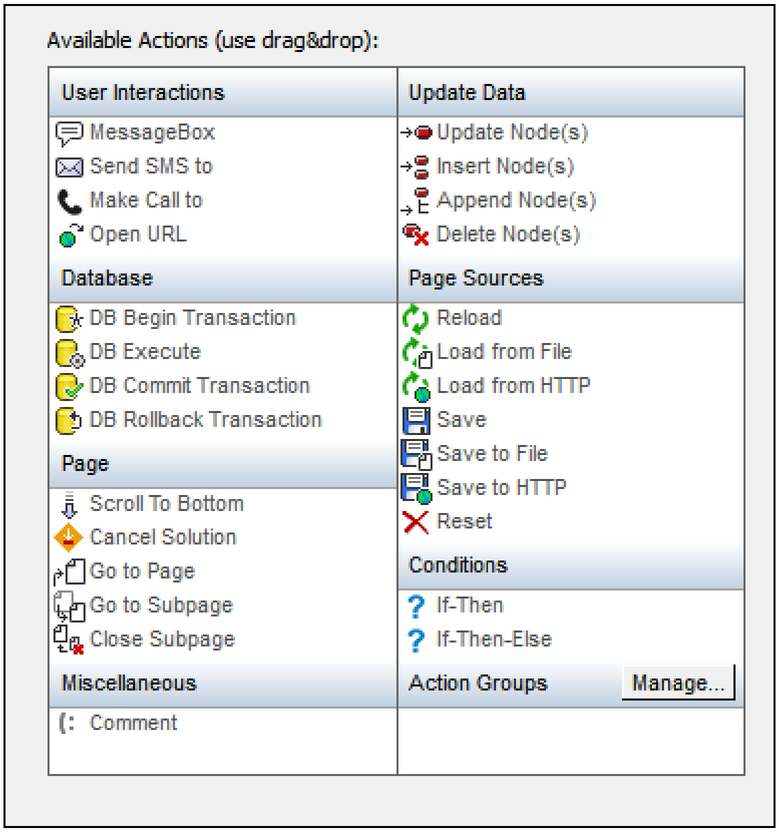
Resets the data source that is selected in the action's combo box to the default values defined in the Page Sources Pane.



Conditions

The following actions are available in the Conditions group of the Actions dialog (*screenshot below*):

- If-Then
- If-Then-Else



Available Actions (use drag&drop):

User Interactions
MessageBox
Send SMS to
Make Call to
Open URL

Database
DB Begin Transaction
DB Execute
DB Commit Transaction
DB Rollback Transaction

Page
Scroll To Bottom
Cancel Solution
Go to Page
Go to Subpage
Close Subpage

Miscellaneous
Comment

Update Data
Update Node(s)
Insert Node(s)
Append Node(s)
Delete Node(s)

Page Sources
Reload
Load from File
Load from HTTP
Save
Save to File
Save to HTTP
Reset

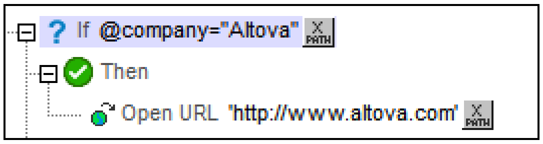
Conditions
If-Then
If-Then-Else

Action Groups
Manage...

The actions in this group are available for page events and control events. The fastest way to access the Actions dialog (*screenshot above*) is to right-click the page or control and select the page/control actions command. *See also Page Events and Control Events.*

If-Then

Sets the action to be carried out if the `IF` condition evaluates to `true`. The action to be carried out is dropped as a child of the `THEN` clause.

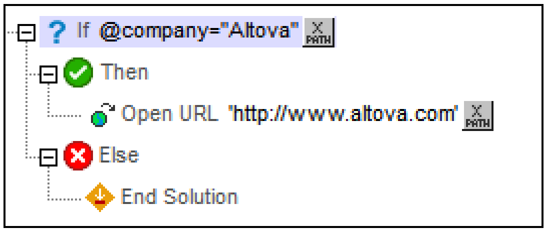


If @company="Altova"
Then
Open URL 'http://www.altova.com'

The condition in the definition above tests whether the current element has an attribute called `company` that has a value of `Altova`. If `true`, the Altova website URL is opened.

If-Then-Else

Sets the actions to be carried out for both the `true` and `false` evaluations of the `IF` condition. The action to be carried out for a `true` evaluation is dropped as a child of the `THEN` clause. The action to be carried out for a `false` evaluation is dropped as a child of the `ELSE` clause.



```
? If @company="Altova"
    Then
        Open URL 'http://www.altova.com'
    Else
        End Solution
```

The condition in the definition above tests whether the current element has an attribute called `company` that has a value of `Altova`. If `true`, the Altova website URL is opened. If `false`, the solution is exited.

7.6 Charts

Charts provide a graphical representation of data in the source document. A chart is set up by defining XPath expressions to specify a sequence of items for each axis of the chart. MobileTogether Designer then automatically generates the chart. The table below shows the types of charts that can be created, and the kind of data items that are required for each of the chart's axes.

Chart type	X-Axis (Category)	Y-Axis (Value)	Number of Series (on Z-Axis)
Pie Charts (2D, 3D)	Text	Numeric	1
Bar Charts Ungrouped (2D, 3D)	Text	Numeric	1
Bar Charts Grouped (2D, 3D)	Text	Numeric	> 1
Category Line Graphs	Text	Numeric	1 line = 1 series
Value Line Graphs	Numeric	Numeric	1 line = 1 series
Area and Stacked Area Charts	Text	Numeric	1 area = 1 series
Candlestick Charts	Text	Numeric	3 or 4
Gauge	—	Numeric	1
Overlay Charts	Text	Numeric	= 1 or > 1 per chart

This section is organized as follows:

- Chart Data Selection explains how to select data for the different axes
- Chart Settings and Appearance describes how to define the properties of charts

Creating and Configuring Charts

This section:

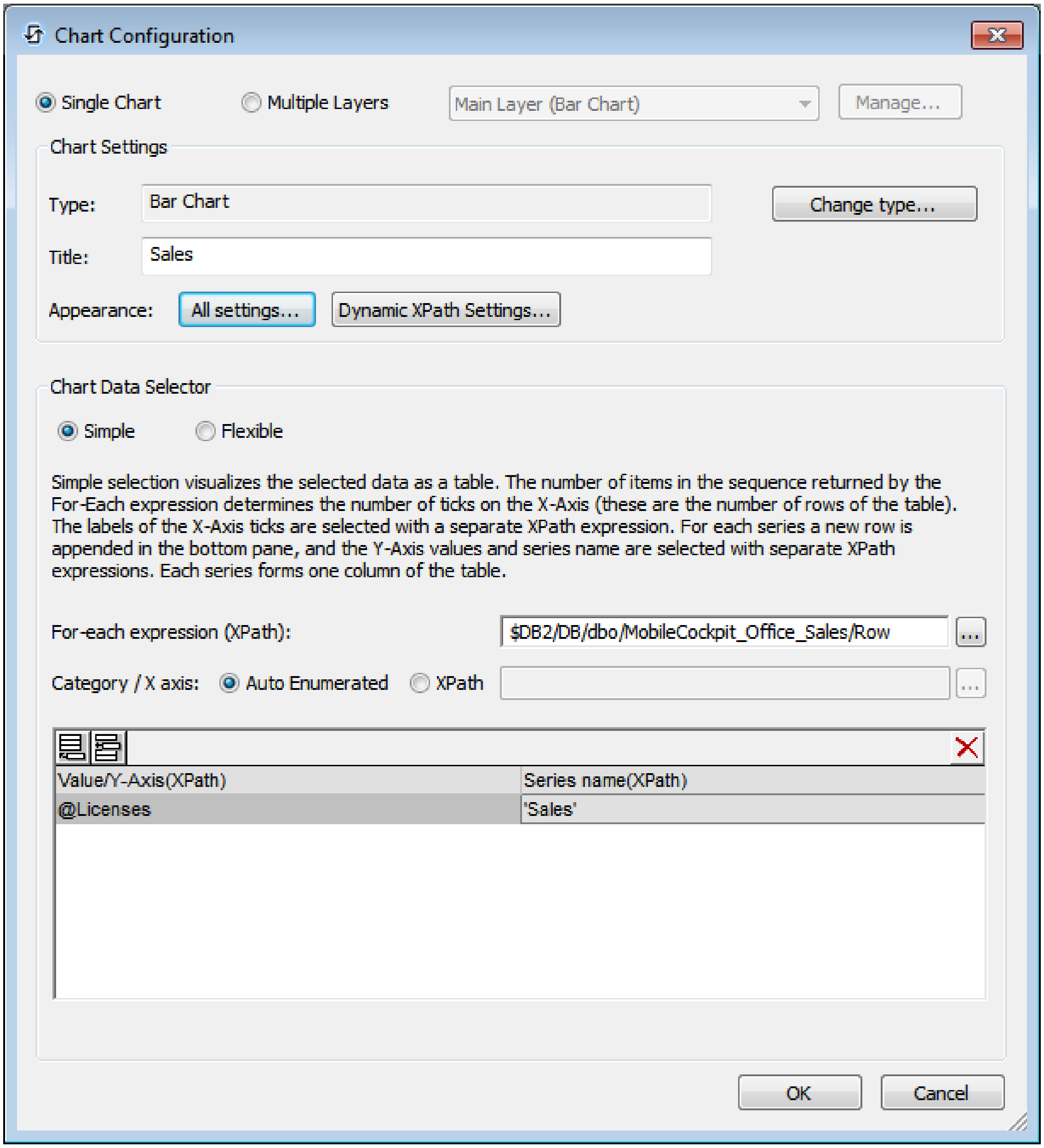
- Creating a chart
- The context node
- The Chart Configuration dialog
- Editing chart settings and data selection

Creating a chart
To insert a chart in the design, drag the Chart control from the Controls Pane to the location in the design where you wish to insert the chart.

The context node
Drag an XML node from the Page Sources Pane onto the chart in the design to make this XML node the context node of the chart's XPath expressions. You can change the chart's context node at any time by dragging a new XML node onto the chart. It is important to be aware of the chart's context node because this context node is the starting point of path locators in XPath expressions.

The Chart Configuration dialog
A chart can be configured in the Chart Configuration dialog (*screenshot below*) at the time that the chart is created. Configuring a chart involves: (i) specifying data selection for the charts's axes, and (ii) defining the chart's properties. These settings can be edited at any later time. After assigning an XML context node to the chart, do either of the following to bring up the Chart Configuration dialog:

- Double-click the chart
- Select the chart in the design, then click the Edit XPath button of the *Chart Settings* property in the Styles & Properties Pane



Type: Bar Chart

Title: Sales

Simple selection visualizes the selected data as a table. The number of items in the sequence returned by the For-Each expression determines the number of ticks on the X-Axis (these are the number of rows of the table). The labels of the X-Axis ticks are selected with a separate XPath expression. For each series a new row is appended in the bottom pane, and the Y-Axis values and series name are selected with separate XPath expressions. Each series forms one column of the table.

For-each expression (XPath): $DB2/DB/dbo/MobileCockpit_Office_Sales/Row

Category / X axis: ● Auto Enumerated ○ XPath

Value/Y-Axis(XPath)	Series name(XPath)
@Licenses	'Sales'

The Chart Configuration dialog has three parts:

- *Single or Multiple layers:* Multiple layers can be selected to create overlay charts, in which multiple charts can be presented, one overlaid upon the other in a sequence of layers
- *Chart Settings:* To select the chart type and define the appearance of the chart
- *Chart Data Selection:* To select the data for the various axes of the chart using either the simple option or flexible option

Editing chart settings and data selection

If you wish to change chart settings or a chart's data selection after a chart has been created, right-click the chart in the design and select **Edit Chart Settings**. This pops up the Chart Configuration dialog (*screenshot above*) of that chart. You can edit the chart's settings or data selection in the dialog, then click **OK** to finish.

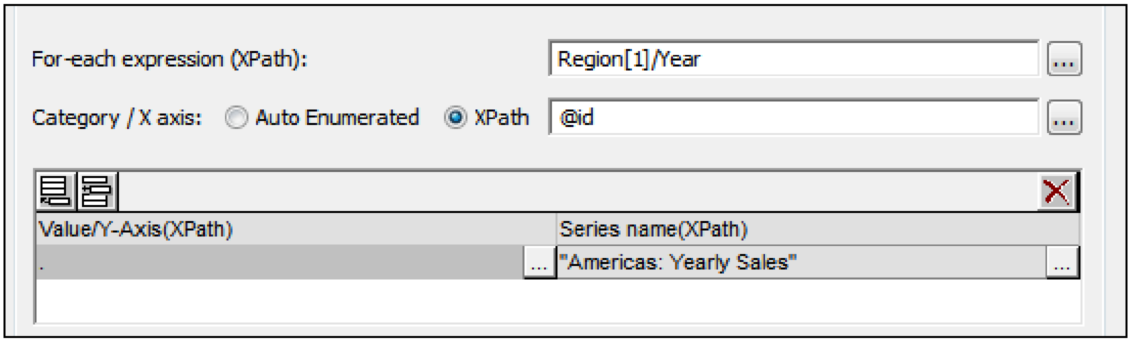
Chart Data Selection

This topic contains simple examples to illustrate how chart data selection works.

⊟ XML file used in chart examples: YearlySales.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<Data xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xsi:noNamespaceSchemaLocation="YearlySales.xsd">
       <ChartType>Pie Chart 2D</ChartType>
       <Region id="Americas">
              <Year id="2005">30000</Year>
              <Year id="2006">90000</Year>
              <Year id="2007">120000</Year>
              <Year id="2008">180000</Year>
              <Year id="2009">140000</Year>
              <Year id="2010">100000</Year>
       </Region>
       <Region id="Europe">
              <Year id="2005">50000</Year>
              <Year id="2006">60000</Year>
              <Year id="2007">80000</Year>
              <Year id="2008">100000</Year>
              <Year id="2009">95000</Year>
              <Year id="2010">80000</Year>
       </Region>
       <Region id="Asia">
              <Year id="2005">10000</Year>
              <Year id="2006">25000</Year>
              <Year id="2007">70000</Year>
              <Year id="2008">110000</Year>
              <Year id="2009">125000</Year>
              <Year id="2010">150000</Year>
       </Region>
</Data>
```

Chart data selection with four XPath expressions
The screenshot below shows the Chart Configuration dialog, at the bottom of which is the Chart Data Selector pane with fields for entering the four XPath expressions for data selection.



For-each expression (XPath): Region[1]/Year

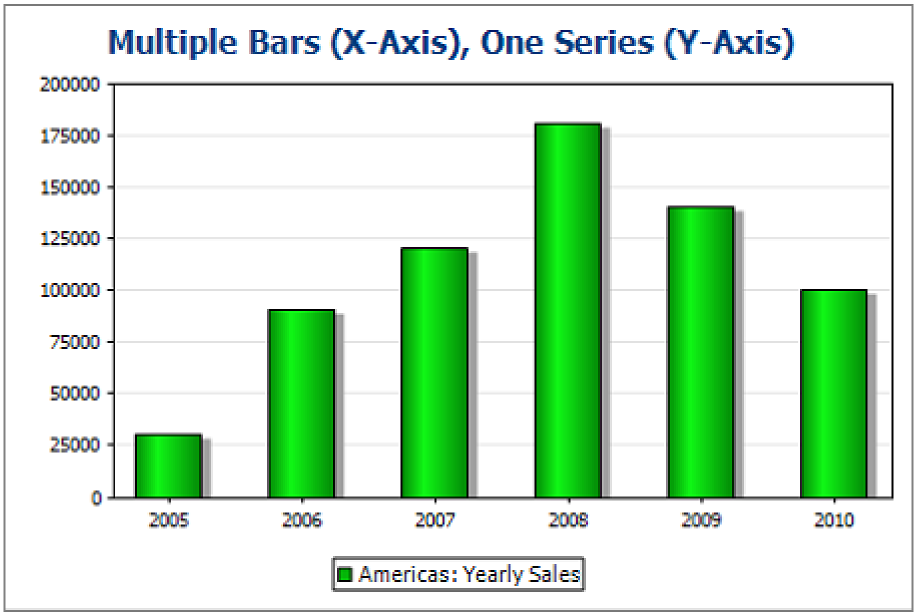
Category / X axis: ○ Auto Enumerated ● XPath @id

Value/Y-Axis(XPath)	Series name(XPath)
.	"Americas: Yearly Sales"

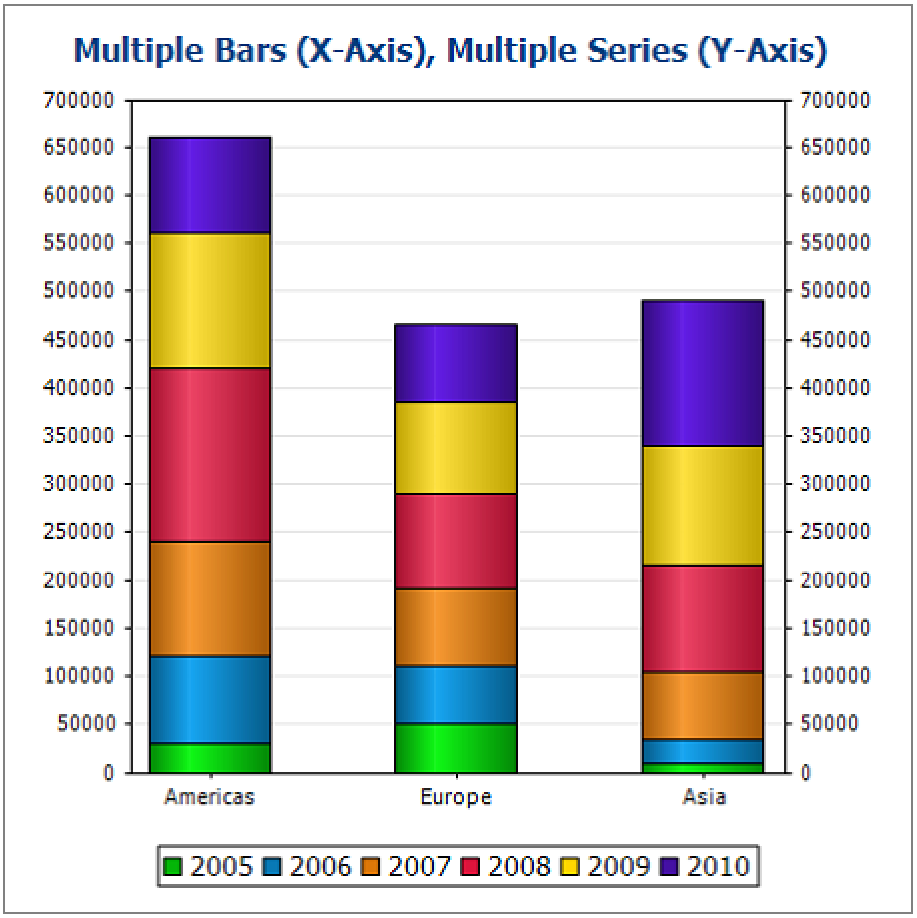
The four XPath expressions in the Chart Data Selector pane work together and do the following:

XPath	Decsription
For-Each	• Sets the context for the other three XPath expressions • Sets the number of items in the returned sequence as the number of ticks on the X-Axis. • In the case of the screenshot above, the `Region[1]/Year` expression returns six node items; so there will be six ticks on the X-Axis (*see screenshot below*).
X-Axis	• The items in the returned sequence provide the label text for the corresponding ticks on the X-Axis. • In the example shown above, the `@id` expression returns the `id` attribute value of each `Year` element. These values become the labels of the corresponding ticks (*see screenshot below*). • Since we have specified that this will be a bar chart, bars are drawn at the ticks.
Y-Axis	• The Y-Axis can display multiple series, each of which is defined in one row of the Y-Axis table. • Each series is defined by two XPath expressions: one for the series value, the other for the series name. • In our example, the `self::node()` XPath expression (indicated by its abbreviated form of a period) selects the current node, which is the `Year` element that is the context node. So, for each `Year` element (represented by a bar on the X-Axis), the `Year` element's content will be read as the Y-Axis value of that year, and therefore plotted as the height of the bar (*see screenshot below*). The screenshot further below shows a chart with multiple series on the Y-Axis.
Series name	• This expression provides the legend text for the series. In our example, the legend text (which appears at the bottom of the chart, *screenshot below*) is obtained from an XPath expression that is a text string (*see screenshot above*).

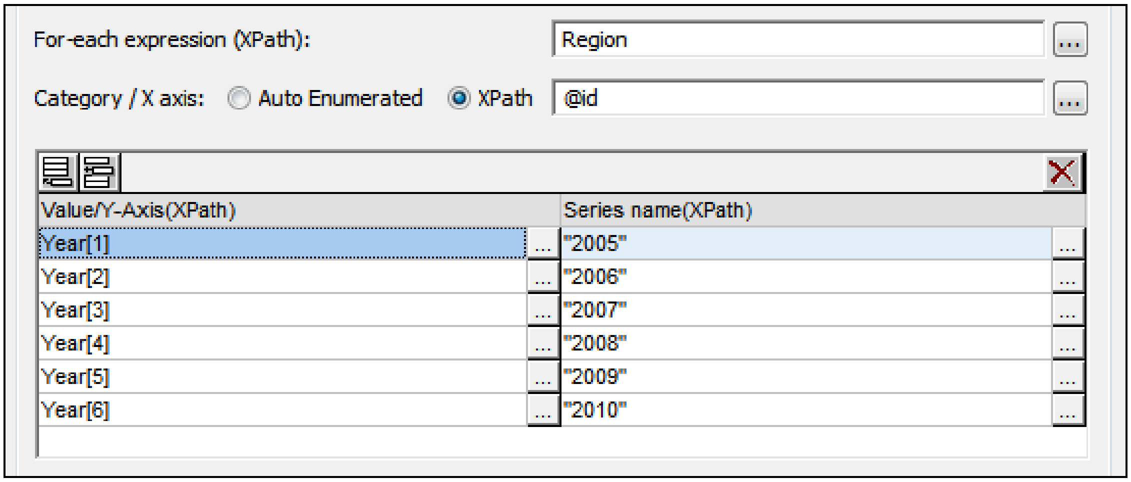
A bar chart that is generated for the data selection shown in the screenshot above and the XML data in `YearlySales.xml` looks like the chart in the screenshot below.



The screenshot above shows a bar chart with a single series, while that below is of a stacked bar chart with multiple series. In the latter example, the value of each series is stacked on to the bar.



The XPath expressions of this chart are shown in the screenshot below.



For-each expression (XPath): Region

Category / X axis: ○ Auto Enumerated ● XPath @id

Value/Y-Axis(XPath)	Series name(XPath)
Year[1]	"2005"
Year[2]	"2006"
Year[3]	"2007"
Year[4]	"2008"
Year[5]	"2009"
Year[6]	"2010"

Note: Pie charts and gauge charts have a single nominal series, which requires no name. So if a series name is entered in the data selection, it is ignored. For ungrouped bar charts, however, the name of the single series, if present, is used for the legend. For gauge charts, in addition to any Series Name entry being ignored, the X-Axis data selection is also ignored; only the Y-Axis selection is used for gauge charts.

Chart Data Selection: Simple

This section:

- Introduction
- The context node
- Data selection for the X and Y axes
- If the For-Each expression returns items that are not nodes

Introduction

In the Chart Data Selector pane of the Chart Configuration dialog, the Simple option enables the data selection to be visualized as a table. We use the XML document that is listed below to explain the visualization.

⊟ XML file used in chart examples: YearlySales.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<Data xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xsi:noNamespaceSchemaLocation="YearlySales.xsd">
       <ChartType>Pie Chart 2D</ChartType>
       <Region id="Americas">
              <Year id="2005">30000</Year>
              <Year id="2006">90000</Year>
              <Year id="2007">120000</Year>
              <Year id="2008">180000</Year>
              <Year id="2009">140000</Year>
              <Year id="2010">100000</Year>
       </Region>
       <Region id="Europe">
              <Year id="2005">50000</Year>
              <Year id="2006">60000</Year>
              <Year id="2007">80000</Year>
              <Year id="2008">100000</Year>
              <Year id="2009">95000</Year>
              <Year id="2010">80000</Year>
       </Region>
       <Region id="Asia">
              <Year id="2005">10000</Year>
              <Year id="2006">25000</Year>
              <Year id="2007">70000</Year>
              <Year id="2008">110000</Year>
              <Year id="2009">125000</Year>
              <Year id="2010">150000</Year>
       </Region>
</Data>
```

The context node

In the design, drag an XML node from the Page Sources Pane to make this node the context node of the chart's XPath expressions. You can change the charts's context node by dragging a new XML node onto the chart. It is important to be aware of the chart's context node, since this context node is the starting point of path locators in XPath expressions.

Selecting data for the X and Y axes

In the Chart Data Selector pane (*screenshot below*) we make the data selection as shown in the screenshot. Since the chart has been inserted within the `Data` node, the context node for the For-Each expression is the `Data` node.

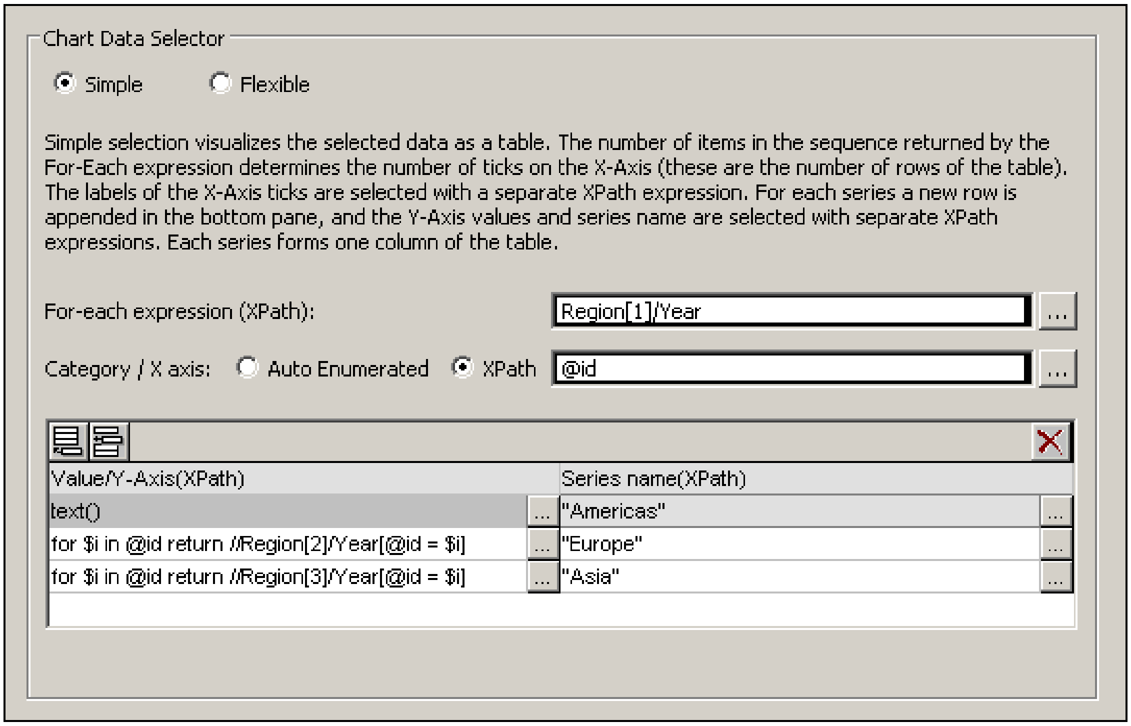


Chart Data Selector

● Simple ○ Flexible

Simple selection visualizes the selected data as a table. The number of items in the sequence returned by the For-Each expression determines the number of ticks on the X-Axis (these are the number of rows of the table). The labels of the X-Axis ticks are selected with a separate XPath expression. For each series a new row is appended in the bottom pane, and the Y-Axis values and series name are selected with separate XPath expressions. Each series forms one column of the table.

For-each expression (XPath): Region[1]/Year

Category / X axis: ○ Auto Enumerated ● XPath @id

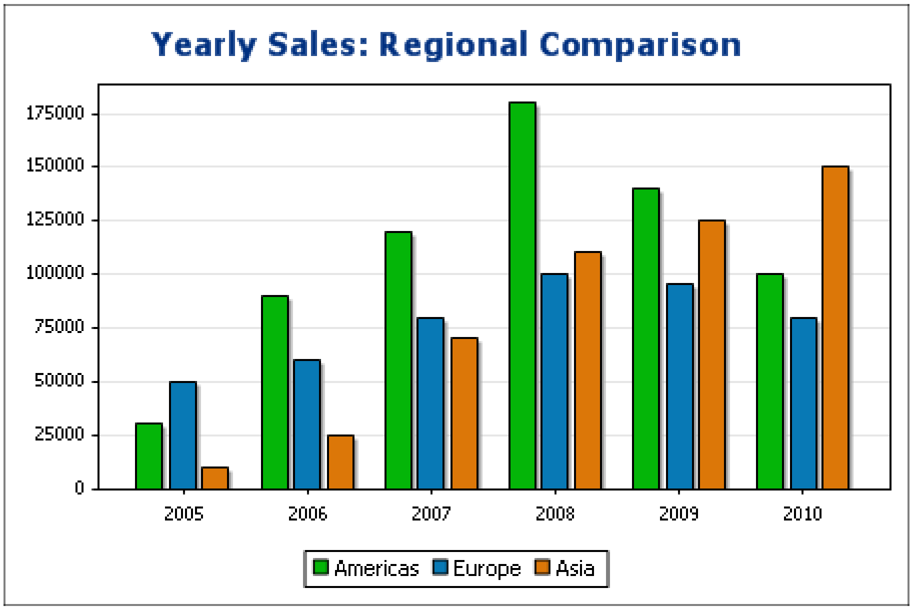
Value/Y-Axis(XPath)	Series name(XPath)
text()	"Americas"
for $i in @id return //Region[2]/Year[@id = $i]	"Europe"
for $i in @id return //Region[3]/Year[@id = $i]	"Asia"

The chart data table can be visualized as the table below. What happens is that for each `Region[1]/Year` element a row is created and the X-Axis and Y-Axis XPath expressions are evaluated within the respective `Region[1]/Year` element's context.

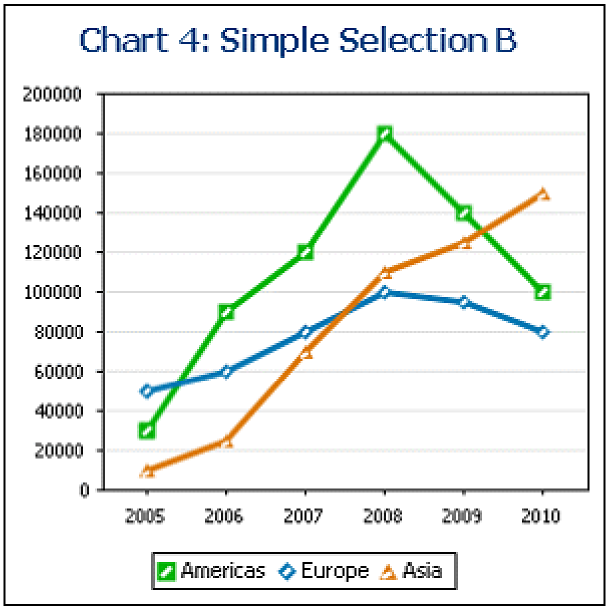
For-Each XPath	X-Axis	Y-Axis for Series		
		Americas	Europe	Asia
`Region[1]/Year[1]`	`@id`	`text()`	`XPath-1`	`XPath-2`
`Region[1]/Year[2]`	`@id`	`text()`	`XPath-1`	`XPath-2`
`Region[1]/Year[3]`	`@id`	`text()`	`XPath-1`	`XPath-2`
`Region[1]/Year[4]`	`@id`	`text()`	`XPath-1`	`XPath-2`
`Region[1]/Year[5]`	`@id`	`text()`	`XPath-1`	`XPath-2`
`Region[1]/Year[6]`	`@id`	`text()`	`XPath-1`	`XPath-2`

- The For-Each expression `Region[1]/Year` returns six nodes (which become the rows of the table). The number of items in the sequence returned by the For-Each expression determines the **number of ticks** on the X-Axis.
- The XPath expression for the X-Axis returns the `@id` attribute value of each `Region[1]/Year` element. These values will be the **labels of the X-Axis ticks**. If there are more labels than ticks then extra ticks will be generated so that all labels are plotted. If there are fewer labels than ticks, then the latter ticks (for which no corresponding labels exist) will be unlabelled. The Auto-Enumerated option generates a sequence of integers starting with 1, and assigns each integer sequentially to an X-Axis tick.
- The XPath expression for the `Americas` series (`text()`) returns the content of each of the `Region[1]/Year` elements. This expression could also have been one similar to that for the Europe and Asia series (explained below)—as long as it efficiently returns the values we want.
- The XPath expression for the `Europe` series is: `for $i in @id return //Region[2]/Year[@id=$i]`. This expression does the following: (i) looks for the current `Region[1]/Year/@id` attribute value, (ii) returns the content of the `Region[2]/Year` element that has the same `@id` value as the `@id` value of the current `Region[1]/Year` element.
- The XPath expression for the `Asia` series works in a similar way to the XPath expression for the `Europe` series.

The bar chart generated with this data selection would look something like this:



The line graph chart for this data selection would look like this:



If the For-Each expression returns items that are not nodes
Since the number of X-Axis ticks is primarily dependent on the number of items returned by the For-Each XPath expression, the XPath expression in the screenshot below (`distinct-values(//Year/@id)`), which returns the six unique year values, will also generate six ticks on the X-Axis. The items returned by the sequence, however, are atomic values, not nodes. Consequently, although they can be used as context items, they cannot be used as context nodes for locating nodes in the XML tree. They can, however, be used to locate nodes on the basis of the equality of values—which is how we will use them.

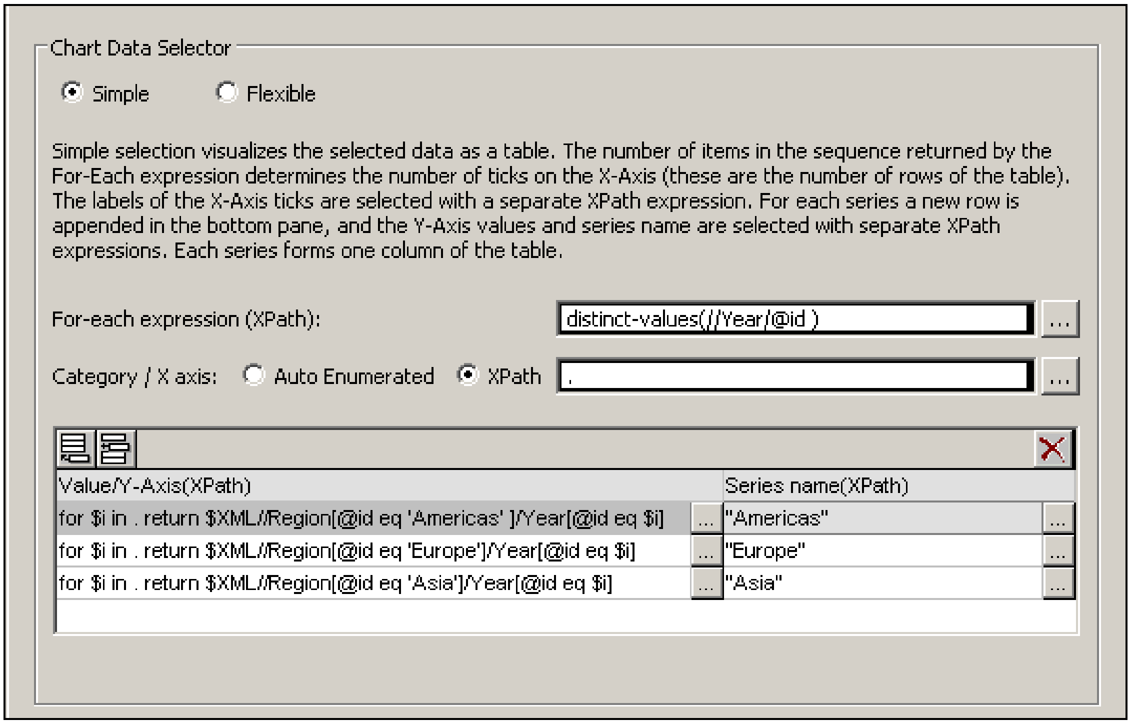


Chart Data Selector

● Simple ○ Flexible

Simple selection visualizes the selected data as a table. The number of items in the sequence returned by the For-Each expression determines the number of ticks on the X-Axis (these are the number of rows of the table). The labels of the X-Axis ticks are selected with a separate XPath expression. For each series a new row is appended in the bottom pane, and the Y-Axis values and series name are selected with separate XPath expressions. Each series forms one column of the table.

For-each expression (XPath): distinct-values(//Year/@id)

Category / X axis: ○ Auto Enumerated ● XPath .

Value/Y-Axis(XPath)	Series name(XPath)
for $i in . return $XML//Region[@id eq 'Americas']/Year[@id eq $i]	"Americas"
for $i in . return $XML//Region[@id eq 'Europe']/Year[@id eq $i]	"Europe"
for $i in . return $XML//Region[@id eq 'Asia']/Year[@id eq $i]	"Asia"

In the data selection shown in the screenshot above, note the following:

- The X-Axis and Y-Axis data selections use the atomic values returned by the For-Each expression, respectively, as direct output and as filter test values.
- Location steps in XPath expressions start at the document node (the `$XML` in `$XML//Region...`). This is necessary because the atomic values provide no locational context.

The chart data table would evaluate to the following:

For-Each XPath	X-Axis	Y-Axis for Series		
		Americas	Europe	Asia
2005	2005	XPath-1	XPath-2	XPath-3
2006	2006	XPath-1	XPath-2	XPath-3
2007	2007	XPath-1	XPath-2	XPath-3
2008	2008	XPath-1	XPath-2	XPath-3
2009	2009	XPath-1	XPath-2	XPath-3
2010	2010	XPath-1	XPath-2	XPath-3

Chart Data Selection: Flexible

This section:

- About flexible chart data selection
- One row, one series
- Three rows, three series, category values not merged
- Three rows, three series, category values merged
- One row, three series
- Rules for chart data selection

About flexible chart data selection

In the Chart Data Selector pane (*screenshot below*) of the Chart Configuration dialog, the Flexible option enables the Series Axis (Z-Axis), X-Axis, and Y-Axis data to be selected freely using XPath expressions. The XPath expression for an axis returns the sequence of items that are to be plotted on that axis. These sequences (of items) for the axes are then collated to generate the chart.

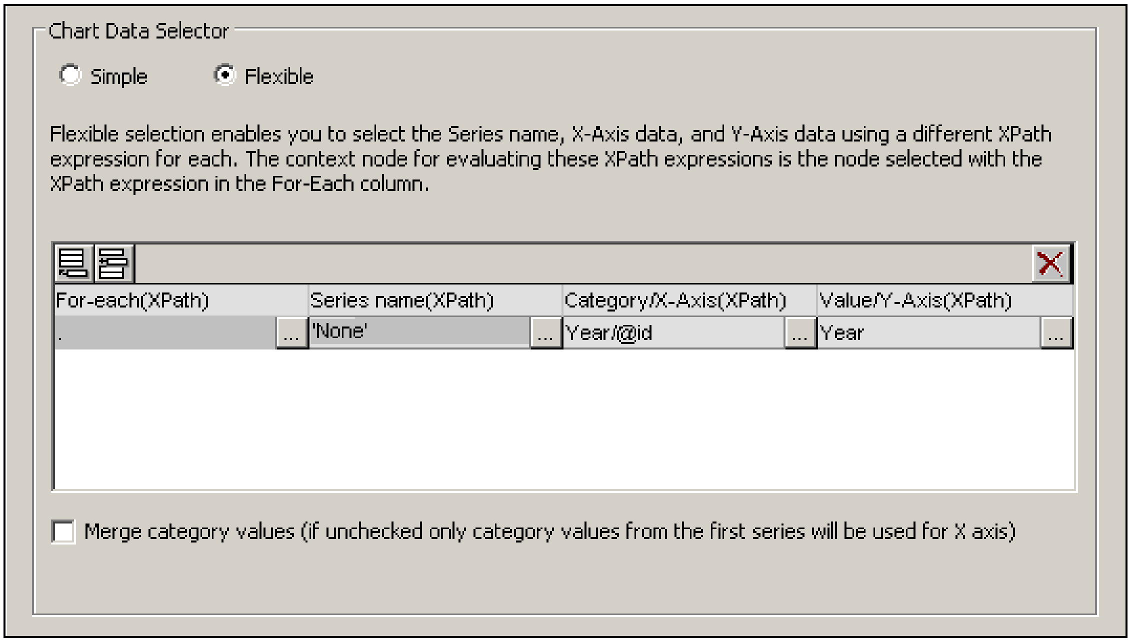


Chart Data Selector

○ Simple ● Flexible

Flexible selection enables you to select the Series name, X-Axis data, and Y-Axis data using a different XPath expression for each. The context node for evaluating these XPath expressions is the node selected with the XPath expression in the For-Each column.

For-each(XPath)	Series name(XPath)	Category/X-Axis(XPath)	Value/Y-Axis(XPath)
.	'None'	Year/@id	Year

□ Merge category values (if unchecked only category values from the first series will be used for X axis)

Note the following points:

- A series refers to a series of values plotted for a set of X-Axis (Category Axis) ticks. A second series would plot a second set of values on the same X-Axis ticks. For example, if the X-Axis represented the years 2008, 2009, and 2010, and the Y-Axis represented sales turnover, then Series 1 could represent America (sales in America for those three years) while Series 2 could represent Europe (sales in Europe for those three years). If this data were selected for a bar chart, then for each year (2008, 2009, 2010) on the X-Axis, there would be two bars (America and Europe), one for each series. In the case of pie charts and single-bar charts, only one series is possible. See the chart type table for more information about each chart type.

- Each row in the Chart Data Selector pane represents a series.
- The XPath expression in the For-Each column provides the context for the evaluation of each of the other three XPath expressions. The For-Each XPath expression is itself evaluated in the context of the node in the design within which it was inserted.

The following examples illustrate important points to consider when selecting data for the axes. They reference the XML document listed below.

⊟ XML file used in chart examples: YearlySales.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<Data xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xsi:noNamespaceSchemaLocation="YearlySales.xsd">
       <ChartType>Pie Chart 2D</ChartType>
       <Region id="Americas">
              <Year id="2005">30000</Year>
              <Year id="2006">90000</Year>
              <Year id="2007">120000</Year>
              <Year id="2008">180000</Year>
              <Year id="2009">140000</Year>
              <Year id="2010">100000</Year>
       </Region>
       <Region id="Europe">
              <Year id="2005">50000</Year>
              <Year id="2006">60000</Year>
              <Year id="2007">80000</Year>
              <Year id="2008">100000</Year>
              <Year id="2009">95000</Year>
              <Year id="2010">80000</Year>
       </Region>
       <Region id="Asia">
              <Year id="2005">10000</Year>
              <Year id="2006">25000</Year>
              <Year id="2007">70000</Year>
              <Year id="2008">110000</Year>
              <Year id="2009">125000</Year>
              <Year id="2010">150000</Year>
       </Region>
</Data>
```

One row, one series

Say we wish to generate a 2D bar chart for each `Region` element (there are three such elements: for the Americas, Europe, and Asia). Let us create the chart in the design within the `Region` element node. The context node for the For-Each XPath expressions in the Chart Data Selector will therefore be the `Region` element.

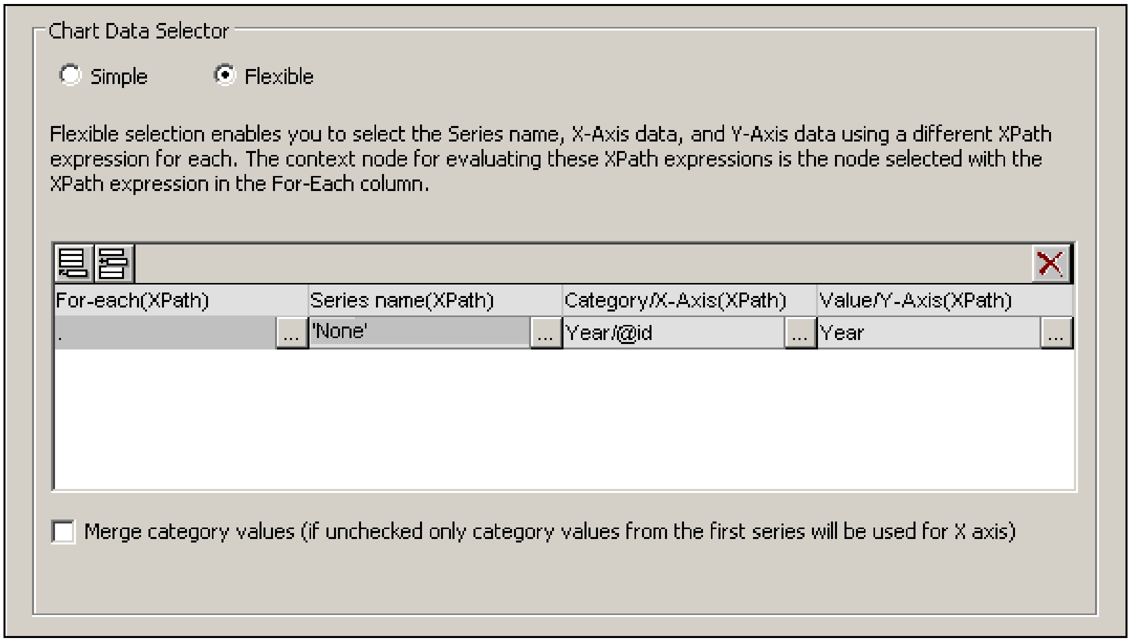


Chart Data Selector

○ Simple ● Flexible

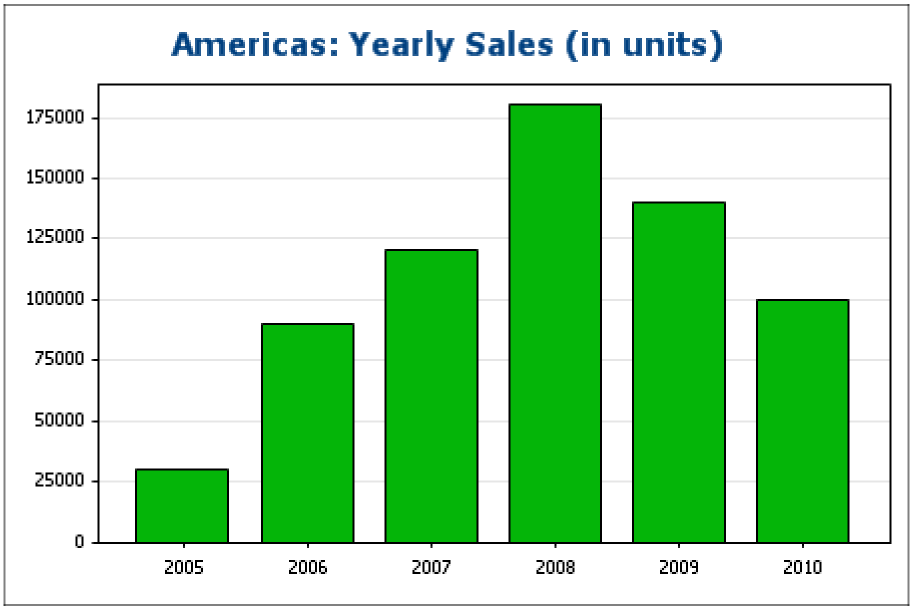
Flexible selection enables you to select the Series name, X-Axis data, and Y-Axis data using a different XPath expression for each. The context node for evaluating these XPath expressions is the node selected with the XPath expression in the For-Each column.

For-each(XPath)	Series name(XPath)	Category/X-Axis(XPath)	Value/Y-Axis(XPath)
.	'None'	Year/@id	Year

□ Merge category values (if unchecked only category values from the first series will be used for X axis)

In the chart data selection shown in the screenshot above, the For-Each expression returns the current node (which is the `Region` element), so the `Region` element will be the context node for all the other three XPath expressions (series, X-Axis, and Y-Axis). Since there is only one series in this chart, we do not need a series name and so we leave this column blank. The X-Axis selection returns six values. Six will therefore be the number of ticks on the X-Axis and the six items of the sequence will be the respective labels of the X-Axis ticks. The Y-Axis selection also returns six items, each of which is plotted on the Y-Axis for its corresponding X-Axis tick. Since the chart was created within the `Region` element, the XSLT engine will iterate over the three `Region` elements to produce three charts, one for each `Region` element. For each chart, that particular `Region` element's descendant nodes will be used.

The chart for the `Americas` region would look something like this in the output:



Three rows, three series, category values not merged

To create multiple series, additional rows can be added to the chart data selection, as shown in the screenshot below.

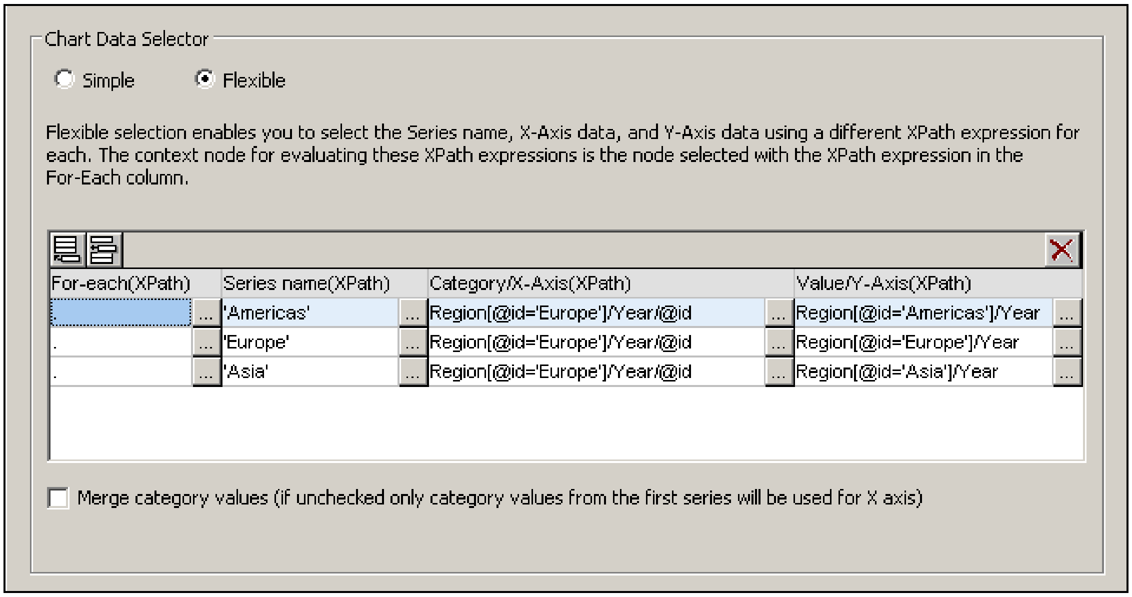


Chart Data Selector

○ Simple ● Flexible

Flexible selection enables you to select the Series name, X-Axis data, and Y-Axis data using a different XPath expression for each. The context node for evaluating these XPath expressions is the node selected with the XPath expression in the For-Each column.

For-each(XPath)	Series name(XPath)	Category/X-Axis(XPath)	Value/Y-Axis(XPath)
.	'Americas'	Region[@id='Europe']/Year/@id	Region[@id='Americas']/Year
.	'Europe'	Region[@id='Europe']/Year/@id	Region[@id='Europe']/Year
.	'Asia'	Region[@id='Europe']/Year/@id	Region[@id='Asia']/Year

□ Merge category values (if unchecked only category values from the first series will be used for X axis)

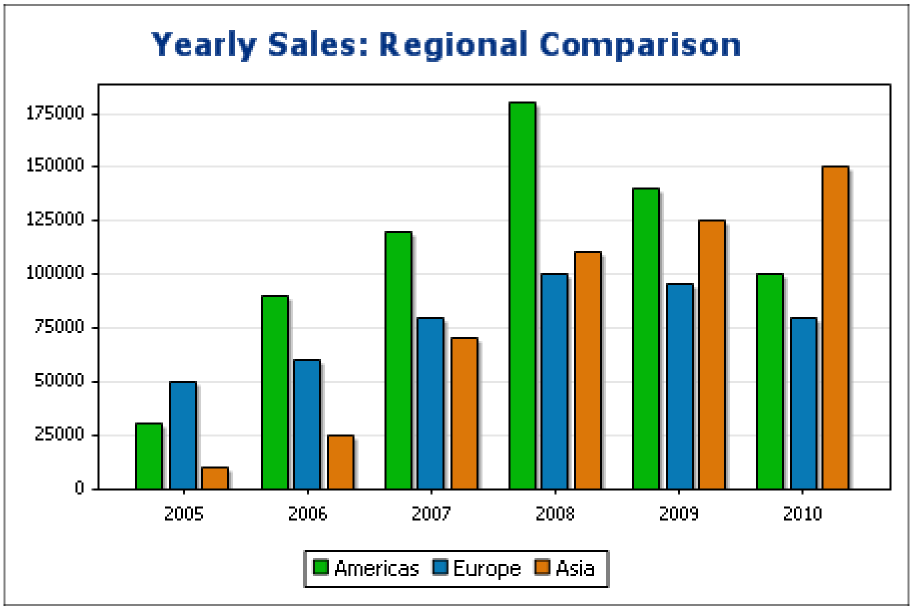
The important points to note about the data selection above are:

- Each row defines a series and all rows have the `Region` element as its context node (since the chart is created inside a `Region` node).
- The first row is set to define the Americas series and is given a string expression as its series name. The X-Axis values are selected using the `Year/@id` values of the Europe

region (it doesn't matter which region is selected since all have the same `Year/@id` values). The Y-Axis values of the first (Americas) series are selected for the Americas region using a predicate filter.

- The second and third series follow the same pattern as the first series. Note, however, that the X-Axis selection for each series is identical. But since the *Merge Category Values* check box is not checked, the second and third expressions are ignored. (Even if the values were merged, it would not make a difference because the values of each series are identical; only new distinct values would be added to the category values.)

The chart generated with the data selection above would look something like this:



Three rows, three series, category values merged

The data selection in this example (*see screenshot below*) is different from the previous example in two respects: (i) the X-Axis selection for the third series has an extra item (`2011`) added to the series, and (ii) the *Merge Category Values* check box has been checked, and (iii) the Y-Axis tick interval has been manually set to `20000`.

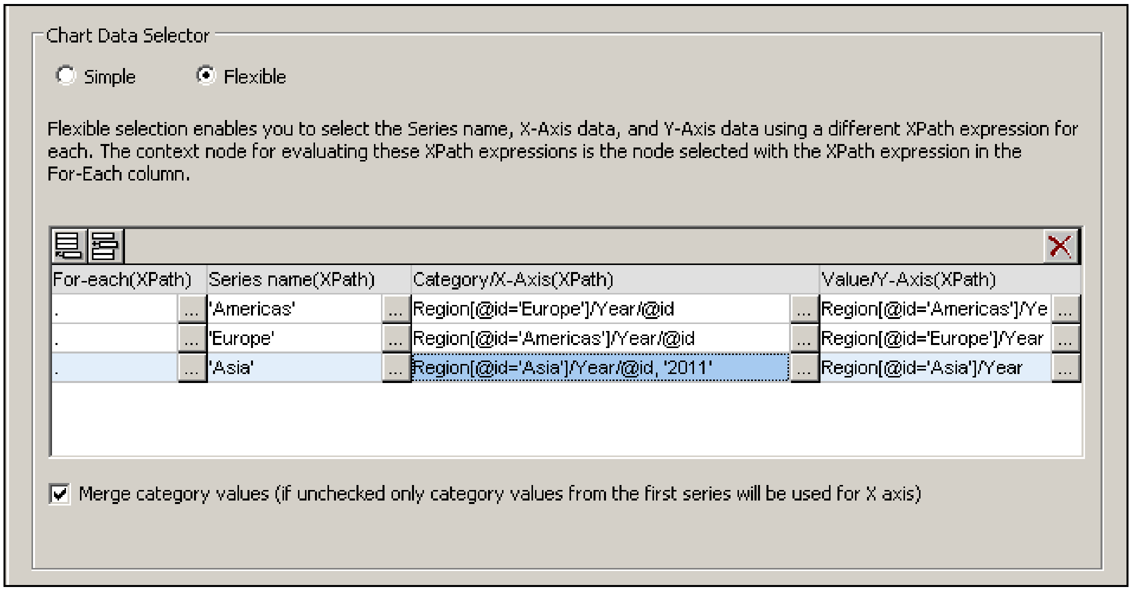


Chart Data Selector

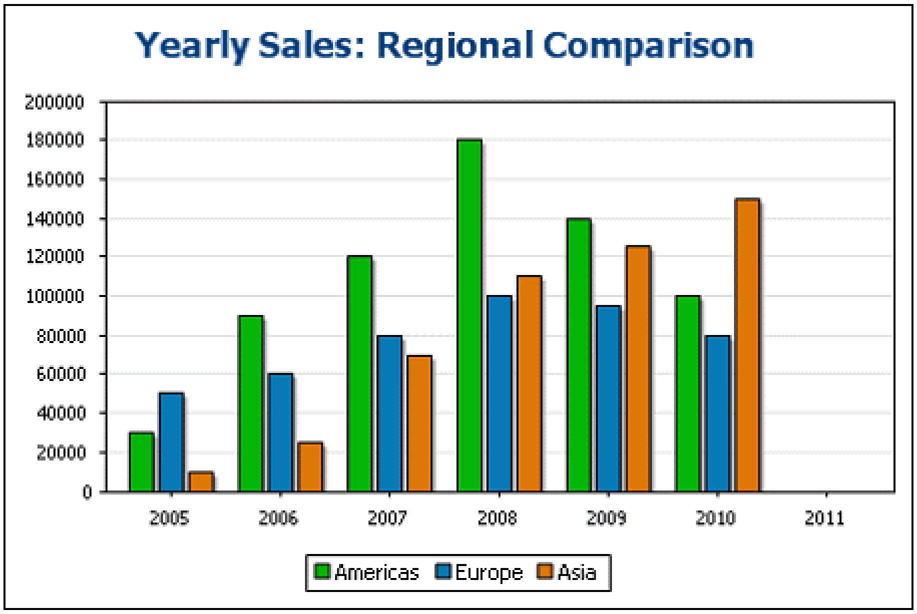
○ Simple ● Flexible

Flexible selection enables you to select the Series name, X-Axis data, and Y-Axis data using a different XPath expression for each. The context node for evaluating these XPath expressions is the node selected with the XPath expression in the For-Each column.

For-each(XPath)	Series name(XPath)	Category/X-Axis(XPath)	Value/Y-Axis(XPath)
.	'Americas'	Region[@id='Europe']/Year/@id	Region[@id='Americas']/Ye
.	'Europe'	Region[@id='Americas']/Year/@id	Region[@id='Europe']/Year
.	'Asia'	Region[@id='Asia']/Year/@id, '2011'	Region[@id='Asia']/Year

☑ Merge category values (if unchecked only category values from the first series will be used for X axis)

The effect of this change is to add one new item (2011) to the X-Axis result sequence. The chart would look something like this:



One row, three series

The chart in this example is created, not within the `Region` node as in the previous examples, but one node higher, in the `Data` node (see XML document above). Only one row is used for data selection, but it generates three series. This is because the XPath expression in the For-Each column returns a sequence of three items, thus implicitly creating three series.

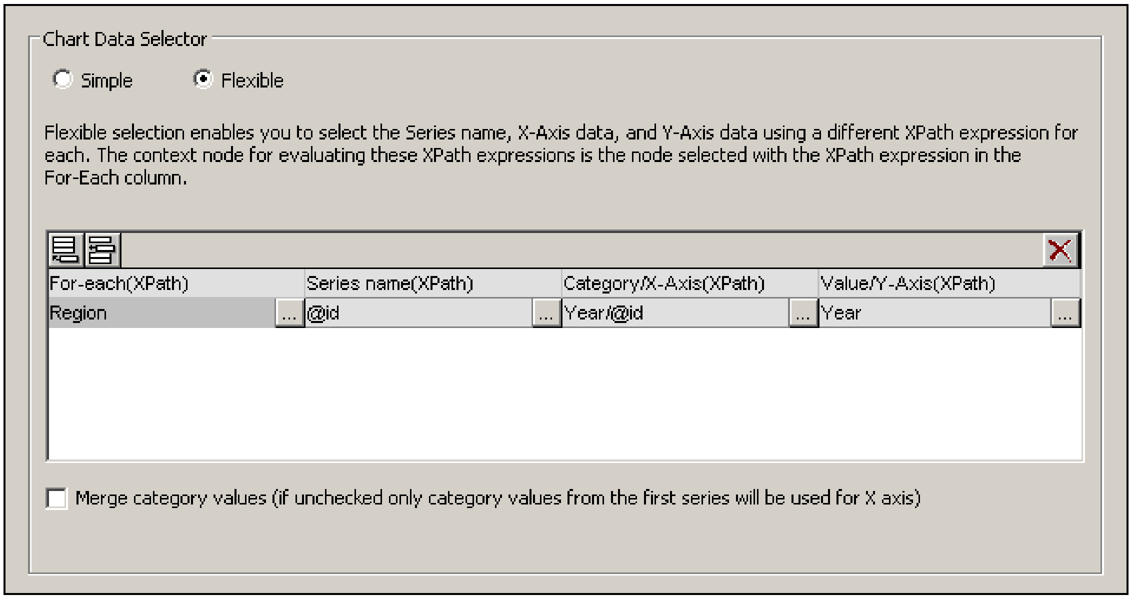


Chart Data Selector

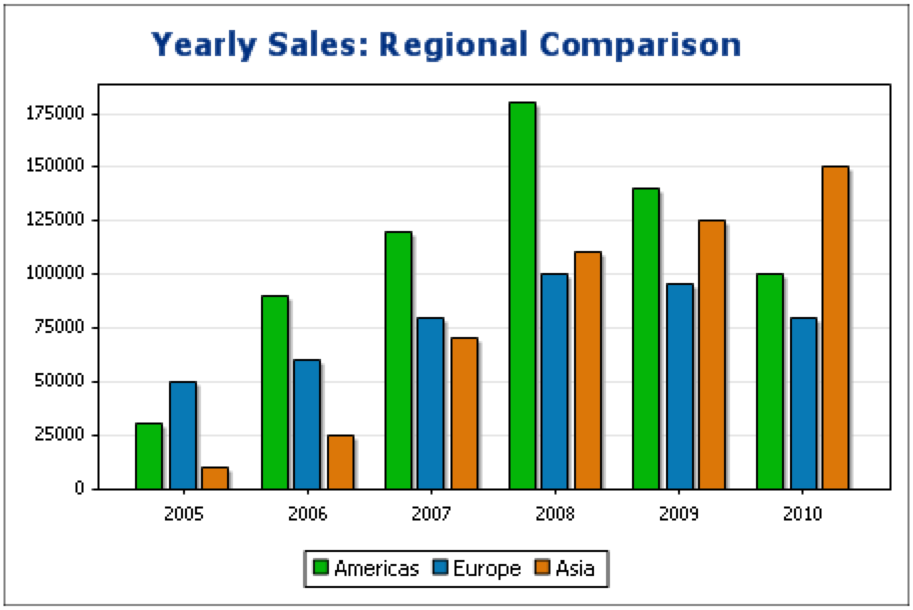
○ Simple ● Flexible

Flexible selection enables you to select the Series name, X-Axis data, and Y-Axis data using a different XPath expression for each. The context node for evaluating these XPath expressions is the node selected with the XPath expression in the For-Each column.

For-each(XPath)	Series name(XPath)	Category/X-Axis(XPath)	Value/Y-Axis(XPath)
Region	@id	Year/@id	Year

□ Merge category values (if unchecked only category values from the first series will be used for X axis)

For each series, the series name, X-Axis, and Y-Axis selections will be correspond to the different regions because each series has a different `Region` element as its context node. The chart for this data selection will look something like this:



Rules for chart data selection

The following points should be noted when using the Chart Data Selector to select data for the various chart axes:

1. The number of bars (or pie chart slices, etc) is equal to the number of items in the larger of the X-Axis or Y-Axis sequences of a single data row selection. So if the X-Axis (which

gives labels) has five items and the Y-Axis (which gives values) has six items, then six bars will be plotted with the last one being unlabelled. If the X-Axis has six items and the Y-Axis five items, then six bars will be plotted with the last one being labeled but having a value of zero.

2. The number of series is equal to the cumulative number of items in all the sequences returned by expressions in the For-Each column.
3. The name of a series is selected with the XPath expression of the Z-Axis (or Series Name Axis). If, in a data selection row, this XPath expression is left empty, then a no-name series is created. Also if the XPath expression returns a sequence with a lesser number of items than the number of series, then some series will have no name.

Chart Settings and Appearance

Chart settings are organized as follows:

- Basic Chart Settings, which enable you to select the type of chart and its title. Basic chart settings are defined in the Chart Settings pane of the Chart Configuration dialog (*see screenshot below*).
- Advanced Chart Settings,which enable you to change the appearance of a chart (its title, legend, colors, fonts, etc). Advanced settings are defined in the Change Appearance dialog. To access this dialog, click **All Settings** in the Chart Configuration dialog (*see screenshot below*).
- Dynamic XPath Settings, which can be accessed by clicking **Dynamic XPath Settings** in the Chart Settings pane (*see screenshot below*).

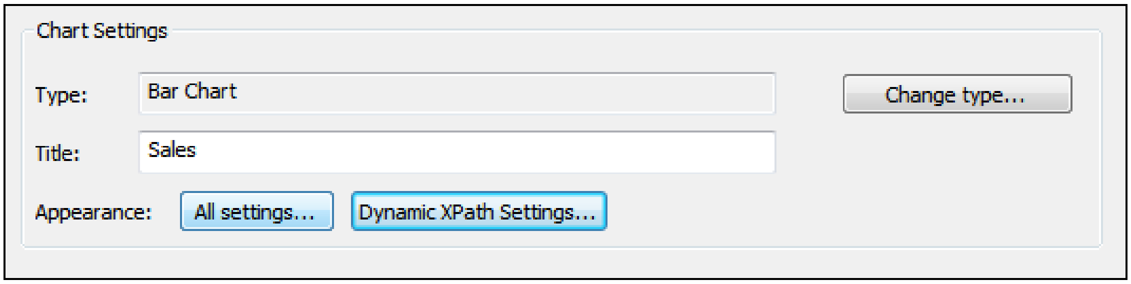


Chart Settings

Type: Bar Chart

Title: Sales

Basic Chart Settings

This section:

- Setting the chart type
- List of chart types
- Other basic settings

Setting the chart type

The most basic chart setting is the chart type. To select the chart type, click **Change Type** in the Chart Settings pane of the Chart Configuration dialog.

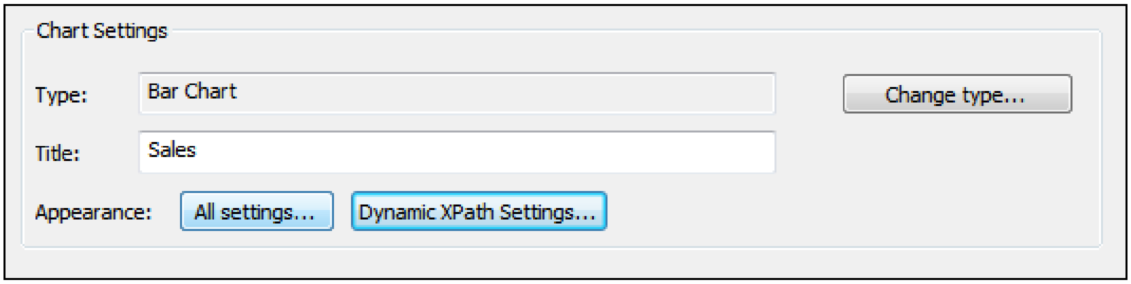
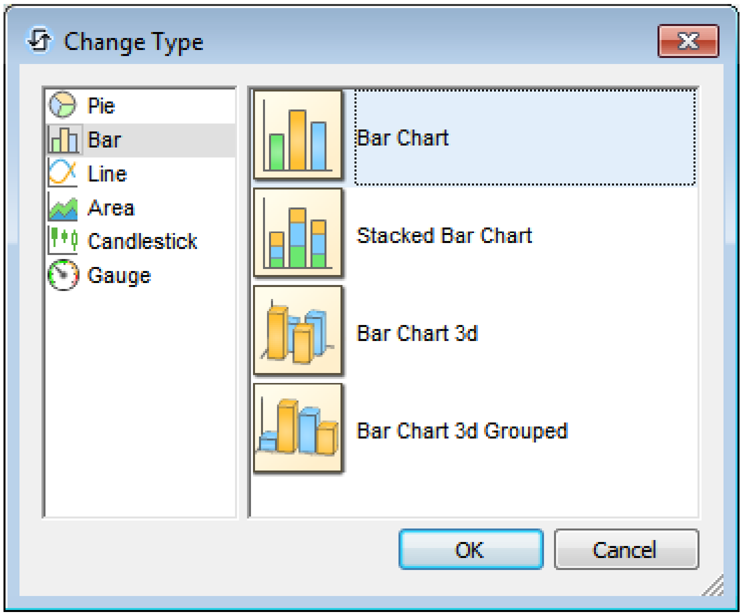


Chart Settings

Type: Bar Chart

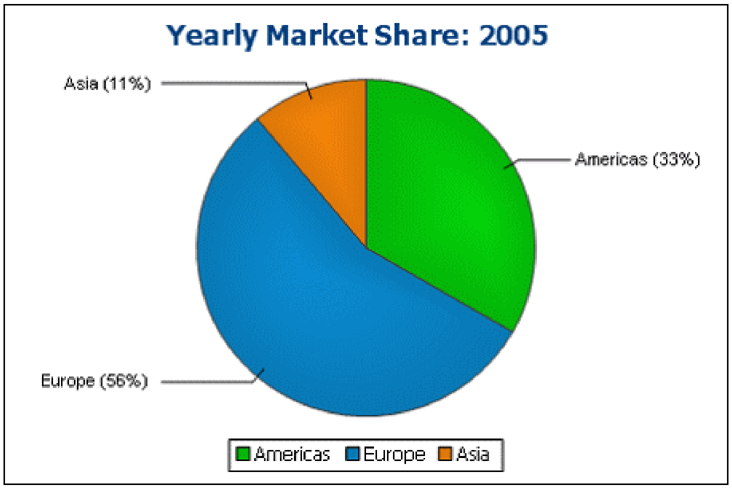
Title: Sales



The various types of charts that are available are listed below. In the Change Type dialog (*screenshot above*), select the chart type you want and click **OK**.

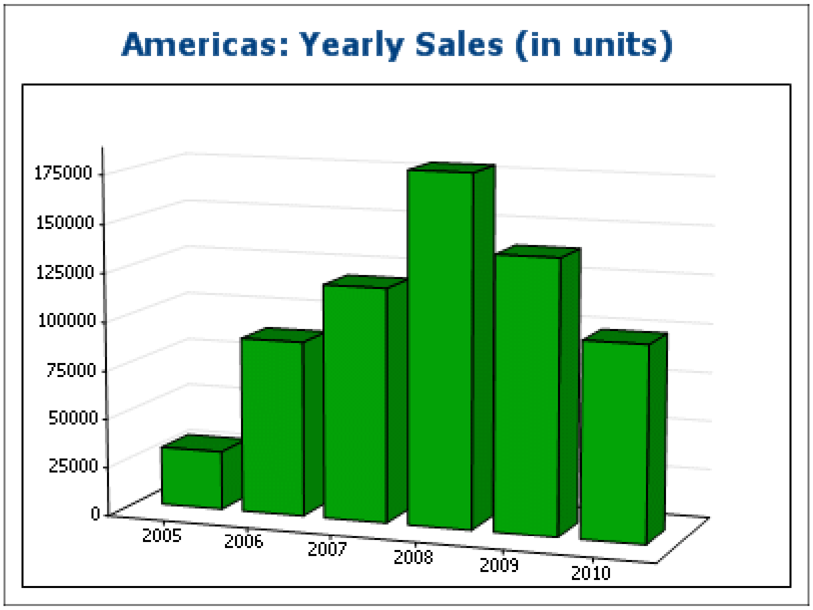
▼ *Pie charts*

In pie charts, one column/axis provides the values, another column/axis provides labels for these values. The labeling column/axis can take non-numeric values.

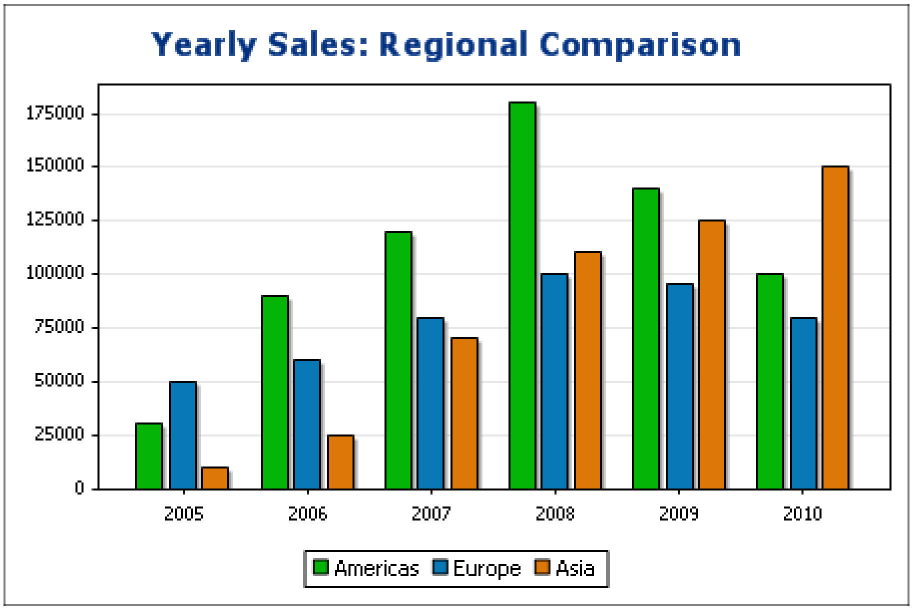


Bar charts

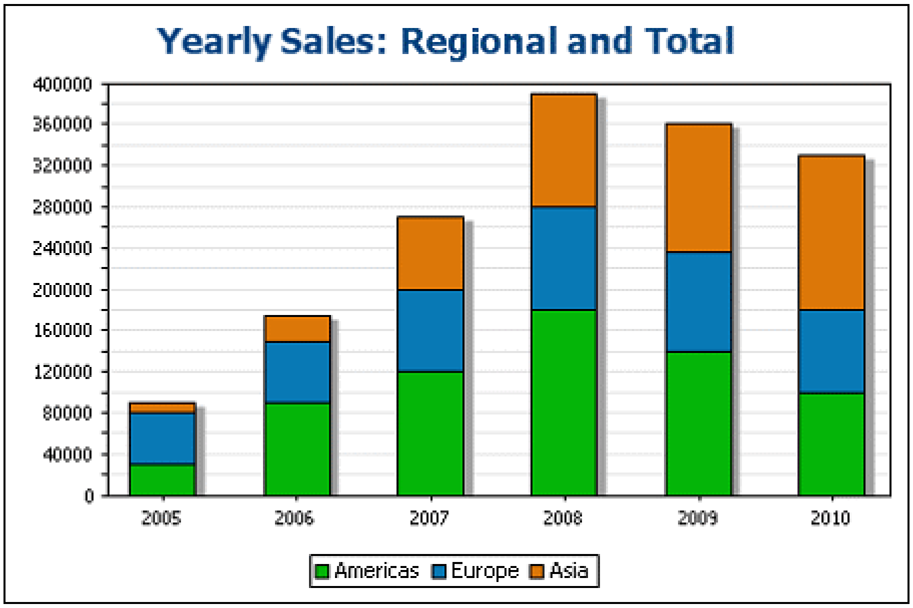
Bar charts can have two sets of values used along two axes (*below*).



They can also use three sets of values, as in the example below: (i) continent, (ii) year, (iii) sales volume. Bar charts can be displayed in 2D (*below*) or 3D (*above*).

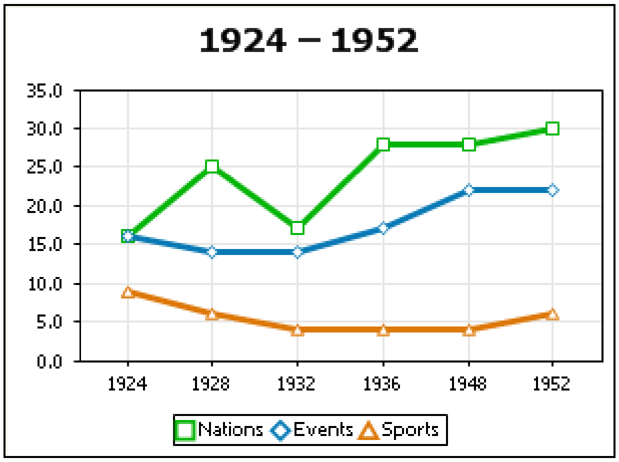


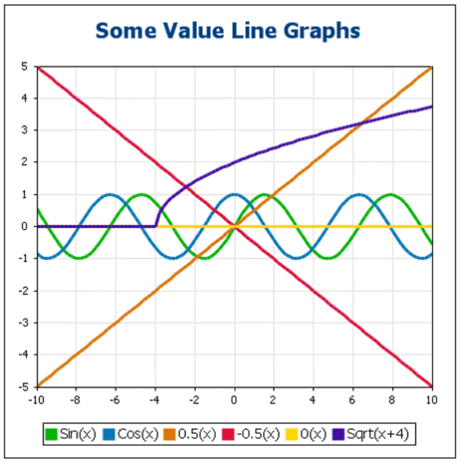
A three-axis bar chart can also be stacked if you need to show totals. Compare the stacked chart below with the chart above. The stacked chart shows the total of sales on all continents.



▼ *Line charts*

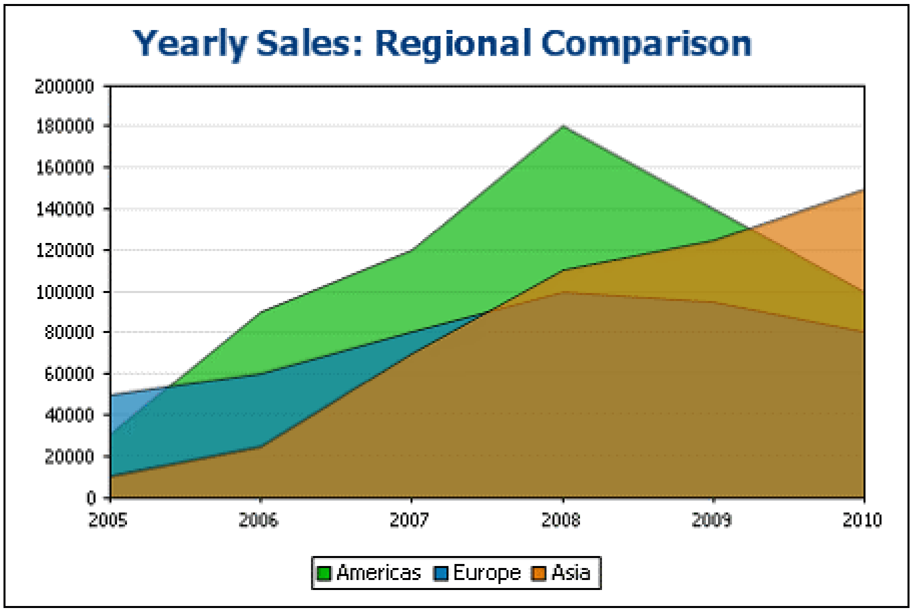
The difference between a line chart (*below left*) and a value line chart (*below right*) is that value line charts only take numerical values for the X-axis. If you need to display line charts with text values on the X-axis, use line charts.



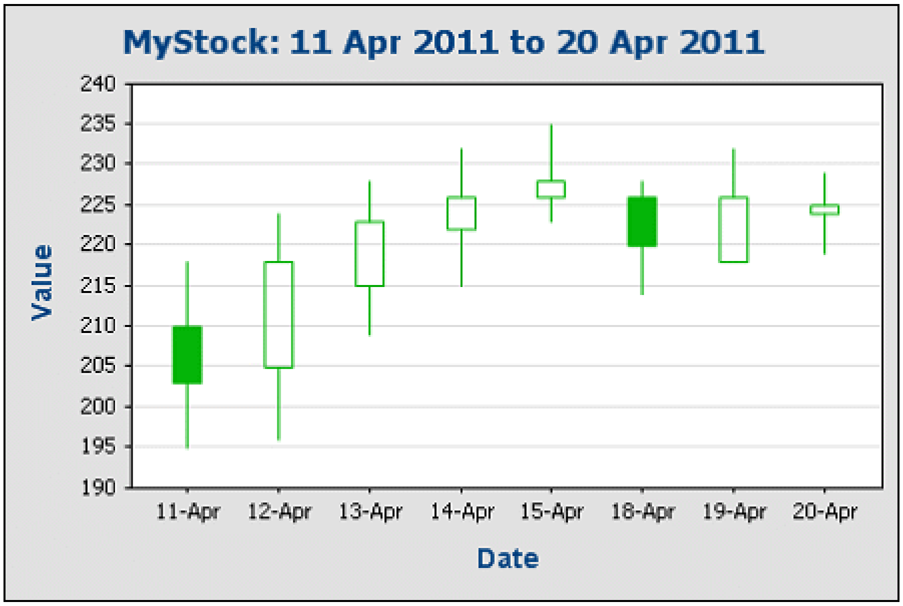


Area charts

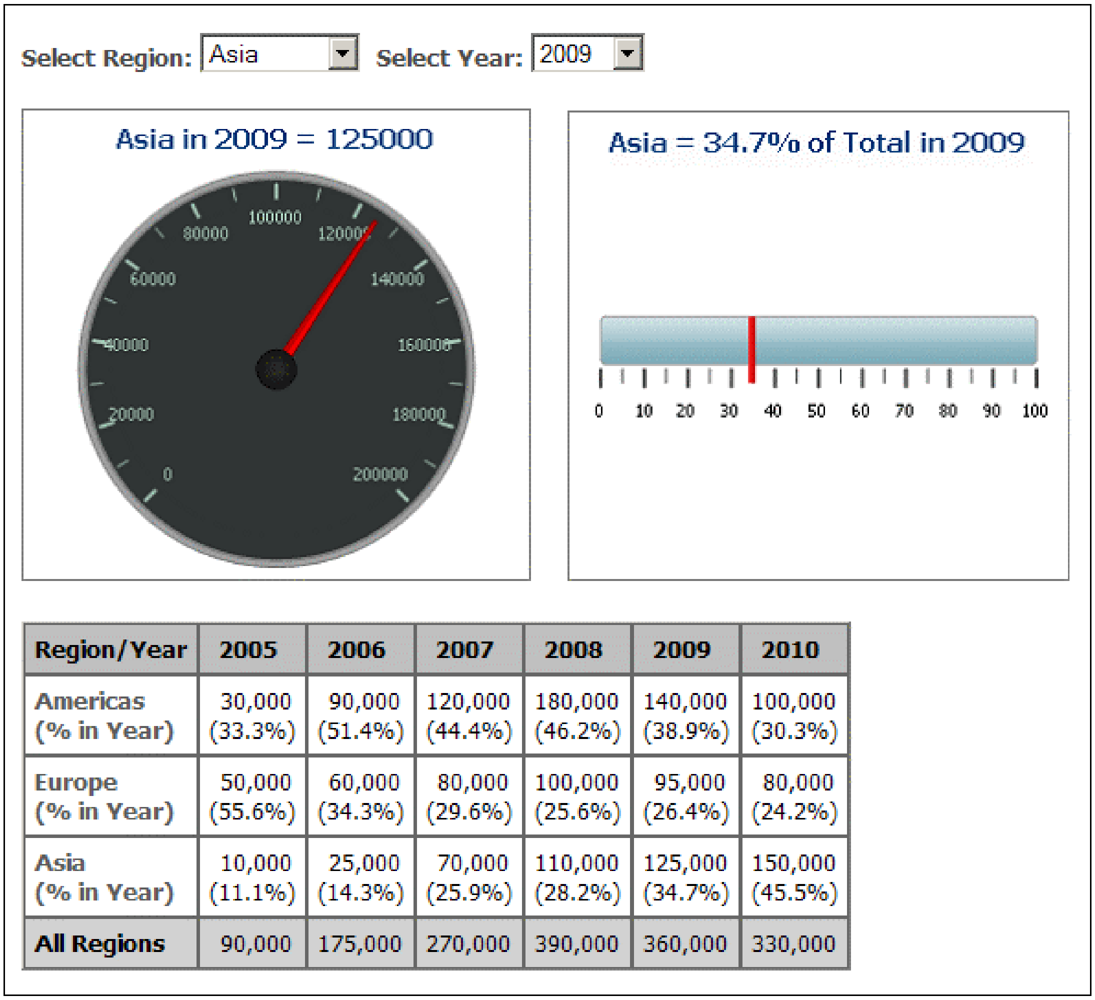
Area charts are a variation of line charts, in which the areas below the lines are also colored. Note that area charts can also be stacked (*see bar graphs above*).



- *Candlestick charts*
  A candlestick chart can be used to depict price movements of securities, commodities, currencies, etc over a period of time. The chart indicates not only how prices developed over time, but also the daily close, high, low, and (optionally) open. The Y-axis takes three or four series (close, high, low, and (optionally) open). The screenshot below shows a four-series candlestick chart.



- *Gauge charts*
  Gauge charts are used to illustrate a single value and show its relation to a minimum and a maximum value.



Select Region: Asia

Select Year: 2009

Asia in 2009 = 125000

Asia = 34.7% of Total in 2009

Region/Year	2005	2006	2007	2008	2009	2010
Americas (% in Year)	30,000 (33.3%)	90,000 (51.4%)	120,000 (44.4%)	180,000 (46.2%)	140,000 (38.9%)	100,000 (30.3%)
Europe (% in Year)	50,000 (55.6%)	60,000 (34.3%)	80,000 (29.6%)	100,000 (25.6%)	95,000 (26.4%)	80,000 (24.2%)
Asia (% in Year)	10,000 (11.1%)	25,000 (14.3%)	70,000 (25.9%)	110,000 (28.2%)	125,000 (34.7%)	150,000 (45.5%)
All Regions	90,000	175,000	270,000	390,000	360,000	330,000

Other basic settings

In the Chart Settings pane, you can also set the title of the chart (*see screenshot below*).

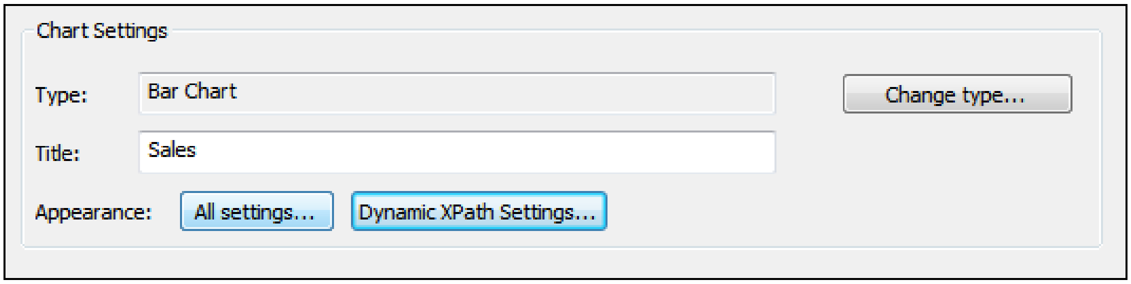


Chart Settings

Type: Bar Chart

Title: Sales

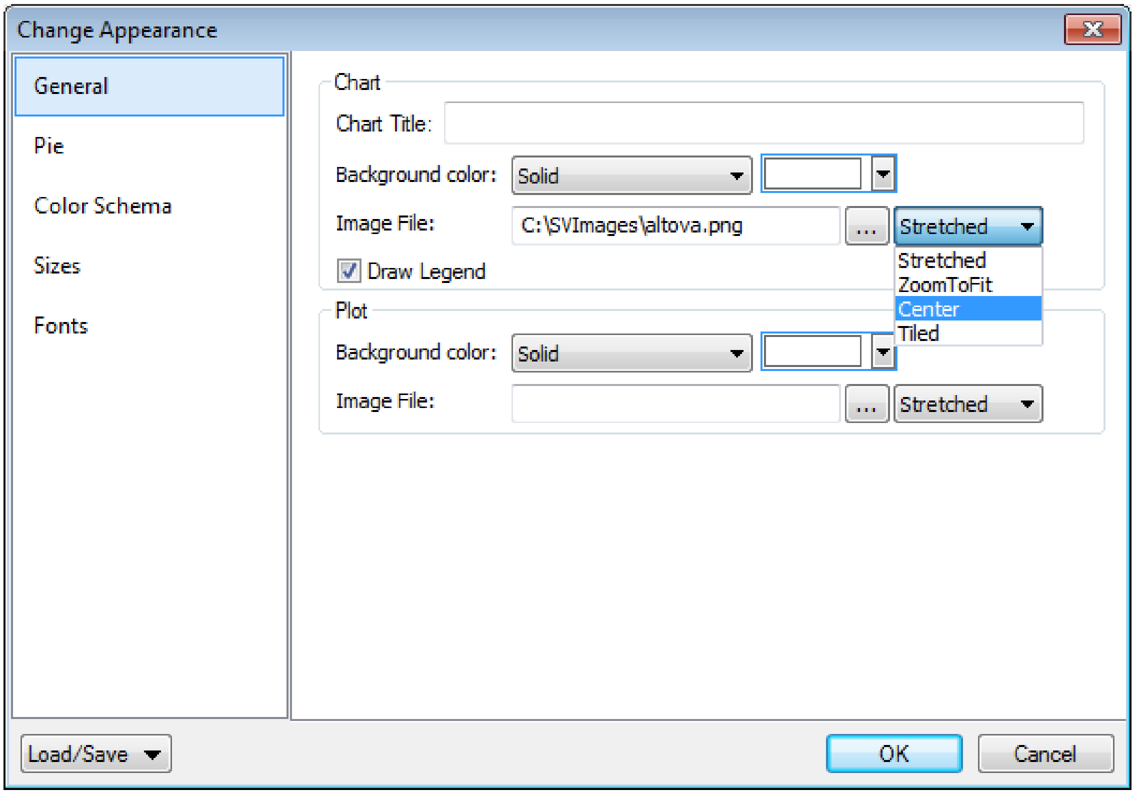
Advanced Chart Settings

This section:

- Accessing the advanced settings
- Overview of advanced settings
- Loading, saving, resetting chart settings

Accessing the advanced settings

To access a chart's advanced settings do the following: Click **All Settings** in the Chart Configuration dialog. This displays the Change Appearance dialog for that particular chart type (*the screenshot below shows the Change Appearance dialog of a pie chart*).



Chart

Chart Title:

Background color: Solid

Image File: C:\SVImages\altova.png ... Stretched

☑ Draw Legend

Plot

Background color: Solid

Image File: ... Stretched

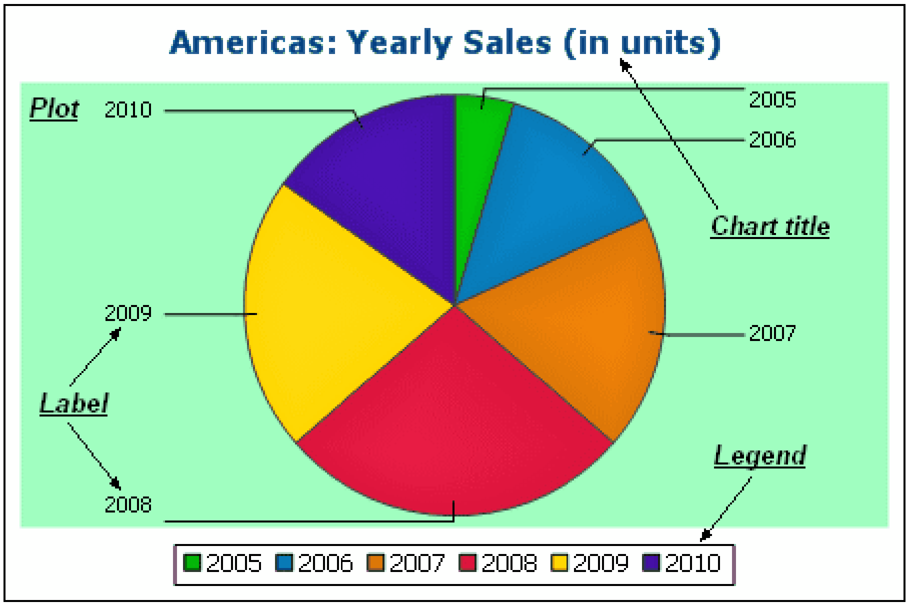
Overview of advanced settings

The advanced settings are organized into tabs that are common to all chart types and those that are specific to a single chart type.

Common chart settings

▾ *General*

The chart title (*see screenshot below*) is the same as the basic setting (*see above*) and can be edited as an advanced setting also. Other settings in this dialog are the background color of the chart and the plot. In the screenshot below, the plot has been given a pale green background color. An image file can also be set as the background image of the chart and/or the plot. This image can be stretched to cover the entire area of the chart or plot; zoomed to fit so that the zoom matches one of the two dimensions (of chart/plot); centered; or tiled. The legend is the key to the color codes in the chart, and it can be turned on or off.



- *Color scheme*
  Four predefined color schemes are available plus a user-defined color scheme. You can modify any of the color schemes by adding colors to and/or deleting colors from a scheme. The color scheme selected in this tab will be used in the chart.

- *Sizes*
  Sizes of various aspects of the chart can be set, either as pixels or as a percentage ratio.

- *Font*
  The font properties of the chart title and of legends and labels can be specified in this tab. Sizes can be set as a percentage of the chart size or absolutely as points.

- *Load/Save button*
  Settings can be saved to an XML file and can be loaded from an XML file having the correct structure. To see the structure, save the settings of a chart and then open the XML file. Clicking this button also gives you the option of resetting chart settings to the default.

Type-specific chart settings

- *Pie charts*

Settings for: (i) the angle from which the first slice should be drawn; (ii) the direction in which slices should be drawn; (iii) the outline color; (iv) whether the colors receive highlights (in 3D pie charts: whether dropshadows and transparency are used); (v) whether labels should be drawn; and (vi) whether values and percentages should be added to labels and how many decimal places should be added to the percentages.

▼ *Bar charts*
Settings for: *(General)* Drawing the X and Y axes exchanged generates a horizontal bar chart; *(Bar)* Bar outlines and dropshadows (dropshadows in 2D bar charts only); *(X-Axis)* Label and color of the x-axis, and vertical gridlines; *(Y-Axis)* Label and color of the y-axis, horizontal gridlines, the range of values to be displayed, and the tick marks on the y-axis; *(Z-Axis, 3D only)* Label and color of the z-axis; *(3D)* the vertical tilt, horizontal rotation, and the width of the view.

▼ *Line graphs*
Settings for: *(General)* Drawing the X and Y axes exchanged; *(Line)* including the plot points or not; *(X-Axis)* Label and color of the x-axis, and vertical gridlines; *(Y-Axis)* Label and color of the y-axis, horizontal gridlines, the range of values to be displayed, and the tick marks on the y-axis.

▼ *Gauge charts*
Settings for: (i) the angle at which the gauge starts and the angular sweep of the scale; (ii) the range of the values displayed; (iii) the interval and color of major and minor ticks; (iv) colors of the dial, the needle, and the border.

▼ *Area charts*
The transparency of areas can be set as a value from 0 (no transparency) to 255 (maximum transparency). In the case of non-stacked area charts transparency makes parts of areas that lie under other areas visible top the viewer. Outlines for the areas can also be specified.

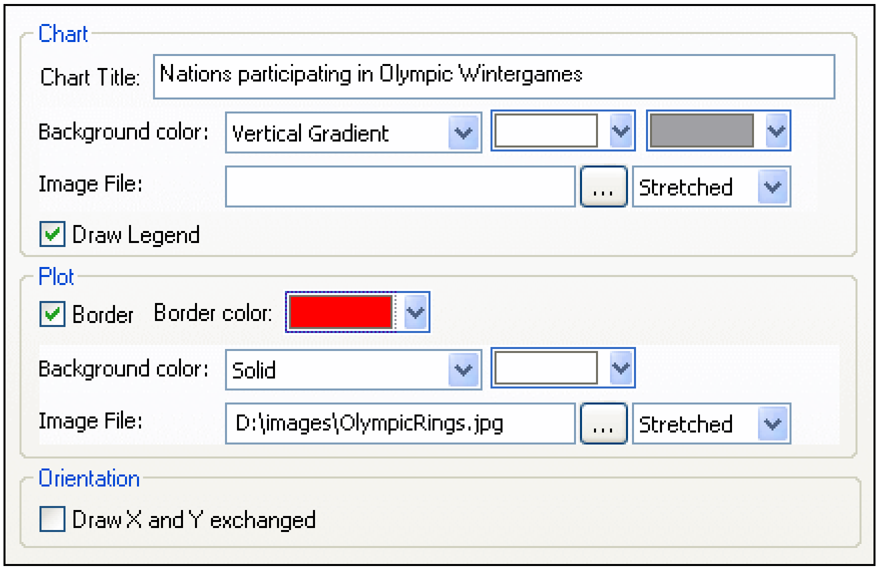
▼ *Candlestick charts*
The fill color can be specified for the two situations: (i) when the closing value is greater than the opening value, and (ii) when the opening value is greater than the closing value. In the latter case, the Series color is also available as an option. The Series color is specified in the Color Schema tab of the Change Appearance dialog.

Loading, saving, resetting chart settings
Chart settings that are different from the default settings can be saved in an XML file. These settings can subsequently loaded as the settings of a chart, which can help you save time and effort. The **Load/Save** button (see first screenshot in this section) provides the following options when clicked:

- **Set to default**: Rejects changes made to the settings, and restores the default settings to **all** settings sections.
- **Load from file**: Enables settings to be imported that have been previously saved in an XML file (*see next command*). The command displays the **Open** dialog, in which you enter the location of the required file.
- **Save to file**: Opens the **Save As** dialog box. You can specify an XML file in which to save the settings. This file lists those settings that are different from the default settings.

General

The General section of the **Change Appearance** dialog box lets you define the title of the chart, add or remove a legend, and define background pictures and colors and—for bar, line, area, and candlestick charts—orientation of the chart.



Chart

Chart Title: Nations participating in Olympic Wintergames

Background color: Vertical Gradient

Image File: Stretched

☑ Draw Legend

Plot

☑ Border Border color: 🟥

Background color: Solid

Image File: D:\images\OlympicRings.jpg Stretched

Orientation

□ Draw X and Y exchanged

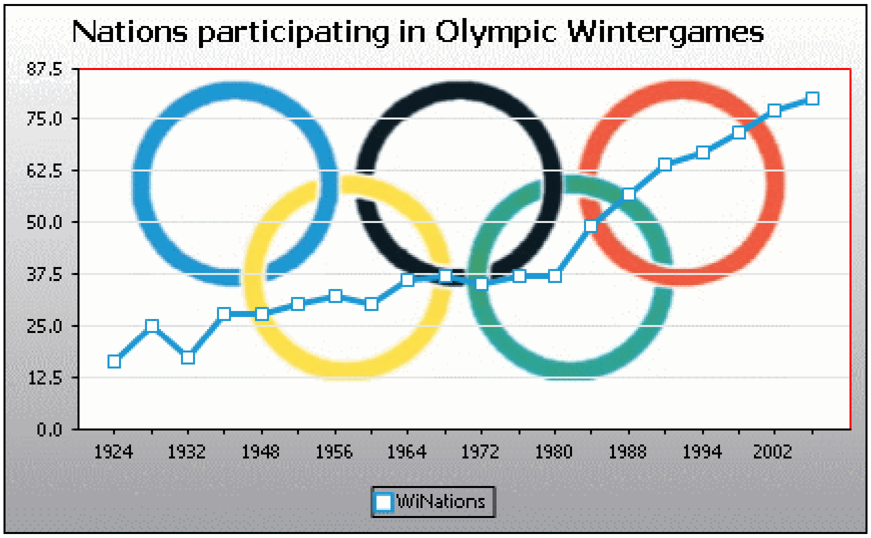
Chart
Enter a descriptive title for your chart into the `Chart Title` field and select a background color for the entire chart from the drop-down list. You can choose a solid background, vertical gradient, or horizontal gradient and define start and end colors for the gradient, if applicable. In addition, or instead of a colored background, you can also define a background image and choose one of the available display options from the drop-down list:

- Stretched: the image will be stretched to the height and width of the chart
- Zoom to Fit: the image will be fit into the frame of the chart and the aspect ratio of the image will be maintained
- Center: the image will be displayed in its original size in the center of the chart
- Tiled: if the image is smaller than the chart, duplicates of the image will be displayed to fill the background area

The `Draw Legend` check box is activated by default, clear the check box if you do not want to display a legend in your chart.

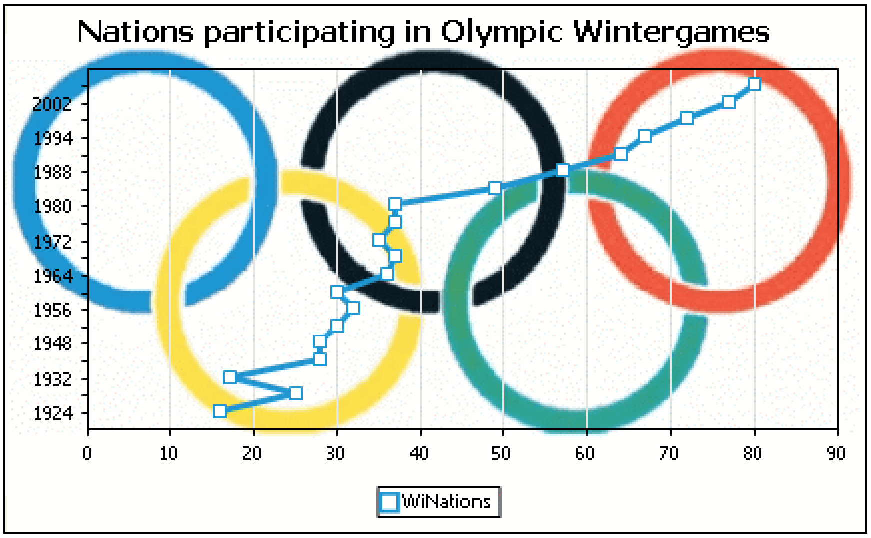
Plot

The Plot is the area where the actual data of the chart is displayed. You can draw a border around the plot and specify a different background color and/or image for the plot area. In the screenshot below, the background color of the chart has been changed to gray (vertical gradient) whereas the plot is still white, a red border has been drawn around it, and a background image has been added.



Orientation

If you have a small series of large values it may be convenient to swap the X and Y axis for a better illustration. Note that in the screenshot below also the background color of the plot has been set to "Transparent" and the background image has been applied to the chart.



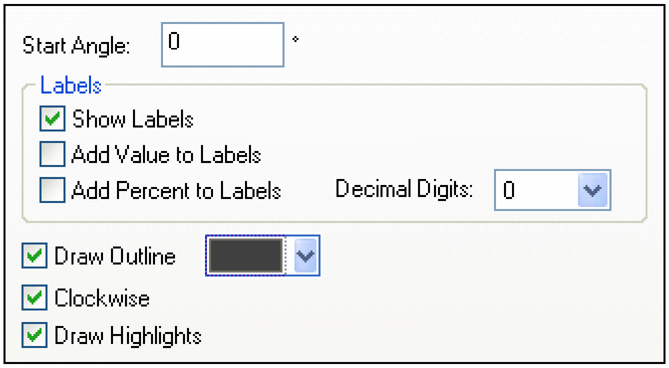
Note that this option is not available for pie and gauge charts.

Type-Related Features

For each of the chart types, and even for the various sub-types, the **Change Appearance** dialog box provides a section where you can define the type-related features of the chart.

Pie chart

Most settings are the same for the 2d and 3d versions. In 2d pie charts, you can additionally draw highlights.



Start Angle: 0 °

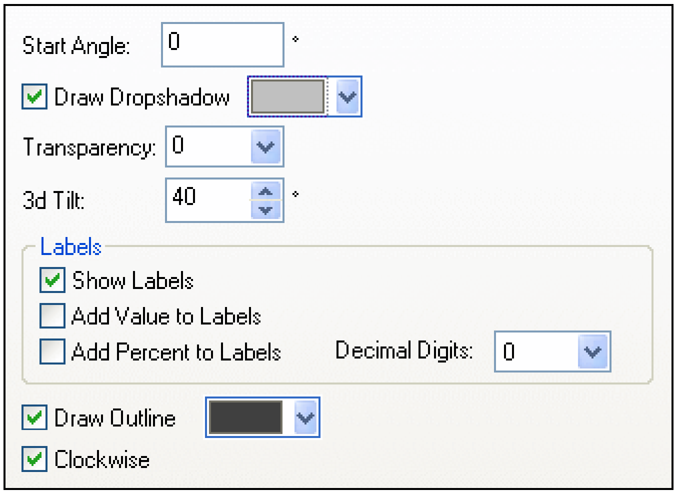
Labels
- ☑ Show Labels
- ☐ Add Value to Labels
- ☐ Add Percent to Labels Decimal Digits: 0

☑ Draw Outline

☑ Clockwise

☑ Draw Highlights

In 3d pie charts, you can display drop shadows, add transparency and define the 3d tilt.



Start Angle: 0 °

☑ Draw Dropshadow

Transparency: 0

3d Tilt: 40 °

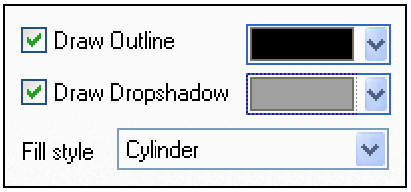
Labels
- ☑ Show Labels
- ☐ Add Value to Labels
- ☐ Add Percent to Labels Decimal Digits: 0

☑ Draw Outline

☑ Clockwise

The `Start Angle` value defines where the first row of the selected column will be displayed in the chart. An angle of 0 degrees corresponds to 12 o'clock on a watch.
You can show labels in addition to, or instead of, the legend, add values and/or percentage to the labels, and define for the percentage values the number of decimal digits to be displayed.
The color that you can select next to the `Draw Outline` check box is used for the optional border drawn around the chart and the individual pie segments. The `Clockwise` check box allows you to specify whether the rows should be listed clockwise or counter-clockwise.
In 3d pie charts, you can draw a drop shadow and define its color, add transparency to the chart, and define the 3d tilt. In 2d charts, the `Draw Highlights` option adds additional structure to the chart.

Bar chart



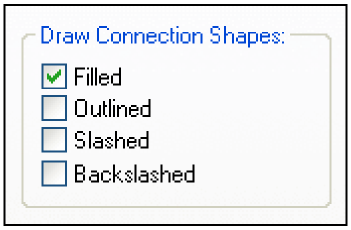
☑ Draw Outline

☑ Draw Dropshadow

Fill style: Cylinder

For bar charts, you can add an outline to the bars and define its color. In 2d bar charts, you can also draw a drop shadow and define its color (this option is not available for 3d bar charts). By default, the shape of the bars resembles a cylinder, however you can also choose "Vertical Gradient" or "Solid" from the `Fill style` drop-down list.

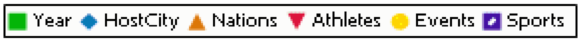
Line chart



Draw Connection Shapes:

☑ Filled
☐ Outlined
☐ Slashed
☐ Backslashed

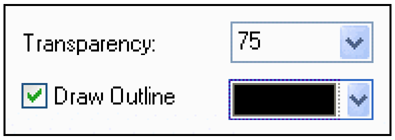
To draw connection shapes that mark the values in line charts, you need to activate at least one check box in the Draw Connection Shapes group box. You can use five different shapes to mark a series: square, rhomb, triangle, inverted triangle, and circle. If there are more than five series in your chart you can combine the connection shapes by selecting more than one option in the Draw Connection Shapes group box. In the screenshot below, both `Filled` and `Slashed` have been selected and the `Slashed` type is used for the sixth series and beyond.



🟩 Year ◆ HostCity ▲ Nations ▼ Athletes ● Events ▨ Sports

Connection shapes are available for both line charts and value line charts.

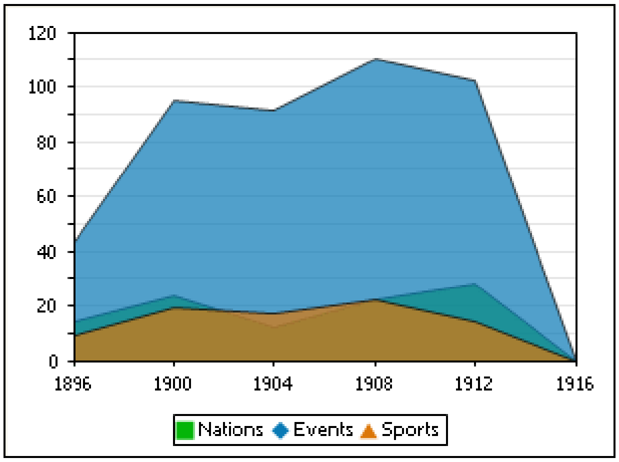
Area chart



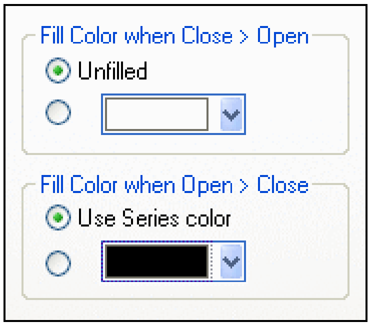
Transparency: 75

☑ Draw Outline

Among the properties that you can change for area charts is transparency; this way you can prevent that one series is hidden by another series in the chart. In addition, you can add an outline to the individual data areas and define its color (*see screenshot below*).



Candlestick chart



Fill Color when Close > Open

● Unfilled

○

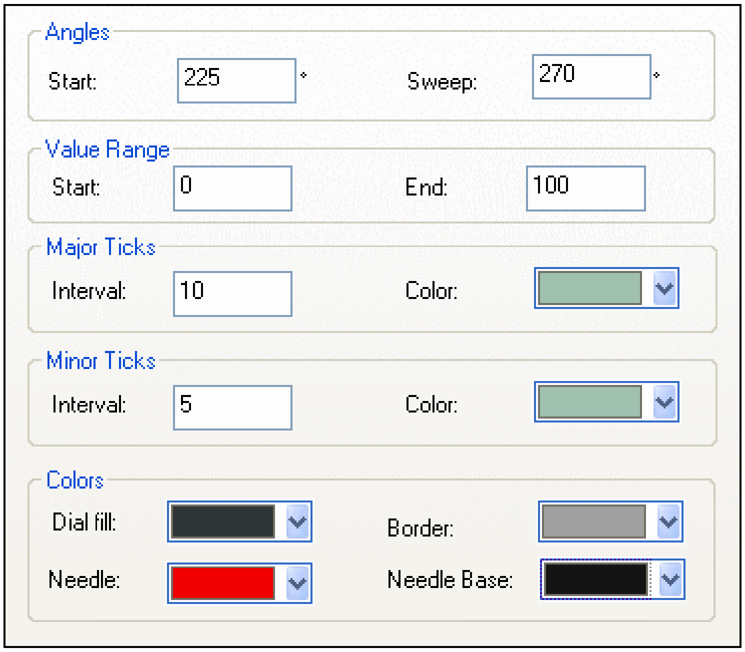
Fill Color when Open > Close

● Use Series color

○

If both opening and closing value are defined as series, you can choose the colors and whether or not the candle should be filled if the closing value is greater that the opening value.

Gauge chart



Angles

Start: 225 ° Sweep: 270 °

Value Range

Start: 0 End: 100

Major Ticks

Interval: 10 Color:

Minor Ticks

Interval: 5 Color:

Colors

Dial fill: Border:

Needle: 🟥 Needle Base: ⬛

The `Start` value in the Angles group box defines the position of the 0 mark and the `Sweep` value is the angle that is used for display. In the Value Range group box you can define the minimum and maximum values to be displayed. Tick marks are displayed with (major ticks) or without (minor ticks) the corresponding value; you can define separate colors for them. In the Colors group box you can define colors for the dial fill, needle, needle base (hides the first part of the needle in the center of the chart), and the border that surrounds the chart.

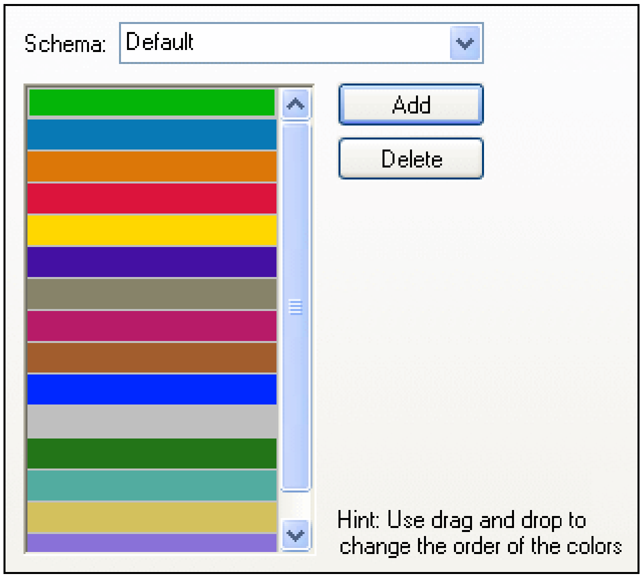
Colors

Depending on the chart type you have selected, MobileTogether Designer provides two different sections for the definition of colors to be used in charts:

- Color Schema for pie, bar, line, area, and candlestick charts
- Color Range for gauge charts

Color Schema

The Color Schema section of the **Change Appearance** dialog box provides four predefined color schemas (i.e., default, grayscale, colorful, and pastel) that can be customized; in addition you can also define your own color schema from scratch.

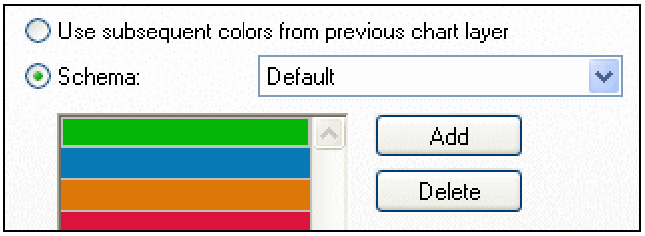


Schema: Default

Hint: Use drag and drop to change the order of the colors

The top color will be used for the first series, then the second color and so on. You can change the order or the colors by selecting a color and dragging it to its new positions with the mouse. To add a new or delete an unwanted color, click the corresponding button. In candlestick charts, only the first color will be used.

If you have appended one or several layers of overlay charts to a Charts window, the Color Schema section of the **Change Appearance** dialog box contains the additional radio button `Use subsequent colors from previous chart layer` which is activated by default.



○ Use subsequent colors from previous chart layer

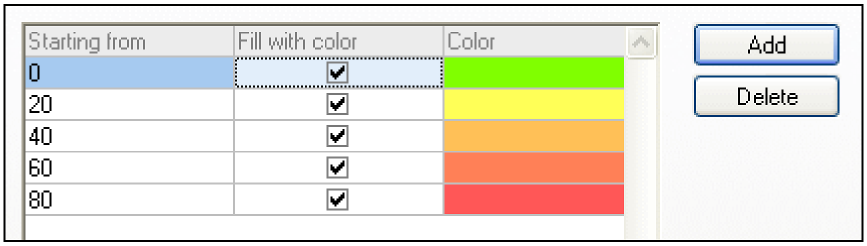
● Schema: Default

When the radio button is activated, the color schema from the previous layer will be used and you cannot choose a separate color schema for the overlay. The series of the active layer will be displayed using subsequent colors from the color schema of the previous layer. This way, all series of the Charts window have different colors and can therefore be distinguished more easily.

You can break this link on any additional layer that you add and choose a different color schema that then can also be re-used in subsequent layers.

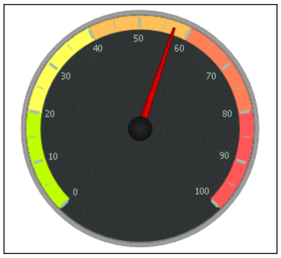
Color Range

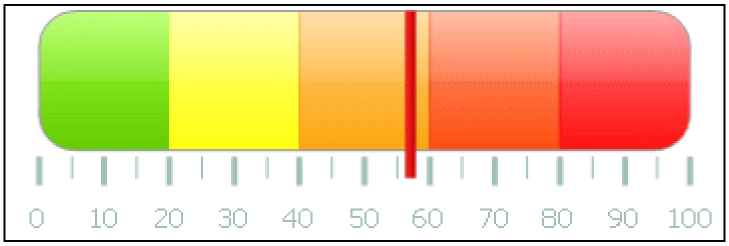
In gauge charts, you can customize the appearance of the gauge by applying colors to certain value ranges.



Starting from	Fill with color	Color
0	☑	🟩
20	☑	🟨
40	☑	🟧
60	☑	🟧
80	☑	🟥

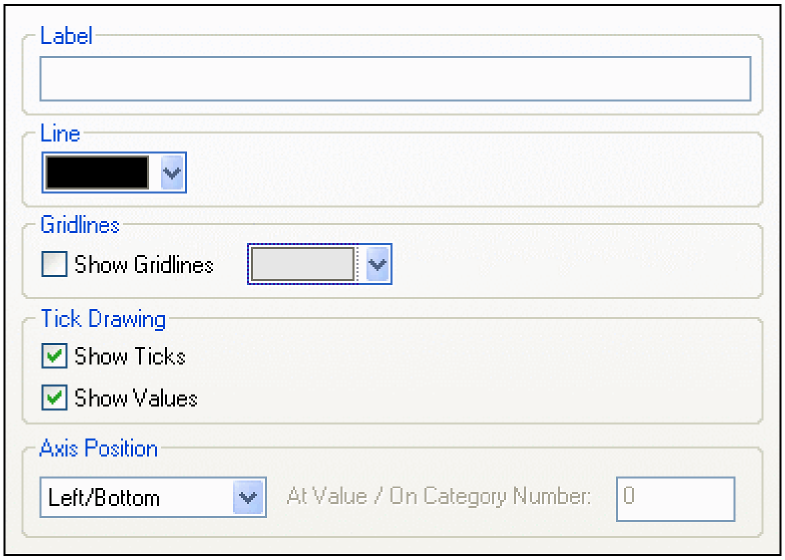
The definition shown in the screenshot above will appear in the gauge charts as follows:





X-Axis

In the X-axis section of the **Change Appearance** dialog box, you can enter a label for the axis, and define colors for the axis and the grid lines (if displayed). You can also define whether or not you want to display tick marks and axis values. This section is the same for all bar, line, area, and candlestick charts.



Label

Line

Gridlines

☐ Show Gridlines

Tick Drawing

☑ Show Ticks

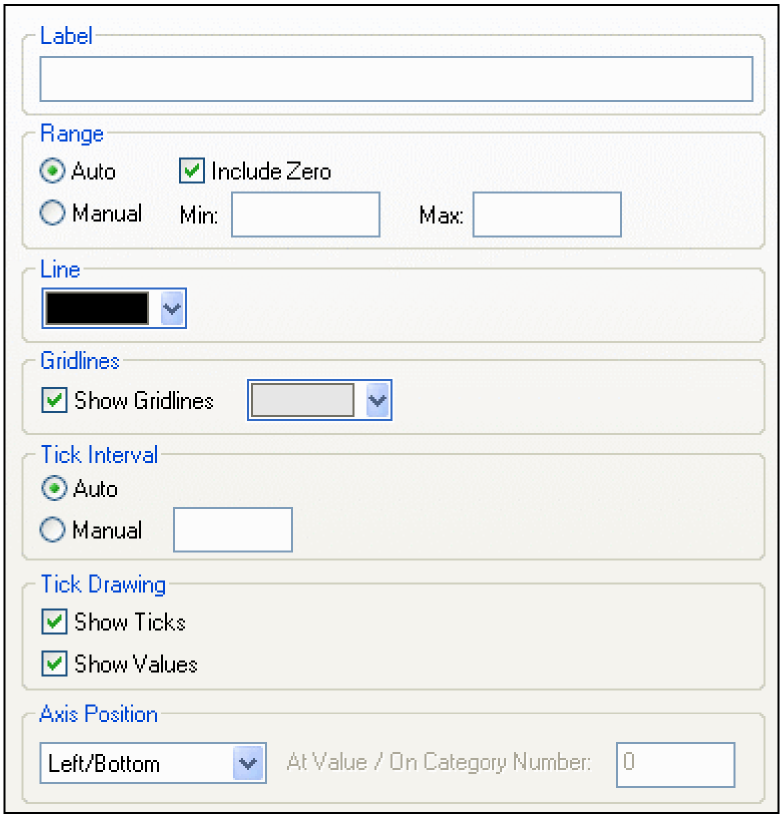
☑ Show Values

Axis Position

Left/Bottom

At Value / On Category Number: 0

In Value Line Charts however, you can also define the value range, and define at what interval tick marks should be displayed.



Label

Range

● Auto ☑ Include Zero

○ Manual Min: Max:

Line

Gridlines

☑ Show Gridlines

Tick Interval

● Auto

○ Manual

Tick Drawing

☑ Show Ticks

☑ Show Values

Axis Position

Left/Bottom At Value / On Category Number: 0

Label

The text entered into the `Label` field will be printed below the axis as a description of the X-axis.

Range

By default, the `Auto` radio button is selected in the Range group box. If you want to display a fragment of the chart in greater detail, activate the `Manual` radio button and enter minimum and maximum values into the respective fields.

If the column that is used for the X-axis does not include zero, you can deactivate the `Include zero` check box and the X-axis will start with the minimum value that is available in the series.

Line

The axis is displayed in the color that you choose from the `Line` drop-down-list. You can use one of the preselected colors, or click the **Other color...** button to choose a standard color or define a custom color. Click the **Select...** button on the Custom tab and use the pipette to pick a color that is displayed somewhere on your screen.

Grid lines

If the `Show Grid lines` check box is activated, you can choose a color from the corresponding drop-down list box.

Tick Interval

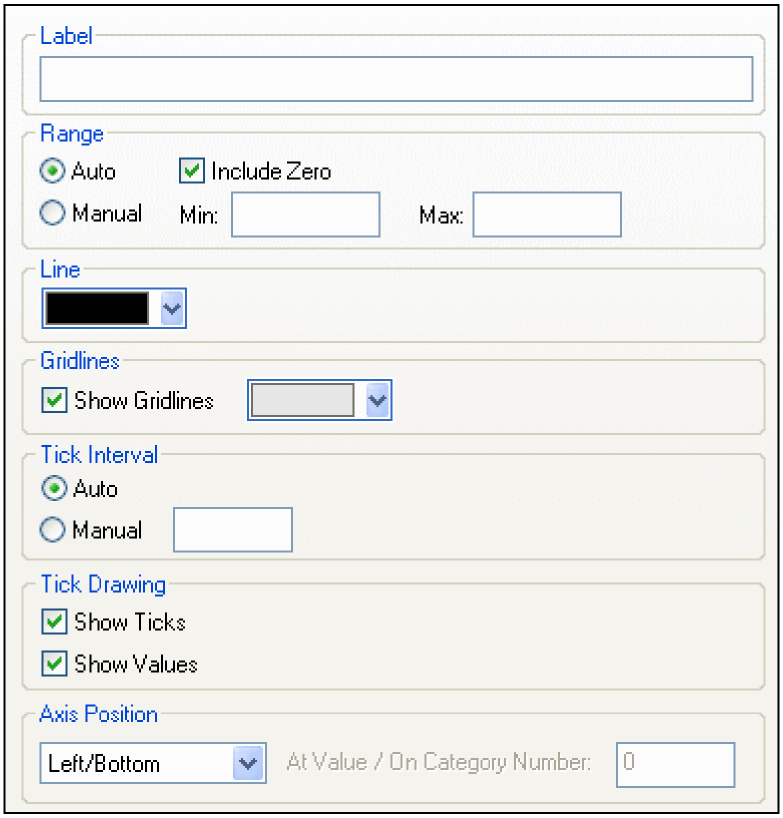
If you are not satisfied with the default tick marks, you can activate the `Manual` radio button in the Tick Interval group box and enter the difference between the individual tick marks into the corresponding field.

Tick Drawing
You can switch the display of tick marks on the axis and/or axis values on or off.

Axis Position
From the drop-down list, you can choose the position where the axis is to be displayed. When selecting "At Value / On Category Number", you can also position the axis anywhere within the plot.

Y-Axis

In the Y-axis section of the **Change Appearance** dialog box, you can enter a label for the axis, define colors for the axis and the grid lines (if displayed), define the value range, and decide if and where tick marks should be displayed and whether or not you want to show the axis values. This section is the same for all bar and line charts.



Label

Range
● Auto ☑ Include Zero
○ Manual Min: Max:

Line

Gridlines
☑ Show Gridlines

Tick Interval
● Auto
○ Manual

Tick Drawing
☑ Show Ticks
☑ Show Values

Axis Position
Left/Bottom At Value / On Category Number: 0

Label
The text entered into the `Label` field will be printed to the left of the axis as a description of the Y-axis.

Range
By default, the `Auto` radio button is selected in the Range group box. If you want to display a fragment of the chart in greater detail, activate the `Manual` radio button and enter minimum and maximum values into the respective fields.
If the column that is used for the X-axis does not include zero, you can deactivate the `Include zero` check box and the X-axis will start with the minimum value that is available in the series.

Line
The axis is displayed in the color that you choose from the `Line` drop-down-list. You can use one of the preselected colors, or click the **Other color...** button to choose a standard color or define a custom color. Click the **Select...** button on the Custom tab and use the pipette to pick a color that is displayed somewhere on your screen.

Grid lines
If the `Show Grid lines` check box is activated, you can choose a color from the corresponding drop-down list box.

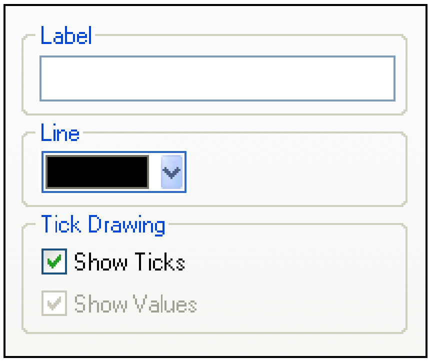
Tick Interval
If you are not satisfied with the default tick marks, you can activate the `Manual` radio button in the Tick Interval group box and enter the difference between the individual tick marks into the corresponding field.

Tick Drawing
You can switch the display of tick marks on the axis and/or axis values on or off.

Axis Position
From the drop-down list, you can choose the position where the axis is to be displayed. When selecting "At Value / On Category Number", you can also position the axis anywhere within the plot.

Z-Axis

In the Z-axis section of the **Change Appearance** dialog box, you can enter a label for the axis, define colors for the axis, and decide whether or not you want to show tick marks on the axis. This section is the same for all 3d bar charts (Bar Chart 3d and Bar Chart 3d Grouped).



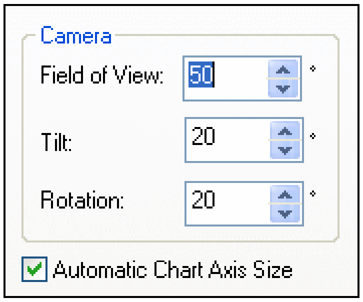
Label
The text entered into the `Label` field will be printed to the right of the axis as a description of the Z-axis.

Line
The axis is displayed in the color that you choose from the `Line` drop-down-list. You can use one of the preselected colors, or click the **Other color...** button to choose a standard color or define a custom color. Click the **Select...** button on the Custom tab and use the pipette to pick a color that is displayed somewhere on your screen.

Tick Drawing
You can switch the display of tick marks on the axis on or off.

3D Angles

In 3d bar charts you can customize the 3d appearance of the chart in the 3d Angles section of the **Change Appearance** dialog box.



Camera

Field of View: 50 °

Tilt: 20 °

Rotation: 20 °

☑ Automatic Chart Axis Size

The `Tilt` value determines the rotation around the X-axis, whereas the `Rotation` value defines the rotation around the Y-axis. You can automatically adapt the size of the chart axis to the Chart window width by activating the corresponding check box.

If the `Automatic Chart Axis Size` check box is activated, MobileTogether Designer will automatically calculate the optimum size of the X-axis as well as the Y-axis for the current Chart window size. The width and height of the chart will change dynamically when you resize the Chart window.

Sizes

In the Sizes section of the **Change Appearance** dialog box, you can define different margins as well as the size of axis and gauge ticks. Note that not all the properties listed below are available for all chart types.

General	
Outside margin	Space between the plot and the edge of the Chart window.
Title to Plot	Space between the chart title and the upper edge of the plot.
Legend to Plot	Space between the lower edge of the plot and the legend.
Pie	
Plot to Label	In pie charts, the space between the most left and right edge of the pie and its labels.
Pie Height	In 3d pie charts, the height of the pie.
Pie Drop Shadow	In 3d pie charts, the length of the shadow (if it is activated in the Pie section).
X-Axis	
X-Axis to Axis Label	In bar and line charts, the space between the X-axis and its label.
X-Axis to Plot	In 2d bar charts and line charts, the space between the X-axis and the plot.
X-Axis Tick Size	In bar and line charts, the length of the ticks on the X-axis.
Y-Axis	
Y-Axis to Axis Label	In bar and line charts, the space between the Y-axis and its label.
Y-Axis to Plot	In 2d bar and line charts, the space between the Y-axis and the plot.
Y-Axis Tick Size	In bar and line charts, the length of the ticks on the Y-axis.
Z-Axis	
Z-Axis to Axis Label	In 3d bar charts, the space between the Z-axis and its label.
Z-Axis Tick Size	In 3d bar charts, the length of the ticks on the Z-axis.
Line Drawing	

Connection Shape Size

In line charts, the size of the squares that mark the values in the chart.

3d Axis Sizes

Manual X-Axis Size of Base

In 3d bar charts, defines the relation between the length of the X-axis and the Chart window size. Please note that the `Automatic Chart Axis Size` check box in the 3d Angles section must be deactivated, otherwise the size will still be calculated automatically.

Manual Y-Axis Size of Base

In 3d bar charts, defines the relation between the length of the Y-axis and the Chart window size. Please note that the `Automatic Chart Axis Size` check box in the 3d section must be deactivated, otherwise the size will still be calculated automatically.

Z-Axis Series Margin

In 3d bar charts, the distance on the Z-axis between the individual series.

Gauge

Border Width

In round gauge charts, the width of the border around the gauge.

Gauge Ticks

Border to Tick Distance

In round gauge charts, the space between the inner edge of the border and the ticks that mark the values.

Major Tick Length

In round gauge charts, the length of the major ticks (i.e., ticks that show a label).

Major Tick Width

In round gauge charts, the width of the major ticks (i.e., ticks that show a label).

Minor Tick Length

In round gauge charts, the length of ticks that do not have a value displayed.

Minor Tick Width

In round gauge charts, the width of ticks that do not have a value displayed.

Gauge Needle

Needle Length

In round gauge charts, the length of the needle. (Note that the percentage is calculated from the diameter of the gauge, so if you choose a value greater that 50%, the needle will point to somewhere outside the gauge!)

Needle Width at Base

In round gauge charts, the width of the needle at the center of the gauge.

Needle Base Radius

In round gauge charts, the radius of the base that covers the center of the gauge.

Gauge Color Range

Border to Color Range Distance

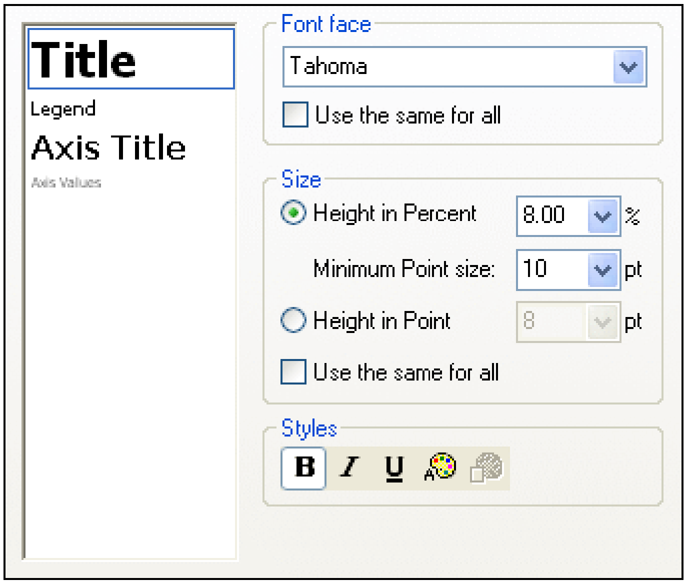
In round gauge charts, the space between the inner edge of the border and the outer edge of the color range.

Color Range Width

In round gauge charts, the width of the customizable color range. (Note that the percentage is calculated from the diameter of the gauge!)

Fonts

The Fonts section of the **Change Appearance** dialog box lets you configure fonts for objects in the Chart window.



Font face: Tahoma

□ Use the same for all

● Height in Percent 8.00 %

Minimum Point size: 10 pt

○ Height in Point 8 pt

□ Use the same for all

Font settings

You can choose the font face, size, and style for the individual elements displayed in the Chart window. You can define the size as a percentage of the chart size and define a minimum size in points, or specify an absolute value (in points). To apply the same font and/or size to all text elements, activate the respective `Use the same for all` check box.

The element names in the list box are defined as follows:

- **Title**: The name of a chart
- **Legend**: The key to the colors used in the chart
- **Labels**: The designation of the pies of a pie chart
- **Axis Title**: The name of the X, Y, and Z axis in a bar or line chart
- **Axis Values**: The units displayed on an axis in a bar or line chart
- **Tick Values**: The units displayed on a gauge chart

Dynamic XPath Settings

Dynamic XPath Settings are very useful if you wish to use dynamic data from the XML document in the settings of the chart. For example, the title of a chart about a `Region` element might need to have data about that `Region` element (such as its name) in the title of the chart. If there are several Region elements, any one of which will be used at a time for the chart, then the data for the chart title can only be obtained dynamically via an XPath expression. The Dynamic XPath Settings dialog enables such data to be accessed via XPath expressions.

To access the chart's Dynamic XPath Settings dialog, click **Dynamic XPath Settings** in the Chart Settings pane of the Chart Configuration dialog (*see screenshot below*).

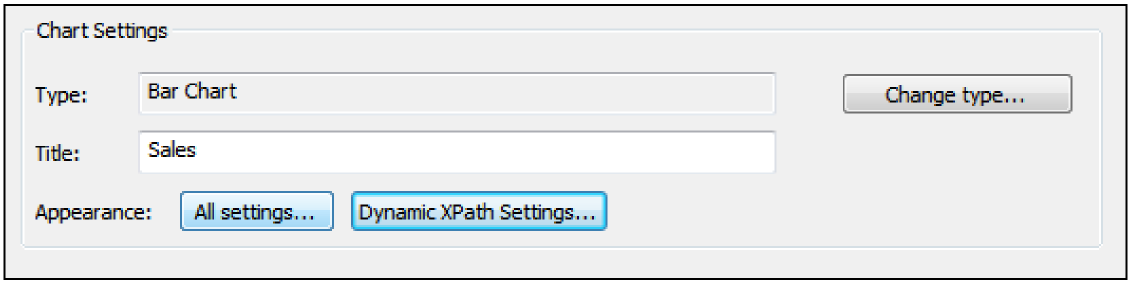
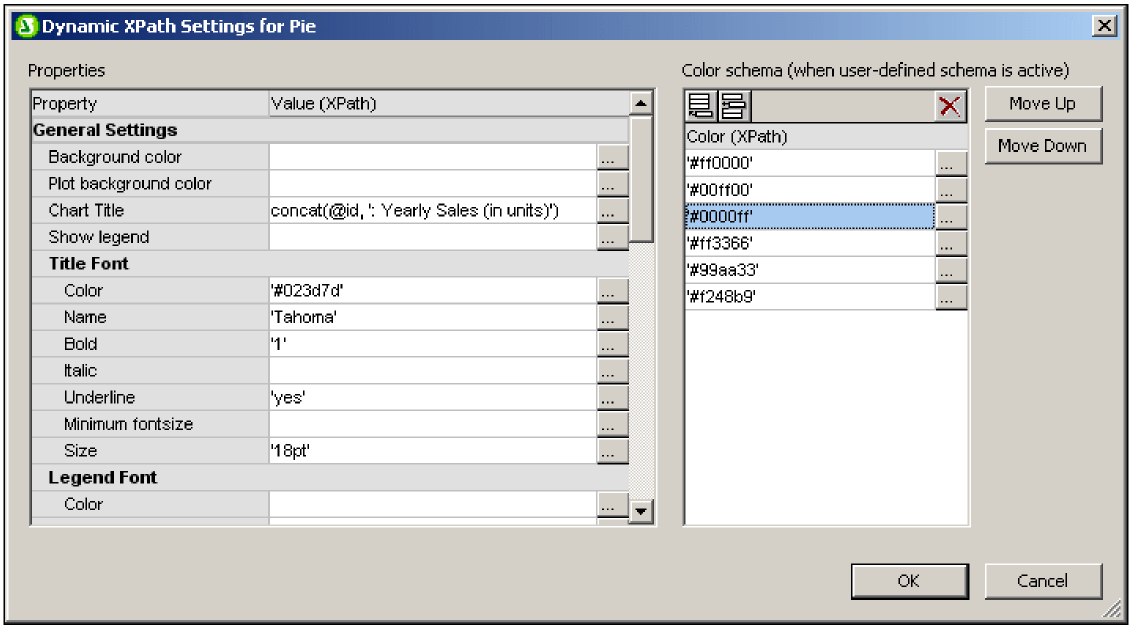


Chart Settings

Type:	Bar Chart
Title:	Sales

Dynamic XPath Settings

The Dynamic XPath Settings dialog (*screenshot below*) is accessed by clicking the **Dynamic XPath Settings** button in the Chart Settings pane of the Chart Configuration dialog. A number of chart settings can be entered in this dialog.



Dynamic XPath Settings for Pie

Properties

Property	Value (XPath)
General Settings	
Background color	
Plot background color	
Chart Title	concat(@id, ': Yearly Sales (in units)')
Show legend	
Title Font	
Color	'#023d7d'
Name	'Tahoma'
Bold	'1'
Italic	
Underline	'yes'
Minimum fontsize	
Size	'18pt'
Legend Font	
Color	

Color schema (when user-defined schema is active)

Color (XPath)
'#ff0000'
'#00ff00'
'#0000ff'
'#ff3366'
'#99aa33'
'#f248b9'

Note the following points:

- All entries in this dialog are evaluated as XPath expressions, so string literals must be enclosed in quotes. For example: `'Tahoma'`, `'1'`, `'18pt'`, and `'#FF3366'`.

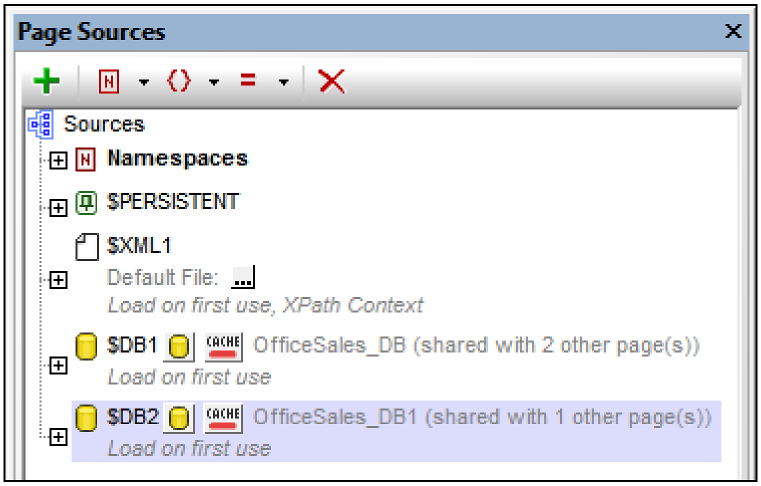
- On hovering over a setting or its value, a tooltip appears giving information about enumerations and formats.
- Dynamic XPath settings have precedence over settings made in the Chart Configuration dialog or Change Appearance dialog. For example, a chart title made as a dynamic XPath setting has precedence over one set in the Chart Configuration dialog.
- The colors of the color schema will be used when the user-defined color schema is selected in the Change Appearance dialog as the color schema to use. Colors are set in the RGB hexadecimal format: `#RRGGBB`. So an XPath expression to specify the color red would be: `'#FF0000'`.

Chapter 8

Databases

8 Databases

You can use databases (DBs) as data sources of MobileTogether designs. This allows data from DBs to be presented in MobileTogether solutions as well as allows end users to edit data in DBs from their mobile devices. You can use multiple editable DB data sources, and then use XPath and XQuery expressions to retrieve and modify data.



Page Sources

Sources
- Namespaces
- $PERSISTENT
- $XML1
  Default File: ...
  Load on first use, XPath Context
- $DB1 OfficeSales_DB (shared with 2 other page(s))
  Load on first use
- $DB2 OfficeSales_DB1 (shared with 1 other page(s))
  Load on first use

This section
This section is organized as follows:

- DBs as Data Sources
- Displaying DB Data
- Saving Data to the DB
- Connecting to a DB
- Database Query

See the database tutorial, Database-And-Charts, for a detailed description of a MobileTogether design which uses multiple data sources that can be edited in the MobileTogether Client solution.

Database support
The following databases are supported. The available root object for each database is also listed. While Altova endeavors to support other ODBC/ADO databases, successful connection and data processing have only been tested with the databases listed below. If your Altova application is a 64-bit version, ensure that you have access to the 64-bit database drivers needed for the specific database you are connecting to. Your Altova application supports logical files of the IBM iSeries database and shows logical files as views.

Database (natively supported)	Root Object

MS Access 2003, 2007, 2010, 2013	database
MS SQL Server 2005, 2008, 2012	database/schema
Oracle 9i, 10g, 11g	schema
MySQL 5.x, 5.5.28	database
PostgreSQL 8.0, 8.1, 8.2, 8.3, 9.0.10, 9.1.6, 9.2.1	database
Sybase ASE12, ASE15	database
IBM DB2 8.x, 9.5, 9.7, 10.1	schema
IBM DB2 for i 6.1, 7.1	schema
IBM Informix 11.70	database

MobileTogether also supports SQLite.

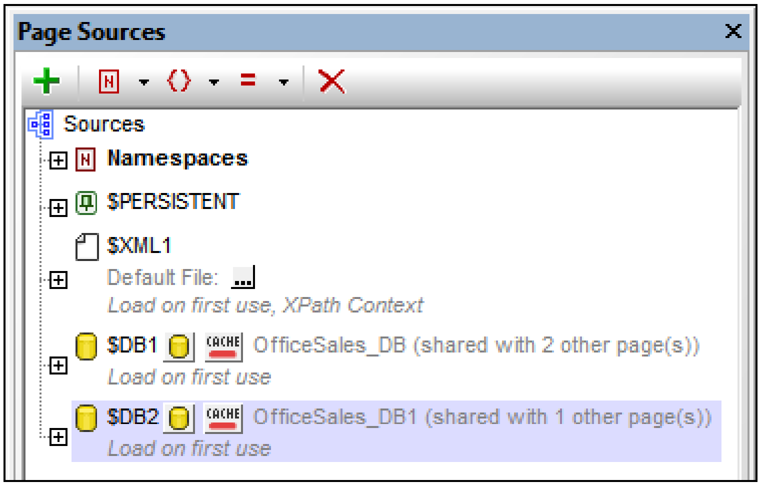
8.1 DBs as Data Sources

This section:

- About DB data sources
- Tree structure of the DB data source
- About OriginalRowSet
- Primary Keys in MobileTogether Designer

About DB data sources

Any number of DB data sources can be added to the design of a page and then used within it. Whether a DB data source is editable or not is defined at the time the data source is added.



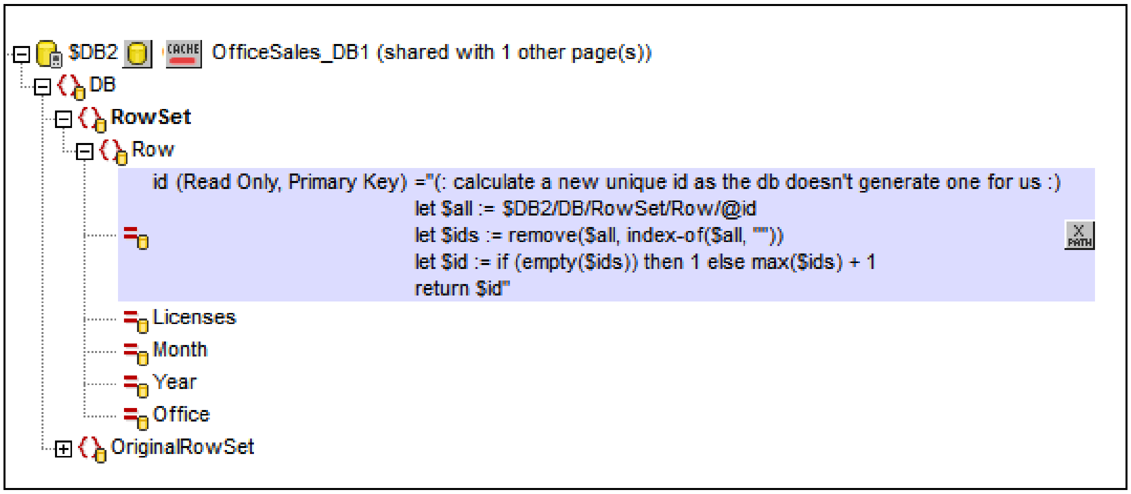
$PERSISTENT

$XML1
Default File:
Load on first use, XPath Context

$DB1 OfficeSales_DB (shared with 2 other page(s))
Load on first use

$DB2 OfficeSales_DB1 (shared with 1 other page(s))
Load on first use

When a DB source is added, a data tree is generated (*see screenshot below*). Each DB table is added as a `Row` element and the table's columns are added as attributes of the `Row` element. If the data source is used on other pages also, then you can synchronize the several instance of this data source on all pages so that they are identical (in terms of structure and data). This synchronization possibility is available each time you add the data source or modify any of the instances in any way.



```
$DB2   OfficeSales_DB1 (shared with 1 other page(s))
  DB
    RowSet
      Row
        id (Read Only, Primary Key) ="(: calculate a new unique id as the db doesn't generate one for us :)
                                     let $all := $DB2/DB/RowSet/Row/@id
                                     let $ids := remove($all, index-of($all, ""))
                                     let $id := if (empty($ids)) then 1 else max($ids) + 1
                                     return $id"
        Licenses
        Month
        Year
        Office
    OriginalRowSet
```

Tree structure of the DB data source

Every DB data source has the following structure:

```
$DBX (the root node)
|
|--DB (the root element)
|  |
|  |--RowSet (a container element for the DB table)
|  |  |
|  |  |--Row (the DB table)
|  |  |  |
|  |  |  |--<Attributes> (the columns of the DB table)
```

The nodes in the tree can be addressed using XPath expressions. If a node is set as the *XPath context node for the page* (via the node's context menu), then XPath expressions can be built relative to the XPath context node. Otherwise nodes ca be addressed using absolute paths starting at the root node: `$DBX/DB/RowSet/Row/MyAttribute`.

You can also use XQuery expressions to retrieve or manipulate data in the DB tree. See the section on primary keys below for an example.

About `OriginalRowSet`

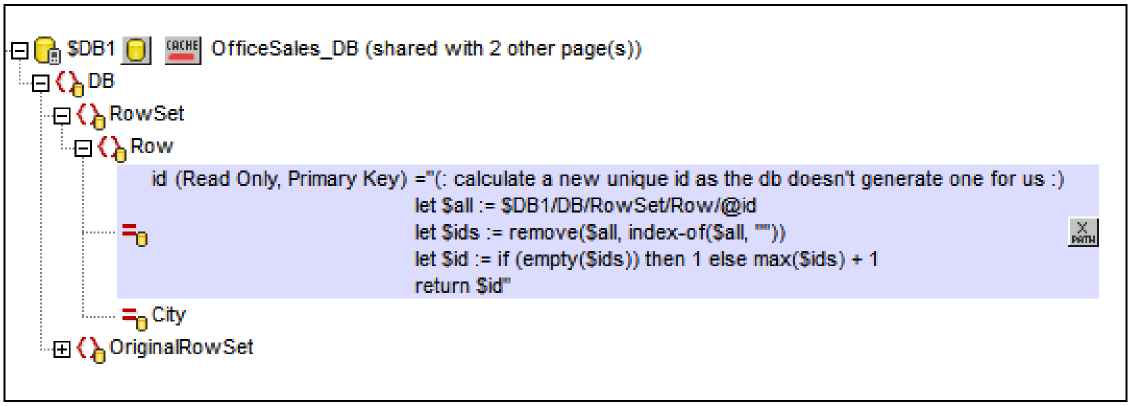
In order for data to be edited and saved, the tree of the data source must also include an `OriginalRowSet` element, which is a copy of the `RowSet` element. The original data is saved in the `OriginalRowSet` element, while edited data is saved in the `RowSet` element. When the data source is saved, the difference between the two trees, `OriginalRowSet` and `RowSet`, is calculated, and the data source is modified accordingly. If the modification is successful, then the modified data from `RowSet` is copied to `OriginalRowSet` so that `OriginalRowSet` is again current.

To create an `OriginalRowSet` for a data source, right-click the root node of the data source and toggle on the command **Create OriginalRowSet**.

The **Create OriginalRowSet** command is enabled for database type (`$DB`) root nodes. It is a toggle command that creates/removes an `OriginalRowSet` data structure that contains the original data of the data source. User modified data is saved in the main structure created from the data source. When modified data is saved back to the DB, the `OriginalRowSet` structure is modified to contain the data newly saved to the DB. Till the time modified data is saved to the DB, the original DB data is retained in the `OriginalRowSet` structure. This ensures that the original DB data is still available in the tree.

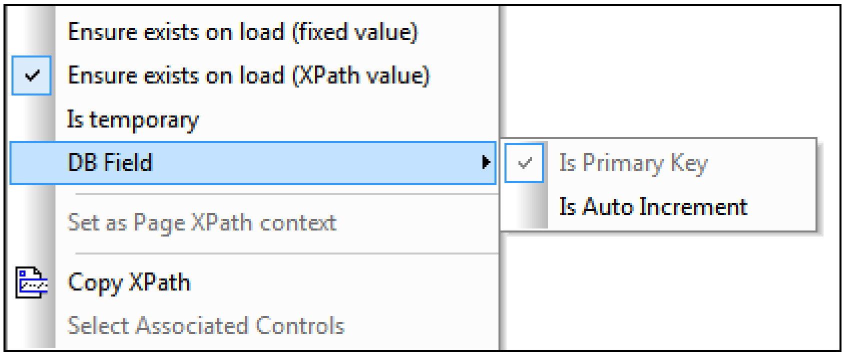
Primary keys in MobileTogether Designer

Primary keys in DBs typically are auto-incrementing. When this is the case and a new row is added to a table, the primary key column of the added row is automatically incremented. In MobileTogether Designer, when a table is retrieved the primary key and auto-increment information is automatically retrieved and displayed in the Page Sources Pane (*see screenshot below*).



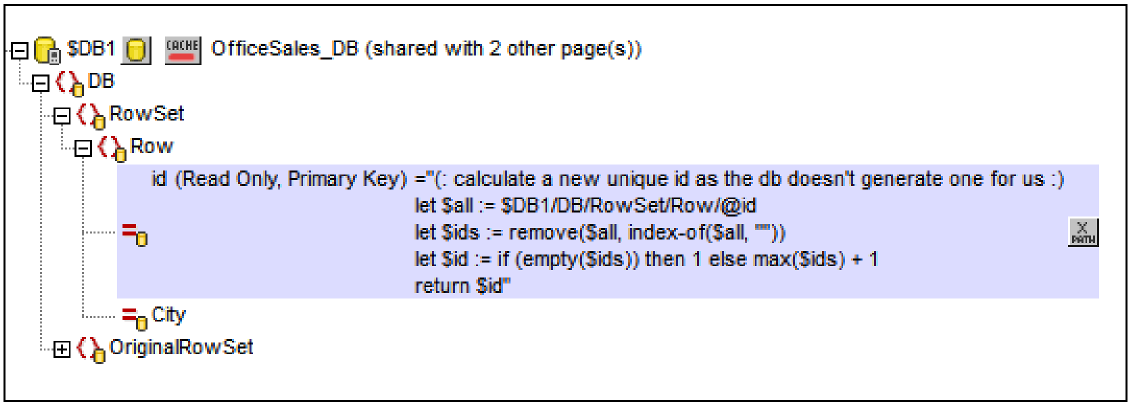
```
id (Read Only, Primary Key) ="(: calculate a new unique id as the db doesn't generate one for us :)
                               let $all := $DB1/DB/RowSet/Row/@id
                               let $ids := remove($all, index-of($all, ""))
                               let $id := if (empty($ids)) then 1 else max($ids) + 1
                               return $id"
```

If auto-retrieval of this information was not successful, the context menu of tree nodes contains toggle commands that enable you to correctly annotate nodes (*see screenshot below*).


Ensure exists on load (fixed value)
✓ Ensure exists on load (XPath value)
Is temporary
DB Field
✓ Is Primary Key
Is Auto Increment
Set as Page XPath context
Copy XPath
Select Associated Controls

If the primary key column is not auto-incrementing, new primary key values for appended rows must be automatically generated using an XQuery expression. This is because primary key columns cannot be edited. The XQuery expression is inserted by using the primary-key node's context menu command, **Ensure Exists before Page Load (XPath Value)**. In the example below, a new value is generated for the primary key `@id` by using the following XQuery expression:

```
let $all := $DB1/DB/RowSet/Row/@id
let $ids := remove($all, index-of($all, ""))
let $id := if (empty($ids)) then 1 else max($ids) + 1
return $id
```



```
$DB1 OfficeSales_DB (shared with 2 other page(s))
  DB
    RowSet
      Row
        id (Read Only, Primary Key) ="(: calculate a new unique id as the db doesn't generate one for us :)
                                      let $all := $DB1/DB/RowSet/Row/@id
                                      let $ids := remove($all, index-of($all, ""))
                                      let $id := if (empty($ids)) then 1 else max($ids) + 1
                                      return $id"
        City
    OriginalRowSet
```

8.2 Editing DB Data

This section:

- About OriginalRowSet
- Editing DB data in tables and other controls
- Updating, inserting, appending, and deleting nodes
- The DB Execute action and $MT_DBExecute_Result variable
- Primary Keys in MobileTogether Designer

About `OriginalRowSet`

In order for data to be edited and saved, the tree of the data source must also include an `OriginalRowSet` element, which is a copy of the `RowSet` element. The original data is saved in the `OriginalRowSet` element, while edited data is saved in the `RowSet` element. When the data source is saved, the difference between the two trees, `OriginalRowSet` and `RowSet`, is calculated, and the data source is modified accordingly. If the modification is successful, then the modified data from `RowSet` is copied to `OriginalRowSet` so that `OriginalRowSet` is again current.

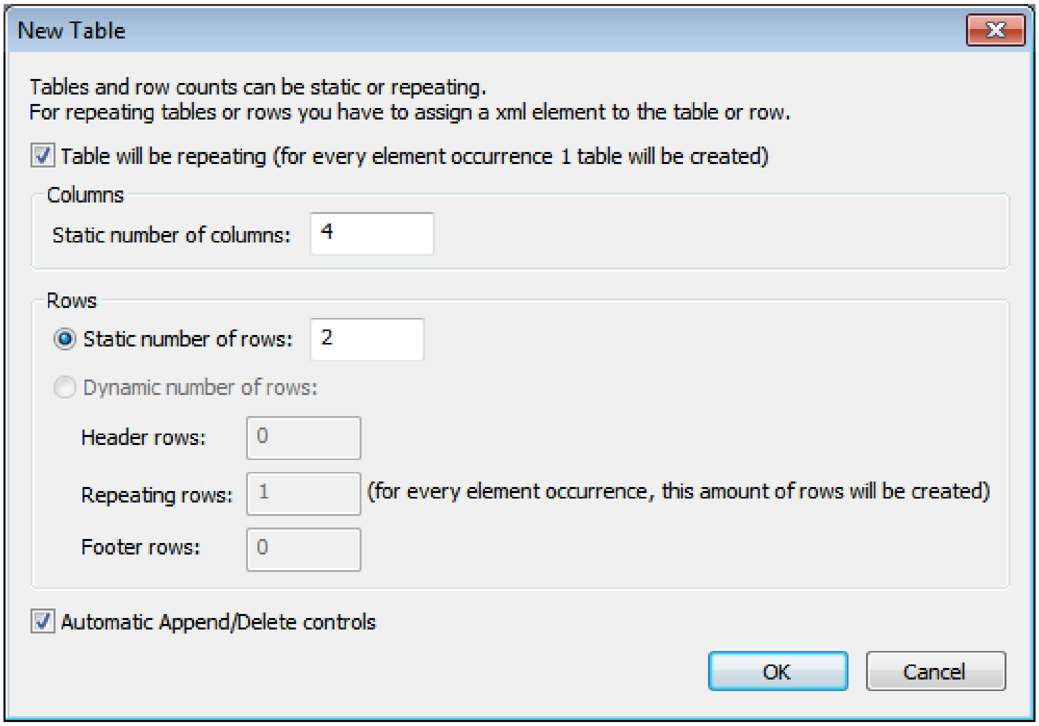
To create an `OriginalRowSet` for a data source, right-click the root node of the data source and toggle on the command **Create OriginalRowSet**.

The **Create OriginalRowSet** command is enabled for database type (`$DB`) root nodes. It is a toggle command that creates/removes an `OriginalRowSet` data structure that contains the original data of the data source. User modified data is saved in the main structure created from the data source. When modified data is saved back to the DB, the `OriginalRowSet` structure is modified to contain the data newly saved to the DB. Till the time modified data is saved to the DB, the original DB data is retained in the `OriginalRowSet` structure. This ensures that the original DB data is still available in the tree.

Editing DB data in tables and other controls

To create a control in which DB data can be edited, do the following:

- Use a control that is editable by the end user, such as a combo box or edit field. A label control, for instance, is not editable. If a table is used to generate repeating rows, add the editable controls within the cells of the table. See the Edit Offices Table in the Database-And-Charts tutorial for an example of how to do this.
- On the control, create a page source link to the data source node that is to be edited. Do this by dragging the data source node onto the control.
- If you use a table with repeating rows, use the option to automatically include Append/ Delete controls when the table is created (*screenshot below*).



New Table

Tables and row counts can be static or repeating.
For repeating tables or rows you have to assign a xml element to the table or row.

☑ Table will be repeating (for every element occurrence 1 table will be created)

Columns

Static number of columns: 4

Rows

● Static number of rows: 2

○ Dynamic number of rows:

Header rows: 0

Repeating rows: 1 (for every element occurrence, this amount of rows will be created)

Footer rows: 0

☑ Automatic Append/Delete controls

The advantage of using a table with repeating rows that are linked to the repeating `Row` elements of a DB data source is that when a table row is added, a DB row is also automatically added. For examples, see the Database-And-Charts tutorial and the MobileTogether Designer DB examples files.

Updating, inserting, appending, and deleting nodes

The Update Data actions enable nodes in DBs to be edited when a page or control event is triggered:

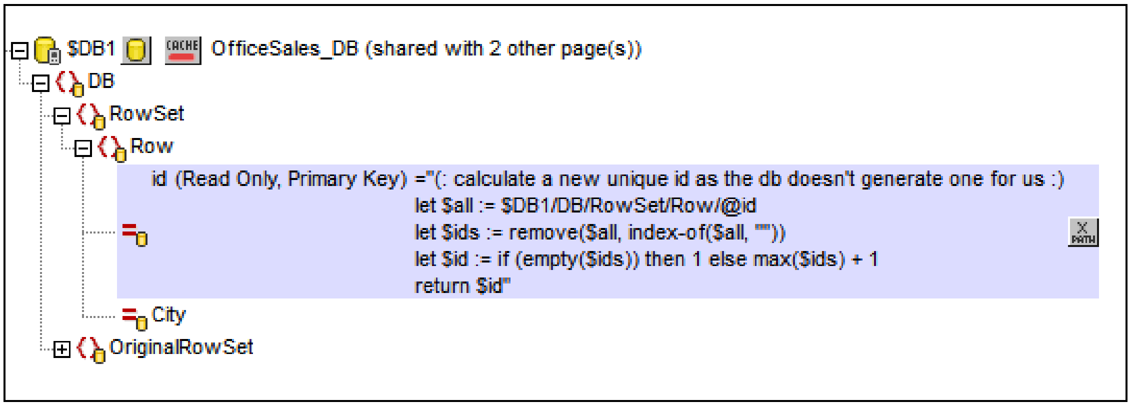
- Update Node(s): Updates one or more nodes, such as DB column, with value/s generated or obtained by the action's XPath expression.
- Insert Node(s): Inserts one or more nodes, such as DB rows, before a node selected by the action's XPath expression. The structure and contents of the inserted node can also be defined.
- Append Node(s): Appends one or more nodes, such as DB rows, as the first or last child of a node selected by the action's XPath expression. The structure and contents of the appended node can also be defined.
- Delete Node(s): Deletes one or more nodes, such as a DB rows, specified by the action's XPath expression.

The DB Execute action and `$MT_DBExecute_Result` variable

The DB Execute action executes an SQL statement and stores the result of that execution in a variable named `$MT_DBExecute_Result`. This variable can then be used in XPath expressions anywhere within the the project. This enables, for example, structures and data from one DB to be selected based on parameters, to be held in storage, modified, and inserted at other locations.

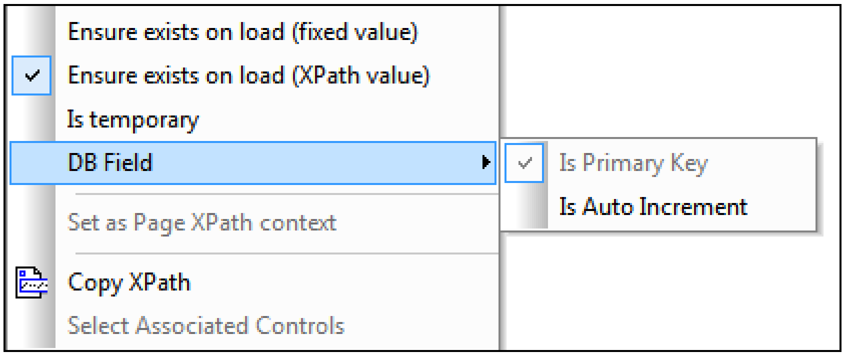
Primary keys in MobileTogether Designer

Primary keys in DBs typically are auto-incrementing. When this is the case and a new row is added to a table, the primary key column of the added row is automatically incremented. In MobileTogether Designer, when a table is retrieved the primary key and auto-increment information is automatically retrieved and displayed in the Page Sources Pane (*see screenshot below*).



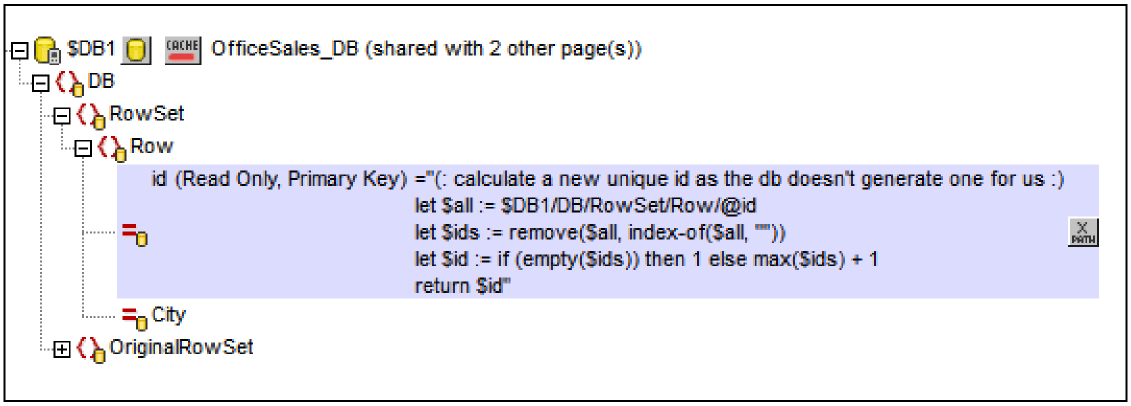
```
id (Read Only, Primary Key) ="(: calculate a new unique id as the db doesn't generate one for us :)
                              let $all := $DB1/DB/RowSet/Row/@id
                              let $ids := remove($all, index-of($all, ""))
                              let $id := if (empty($ids)) then 1 else max($ids) + 1
                              return $id"
```

If auto-retrieval of this information was not successful, the context menu of tree nodes contains toggle commands that enable you to correctly annotate nodes (*see screenshot below*).


Ensure exists on load (fixed value)
✓ Ensure exists on load (XPath value)
Is temporary
DB Field
✓ Is Primary Key
Is Auto Increment
Set as Page XPath context
Copy XPath
Select Associated Controls

If the primary key column is not auto-incrementing, new primary key values for appended rows must be automatically generated using an XQuery expression. This is because primary key columns cannot be edited. The XQuery expression is inserted by using the primary-key node's context menu command, **Ensure Exists before Page Load (XPath Value)**. In the example below, a new value is generated for the primary key `@id` by using the following XQuery expression:

```
let $all := $DB1/DB/RowSet/Row/@id
let $ids := remove($all, index-of($all, ""))
let $id := if (empty($ids)) then 1 else max($ids) + 1
return $id
```



```
$DB1 OfficeSales_DB (shared with 2 other page(s))
  DB
    RowSet
      Row
        id (Read Only, Primary Key) ="(: calculate a new unique id as the db doesn't generate one for us :)
                                    let $all := $DB1/DB/RowSet/Row/@id
                                    let $ids := remove($all, index-of($all, ""))
                                    let $id := if (empty($ids)) then 1 else max($ids) + 1
                                    return $id"
        City
    OriginalRowSet
```

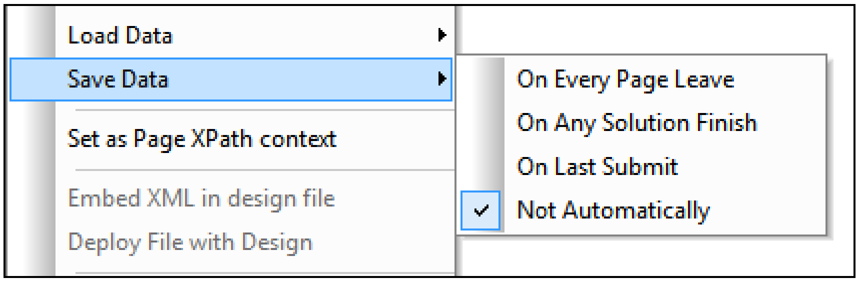
8.3 Saving Data to the DB

This section:

Saving on the basis of solution progress

The context menu of a `$DB` root node has a **Save Data** command (*screenshot below*) which enables the data source represented by the root node to be saved at different points during the progress of the solution. The options are described below. If the default option, *Not Automatically*, is selected, then data is saved only when the *Save* action of an event is triggered.

The **Save Data** command rolls out a sub-menu with the following mutually exclusive options (only one can be selected):

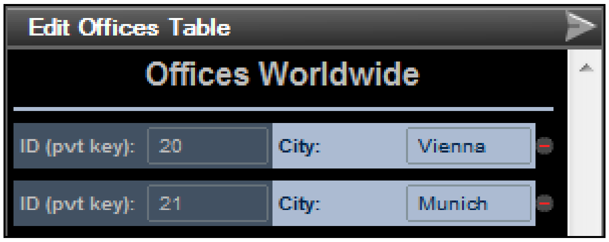


- *On Every Page Leave:* Data in the tree is saved every time a page containing that tree is exited.
- *On Any Solution Finish:* Data in the tree is saved when the solution is exited, no matter at what point or how the solution is exited.
- *On Last Submit:* Data in the tree is saved when the workflow progresses as designed, from first page to last page, and when the last **Submit** button is tapped. If this option is selected and the solution is exited before the last **Submit** button is tapped, then data in the tree will not be saved.
- *Not Automatically:* The tree will not automatically be saved. You must use the Save, Save to File, or Save to HTTP actions to save data.

The default is *Not Automatically*.

The Save action

Data can be saved to the DB when a page event or control event is triggered for which the *Save* action is defined. Such an event could be, for example, the clicking of the **Submit** button by the end user. In the screenshot below, the Submit button is located in the Edit Offices Table bar.



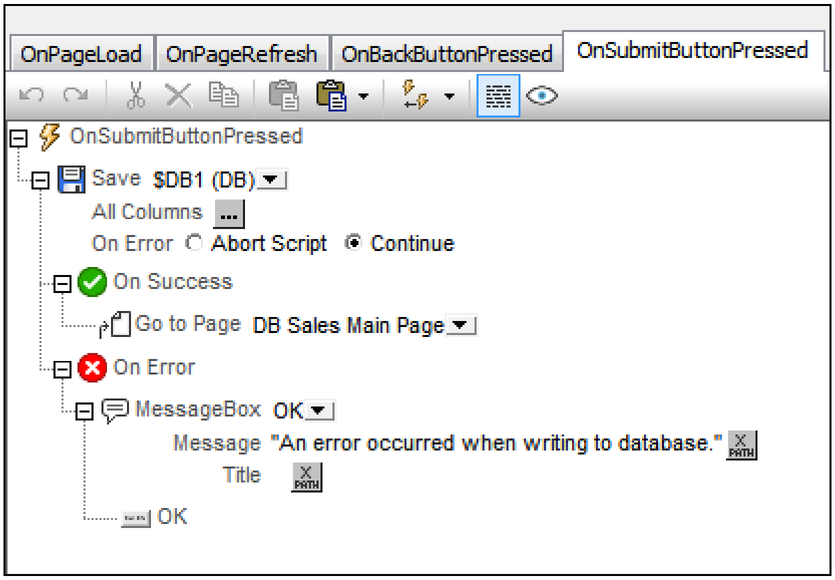
Edit Offices Table

Offices Worldwide

ID (pvt key): 20 City: Vienna

ID (pvt key): 21 City: Munich

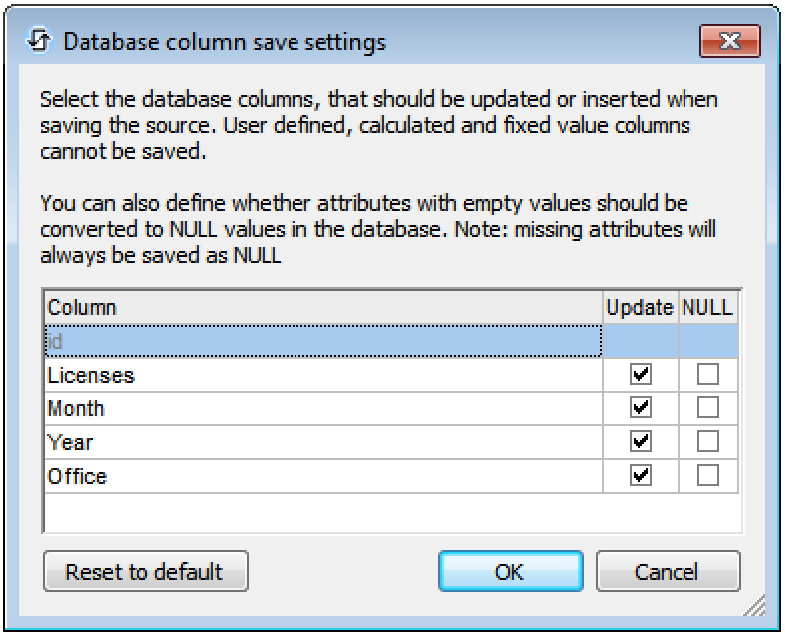
The *Save* action can be defined on a page action or control action. Access the respective Actions dialogs via the All Actions dialog (**Page | Actions Overview**). The screenshot shows a Page Actions dialog (*screenshot below*) with the Save action defined on the **`OnSubmitButtonPressed`** event.



OnSubmitButtonPressed
Save $DB1 (DB)
All Columns ...
On Error ○ Abort Script ● Continue
On Success
Go to Page DB Sales Main Page
On Error
MessageBox OK
Message "An error occurred when writing to database."
Title
OK

Filtering the columns to save

In the context menu of `$DB` root nodes, select the command Filter columns to display the Database Column Save Settings dialog (*screenshot below*) and to select which columns should be updated or inserted.



Database column save settings

Select the database columns, that should be updated or inserted when saving the source. User defined, calculated and fixed value columns cannot be saved.

You can also define whether attributes with empty values should be converted to NULL values in the database. Note: missing attributes will always be saved as NULL

Column	Update	NULL
id		
Licenses	☑	□
Month	☑	□
Year	☑	□
Office	☑	□

The dialog displays the columns of the data source. Columns that cannot be updated (because they are user-defined, fixed-value, or calculated-value) will not have an *Update* option. Deselect the columns you do not want to update. Attributes with empty values can be converted to `NULL` values in the DB by checking that column's *NULL* check box. Note that missing attributes will always be saved as `NULL`. If you wish to reset the Save settings so that all columns are updtaed, click **Reset to default**.

About `OriginalRowSet`

In order for data to be edited and saved, the tree of the data source must also include an `OriginalRowSet` element, which is a copy of the `RowSet` element. The original data is saved in the `OriginalRowSet` element, while edited data is saved in the `RowSet` element. When the data source is saved, the difference between the two trees, `OriginalRowSet` and `RowSet`, is calculated, and the data source is modified accordingly. If the modification is successful, then the modified data from `RowSet` is copied to `OriginalRowSet` so that `OriginalRowSet` is again current.

To create an `OriginalRowSet` for a data source, right-click the root node of the data source and toggle on the command **Create OriginalRowSet**.

The **Create OriginalRowSet** command is enabled for database type (`$DB`) root nodes. It is a toggle command that creates/removes an `OriginalRowSet` data structure that contains the original data of the data source. User modified data is saved in the main structure created from the data source. When modified data is saved back to the DB, the `OriginalRowSet` structure is modified to contain the data newly saved to the DB. Till the time modified data is saved to the DB, the original DB data is retained in the `OriginalRowSet` structure. This ensures that the original DB data is still available in the tree.

Committing transactions
Another way to save data to a DB is to begin an independent transaction and commit it. DB transactions are available as actions for page and control events.

About DB Transactions
For each DB access that needs a transaction, one is automatically created and closed afterwards. This might not be desirable for some setups. For example, when you have two DB page sources that you want to update atomically together: If one table is saved successfully, then the other one is committed. To accommodate this kind of situation, transactions can be created on a connection basis.

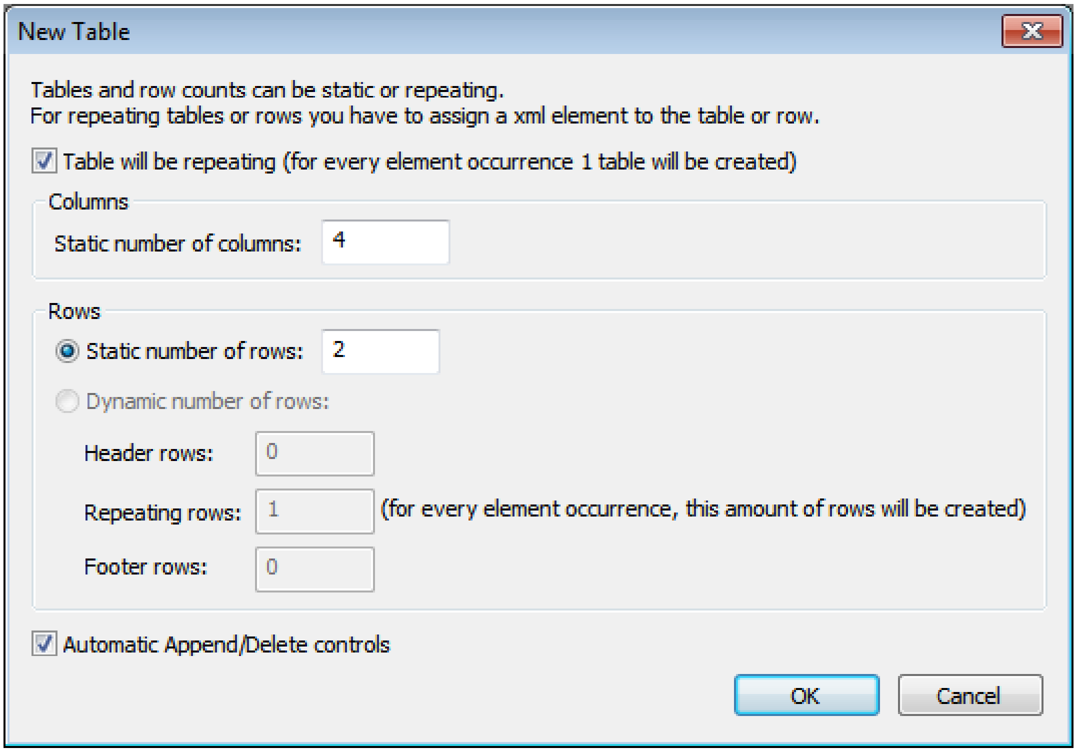
If you begin a transaction, all DB operations belonging to the same DB connection will use this transaction.

Committing a transaction makes changes visible to the world outside your transaction. Changes can be rolled back. In this case, even if you have done a Save on your page source, the changes won't be visible after a rollback! Note that any transaction that is not closed (committed or rolled back) when the end of the action tree has been reached will be rolled back automatically! A warning to this effect will be displayed in the Messages window.

It is important to bear in mind that, while the behavior above refers to explicit transaction actions, this behavior also applies to all DB operations that use the same connection as the transaction.

8.4 Displaying DB Data

Displaying DB data in tables and other controls
DB data can be displayed in a control by creating a page source link from the control to a data source node. Typically the best way to display DB data is in a table with repeating rows. Drag a table control into the design and create a new table as a repeating table (*see screenshot below*). Then, drag controls into the cells of the table and make page source links to the column nodes of the DB row. For an actual demonstration of how this works, see the tutorial, Database-And-Charts.


New Table

Tables and row counts can be static or repeating.
For repeating tables or rows you have to assign a xml element to the table or row.

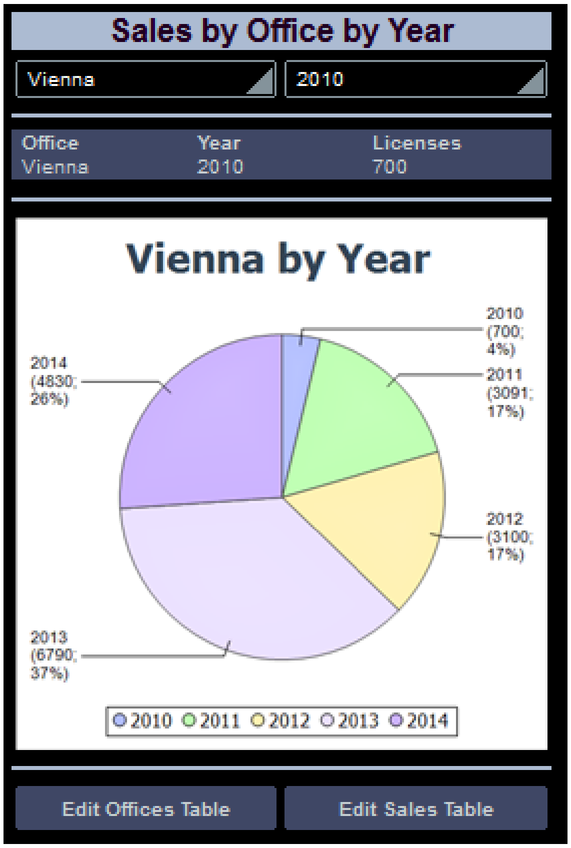
☑ Table will be repeating (for every element occurrence 1 table will be created)

Columns
Static number of columns: 4

Rows
● Static number of rows: 2
○ Dynamic number of rows:
Header rows: 0
Repeating rows: 1 (for every element occurrence, this amount of rows will be created)
Footer rows: 0

☑ Automatic Append/Delete controls

Displaying DB data as charts
In addition to being able to display DB data directly, you can also create charts based on DB data.



Sales by Office by Year

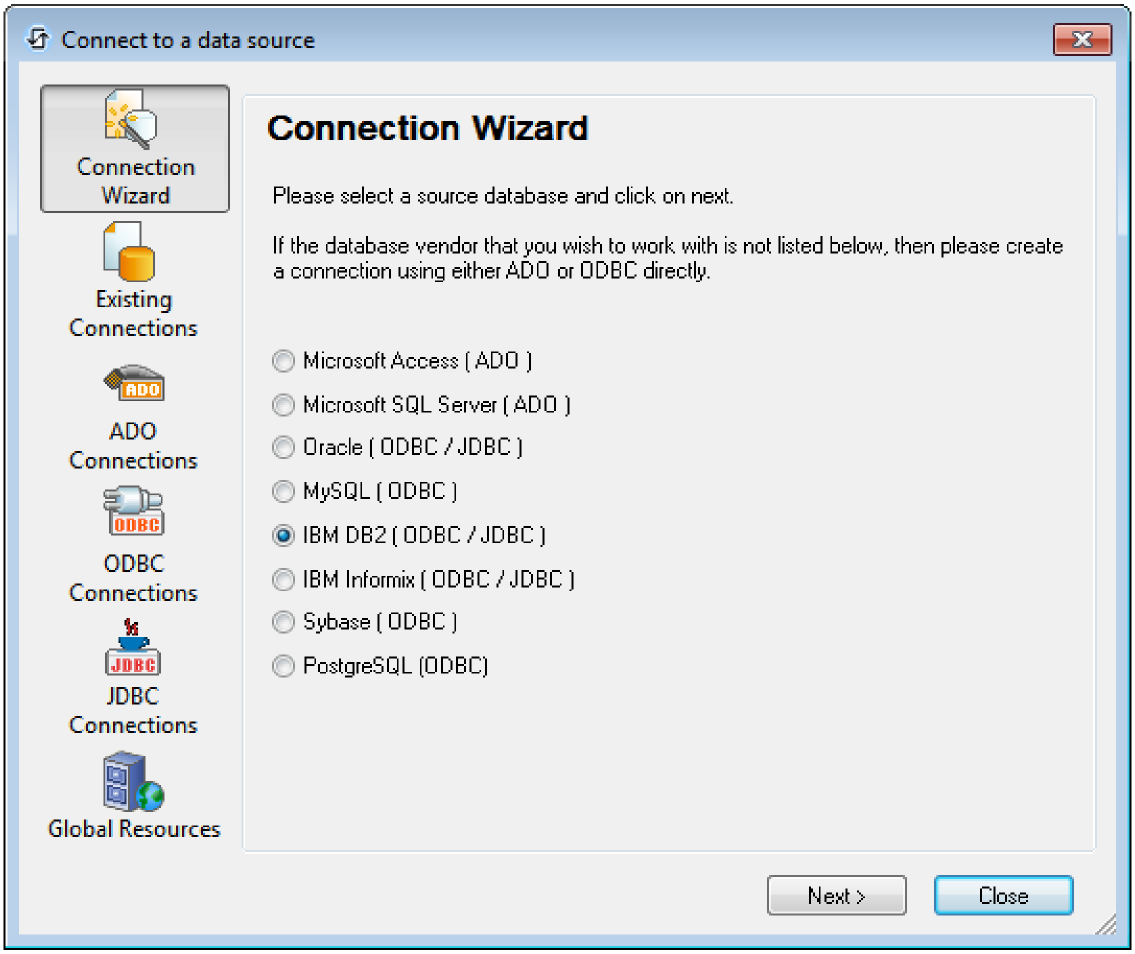
Vienna ▾ 2010 ▾

Office	Year	Licenses
Vienna	2010	700

For an example of how to do this, see the tutorial, Database-And-Charts.

8.5 Connecting to a DB

Connections to a database (DB) are made via the Connect to a Data Source dialog (*screenshot below*).



Connection Wizard

Please select a source database and click on next.

If the database vendor that you wish to work with is not listed below, then please create a connection using either ADO or ODBC directly.

○ Microsoft Access (ADO)
○ Microsoft SQL Server (ADO)
○ Oracle (ODBC / JDBC)
○ MySQL (ODBC)
● IBM DB2 (ODBC / JDBC)
○ IBM Informix (ODBC / JDBC)
○ Sybase (ODBC)
○ PostgreSQL (ODBC)

The Connect to a Data Source dialog is accessed, and connections to DBs are made, for two purposes:

- To add a page data source: In the Page Sources Pane (of Page Design View), click the **Add Source** button and select *New DB Structure*.
- To do a DB Query: In DB Query View, click the **Quick Connect** button at the top left of the view.

In this section

The options for connecting to the database are described in the subsections of this section.

- Using a Connection Wizard that guides you through the connection process
- Using an existing connection
- Using an ADO Connection

- Using an ODBC Connection
- Using a JDBC Connection
- Using a global resource

Note: If you are using the 64-bit version of MobileTogether Designer, ensure that you have access to the 64-bit database drivers needed for the specific database you are connecting to.

Connection Wizard

The Connection Wizard helps you to connect to the most commonly used types of databases. In this section, we go through the connection to a Microsoft Access database and an IBM DB2 database.

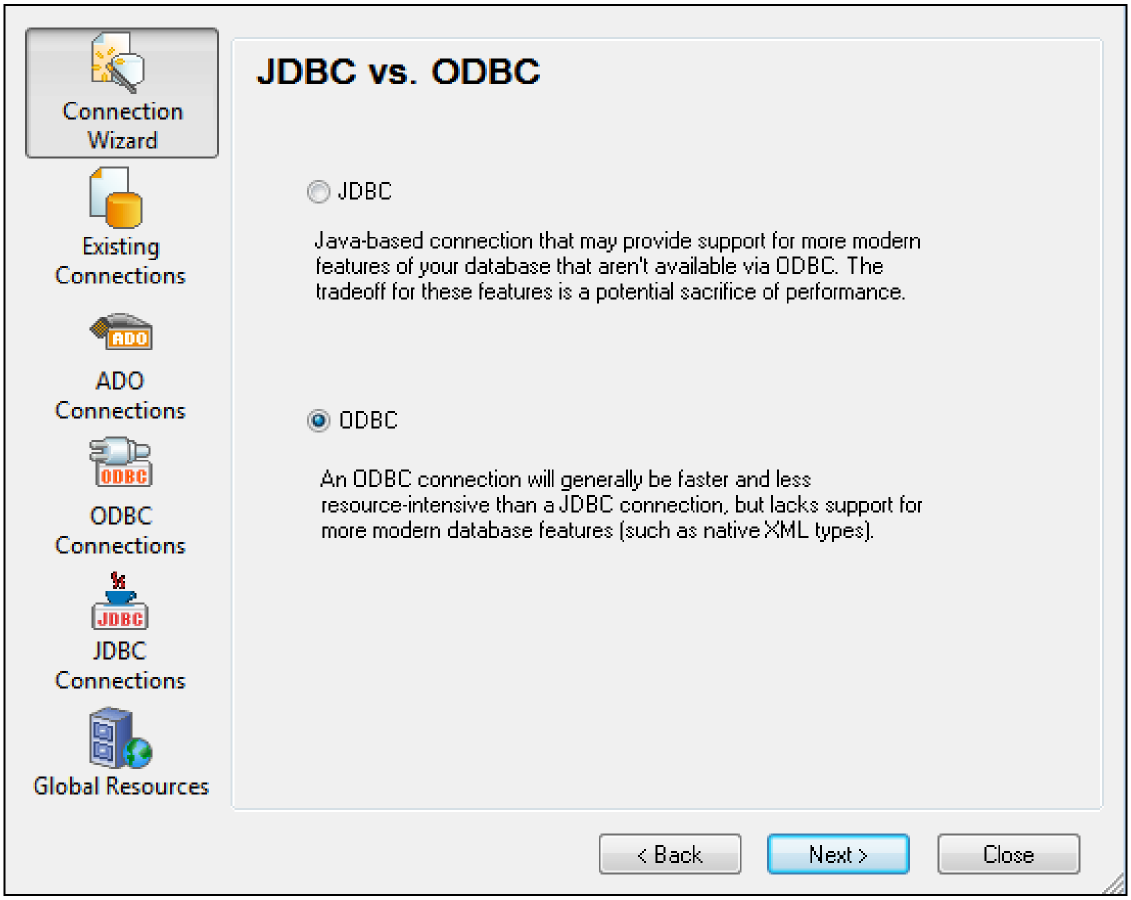
MS Access

1. Select Microsoft Access (ADO) and click **Next**.
2. In the Connect to MS Access dialog that appears, browse for the MS Access database, and click **Next**.

The connection to the data source is established.

Oracle and IBM DB2: ODBC or JDBC

If you select *Oracle (ODBC / JDBC)* or *IBM DB2 (ODBC / JDBC)*, a dialog appears that asks whether you wish to select JDBC or ODBC for the connection (*screenshot below*).



JDBC vs. ODBC

○ JDBC

Java-based connection that may provide support for more modern features of your database that aren't available via ODBC. The tradeoff for these features is a potential sacrifice of performance.

● ODBC

An ODBC connection will generally be faster and less resource-intensive than a JDBC connection, but lacks support for more modern database features (such as native XML types).

ODBC connections are faster and light on resources, while JDBC connections support the newer features of your database. We recommend that you use ODBC connections. If you find that some DB features are not supported when you use ODBC, then try JDBC. The process for connecting to an Oracle database or IBM DB2 database via ODBC is described below. The process for connecting via JDBC is described in the section, JDBC Connections.

Connecting to Oracle or IBM DB2 via ODBC

1. In the JDBC vs. ODBC dialog that appears (*screenshot above*), select ODBC and click **Next**. The wizard will then guide you in configuring the connection to the database and making the connection. These steps consist essentially of the following:
2. Select the appropriate driver to connect to the DB.
3. Locate the DB via a DSN.
4. Enter user information that enables you to connect.

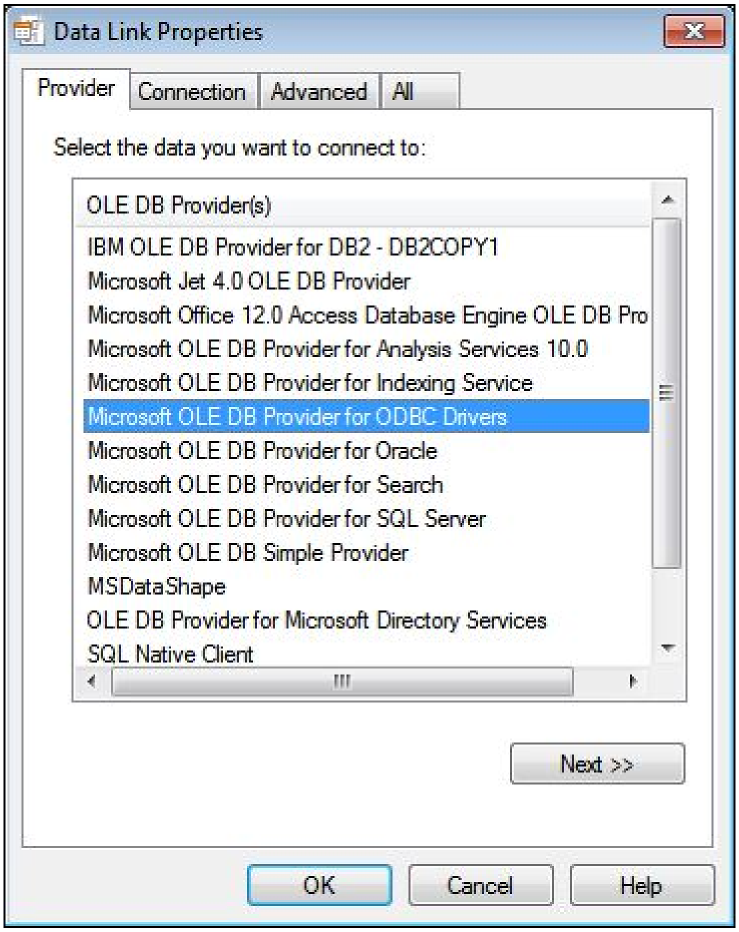
Existing Connections

The Existing Connections pane shows all the connections that are currently established. Select the required database and click **Connect**. The connection to the database is established.

ADO Connections

The connection via ADO is described with an IBM DB2 database as the target DB. Where options are different for other DB types, these differences are noted.

1. In the ADO Connections pane, click **Build**. The Data Link Properties dialog (*screenshot below*) opens.



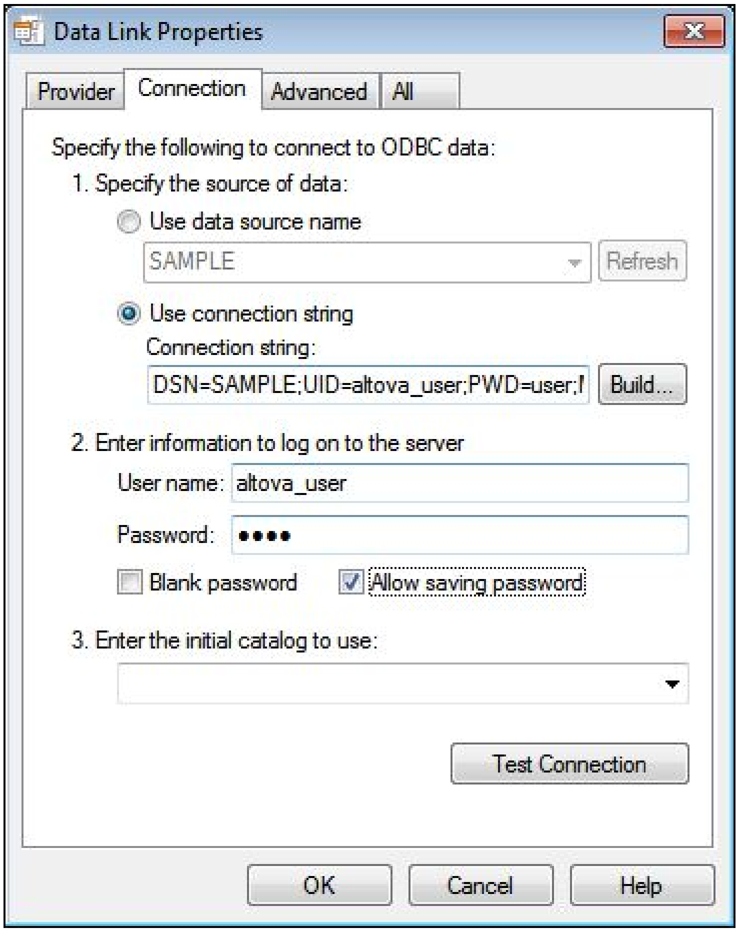
Data Link Properties

Select the data you want to connect to:

OLE DB Provider(s)

IBM OLE DB Provider for DB2 - DB2COPY1
Microsoft Jet 4.0 OLE DB Provider
Microsoft Office 12.0 Access Database Engine OLE DB Pro
Microsoft OLE DB Provider for Analysis Services 10.0
Microsoft OLE DB Provider for Indexing Service
Microsoft OLE DB Provider for ODBC Drivers
Microsoft OLE DB Provider for Oracle
Microsoft OLE DB Provider for Search
Microsoft OLE DB Provider for SQL Server
Microsoft OLE DB Simple Provider
MSDataShape
OLE DB Provider for Microsoft Directory Services
SQL Native Client

2. Select Microsoft OLE DB Provider for ODBC Drivers from the list and click **Next**. The Connection tab is activated (*screenshot below*).
   (For Microsoft SQL DBs, we recommend *Microsoft OLE DB Provider for SQL Server.* For Oracle, MySQL, Sybase, and IBM DB2, we recommend *Microsoft OLE DB Provider for ODBC Drivers*.)


Data Link Properties

Specify the following to connect to ODBC data:

1. Specify the source of data:

○ Use data source name

SAMPLE

● Use connection string

Connection string: DSN=SAMPLE;UID=altova_user;PWD=user;

2. Enter information to log on to the server

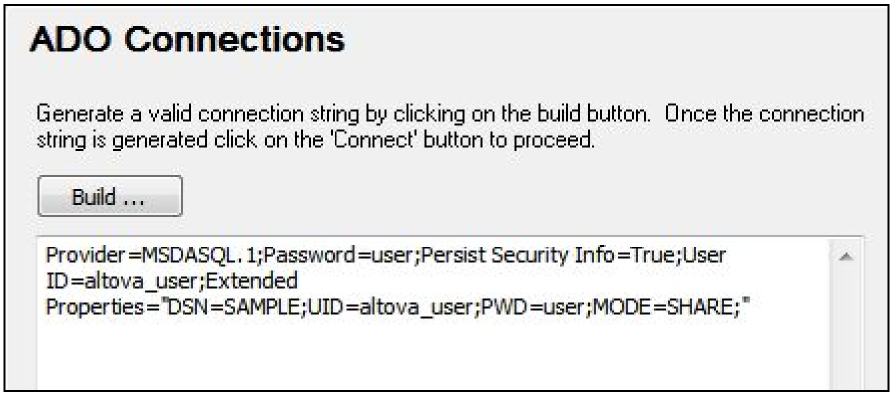
User name: altova_user

Password: ••••

□ Blank password ☑ Allow saving password

3. Enter the initial catalog to use:

3. Click **Build** to build a connection string. (The dialog which pops up prompts you for a data source and user information). In the case of some databases (such as Microsoft SQL Server), you do not build a connection string but instead select the server and database from combo box listings.
4. If login information is required, enter a user name and password, and check the *Allow Saving Password* check box.
5. Click **OK**. The Data Link Properties dialog closes. The connection string appears in the ADO Connections box (*screenshot below*).


ADO Connections

Generate a valid connection string by clicking on the build button. Once the connection string is generated click on the 'Connect' button to proceed.

```
Provider=MSDASQL.1;Password=user;Persist Security Info=True;User
ID=altova_user;Extended
Properties="DSN=SAMPLE;UID=altova_user;PWD=user;MODE=SHARE;"
```

6. Click **Connect**. The connection to the data source is established.

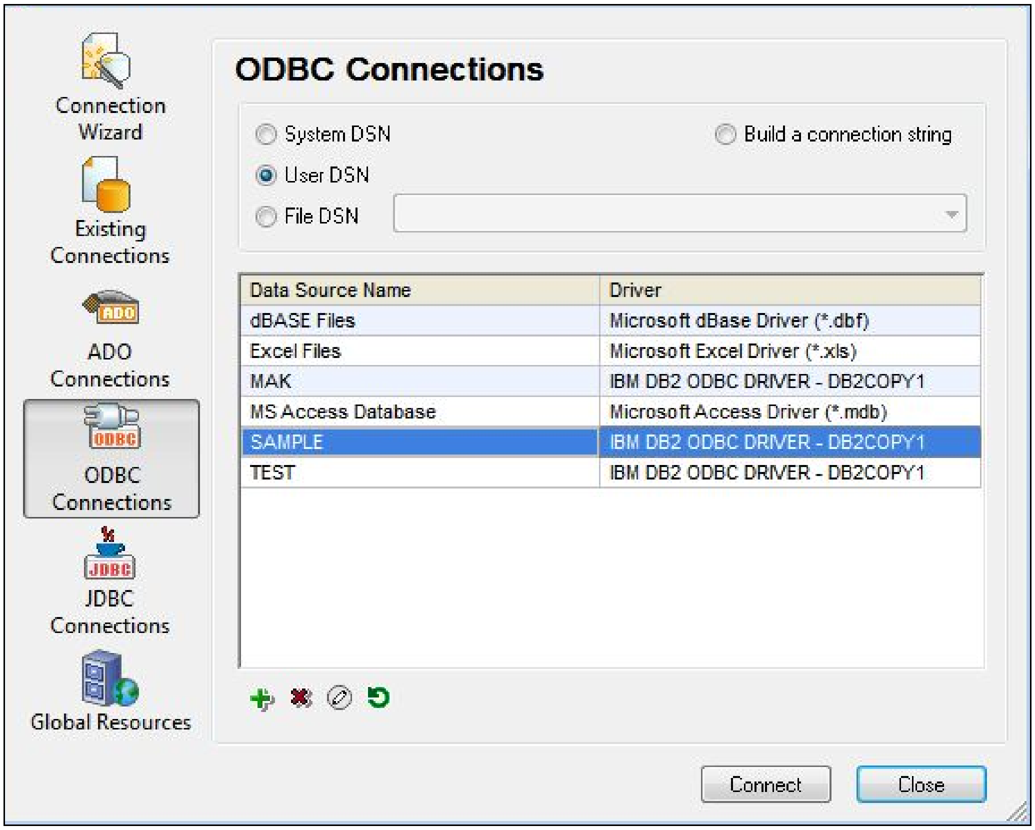
Note: For an ADO connection to MS SQL Server via the SQL Server Native Client 10.0 driver the following property values must be set in the *All* tab of the Data Link Properties dialog:

- Set the property value of Integrated Security to a space character
- Set the property value of Persist Security Info to `true`.

ODBC Connections

The connection via ODBC is described with an IBM DB2 database as the target DB. These steps also apply to other types of DBs that can be connected to via ODBC.

1. In the ODBC Connections pane (*screenshot below*), select one of the options for specifying a Data Source Name (DSN). If you select System DSN or User DSN, the available DSNs are displayed in the Data Source pane. If you select File DSN, you are prompted to browse for the DSN file. Alternatively, you can build a connection string to the DB by selecting the Build a Connection String option..

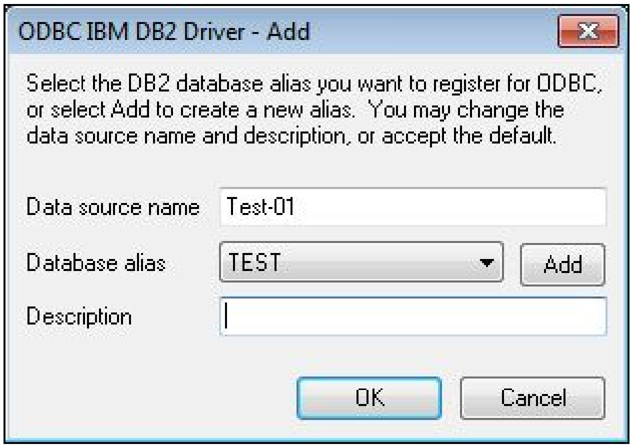


ODBC Connections

○ System DSN ○ Build a connection string
● User DSN
○ File DSN

Data Source Name	Driver
dBASE Files	Microsoft dBase Driver (*.dbf)
Excel Files	Microsoft Excel Driver (*.xls)
MAK	IBM DB2 ODBC DRIVER - DB2COPY1
MS Access Database	Microsoft Access Driver (*.mdb)
SAMPLE	IBM DB2 ODBC DRIVER - DB2COPY1
TEST	IBM DB2 ODBC DRIVER - DB2COPY1

2. If you wish to add a DSN to those in the Data Source pane, click the **Create a New DSN** icon ➕.
3. In the Create an ODBC DSN dialog that appears, select the required driver, then click the **User DSN** or **System DSN** button.
4. In the dialog that appears (*screenshot below*), select the DB alias and give it a DSN.



ODBC IBM DB2 Driver - Add

Select the DB2 database alias you want to register for ODBC, or select Add to create a new alias. You may change the data source name and description, or accept the default.

Data source name: Test-01

Database alias: TEST

Description:

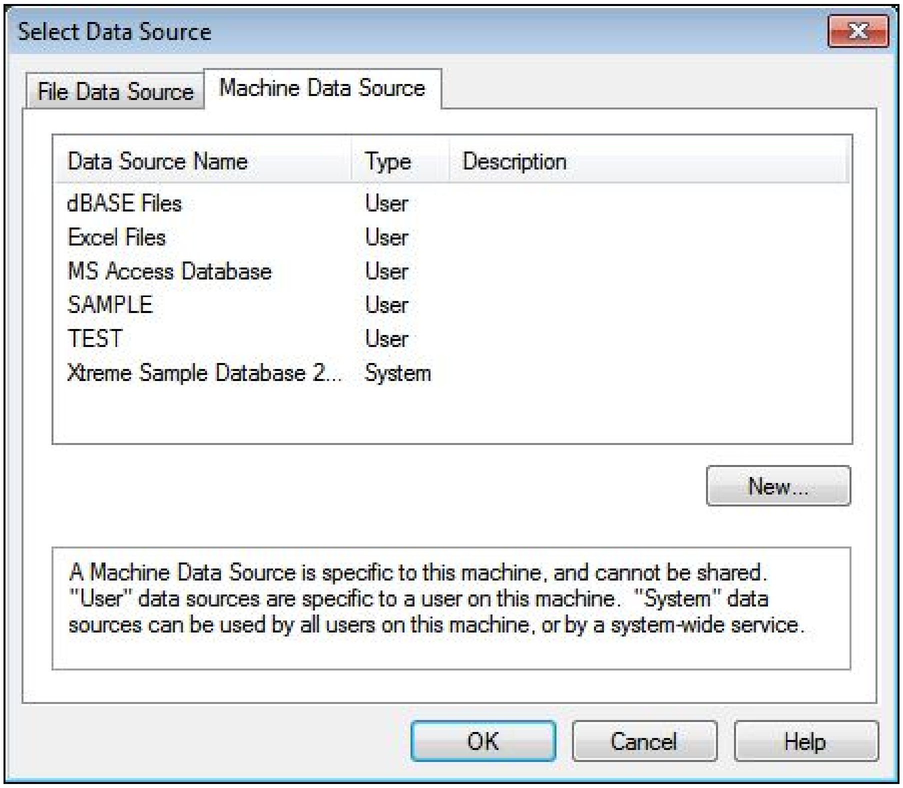
5. Click **OK** to finish. The DB is added as a Data Source to the list in the Data Source pane.
6. After you have selected the DataSource (via the System DSN or User DSN option), or selected a DSN File (File DSN option), or built a connection string (Connection String option), click **Connect**.
7. When you are prompted for your user ID and password, enter these and then click **OK**. The connection is established. For information about subsequent dialogs, refer to the description of the command being executed.

Note: The listing in the data source pane (when the System DSN or User DSN option is selected) can be edited using the *Edit Selected DSN*, *Remove Selected DSN*, and *Refresh Listed DSNs* icons at the bottom of the ODBC Connections screen.

⊘	Edit selected DSN.
✖	Remove selected DSN.
↻	Refresh listed DSNs.

Building a connection string

Select the Build a Connection String radio button and click the **Build** button. The Select Data Source dialog (*screenshot below*) pops up. You can either select a File DSN (in the File Data Source tab), or select a data source that is available on your machine (listed in the Machine Data Source tab).



Data Source Name	Type	Description
dBASE Files	User	
Excel Files	User	
MS Access Database	User	
SAMPLE	User	
TEST	User	
Xtreme Sample Database 2...	System	

A Machine Data Source is specific to this machine, and cannot be shared. "User" data sources are specific to a user on this machine. "System" data sources can be used by all users on this machine, or by a system-wide service.

Clicking **OK** pops up a dialog that prompts for user information, including the User ID and password. When you click **OK** in this dialog, the connection string is entered in the ODBC Connections screen.

JDBC Connections

The connection via JDBC is described with an IBM DB2 database as the target DB. These steps also apply to other types of DBs that can be connected to via JDBC.

Pre-connection steps

Before connecting to the DB via JDBC, you must do the following:

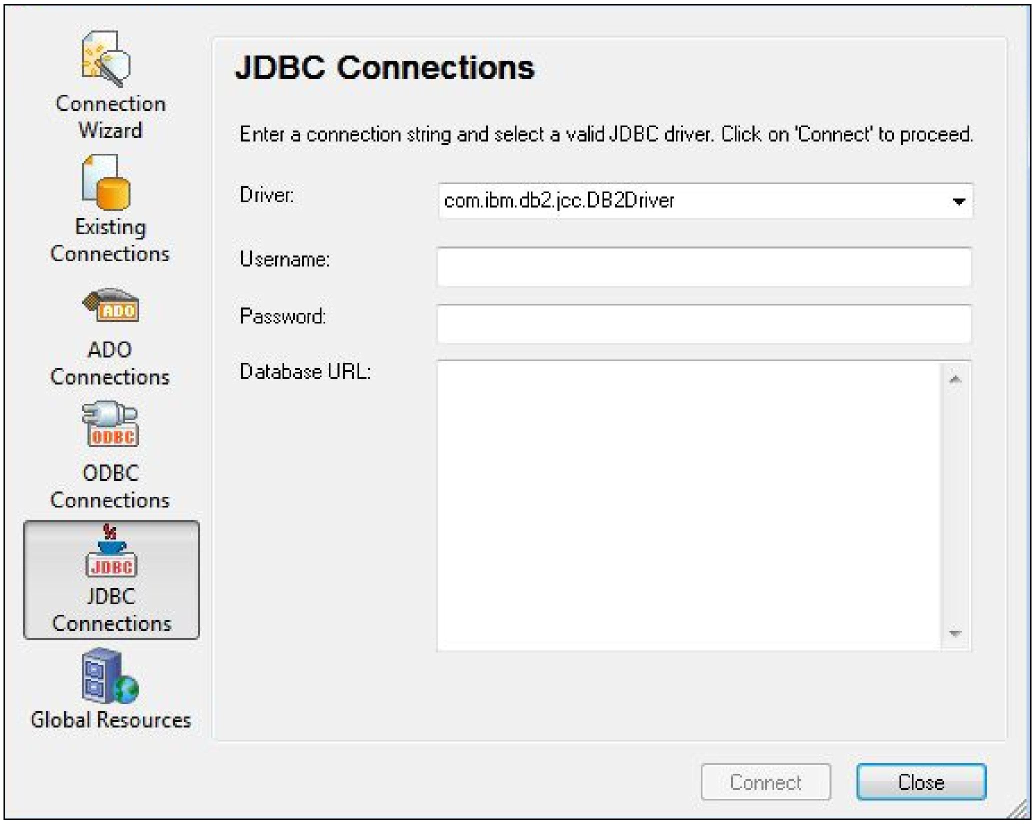
1. Ensure that the Java Runtime Environment (JRE) is installed. If it isn't installed, then install it. Use a 32-bit JRE for a 32-bit machine or a 64-bit JRE for a 64-bit machine.
2. Install a JDBC driver. No special installation setup is required. All you need to do is copy the driver to a local directory, for example, `c:\jdbc`. Note that JDBC drivers (which are Jar files) are platform-independent.
3. Set up the `CLASSPATH` to include the location where the JDBC driver is located. (The application reads the `CLASSPATH` environment variable to locate the JDBC driver.) To access and edit the `CLASSPATH`, do the following: Click **Start | Control Panel | System | Advanced | Environment Variables**. In the Environment Variables dialog that appears, select either the `CLASSPATH` user environment variable or the `CLASSPATH` system environment variable and click the **Edit** button. Add the path to the JDBC driver to the `CLASSPATH`. For example: `CLASSPATH=C:\jdbc\sqljdbc.jar; C:\jdbc\db2jcc.jar;`.
4. Log off and then log on again to make the `CLASSPATH` changes active.

Note: Installing the complete IBM DB2 or Oracle client will automatically populate the `CLASSPATH` with the JDBC drivers of the respective packages. For more details, see the summaries below.

Connecting via JDBC

To connect using a JDBC connection, do the following:

1. In the JDBC Connections pane (*screenshot below*) appears. Select a JDBC driver from the *Driver* dropdown list (all detected drivers, i.e. those in the `CLASSPATH` environment variable, are listed).



JDBC Connections

Enter a connection string and select a valid JDBC driver. Click on 'Connect' to proceed.

Driver: com.ibm.db2.jcc.DB2Driver

Username:

Password:

Database URL:

2. Enter a connection string in the *Database URL* text box and a user name and password as required. Given below is the connection string syntax for commonly used databases, each with an example connection string.

Oracle	`jdbc:oracle:thin:[user/password]@//[host][:port]/SID`
	`jdbc:oracle:thin:@//abcd234/ORA11`

IBM DB2	`jdbc:db2://host_name:port/dbname` `jdbc:db2://MyDB2:50000/boz`
MySQL	`jdbc:mysql://host_name:port/dbname` `jdbc:mysql://MyDB2:3306/moz`

MSSQL	`jdbc:sqlserver://host:port;databasename=name;user=name;password=Pwd`
	`jdbc:sqlserver://abcd38:1433;databasename=coz`
	`jdbc:sqlserver://Q5;DatabaseName=coz;SelectMethod=Cursor`

PostgreSQL	`jdbc:postgresql://host:port/database` `jdbc:postgresql://abc993:5432/qanoz`
Sybase	`jdbc:sybase:Tds:host:port/dbname` `jdbc:sybase:Tds:abc12:2048/QUE`

3. Click the **Connect** button. The connection to the data source is established. For information about subsequent dialogs, refer to the description of the command being executed.

Step-by-step: MSSQL, MySQL, PostGre, and other non-XML DBs
Given below is a step-by-step guide for connecting to a non-XML DB via JDBC:

1. JDBC JAR files (driver files): Copy the files to any local location. Then add the full path and filename to the Windows `CLASSPATH` variable. For example: `C:\jdbc\sqljdbc.jar;C:\jdbc\db2jcc.jar;`.
2. Log off and log on to make the `CLASSPATH` changes active.
3. Start XMLSpy and access the Connect to a Data Source dialog.
4. The JDBC Connections pane lists, in the *Driver* dropdown box, the detected JDBC drivers. If the dropdown box is empty, make sure that the file `altovadb.jar` is present in the folder: `C:\Program Files\Altova\Common2015\jar`.
5. Connect to a database as described in the section *Connecting via JDBC* above.

Step-by-step: Oracle
Given below is a step-by-step guide for connecting to an Oracle DB via JDBC. If you don't need the XML and XDB features of the Oracle DB, follow the instructions in the step-by-step guide for non-XML DBs. The installation folder of the Oracle client is indicated in the steps below by the placeholder: `%ORACLE_HOME%`.

1. Install Oracle client software with OCI and ODBC features enabled. If an Oracle client is already installed check if the two jar files below are present:

   ```
   %ORACLE_HOME%\LIB\xmlparserv2.jar
   %ORACLE_HOME%\RDBMS\jlib\xdb.jar
   ```

2. Add the following files to the Windows `CLASSPATH` environment variable:

   ```
   %ORACLE_HOME%\jdbc\lib\ojdbc6.jar
   %ORACLE_HOME%\LIB\xmlparserv2.jar
   %ORACLE_HOME%\RDBMS\jlib\xdb.jar
   ```

3. Log off and log on to make the `CLASSPATH` changes active.
4. Start XMLSpy and access the Connect to a Data Source dialog.
5. The JDBC Connections pane lists, in the *Driver* dropdown box, the detected JDBC drivers. If the dropdown box is empty, make sure that the file `altovadb.jar` is present in the folder: `C:\Program Files\Altova\Common2015\jar`.
6. Connect to a database as described in the section *Connecting via JDBC* above.

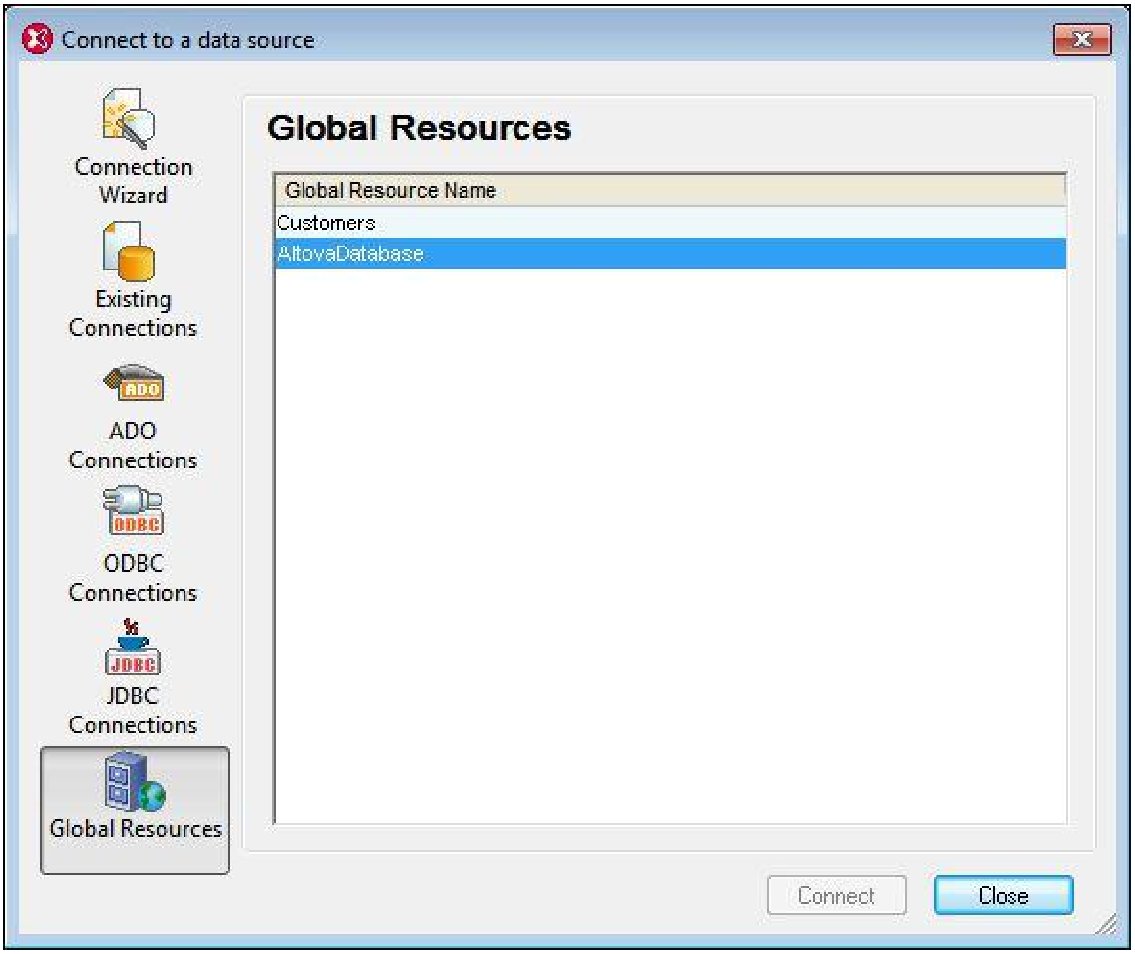
Step-by-step: IBM DB2
Given below is a step-by-step guide for connecting to an IBM DB2 database via JDBC.

1. If an IBM DB2 client has already been installed, nothing needs to be done since the `CLASSPATH` will have been set by the installation process.
2. If no IBM DB2 client has been installed, add the IBM DB2 JDBC driver jar files `db2jcc.jar` and `db2jcc_license_cu.jar` to the Windows `CLASSPATH`. Log off and log on to make the `CLASSPATH` changes active.
3. Start XMLSpy and access the Connect to a Data Source dialog.
4. The JDBC Connections pane lists, in the *Driver* dropdown box, the detected JDBC drivers. If the dropdown box is empty, make sure that the file `altovadb.jar` is present in the folder: `C:\Program Files\Altova\Common2015\jar`.
5. Connect to a database as described in the section *Connecting via JDBC* above.

Note: When databases are connected to via JDBC, due to insufficient information returned by the drivers: (i) data editing is not possible for tables without a primary key; (ii) Execute for data-editing in SQL Editor will not work.

Global Resources

In the Connect to Data Source dialog, the Global Resources pane (*screenshot below*) displays all the database-type global resources that are defined in the currently active Global Resources XML File.

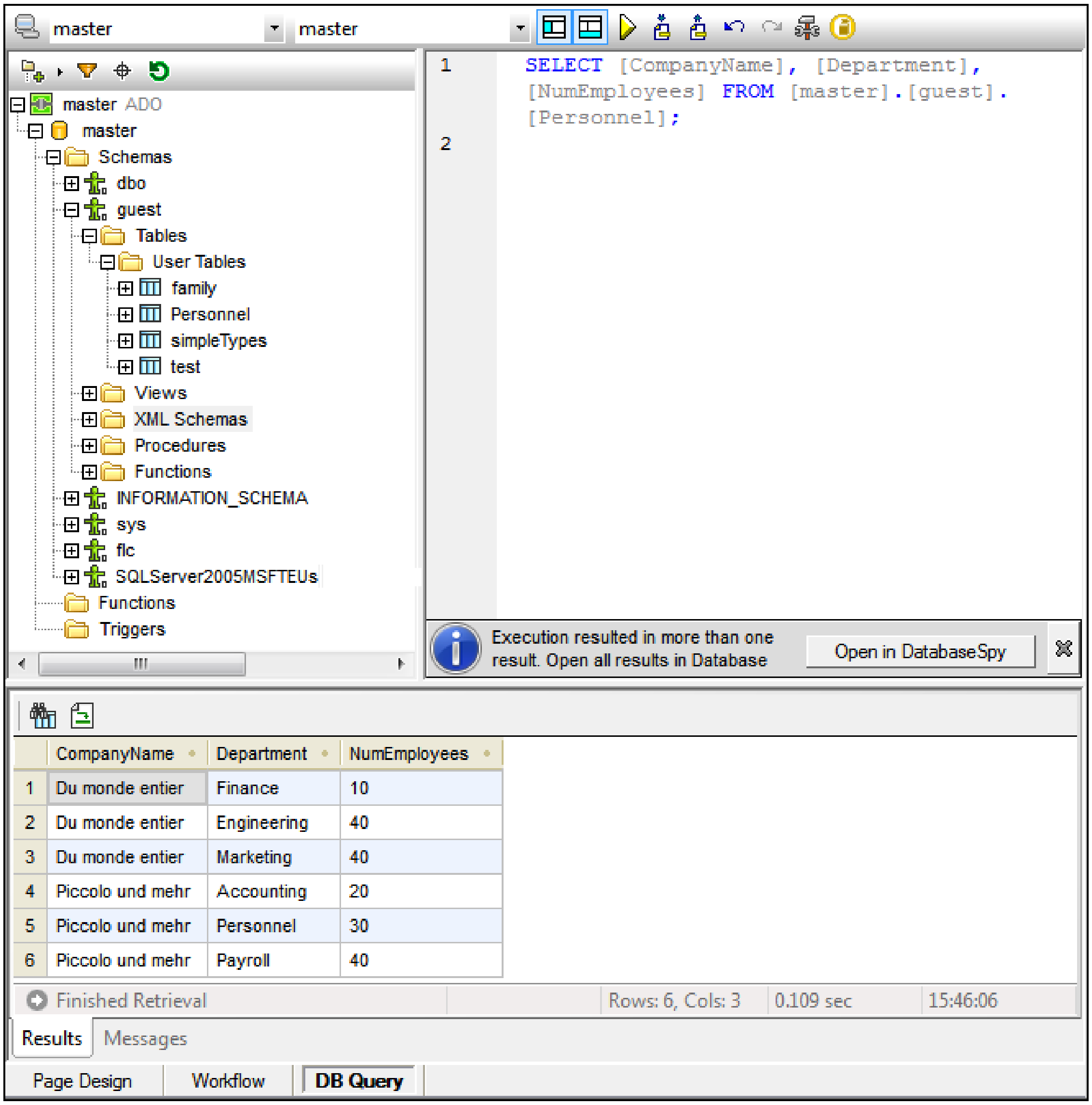


Global Resource Name
Customers
AltovaDatabase

Select the required global resource and click **Connect**. The connection to the database is established.

8.6 Database Query

The **DB Query View** (*screenshot below*) enables you to directly query any major database from within the MobileTogether Designer GUI. The database could be a data source referenced in the active document or an external database. Note that each DB Query pane is associated with the currently active design. The DB Query pane of a design can have connections to multiple databases open at a time. There can also be multiple designs open at a time in MobileTogether Designer. Queries and actions defined in the DB Query View are independent of other MobileTogether Designer tabs, and are not saved as part of the design file.



```
SELECT [CompanyName], [Department],
[NumEmployees] FROM [master].[guest].
[Personnel];
```

Execution resulted in more than one result. Open all results in Database

	CompanyName	Department	NumEmployees
1	Du monde entier	Finance	10
2	Du monde entier	Engineering	40
3	Du monde entier	Marketing	40
4	Piccolo und mehr	Accounting	20
5	Piccolo und mehr	Personnel	30
6	Piccolo und mehr	Payroll	40

Finished Retrieval Rows: 6, Cols: 3 0.109 sec 15:46:06

The Database Query mechanism

The Database Query mechanism is as follows. (It is described in detail in the sub-sections of this section.)

1. A connection to the database is established via the *Database Query* | *Connect to a Data Source* window.
2. The connected database or parts of it are displayed in the Browser pane, which can be configured to suit viewing requirements.
3. A query written in a syntax appropriate to the database to be queried is entered in the Query pane, and the query is executed.
4. The results of the query can be viewed through various filters.

Supported databases
The following databases are supported. The available root object for each database is also listed. While Altova endeavors to support other ODBC/ADO databases, successful connection and data processing have only been tested with the databases listed below. If your Altova application is a 64-bit version, ensure that you have access to the 64-bit database drivers needed for the specific database you are connecting to. Your Altova application supports logical files of the IBM iSeries database and shows logical files as views.

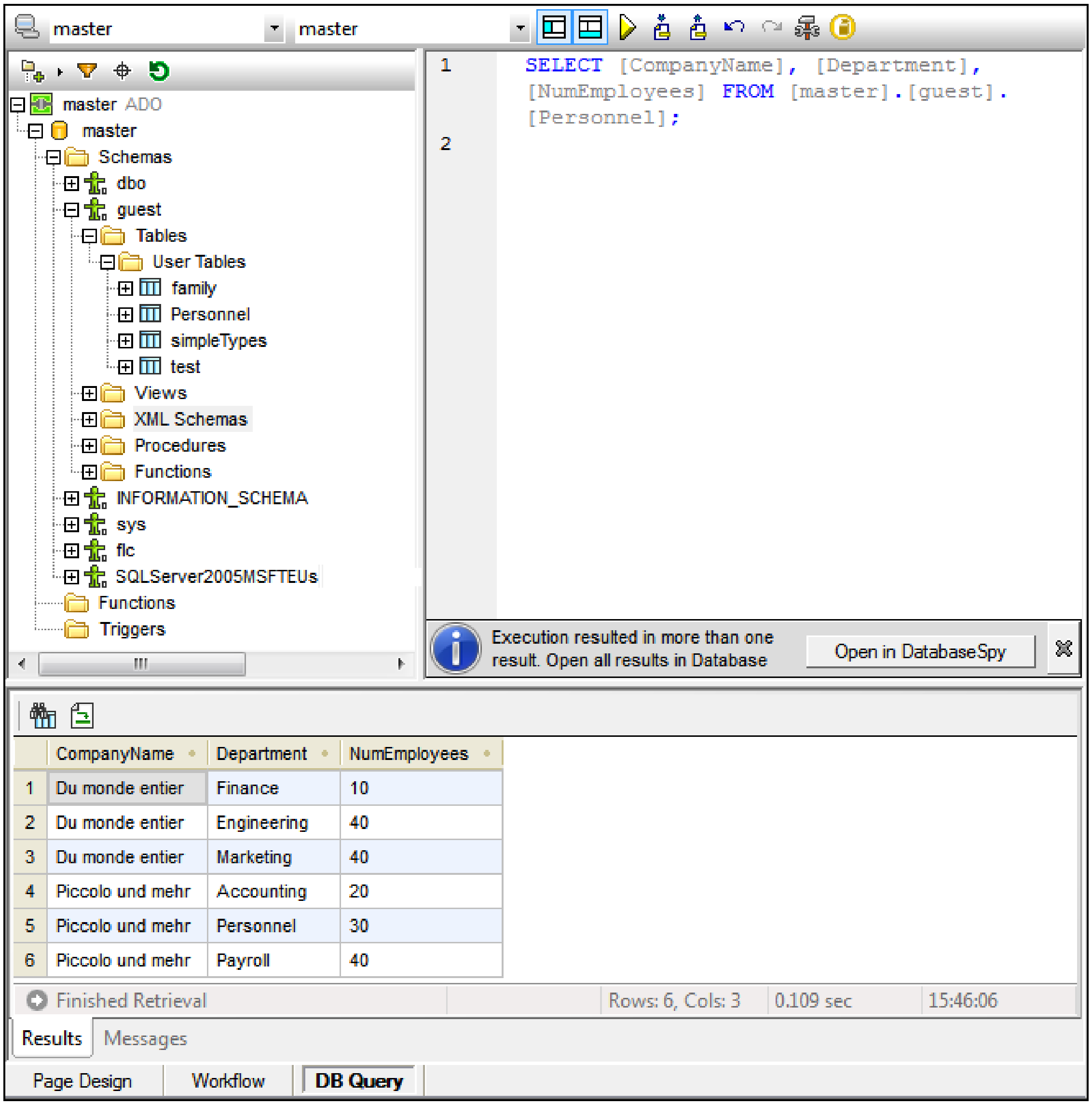
Database (natively supported)	Root Object
MS Access 2003, 2007, 2010, 2013	database
MS SQL Server 2005, 2008, 2012	database/schema
Oracle 9i, 10g, 11g	schema
MySQL 5.x, 5.5.28	database
PostgreSQL 8.0, 8.1, 8.2, 8.3, 9.0.10, 9.1.6, 9.2.1	database
Sybase ASE12, ASE15	database
IBM DB2 8.x, 9.5, 9.7, 10.1	schema
IBM DB2 for i 6.1, 7.1	schema
IBM Informix 11.70	database

GUI Overview and Toolbar

The **DB Query View** (*screenshot below*) is divided into the following parts:

- The Browser window at left, which displays connection info and database tables
- The SQL Editor window (Query window), to the right of the Browser window, contains your SQL queries
- The Results tab of the Result/Messages windows displays the query results in tabular form
- The Messages tab of the Result/Messages windows displays warnings or error messages

These windows are described in detail in the sub-sections of this section.



```
SELECT [CompanyName], [Department],
[NumEmployees] FROM [master].[guest].
[Personnel];
```

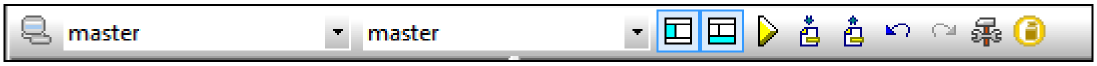
Execution resulted in more than one result. Open all results in Database

	CompanyName	Department	NumEmployees
1	Du monde entier	Finance	10
2	Du monde entier	Engineering	40
3	Du monde entier	Marketing	40
4	Piccolo und mehr	Accounting	20
5	Piccolo und mehr	Personnel	30
6	Piccolo und mehr	Payroll	40

Finished Retrieval Rows: 6, Cols: 3 0.109 sec 15:46:06

The DB Query View toolbar

The DB Query View toolbar (*screenshot below*) is located at the top of the view and provides icons for important commands related to the view.



master master

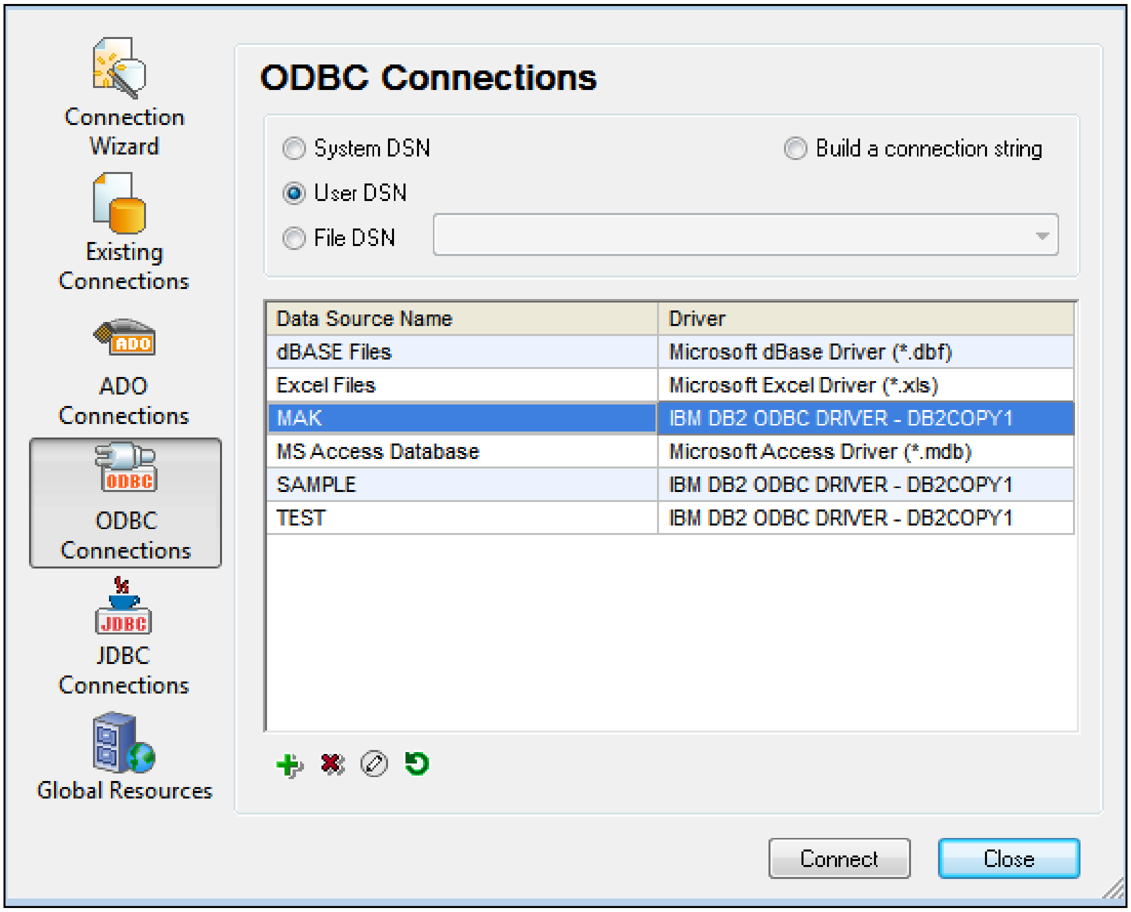
Icon	Command	Does this
🖥	**Quick Connect**	Starts the database connection wizard.
◫	**Toggle Browser**	Toggles the Browser window on and off.
⊟	**Toggle Results**	Toggles the Results/Messages window on and off.
▷	**Execute Query**	Executes currently selected SQL statement. If script contains multiple statements and none is selected, then all are executed.
⇩	**Import SQL File**	Opens an SQL file in the SQL Editor.
⇧	**Export SQL File**	Saves SQL queries to an SQL file.
↶	**Undo**	Undoes an unlimited number of edits in SQL Editor.
↷	**Redo**	Redoes an unlimited number of edits in SQL Editor.
⚙	**Options**	Open the Options dialog of SQL Editor.
●	**Open Query in DatabaseSpy**	Opens the SQL script in Altova's DatabaseSpy application.

Connecting to Data Sources

In order to query a database, you have to first connect to the required database, and then select the required data source and root object from among multiple existing connections.

Connecting to a database

When you switch to DB Query View, any database that the active design uses as a data source is displayed in the Browser pane. If you wish to query some other database, click **Quick Connect** in the DB Query View toolbar.



ODBC Connections

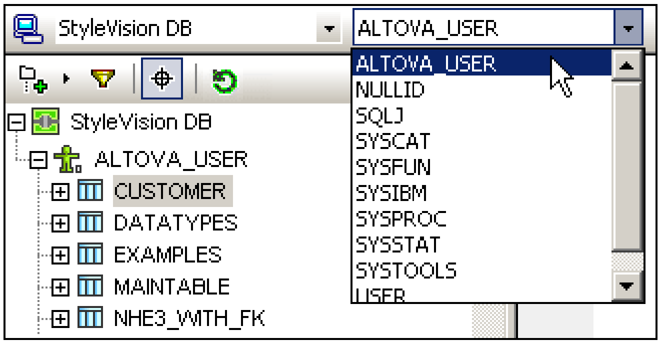
○ System DSN ○ Build a connection string
● User DSN
○ File DSN

Data Source Name	Driver
dBASE Files	Microsoft dBase Driver (*.dbf)
Excel Files	Microsoft Excel Driver (*.xls)
MAK	IBM DB2 ODBC DRIVER - DB2COPY1
MS Access Database	Microsoft Access Driver (*.mdb)
SAMPLE	IBM DB2 ODBC DRIVER - DB2COPY1
TEST	IBM DB2 ODBC DRIVER - DB2COPY1

For a list of supported databases, see Database Query | Supported databases.

Selecting the required data source

All the existing connections and the root objects of each are listed, respectively, in two combo boxes in the toolbar of the Database Query window (*screenshot below*). After selecting the required data source in the left-hand combo box, you can select the required root object from the right-hand combo box.



In the screenshot above, the database with the name `StyleVision DB` has been selected. Of the available root objects for this database, the root object `ALTOVA_USER` has been selected. The database and the root object are then displayed in the Browser pane.

Browser Pane

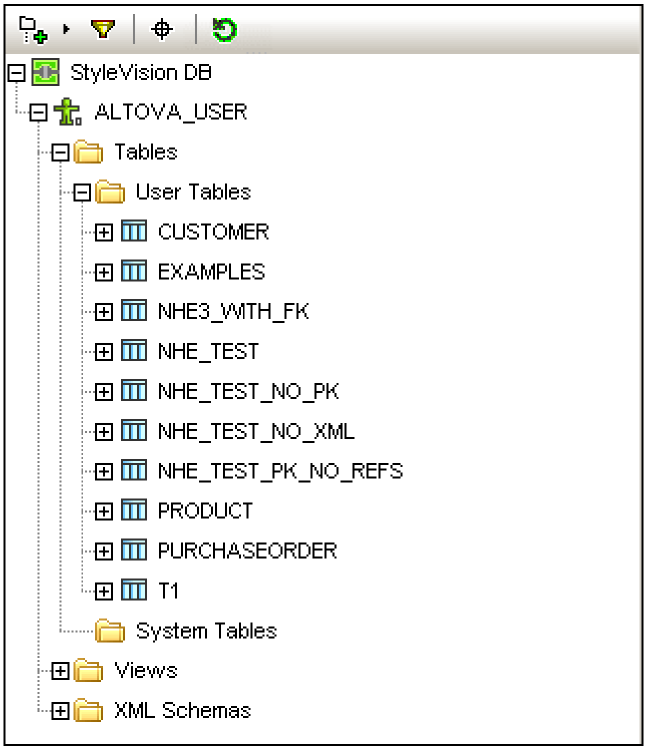
The Browser pane provides an overview of objects in the selected database. This overview includes database constraint information, such as whether a column is a primary or foreign key. In IBM DB2 version 9 and higher databases, the Browser additionally shows registered XML schemas in a separate folder (*see screenshot below*).

This section describes the following:

- The layouts available in the Browser pane.
- How to filter database objects.
- How to find database objects.
- How to refresh the root object of the active data source.

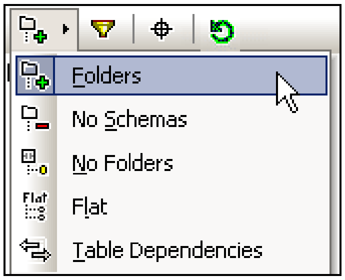
Browser pane layouts

The default Folders layout displays database objects hierarchically. Depending on the selected object, different context menu options are available when you right-click an item.



```
StyleVision DB
  ALTOVA_USER
    Tables
      User Tables
        CUSTOMER
        EXAMPLES
        NHE3_WITH_FK
        NHE_TEST
        NHE_TEST_NO_PK
        NHE_TEST_NO_XML
        NHE_TEST_PK_NO_REFS
        PRODUCT
        PURCHASEORDER
        T1
      System Tables
    Views
    XML Schemas
```

To select a layout for the Browser, click the Layout icon in the toolbar of the Browser pane (*screenshot below*) and select the layout from the drop-down list. Note that the icon changes with the selected layout.



The available layouts are:

- *Folders:* Organizes database objects into folders based on object type in a hierarchical tree, this is the default setting.
- *No Schemas:* Similar to the Folders layout, except that there are no database schema folders; tables are therefore not categorized by database schema.
- *No Folders:* Displays database objects in a hierarchy without using folders.
- *Flat:* Divides database objects by type in the first hierarchical level. For example, instead of columns being contained in the corresponding table, all columns are displayed in a separate Columns folder.
- *Table Dependencies:* Categorizes tables according to their relationships with other tables. There are categories for tables with foreign keys, tables referenced by foreign keys and tables that have no relationships to other tables.

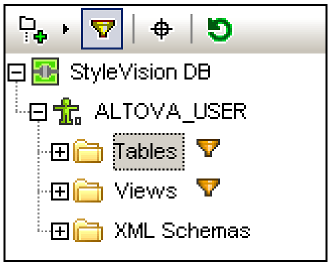
To sort tables into User and System tables, switch to Folders, No Schemas or Flat layout, then right-click the Tables folder and select **Sort into User and System Tables**. The tables are sorted alphabetically in the User Tables and System Tables folders.

Filtering database objects

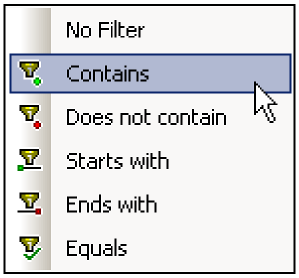
In the Browser pane (in all layouts except No Folders and Table Dependencies), schemas, tables, and views can be filtered by name or part of a name. Objects are filtered as you type in the characters, and filtering is case-insensitive by default.

To filter objects in the Browser, do the following:

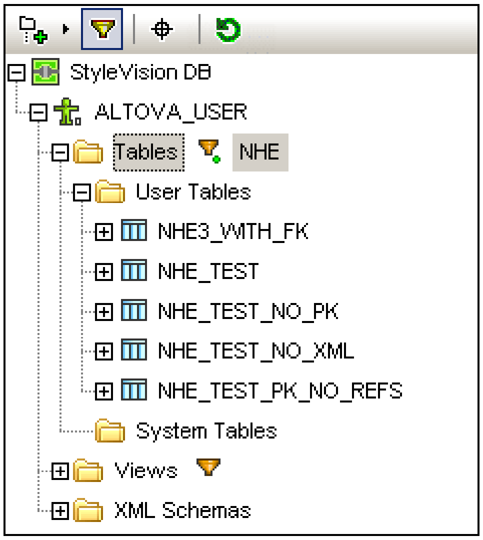
1. Click the Filter Folder Contents icon in the toolbar of the Browser pane. Filter icons appear next to the Tables and Views folders in the currently selected layout (*screenshot below*).



2. Click the filter icon next to the folder you want to filter, and select the filtering option from the popup menu, for example, `Contains`.

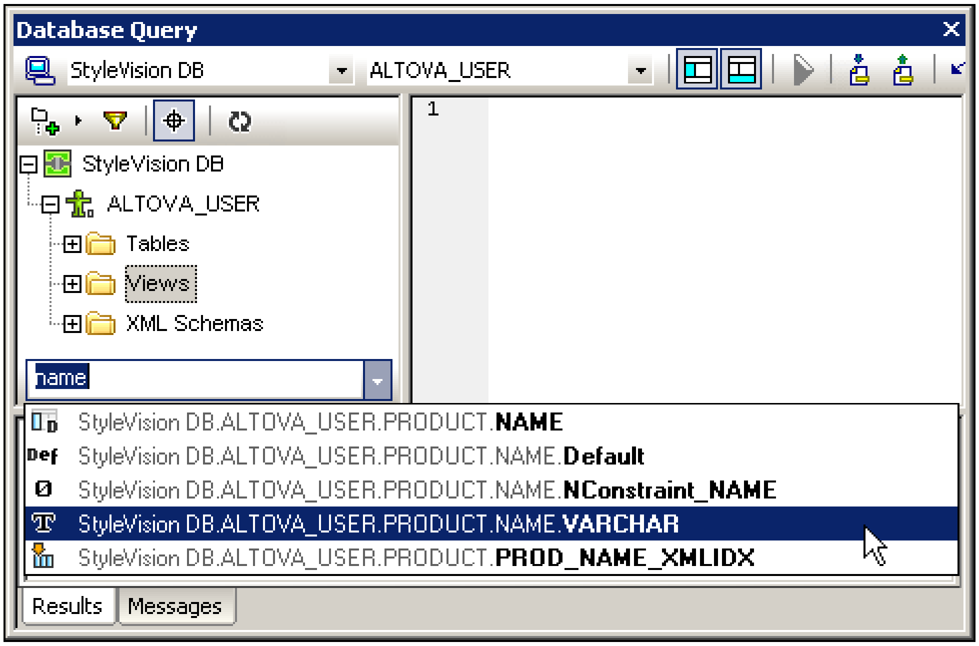


3. In the entry field that appears, enter the filter string (in the screenshot below, the filter string on the `Tables` folder is `NHE`). The filter is applied as you type.



Finding database objects
To find a specific database item by its name, you can use the Browser pane's Object Locator. This works as follows:

1. In the toolbar of the Browser pane, click the Object Locator icon. A drop-down list appears at the bottom of the Browser pane.
2. Enter the search string in the entry field of this list, for example `name` (*screenshot below*). Clicking the drop-down arrow displays all objects that contain the search string.



name

StyleVision DB.ALTOVA_USER.PRODUCT.**NAME**
StyleVision DB.ALTOVA_USER.PRODUCT.NAME.**Default**
StyleVision DB.ALTOVA_USER.PRODUCT.NAME.**NConstraint_NAME**
StyleVision DB.ALTOVA_USER.PRODUCT.NAME.**VARCHAR**
StyleVision DB.ALTOVA_USER.PRODUCT.**PROD_NAME_XMLIDX**

3. Click the object in the list to see it in the Browser pane.

Refreshing the root object

The root object of the active data source can be refreshed by clicking the **Refresh** button of the Browser pane's toolbar.

Query Pane: Description

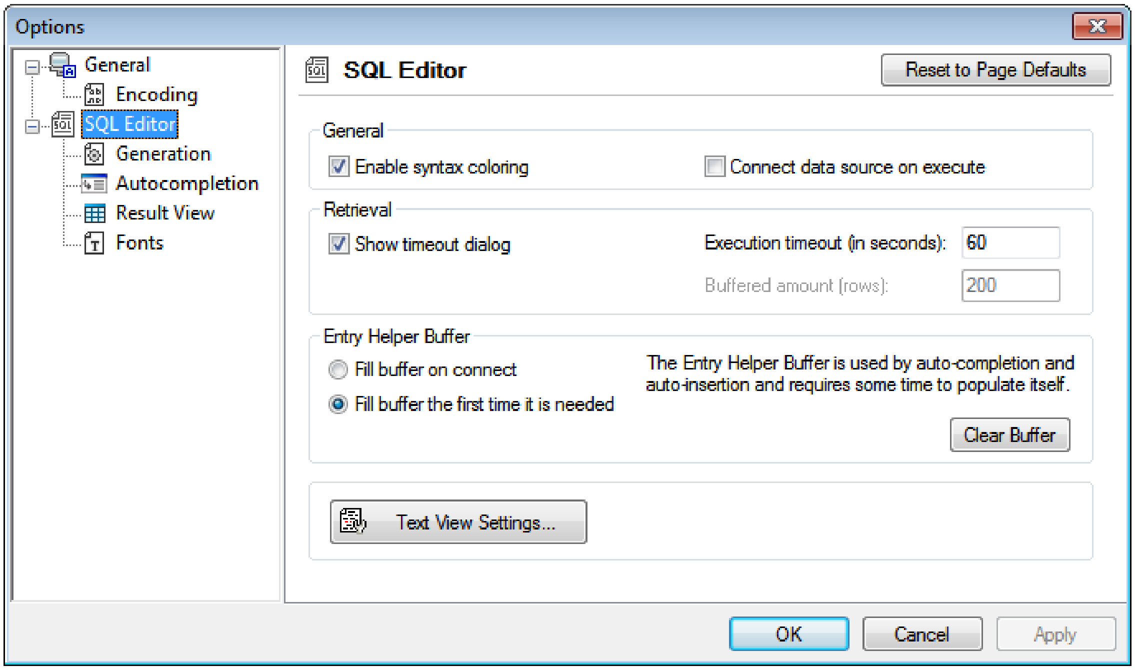
The Query pane is an intelligent SQL editor for entering queries to the selected database. After entering the query, clicking the Execute command of the Database Query window executes the query and displays the result and execution messages in the Results/Messages pane. How to work with queries is described in the next section, Query Pane: Working with Queries. In this section, we describe the main features of the Query pane:

- SQL Editor icons in the Database Query toolbar
- SQL Editor options
- Auto-completion of SQL statements
- Definition of regions in an SQL script
- Insertion of comments in an SQL script
- Use of bookmarks

SQL Editor icons in the Database Query toolbar
The following icons in the toolbar of the Database Query window are used when working with the SQL Editor:

Icon	Command	Does this
▷	**Execute Query**	Executes currently selected SQL statement. If script contains multiple statements and none is selected, then all are executed.
	Import SQL File	Opens an SQL file in the SQL Editor.
	Export SQL File	Saves SQL queries to an SQL file.
↶	**Undo**	Undoes an unlimited number of edits in SQL Editor.
↷	**Redo**	Redoes an unlimited number of edits in SQL Editor.
	Options	Open the Options dialog of SQL Editor.
	Open Query in DatabaseSpy	Opens the SQL script in Altova's DatabaseSpy application.

SQL Editor options
Clicking the **Options** icon in the Database Query toolbar pops up the Options dialog (*screenshot below*). A page of settings can be selected in the left-hand pane, and the options on that page can be selected. Click the **Reset to Page Defaults** button to reset the options on that page to their original settings.



General

☑ Enable syntax coloring

□ Connect data source on execute

Retrieval

☑ Show timeout dialog

Execution timeout (in seconds): 60

Buffered amount (rows): 200

Entry Helper Buffer

○ Fill buffer on connect

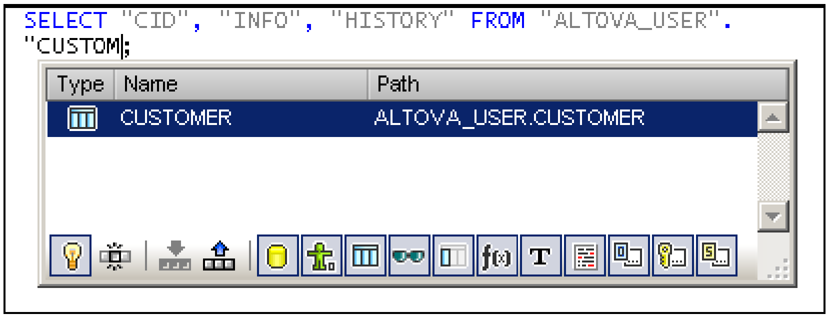
● Fill buffer the first time it is needed

The Entry Helper Buffer is used by auto-completion and auto-insertion and requires some time to populate itself.

The key settings are as follows:

- ***General* | *Encoding:*** Options for setting the encoding of new SQL files, of existing SQL files for which the encoding cannot be detected, and for setting the Byte Order Mark (BOM). (If the encoding of existing SQL files can be detected, the files are opened and saved without changing the encoding.)
- ***SQL Editor:*** Options for toggling syntax coloring and data source connections on execution on/off. A timeout can be set for query execution, and a dialog to change the timeout can also be shown if the specified time is exceeded. The buffer for the entry helper information can be filled on connection to the data source or the first time it is needed.
- ***SQL Editor* | *SQL Generation:*** The application generates SQL statements when you drag objects from the Browser pane into the Query pane. Options for SQL statement generation can be set in the SQL generation tab. Use the *Database* list box to select a database kind and set the statement generation options individually for the different database kinds you are working with. Activating the *Apply to all databases* check box sets the options that are currently selected for all databases. Options include appending semi-colons to statements and surrounding identifiers with escape characters.
- ***SQL Editor* | *Auto-completion:*** The Auto-Completion feature works by suggesting, while you type, relevant entries from various SQL syntax categories. It is available for the following databases: MS SQL Server 2000, 2005, and 2008, MS Access 2003 and 2007, and IBM DB2 v.9. When the Auto-Completion option is switched on, the Auto-Completion window (*screenshot below*) appears, containing suggestions for auto-completion. Select the required entry to insert it. You can define whether the autocompletion popup should be triggered automatically after a delay which you can set in the *Triggering Auto-completion* pane, or if the popup has to be invoked manually. You can also select the keys to be used to insert the selected completion. The SQL Editor can intelligently suggest autocompletion entries based on language statistics. If this feature is activated, items that are frequently used appear on top of the list of suggested entries. In the Auto-completion window (*screenshot below*) itself, note the buttons at the bottom of the window. The *Context-Sensitive Suggestion* button sets whether only entries that are

relevant to the context are displayed or all possible entries with that spelling. The *Single Mode* button enables you to click a category button to select only that category. The *Set All Categories* button selects all categories. You can then deselect a category by clicking its button. The *Clear All Categories* button de-selects all categories. The other buttons are the various category buttons.



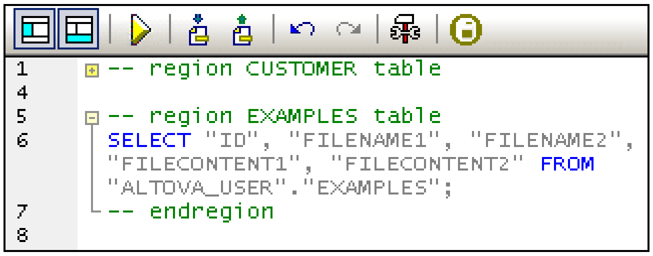
```
SELECT "CID", "INFO", "HISTORY" FROM "ALTOVA_USER".
"CUSTOM;
```

Type	Name	Path
	CUSTOMER	ALTOVA_USER.CUSTOMER

- ***SQL Editor | Result View:*** Options to configure the Result tab.
- ***SQL Editor | Fonts:*** Options for setting the font style of the text in the Text Editor and in the Result View.

Definition of regions in an SQL script

Regions are sections in SQL scripts that are marked and declared to be a unit. Regions can be collapsed and expanded to hide or display parts of the script. It is also possible to nest regions within other regions. Regions are delimited by **`--region`** and **`--endregion`** comments, respectively, before and after the region. Regions can optionally be given a name, which is entered after the **`--region`** delimiter (*see screenshot below*).



```
1   ⊞ -- region CUSTOMER table
4
5   ⊟ -- region EXAMPLES table
6     SELECT "ID", "FILENAME1", "FILENAME2",
      "FILECONTENT1", "FILECONTENT2" FROM
      "ALTOVA_USER"."EXAMPLES";
7     -- endregion
8
```

To insert a region, select the statement/s to be made into a region, right-click, and select **Insert Region**. The expandable/collapsible region is created. Add a name if you wish. In the screenshot above, also notice the line-numbering. To remove a region, delete the two **`--region`** and **`--endregion`** delimiters.

Insertion of comments in an SQL script

Text in an SQL script can be commented out. These portions of the script are skipped when the script is executed.

- To comment out a block, mark the block, right-click, and select **Insert/Remove Block**

Comment. To remove the block comment, mark the comment, right-click and select **Insert/Remove Block Comment**.

- To comment out a line or part of a line, place the cursor at the point where the line comment should start, right-click, and select **Insert/Remove Line Comment**. To remove the line comment, mark the comment, right-click and select **Insert/Remove Line Comment**.

Use of bookmarks

Bookmarks can be inserted at specific lines, and you can then navigate through the bookmarks in the document. To insert a bookmark, place the cursor in the line to be bookmarked, right-click, and select **Insert/Remove Bookmark**. To go to the next or previous bookmark, right-click, and select **Go to Next Bookmark** or **Go to Previous Bookmark**, respectively. To remove a bookmark, place the cursor in the line for which the bookmark is to be removed, right-click, and select **Insert/Remove Bookmark**. To remove all bookmarks, right-click, and select **Remove All Bookmarks**.

Query Pane: Working With

After connecting to a database, an SQL script can be entered in the SQL Editor and executed. This section describes:

- How an SQL script is entered in the SQL Editor.
- How the script is executed in the Database Query window.

The following icons are referred to in this section:

▷	**Execute Query**	Executes currently selected SQL statement. If script contains multiple statements and none is selected, then all are executed.
⇩	**Import SQL File**	Opens an SQL file in the SQL Editor.

Creating SQL statements and scripts in the SQL Editor

The following GUI methods can be used to create SQL statements or scripts:

- *Drag and drop:* Drag an object from the Browser pane into the SQL Editor. An SQL statement is generated to query the database for that object.
- *Context menu:* Right-click an object in the Browser pane and select **Show in SQL Editor | Select**.
- *Manual entry:* Type SQL statements directly in SQL Editor. The Auto-completion feature can help with editing.
- *Import an SQL script:* Click the **Import SQL File** icon in the toolbar of the Database Query window.

Executing SQL statements

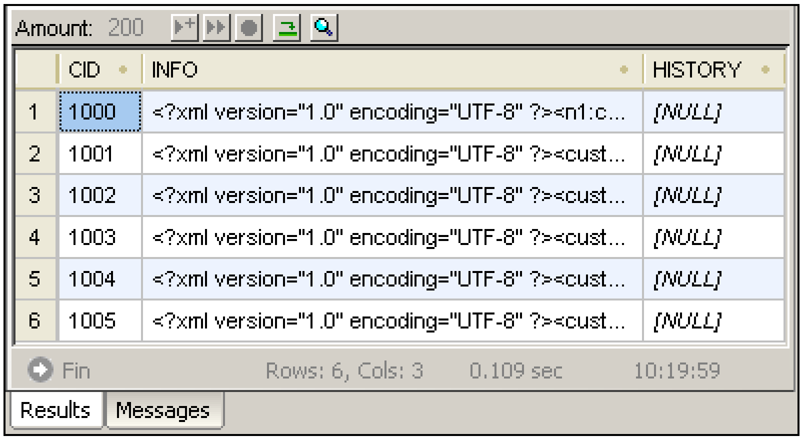
If the SQL script in the SQL Editor has more than one SQL statement, select the statement to execute and click the **Execute** icon in the toolbar of the Database Query window. If no statement in the SQL script is selected, then all the statements in the script are executed. The database data is retrieved and displayed as a grid in the Results tab. Messages about the execution are displayed in the Messages tab.

Results and Messages Pane

The Results/Messages pane has two tabs:

- The Results tab shows the data that is retrieved by the query.
- The Messages tab shows messages about the query execution.

Results tab
The data retrieved by the query is displayed in the form of a grid in the Results tab(*screenshot below*).



	CID	INFO	HISTORY
1	1000	<?xml version="1.0" encoding="UTF-8" ?><n1:c...	[NULL]
2	1001	<?xml version="1.0" encoding="UTF-8" ?><cust...	[NULL]
3	1002	<?xml version="1.0" encoding="UTF-8" ?><cust...	[NULL]
4	1003	<?xml version="1.0" encoding="UTF-8" ?><cust...	[NULL]
5	1004	<?xml version="1.0" encoding="UTF-8" ?><cust...	[NULL]
6	1005	<?xml version="1.0" encoding="UTF-8" ?><cust...	[NULL]

Fin Rows: 6, Cols: 3 0.109 sec 10:19:59

The following operations can be carried out in the Results tab, via the context menu that pops up when you right-click in the appropriate location in the Results tab:

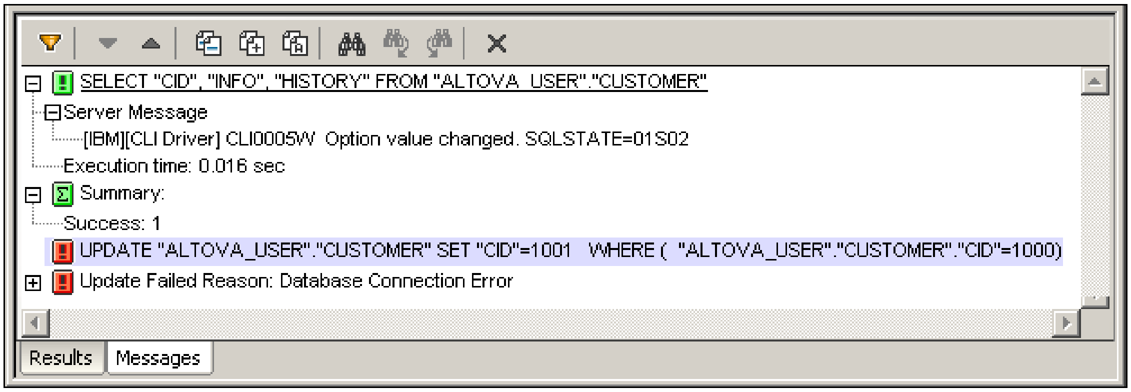
- *Sorting on a column:* Right-click anywhere in the column on which the records are to be sorted, then select **Sorting | Ascending/Descending/Restore Default**.
- *Copying to the clipboard:* This consists of two steps: (i) selecting the data range; and (ii) copying the selection. Data can be selected in several ways: (i) by clicking a column header or row number to select the column or row, respectively; (ii) selecting individual cells (use the **Shift** and/or **Ctrl** keys to select multiple cells); (iii) right-clicking a cell, and selecting **Selection | Row/Column/All**. After making the selection, right-click, and select **Copy Selected Cells**. This copies the selection to the clipboard, from where it can be pasted into another application. To copy the header together with the cells, use the command **Copy Selected Cells with Header**.

The Results tab has the following toolbar icons:

	Go to Statement	Highlights the statement in the SQL Editor that produced the current result.
	Find	Finds text in the Results pane. XML document content is also searched.

Messages tab

The Messages tab provides information on the previously executed SQL statement and reports errors or warning messages.



```
SELECT "CID", "INFO", "HISTORY" FROM "ALTOVA_USER"."CUSTOMER"
  Server Message
    [IBM][CLI Driver] CLI0005W  Option value changed. SQLSTATE=01S02
  Execution time: 0.016 sec
Summary:
  Success: 1
UPDATE "ALTOVA_USER"."CUSTOMER" SET "CID"=1001   WHERE (  "ALTOVA_USER"."CUSTOMER"."CID"=1000)
Update Failed Reason: Database Connection Error
```

The toolbar of the Messages tab contains icons that enable you to customize the view, navigate it, and copy messages to the clipboard. The **Filter** icon enables the display of particular types of messages to be toggled on or off. The **Next** and **Previous** icons lets you step through the list, downwards and upwards, respectively. Messages can also be copied with or without their child components to the clipboard, enabling them to be pasted in documents. The **Find** function enables you to specify a search term and then search up or down the listing for this term. Finally, the **Clear** icon clears the contents of the Report pane.

Note: These toolbar icon commands are also available as context menu commands.

8.7 FAQs about DBs

▼ ***My project uses MySQL over ODBC. If I deploy my solution to MobileTogether Server and run it from there, I get the following error: Database: [Microsoft][ODBC Driver Manager] Data source name not found and no default driver specified... Retrieval from database resulted in error.***

You have to consider that MobileTogether Server is installed as a Windows Service, and therefore by default doesn't run under your Windows account. Try these options to resolve your issue:

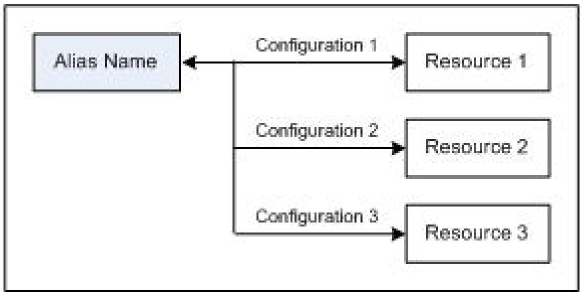
- Your Data Source Name (DSN) is probably defined as a User DSN. Use Windows Control Panel to move/recreate it as a System DSN. Then, if necessary, redo the connection in MobileTogether Designer, and redeploy.
- Assign the MobileTogether Server service to your Windows user account. After installation of MobileTogether Server, use Windows Services app to assign your user account.
- Use MobileTogether's Global Resources mechanism to use a different connection, or even a different database, for MobileTogether Server and MobileTogether Designer.

Chapter 9

Altova Global Resources

9 Altova Global Resources

Altova Global Resources is a collection of aliases for file, folder, and database resources. Each alias can have multiple configurations, and each configuration maps to a single resource (*see screenshot below*). 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 that let you select the active configuration. For example, you can specify a global resource as the default file of a page data source and switch resources by switching the active configuration in the GUI.



Using Altova Global Resources involves two processes:

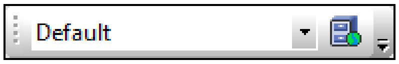
- Defining Global Resources: Resources are defined and the definitions are stored in an XML file. These resources can be shared across multiple Altova applications.
- Using Global Resources: Within MobileTogether Designer, files can be located via a global resource instead of via a file path. The advantage is that the resource can be switched by changing the active configuration in MobileTogether Designer.

Global resources in other Altova products
Currently, global resources can be defined and used in the following individual Altova products: XMLSpy, StyleVision, MapForce, Authentic Desktop, MobileTogether Designer, and DatabaseSpy.

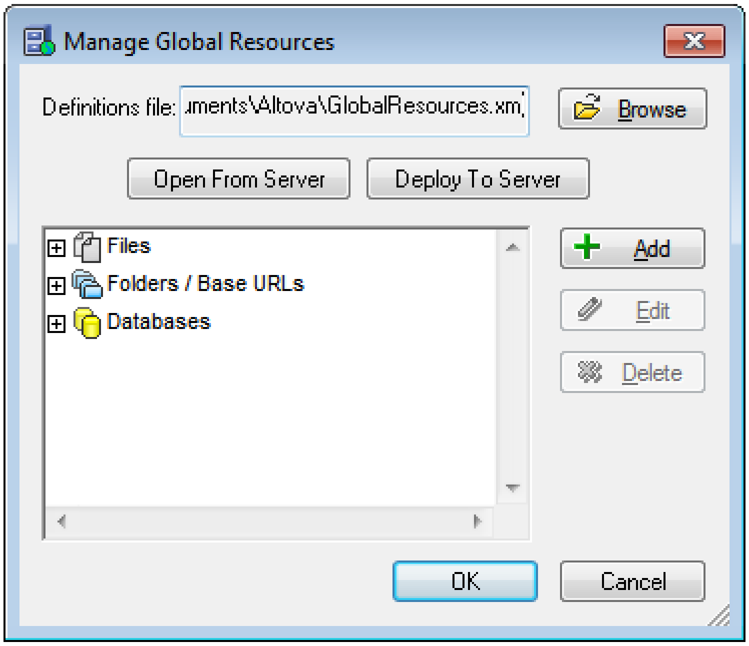
9.1 Defining Global Resources

Altova Global Resources are defined in the Manage Global Resources dialog, which can be accessed in two ways:

- Click the menu command **Tools | Global Resources**.
- Click the **Manage Global Resources** icon in the Global Resources toolbar (*screenshot below*).



The Global Resources Definitions file
Information about global resources is stored in an XML file called the Global Resources Definitions file. This file is created when the first global resource is defined in the Manage Global Resources dialog (*screenshot below*) and saved.



Definitions file: ıments\Altova\GlobalResources.xm

When you open the Manage Global Resources dialog for the first time, the default location and name of the Global Resources Definitions file is specified in the *Definitions File* text box (*see screenshot above*):

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

This file is set as the default Global Resources Definitions file for all Altova applications. So a global resource can be saved from any Altova application to this file and will be immediately available to all other Altova applications as a global resource. To define and save a global resource to the Global Resources Definitions file, add the global resource in the Manage Global Resources dialog and click **OK** to save.

To select an already existing Global Resources Definitions file to be the active definitions file of a particular Altova application, browse for it via the **Browse** button of the *Definitions File* text box (*see screenshot above*).

Note: You can name the Global Resources Definitions file anything you like and save it to any location accessible to your Altova applications. All you need to do in each application, is specify this file as the Global Resources Definitions file for that application (in the *Definitions File* text box). The resources become global across Altova products when you use a single definitions file across all Altova products.

Note: You can also create multiple Global Resources Definitions files. However, only one of these can be active at any time in a given Altova application, and only the definitions contained in this file will be available to the application. The availability of resources can therefore be restricted or made to overlap across products as required.

Managing global resources: adding, editing, deleting, saving
In the Manage Global Resources dialog (*screenshot above*), you can add a global resource to the selected Global Resources Definitions file, or edit or delete a selected global resource. The Global Resources Definitions file organizes the global resources you add into groups: of files, folders, and databases (*see screenshot above*).

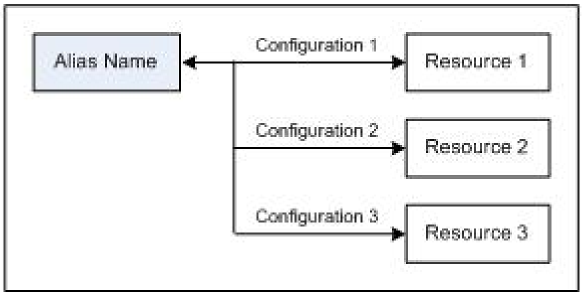
To ***add a global resource***, click the **Add** button and define the global resource in the appropriate **Global Resource** dialog that pops up (*see the descriptions of files, folders, and databases in the sub-sections of this section*). After you define a global resource and save it (by clicking **OK** in the Manage Global Resources dialog), the global resource is added to the library of global definitions in the selected Global Resources Definitions file. The global resource will be identified by an alias.

To ***edit a global resource***, select it and click **Edit**. This pops up the relevant **Global Resource** dialog, in which you can make the necessary changes (*see the descriptions of files, folders, and databases 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 Definitions file.

Relating global resources to alias names via configurations
Defining a global resource involves mapping an alias name to a resource (file, folder, or database). A single alias name can be mapped to multiple resources. Each mapping is called a configuration. A single alias name can therefore be associated with several resources via different configurations (*screenshot below*).



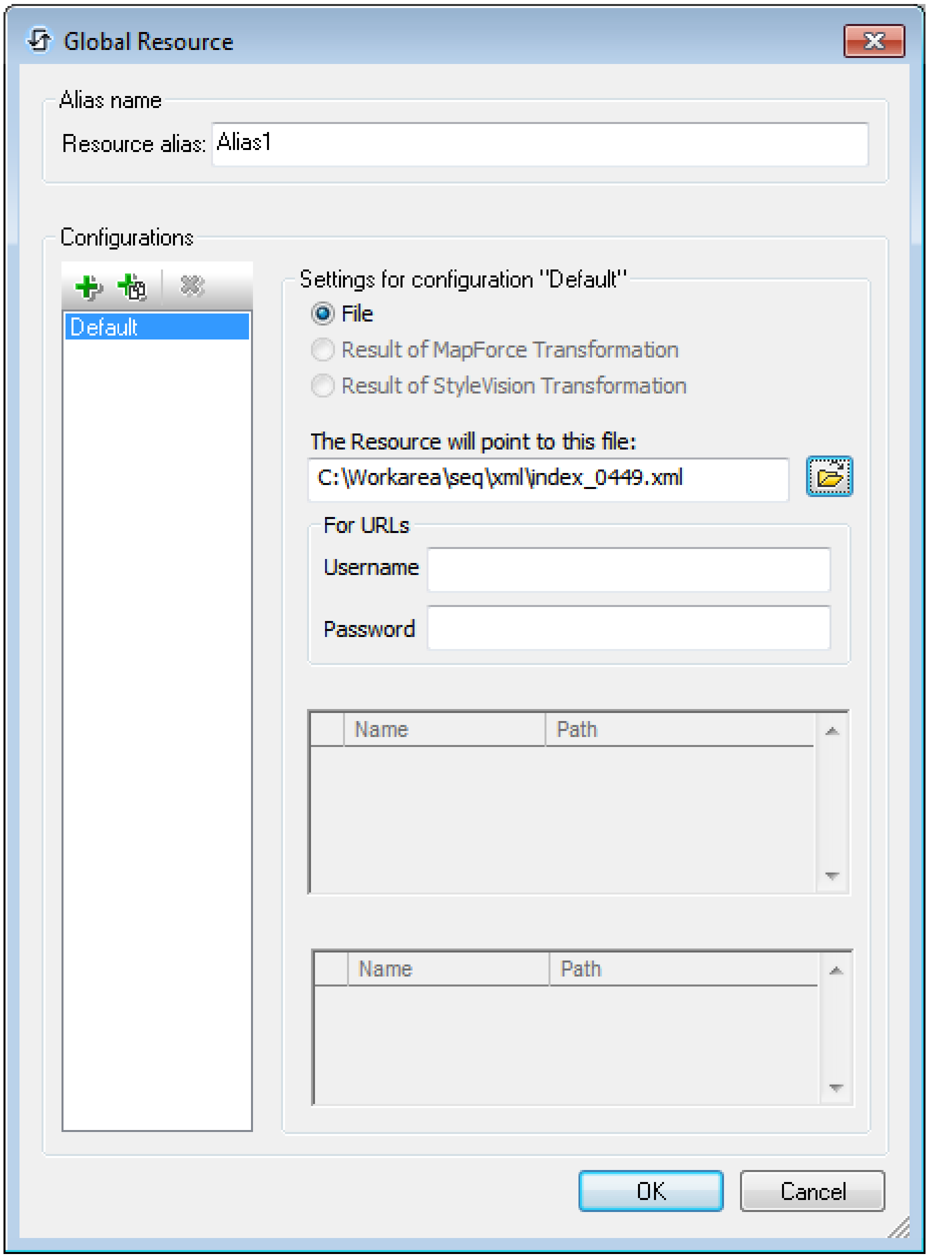
In an Altova application, you can then assign aliases instead of files. For each alias you can switch between the resources mapped to that alias simply by changing the application's active Global Resource configuration (active configuration). For example, in Altova's XMLSpy application, if you wish to run an XSLT transformation on the XML document `MyXML.xml`, you can assign the alias `MyXSLT` to it as the global resource to be used for XSLT transformations. In XMLSpy you can then change the active configuration to use different XSLT files. If `Configuration-1` maps `First.xslt` to `MyXSLT` and `Configuration-1` is selected as the active configuration, then `First.xslt` will be used for the transformation. In this way multiple configurations can be used to access multiple resources via a single alias. This mechanism can be useful when testing and comparing resources. Furthermore, since global resources can be used across Altova products, resources can be tested and compared across multiple Altova products as well.

Files

The Global Resource dialog for Files (*screenshot below*) is accessed via the **Add | Files** command in the Manage Global Resources dialog. In this dialog, you can define configurations of the alias that is named in the *Resource Alias* text box. After specifying the properties of the configurations as explained below, save the alias definition by clicking **OK**.

After saving an alias definition, you can add another alias by repeating the steps given above (starting with the **Add | Files** command in the Manage Global Resources dialog).

Global Resource dialog
An alias is defined in the Global Resource dialog (*screenshot below*).



Alias name

Resource alias: Alias1

Configurations

Default

Settings for configuration "Default"

● File

○ Result of MapForce Transformation

○ Result of StyleVision Transformation

The Resource will point to this file: C:\Workarea\seq\xml\index_0449.xml

For URLs

Username

Password

	Name	Path

	Name	Path

Global Resource dialog icons

Add Configuration: Pops up the Add Configuration dialog in which you enter the name of the configuration to be added.

Add Configuration as Copy: Pops up the Add Configuration dialog in which you can enter the name of the configuration to be created as a copy of the selected configuration.

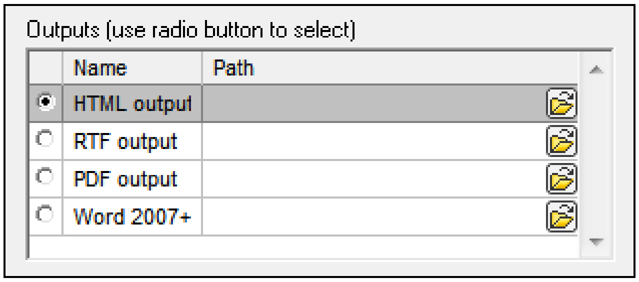
Delete: Deletes the selected configuration.

Open: Browse for the file to be created as the global resource.

Defining the alias

Define the alias (its name and configurations) as follows:

1. *Give the alias a name:* Enter the alias name in the *Resource Alias* text box.
2. *Add configurations:* The Configurations pane will have, by default, a configuration named `Default` (*see screenshot above*), which cannot be deleted or renamed. You can add as many additional configurations as you like by: (i) clicking the **Add Configuration** or **Add Configuration as Copy** icons, and (ii) giving the configuration a name in the dialog that pops up. Each added configuration will be shown in the Configurations list. In the screenshot above, two additional configurations, named `Long` and `Short`, have been added to the Configurations list. The Add Configuration as Copy command enables you to copy the selected configuration and then modify it.
3. *Select a resource type for each configuration:* Select a configuration from the Configurations list, and, in the *Settings for Configuration* pane, specify a resource for the configuration: (i) File, (ii) Output of an Altova MapForce transformation, or (iii) Output of an Altova StyleVision transformation. Select the appropriate radio button. If a MapForce or StyleVision transformation option is selected, then a transformation is carried out by MapForce or StyleVision using, respectively, the `.mfd` or `.sps` file and the respective input file. The result of the transformation will be the resource.
4. *Select a file for the resource type:* If the resource is a directly selected file, browse for the file in the *Resource File Selection* text box. If the resource is the result of a transformation, in the *File Selection* text box, browse for the `.mfd` file (for MapForce transformations) or the `.sps` file (for StyleVision transformations). Where multiple inputs or outputs for the transformation are possible, a selection of the options will be presented. For example, the output options of a StyleVision transformation are displayed according to what edition of StyleVision is installed (*the screenshot below shows the outputs for Enterprise Edition*).

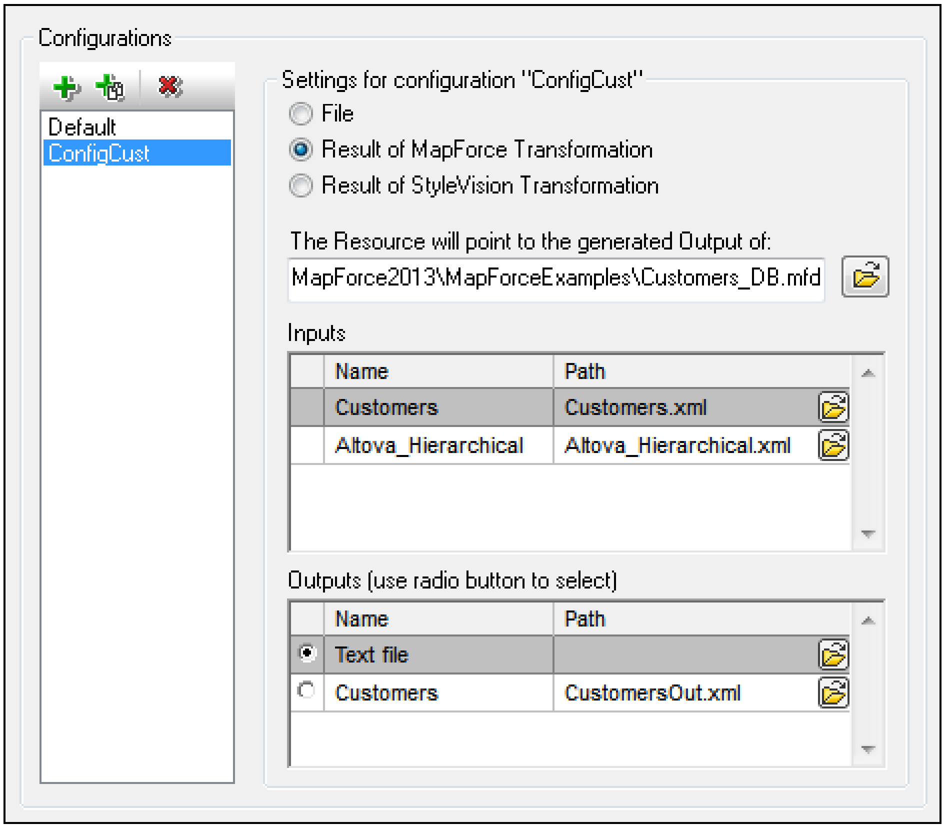
   Outputs (use radio button to select)

	Name	Path
●	HTML output	📂
○	RTF output	📂
○	PDF output	📂
○	Word 2007+	📂

   Select the radio button of the desired option (in the screenshot above, 'HTML output' is selected). If the resource is the result of a transformation, then the output can be saved as a file or itself as a global resource. Click the 📂 icon and select, respectively, Global Resource (for saving the output as a global resource) or Browse (for saving the output as a file). 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.
5. *Define multiple configurations if required:* You can add more configurations and specify a resource for each. Do this by repeating Steps 3 and 4 above for each configuration. You can add a new configuration to the alias definition at any time.
6. *Save the alias definition:* Click **OK** 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.

Result of MapForce transformation
Altova MapForce maps one or more (existing) input document schemas to one or more (new) output document schemas. This mapping, which is created by a MapForce user, is known as a MapForce Design (MFD). XML files, text files, databases, etc, that correspond to the input schema/s can be used as data sources. MapForce generates output data files that correspond to the output document schema. This output document is the *Result of MapForce Transformation* file that will become a global resource.

If you wish to set a MapForce-generated data file as a global resource, the following must be specified in the Global Resource dialog (*see screenshot below*):



Configurations

- Default
- ConfigCust

Settings for configuration "ConfigCust"

○ File
● Result of MapForce Transformation
○ Result of StyleVision Transformation

The Resource will point to the generated Output of:
MapForce2013\MapForceExamples\Customers_DB.mfd

Inputs

Name	Path
Customers	Customers.xml
Altova_Hierarchical	Altova_Hierarchical.xml

Outputs (use radio button to select)

	Name	Path
●	Text file	
○	Customers	CustomersOut.xml

- ***A*** **`.mfd`** ***(MapForce Design) file.*** You must specify this file in the *Resource will point to generated output of* text box (*see screenshot above*).
- ***One or more input data files.*** After the MFD file has been specified, it is analysed and, based on the input schema information in it, default data file/s are entered in the *Inputs* pane (*see screenshot above*). You can modify the default file selection for each input schema by specifying another file.
- ***An output file.*** If the MFD document has multiple output schemas, all these are listed in the *Outputs* pane (*see screenshot above*) and you must select one of them. If the output file location of an individual output schema is specified in the MFD document, then this file location is entered for that output schema in the *Outputs* pane. From the screenshot above we can see that the MFD document specfies that the `Customers` output schema has a default XML data file (`CustomersOut.xml`), while the `Text file` output schema does not have a file association in the MFD file. You can use the default file location in

the Outputs pane or specify one yourself. The result of the MapForce transformation will be saved to the file location of the selected output schema. This is the file that will be used as the global resource

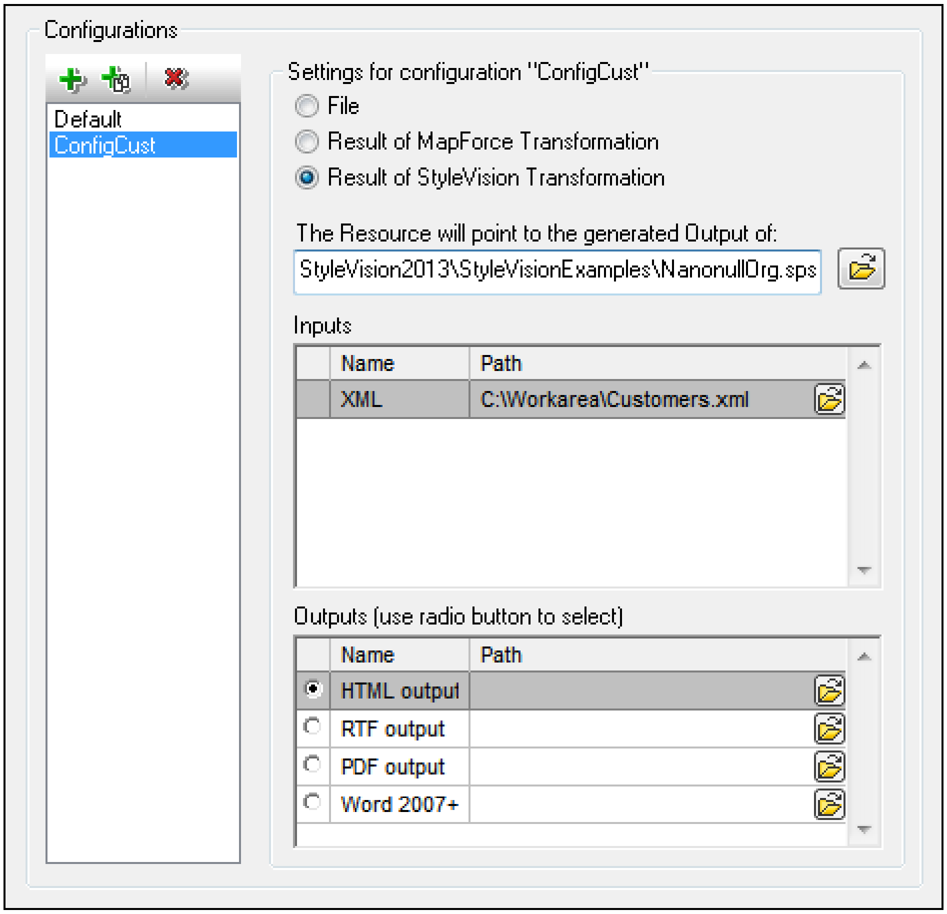
Note: The advantage of this option (Result of MapForce transformation) is that the transformation is carried out at the time the global resource is invoked. This means that the global resource will contain the most up-to-date data (from the input file/s).

Note: Since MapForce is used to run the transformation, you must have Altova MapForce installed for this functionality to work.

Result of StyleVision transformation

Altova StyleVision is used to create StyleVision Power Stylesheet (SPS) files. These SPS files generate XSLT stylesheets that are used to transform XML documents into output documents in various formats (HTML, PDF, RTF, Word 2007+, etc). If you select the option *Result of StyleVision Transformation*, the output document created by StyleVision will be the global resource associated with the selected configuration.

For the *StyleVision Transformation* option in the Global Resource dialog (*see screenshot below*), the following files must be specified.



Configurations

Default
ConfigCust

Settings for configuration "ConfigCust"

○ File
○ Result of MapForce Transformation
● Result of StyleVision Transformation

The Resource will point to the generated Output of:
StyleVision2013\StyleVisionExamples\NanonullOrg.sps

Inputs

Name	Path
XML	C:\Workarea\Customers.xml

Outputs (use radio button to select)

	Name	Path
●	HTML output	
○	RTF output	
○	PDF output	
○	Word 2007+	

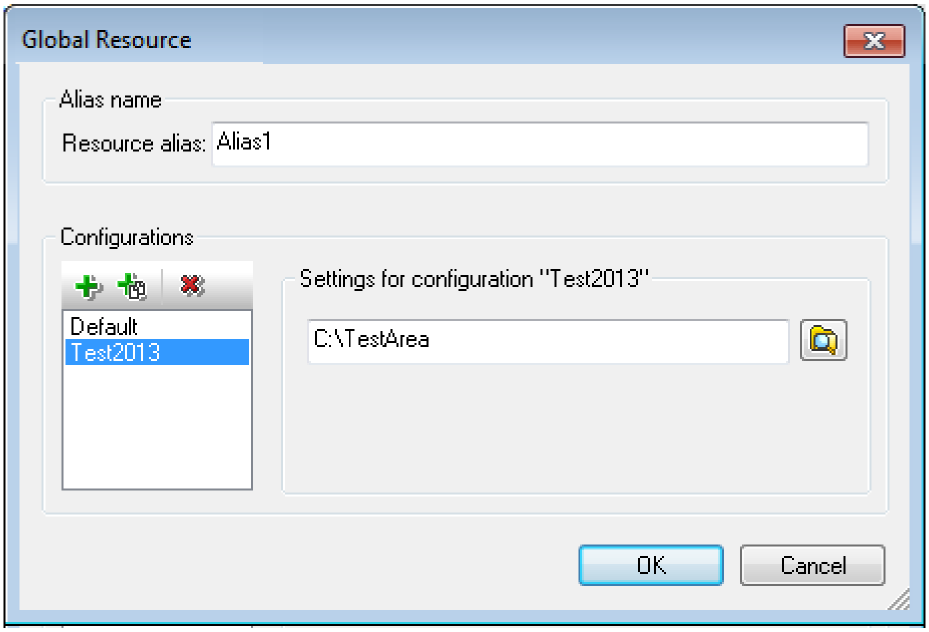
- ***A .sps (SPS) file.*** You must specify this file in the *Resource will point to generated output of* text box (*see screenshot above*).
- ***Input file/s.*** The input file might already be specified in the SPS file. If it is, it will appear automatically in the *Inputs* pane once the SPS file is selected. You can change this entry. If there is no entry, you must add one.
- ***Output file/s.*** Select the output format in the *Outputs* pane, and specify an output file location for that format.

Note: The advantage of this option (Result of StyleVision transformation) is that the transformation is carried out when the global resource is invoked. This means that the global resource will contain the most up-to-date data (from the input file/s).

Note: Since StyleVision is used to run the transformation, you must have Altova StyleVision installed for this functionality to work.

Folders

In the Global Resource dialog for Folders (*screenshot below*), add a folder resource as described below.



Resource alias: Alias1

Configurations: Default, Test2013

C:\TestArea

Global Resource dialog icons

✚ *Add Configuration:* Pops up the Add Configuration dialog in which you enter the name of the configuration to be added.

✚ *Add Configuration as Copy:* Pops up the Add Configuration dialog in which you can enter the name of the configuration to be created as a copy of the selected configuration.

✖ *Delete:* Deletes the selected configuration.

📂 *Open:* Browse for the folder to be created as the global resource.

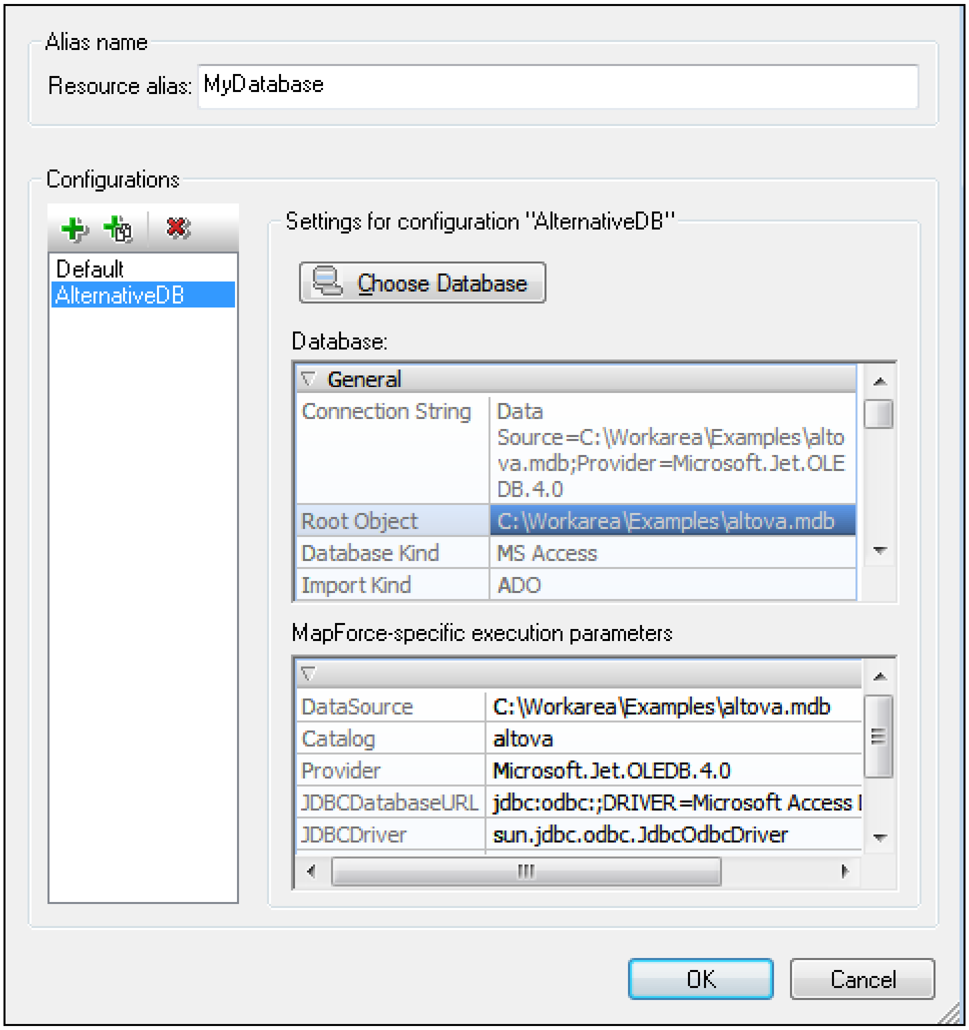
Defining the alias
Define the alias (its name and configurations) as follows:

1. *Give the alias a name:* Enter the alias name in the *Resource Alias* text box.
2. *Add configurations:* The Configurations pane will have a configuration named Default (*see 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** or **Add Configuration as Copy** icons. In the 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 you want.
3. *Select a folder as the resource of a configuration:* Select one of the configurations in the Configurations pane and browse for the folder you wish to create as a global resource.

4. *Define multiple configurations if required:* Specify a folder resource for each configuration you have created (that is, repeat Step 3 above for the various configurations you have created). You can add a new configuration to the alias definition at any time.
5. *Save the alias definition:* 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.

Databases

In the Global Resource dialog for Databases (*screenshot below*), you can add a database resource as follows:



Alias name
Configurations
Settings for configuration "AlternativeDB"
Choose Database
Database:
MapForce-specific execution parameters
OK
Cancel

Resource alias: MyDatabase

Default
AlternativeDB

General	
Connection String	Data Source=C:\Workarea\Examples\altova.mdb;Provider=Microsoft.Jet.OLEDB.4.0
Root Object	C:\Workarea\Examples\altova.mdb
Database Kind	MS Access
Import Kind	ADO

DataSource	C:\Workarea\Examples\altova.mdb
Catalog	altova
Provider	Microsoft.Jet.OLEDB.4.0
JDBCDatabaseURL	jdbc:odbc:;DRIVER=Microsoft Access I
JDBCDriver	sun.jdbc.odbc.JdbcOdbcDriver

Global Resource dialog icons

Add Configuration: Pops up the Add Configuration dialog in which you enter the name of the configuration to be added.

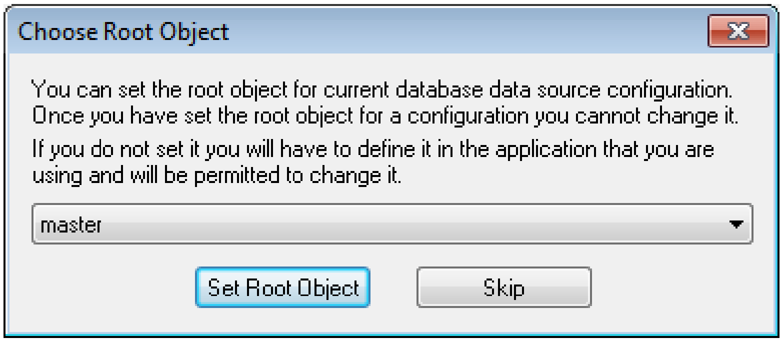
Add Configuration as Copy: Pops up the Add Configuration dialog in which you can enter the name of the configuration to be created as a copy of the selected configuration.

Delete: Deletes the selected configuration.

Defining the alias

Define the alias (its name and configurations) as follows:

1. *Give the alias a name:* Enter the alias name in the *Resource Alias* text box.
2. *Add configurations:* The Configurations pane will have a configuration named Default (*see 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** or **Add Configuration as Copy** icons. In the 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 you want.
3. *Start selection of a database as the resource of a configuration:* Select one of the configurations in the Configurations pane and click the **Choose Database** icon. This pops up the Create Global Resources Connection dialog.
4. *Connect to the database:* Select whether you wish to create a connection to the database using the Connection Wizard, an existing connection, an ADO Connection, an ODBC Connection, or JDBC Connection.
5. *Select the root object:* If you connect to a database server where a root object can be selected, you will be prompted, in the Choose Root Object dialog (*screenshot below*), to select a root object on the server. Select the root object and click **Set Root Object**. The root object you select will be the root object that is loaded when this configuration is used.


Choose Root Object

You can set the root object for current database data source configuration. Once you have set the root object for a configuration you cannot change it.

If you do not set it you will have to define it in the application that you are using and will be permitted to change it.

master

   If you choose not to select a root object (by clicking the **Skip** button), then you can select the root object at the time the global resource is loaded.
6. *Define multiple configurations if required:* Specify a database resource for any other configuration you have created (that is, repeat Steps 3 to 5 above for the various configurations you have created). You can add a new configuration to the alias definition at any time.
7. *Save the alias definition:* 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 databases in the Manage Global Resources dialog.

9.2 Using Global Resources

There are several types of global resources (file-type, folder-type , and database-type). Some scenarios in which you can use global resources in MobileTogether Designer are listed here: Files and Folders and Databases.

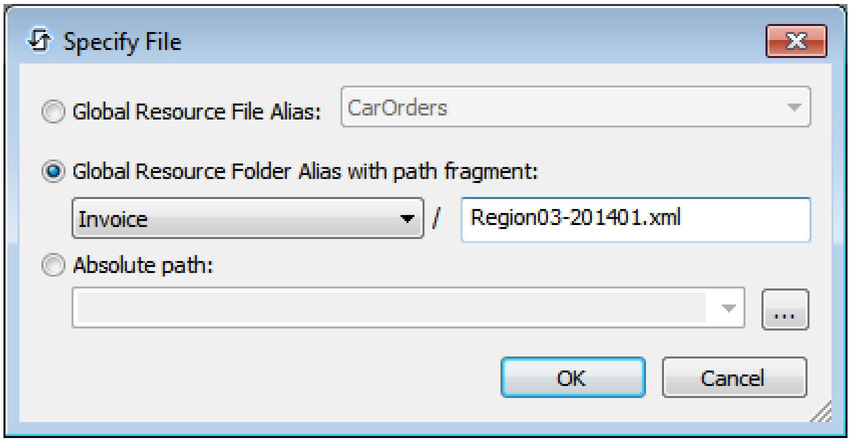
Selections that determine which resource is used

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.

Assigning Files and Folders

In certain usage scenarios, file-type and folder-type global resources can be used to specify the file or folder to use. For example, in the Page Sources Pane, the default file of a data source can be assigned via a global resource. In such scenarios, the Specify File (*screenshot below*) appears.



Specify File

○ Global Resource File Alias: CarOrders

● Global Resource Folder Alias with path fragment:

Invoice / Region03-201401.xml

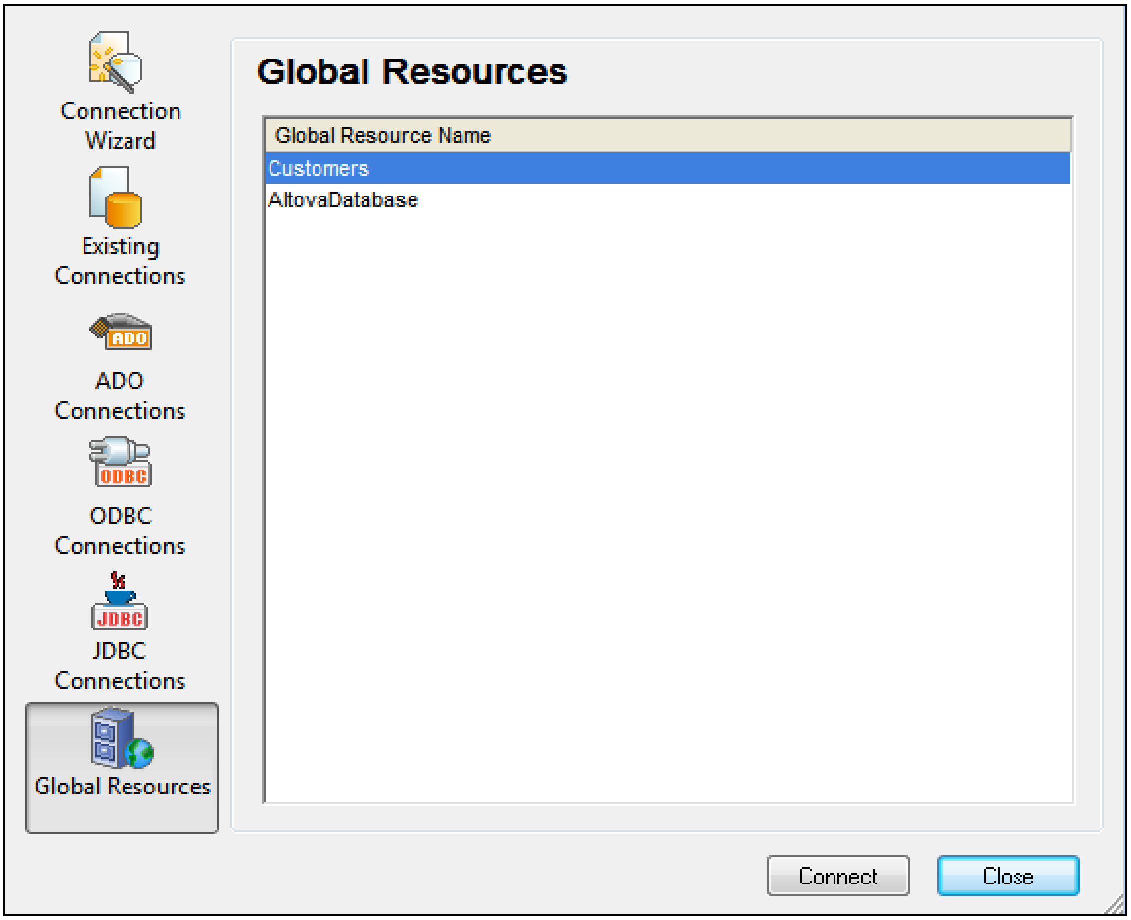
○ Absolute path:

Select whether you wish to use a file-type or folder-type global resource. The combo boxes display, respectively, all the file-type global resources and all the folder-type global resources that have been defined in the currently active Global Resources XML File are. Select the required global resource. In the case of file-type global resources, the selected alias maps to a file. In the case of folder-type global resources, the selected alias maps to a folder, so you will have to enter the rest of the path to locate the resource (*see screenshot above*). Click **OK**.

If the selected global resource has more than one configuration, then the database resource for the currently active configuration is used (check **Tools | Active Configuration** or the Global Resources toolbar).

Assigning Databases

When a command is executed that imports data or a data structure from a DB, you can select the option to use a global resource (*screenshot below*).



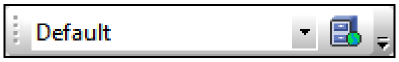
In the Connection dialog (*screenshot above*), all the database-type global resources that have been defined in the currently active Global Resources XML File are displayed. Select the required global resource and click **Connect**. If the selected global resource has more than one configuration, then the database resource for the currently active configuration is used (check **Tools | Active Configuration** or the Global Resources toolbar), and the connection is made.

Changing the Active Configuration

One configuration of a global resource can be active at any time. This configuration is called the active configuration, and it is active application-wide. This means that the active configuration is active for all global resources aliases in all currently open files and data source connections. 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 a file has been assigned via a global resource with multiple configurations. Each configuration maps to a different file. So, which file is selected depends on which configuration is selected as the application's active configuration.

Switching the active configuration can be done in the following ways:

- Via the menu command **Tools | Active Configuration**. Select the configuration from the command's submenu.
- In the combo box of the Global Resources toolbar (*screenshot below*), select the required configuration.



In this way, by changing the active configuration, you can change source files that are assigned via a global resource.

Chapter 10

Simulation

10 Simulation

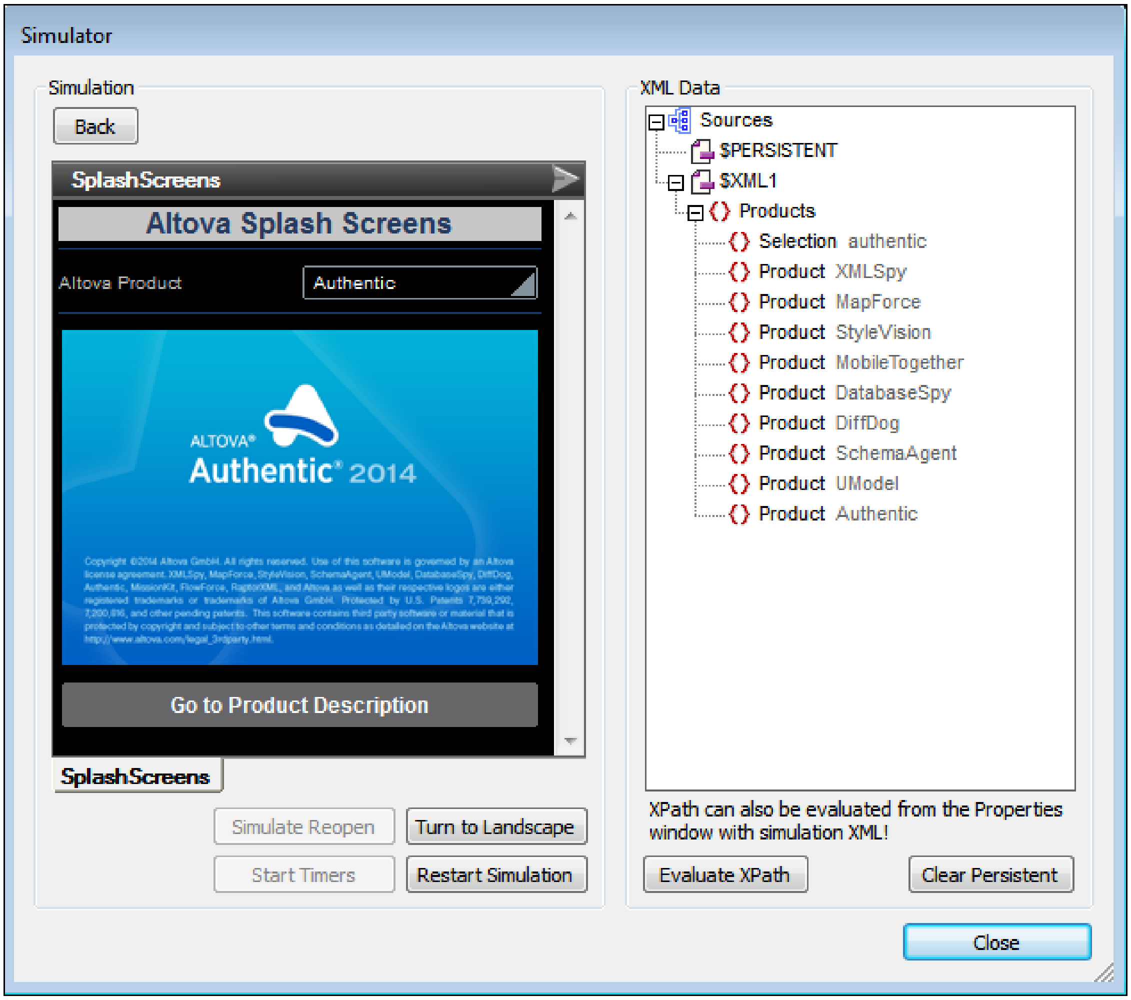
The following simulation methods are available for running and testing a solution:

- Simulation in MobileTogether Designer
- Simulation on Server
- Simulation on Client

Note: You can also deploy a solution to the server, and, in the *Workflows* tab of the Web UI of MobileTogether Server, you can click a solution to run it in the web browser.

10.1 Simulation in MobileTogether Designer

You can run a simulation of the project workflow directly within MobileTogether Designer. The simulation device will be the device currently selected in the Preview Device combo box of the Main toolbar. You can change the preview device to see the simulation on different devices. To run the simulation, select **Project | Simulate Workflow** or **F5**. This opens the simulator and starts the simulation.



Copyright ©2014 Altova GmbH. All rights reserved. Use of this software is governed by an Altova license agreement. XMLSpy, MapForce, StyleVision, SchemaAgent, UModel, DatabaseSpy, DiffDog, Authentic, MissionKit, FlowForce, RaptorXML, and Altova as well as their respective logos are either registered trademarks or trademarks of Altova GmbH. Protected by U.S. Patents 7,739,292, 7,200,816, and other pending patents. This software contains third 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.

XPath can also be evaluated from the Properties window with simulation XML!

The simulator window provides the following features

- The left-hand pane displays the simulation.
- The right-hand pane shows the changing of XML data as the simulation progresses.
- Controls that require user interaction are enabled. In the example above, for example, the combo box is enabled.
- You can switch the view to landscape and you can restart the simulation at any time by using the respective buttons.
- Clicking **Evaluate XPath** opens the Edit XPath/XQuery Expression dialog, in which you can evaluate XPath expressions. XPath expressions can also be evaluated from the Styles & Properties Pane while the simulation is running.
- Clicking **Clear Persistent** clears persistent data and restarts the simulation
- Clicking **Back** close the simulator.

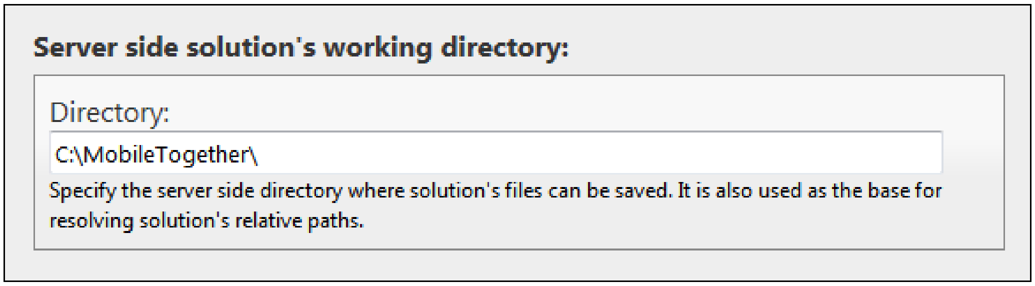
10.2 Simulation on Server

A server simulation uses MobileTogether Server to run the simulation (**Project | Use Server for Workflow Simulation**). Simulating the workflow on the server works as follows:

1. The workflow of the active design file in MobileTogether Designer is temporarily passed to MobileTogether Server. So the design file does not need to be deployed to the server.
2. The server serves the workflow to the simulator of MobileTogether Designer. In this way the simulator plays the role of a client.

Running the simulation

1. Start MobileTogether Server. See the MobileTogether Server user manual for information about how to do this.
2. In the Web UI of MobileTogether Server, you must set the solution's working directory (**Settings | Server Side Solution's Working Directory**, *see screenshot below*). All relative paths in the design will be resolved relative to the directory specified in this setting. In order for server simulation to work correctly, enter the path of the directory where your referenced files are saved.



Server side solution's working directory:

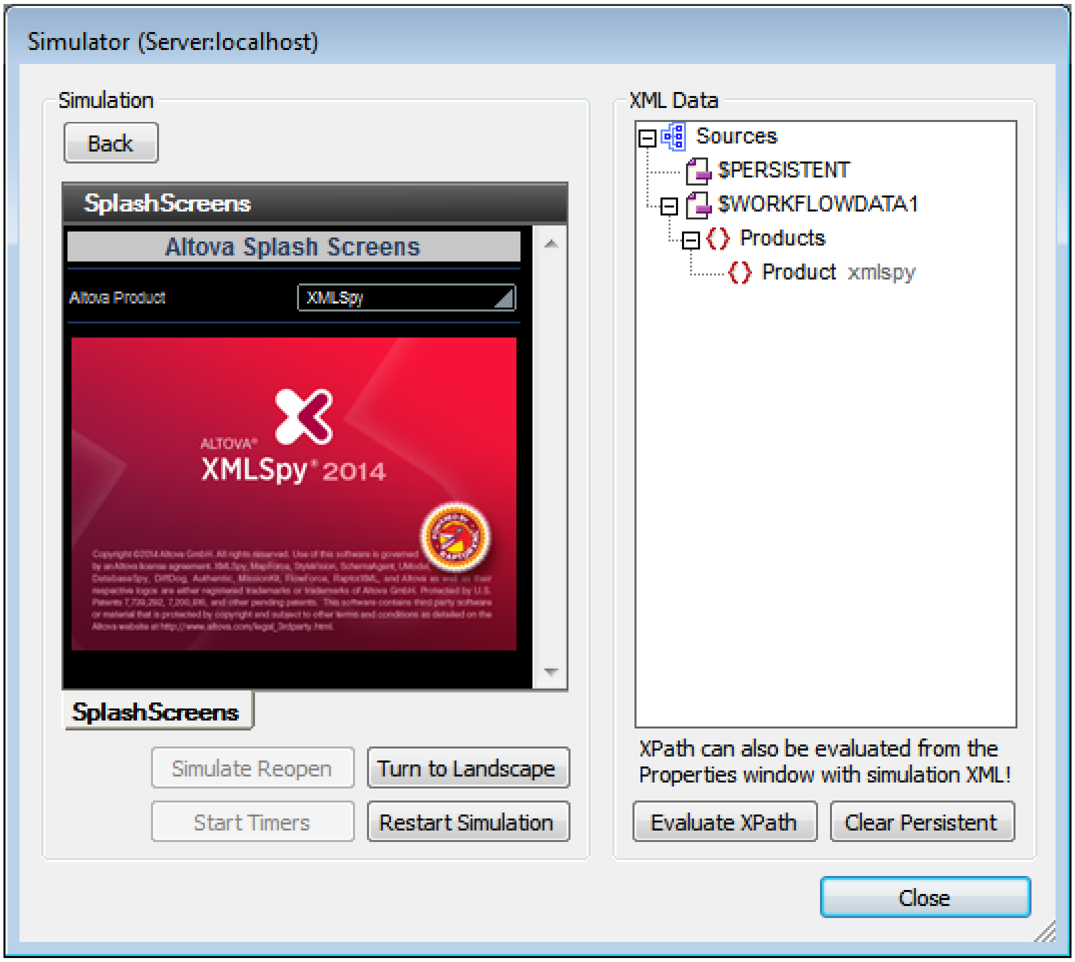
Directory:

C:\MobileTogether\

Specify the server side directory where solution's files can be saved. It is also used as the base for resolving solution's relative paths.

3. In MobileTogether Designer, make sure that the server settings are correctly defined.
4. In MobileTogether Designer, select **Project | Use Server for Workflow Simulation**.

The simulator window is opened and runs the workflow.



XPath can also be evaluated from the Properties window with simulation XML!

The simulator window provides the following features

- The left-hand pane displays the simulation.
- The right-hand pane shows the changing of XML data as the simulation progresses.
- Controls that require user interaction are enabled. In the example above, for example, the combo box is enabled.
- You can switch the view to landscape and you can restart the simulation at any time by using the respective buttons.
- Clicking **Evaluate XPath** opens the Edit XPath/XQuery Expression dialog, in which you can evaluate XPath expressions. XPath expressions can also be evaluated from the Styles & Properties Pane while the simulation is running.
- Clicking **Clear Persistent** clears persistent data and restarts the simulation
- Clicking **Back** close the simulator.

Note: If you have problems connecting to the server, try checking the settings on the server. See the MobileTogether Server user manual for information about how to do this.

10.3 Simulation on Client

This simulation method runs the workflow on your mobile device by using MobileTogether Designer as a server. It assumes that your mobile device can connect to your PC over wireless LAN. Do a trial run on the client (simulation on the client) as follows:

1. Client: Set up a new server
   1. Start MobileTogether on the client and click the **Settings | Server** icon.
   2. Add a new server and enter all the relevant data (server address, port number (`8083` is the default), username, and password).
   3. Save the server settings.

2. PC: Start the simulation
   1. If MobileTogether Server is running on your PC, stop it running as a service.
   2. In MobileTogether Designer, select **Tools | Options**, click the *Trial Run on Client* tab, and enter the PC port number that the client is going to use (`8083` is the default).
   3. Open the project (`.mtd`) file that you want to test on the client.
   4. Select **Project | Trial Run on Client**. This opens a dialog box that will show the page sources and data structure.

3. Client: Run the simulation
   1. Start the MobileTogether Client app on your mobile device if not already started. Only the design files that are currently open on the PC are visible on the mobile client (as solutions) using this type of simulation.
   2. Select the solution that you want to test.
   3. This opens a prompt on the PC asking if you want to start the solution. Click **Yes**. The tree structure now appears in the dialog box of the PC. After a short time, the solution runs on the client.
   4. Click **Back/Return** to stop the current solution. A prompt appears asking if you want to stop. Click **Yes**.

Chapter 11

Menu Commands

11 Menu Commands

MobileTogether Designer menu commands are organized into the following menus:

- File
- Edit
- Project
- Page
- View
- Tools
- Window
- Help

11.1 File

The **File** menu contains the following commands:

- New
- Open
- Reload
- Close
- Close All
- Close All But Active
- Save
- Save As
- Save Copy As
- Save All
- Deploy to MobileTogether Server
- Open from MobileTogether Server
- Delete from MobileTogether Server
- Send by Mail
- Print
- Print Preview
- Print Setup
- Recent Files
- Exit

New

▾ New

⊟ *Icon*

🗋

⊟ *Shortcut*

Ctrl+N

⊟ *Description*

Opens a new document tab in the main window and loads an empty MobileTogether Design file into this tab. The document is stored temporarily in memory. It must be saved to disk with the `.mtd` extension if you want to keep it.

Open

Open

⊟ *Icon*

⊟ *Shortcut*

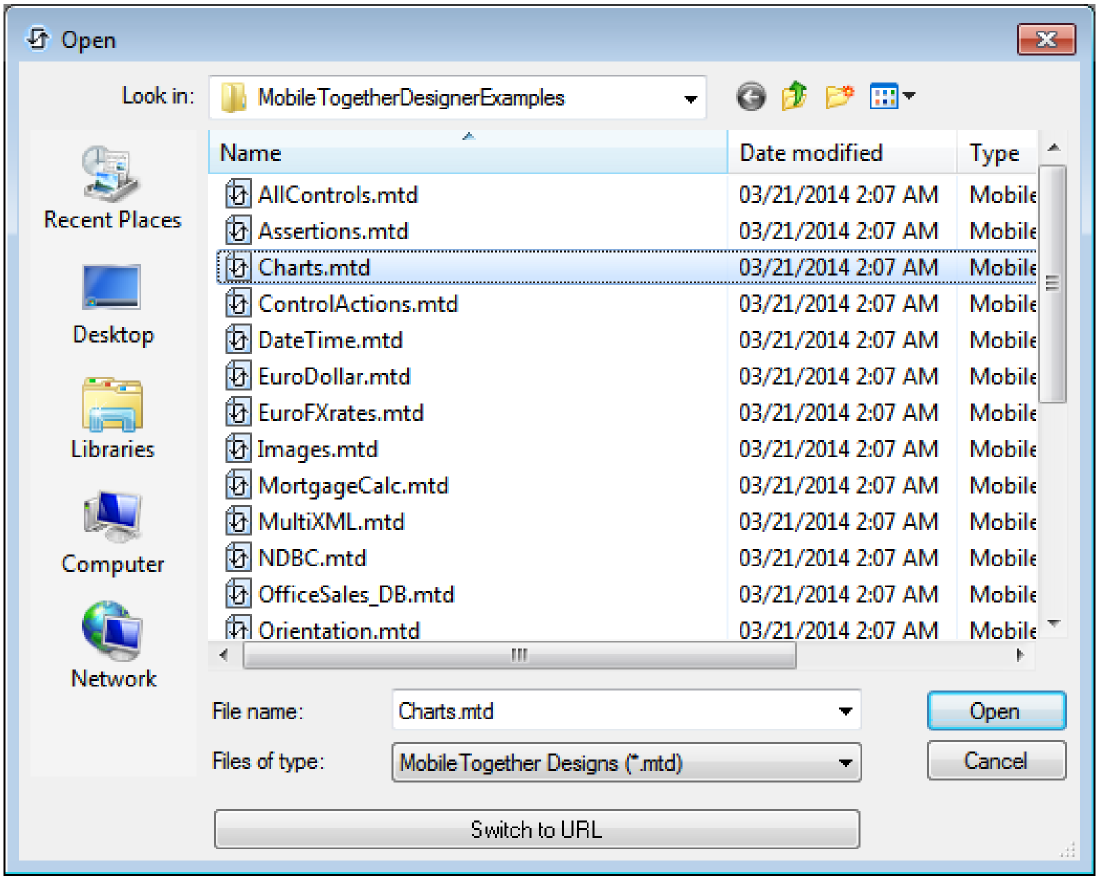
Ctrl+O

⊟ *Description*

Pops up the Open dialog, in which you can select the MobileTogether Design file (`.mtd` file) to open. The MTD file is opened in a new tab of the main window.

Selecting and saving files via URLs

In several File Open and File Save dialogs, you can choose to select the required file or save a file via a URL (*see screenshot below*). Click **Switch to URL** to go to the selection process.



Look in: MobileTogetherDesignerExamples

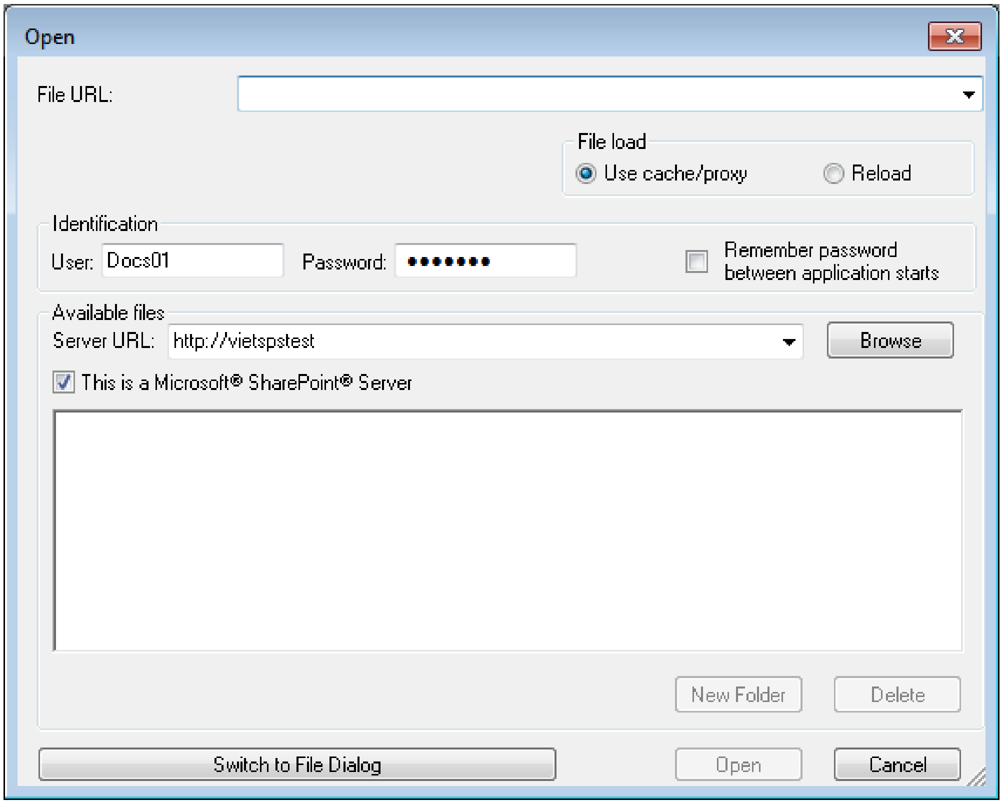
Name	Date modified	Type
AllControls.mtd	03/21/2014 2:07 AM	Mobile
Assertions.mtd	03/21/2014 2:07 AM	Mobile
Charts.mtd	03/21/2014 2:07 AM	Mobile
ControlActions.mtd	03/21/2014 2:07 AM	Mobile
DateTime.mtd	03/21/2014 2:07 AM	Mobile
EuroDollar.mtd	03/21/2014 2:07 AM	Mobile
EuroFXrates.mtd	03/21/2014 2:07 AM	Mobile
Images.mtd	03/21/2014 2:07 AM	Mobile
MortgageCalc.mtd	03/21/2014 2:07 AM	Mobile
MultiXML.mtd	03/21/2014 2:07 AM	Mobile
NDBC.mtd	03/21/2014 2:07 AM	Mobile
OfficeSales_DB.mtd	03/21/2014 2:07 AM	Mobile
Orientation.mtd	03/21/2014 2:07 AM	Mobile

File name: Charts.mtd

Files of type: MobileTogether Designs (*.mtd)

To select a file via a URL (either for opening or saving), do the following:

1. Click the **Switch to URL** command. This switches to the URL mode of the Open or Save dialog (*the screenshot below shows the Open dialog*).



File URL:

File load
● Use cache/proxy ○ Reload

Identification
User: Docs01 Password: •••••••
□ Remember password between application starts

Available files
Server URL: http://vietspstest
☑ This is a Microsoft® SharePoint® Server

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. The **Open** or **Save** button only becomes active at this point.
6. Click **Open** to load the file or **Save** to save it.

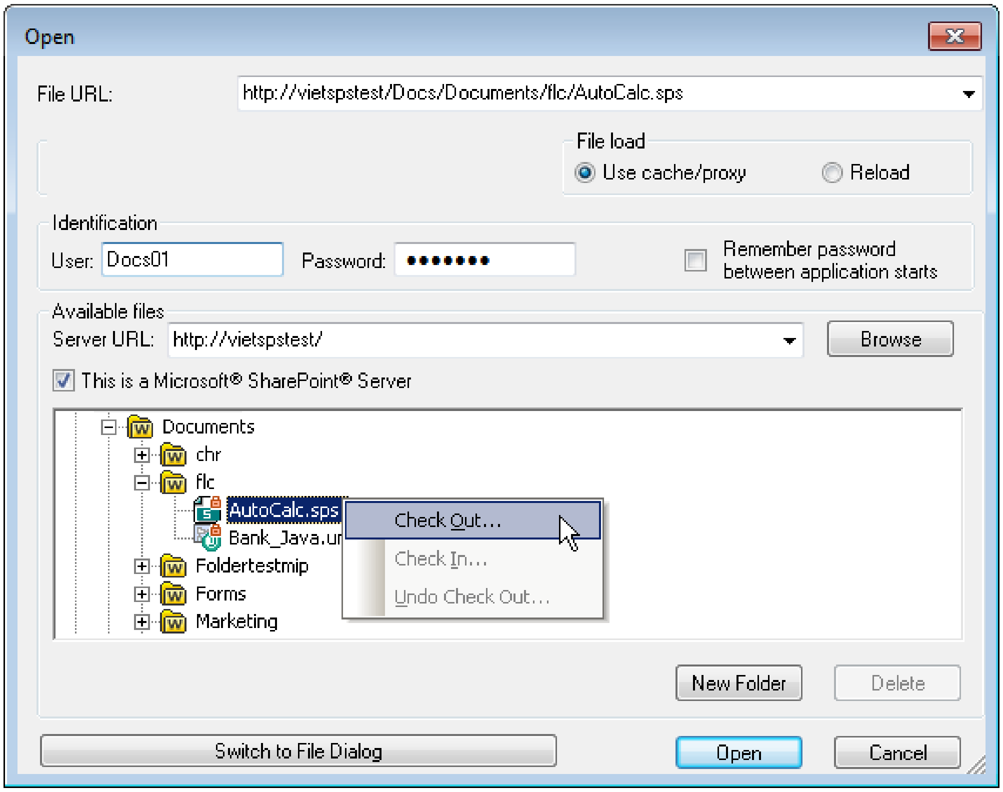
Note the following:

- The Browse function is only available on servers which support WebDAV and on Microsoft SharePoint Servers. The supported protocols are FTP, HTTP, and HTTPS.
- To give you more control over the loading process when opening a file, 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.



File URL: http://vietspstest/Docs/Documents/flc/AutoCalc.sps

● Use cache/proxy ○ Reload

User: Docs01 Password: •••••••

□ Remember password between application starts

Server URL: http://vietspstest/

☑ This is a Microsoft® SharePoint® Server

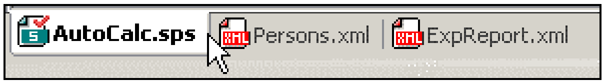
```
⊟ Documents
    ⊞ chr
    ⊟ flc
        AutoCalc.sps
        Bank_Java.ur
    ⊞ Foldertestmip
    ⊞ Forms
    ⊞ Marketing
```

Right-clicking a file pops up a context menu containing commands available for that file (*screenshot above*).

- The various file icons are shown below:

XSD	Checked in. Available for check-out.
XML	Checked out by another user. Not available for check-out.
S	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 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**.

Reload

Reload

Icon

Description

Reloads any open documents that have modified outside MobileTogether Designer. If one or more documents is modified outside MobileTogether Designer, a prompt appears asking whether you wish to reload the modified document/s. If you choose to reload, then any changes you may have made to the file since the last time it was saved will be lost.

Close, Close All, Close All But Active

▼ Close

⊟ *Description*
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.

▼ Close All

⊟ *Description*
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.

▼ Close All But Active

⊟ *Description*
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.

Save, Save As, Save Copy As, Save All

Save

Icon

Shortcut
Ctrl+S

Description
Saves the contents of the active document to the file from which it has been opened.

Save As

Description
Pops up the Save As dialog box, in which you enter the name and location of the file you wish to save the active document as. The newly saved document replaces the original document in the active tab of the main window.

Save Copy As

Icon

Description
Pops up the Save Copy As dialog box, in which you enter the name and location of the file you wish to save the active document as. The saved document will be a copy of the active document. The newly saved document will not be opened in MobileTogether Designer. The original document remains active in the main window.

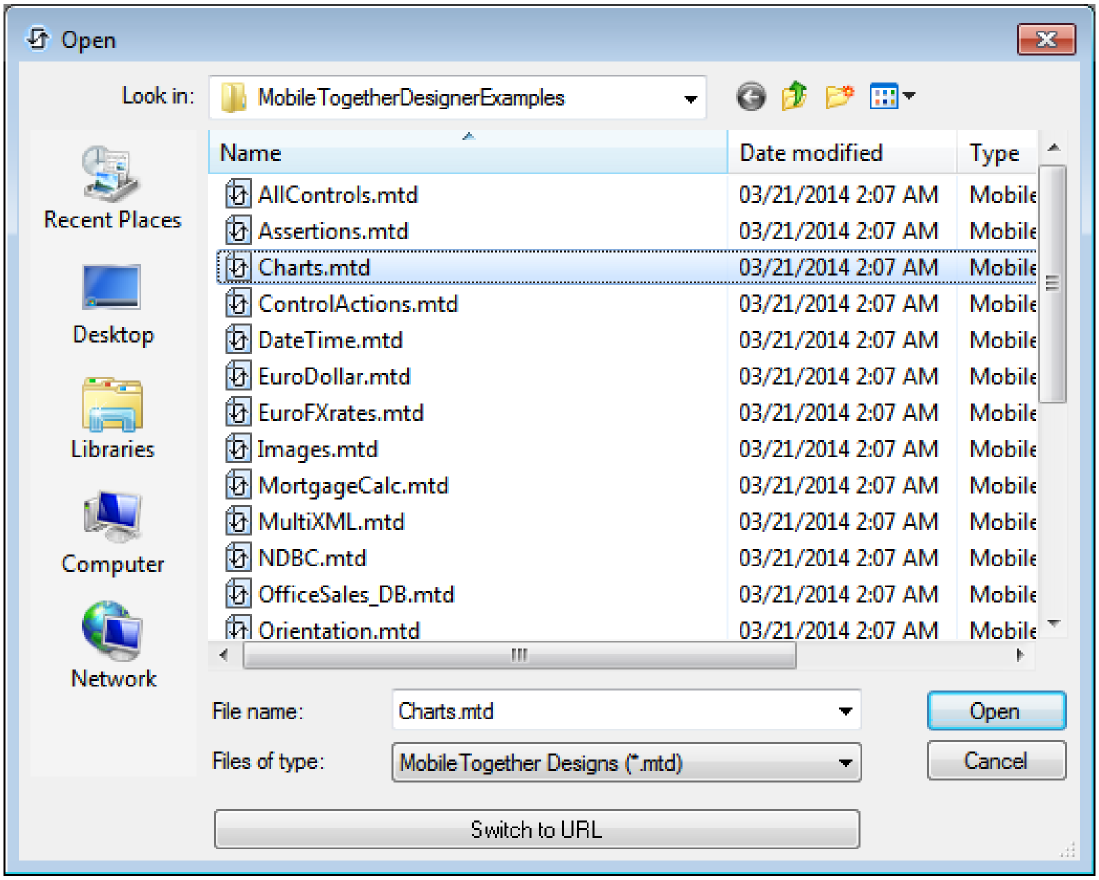
Save All

Icon

Description
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 and saving files via URLs

In several File Open and File Save dialogs, you can choose to select the required file or save a file via a URL (*see screenshot below*). Click **Switch to URL** to go to the selection process.



Look in: MobileTogetherDesignerExamples

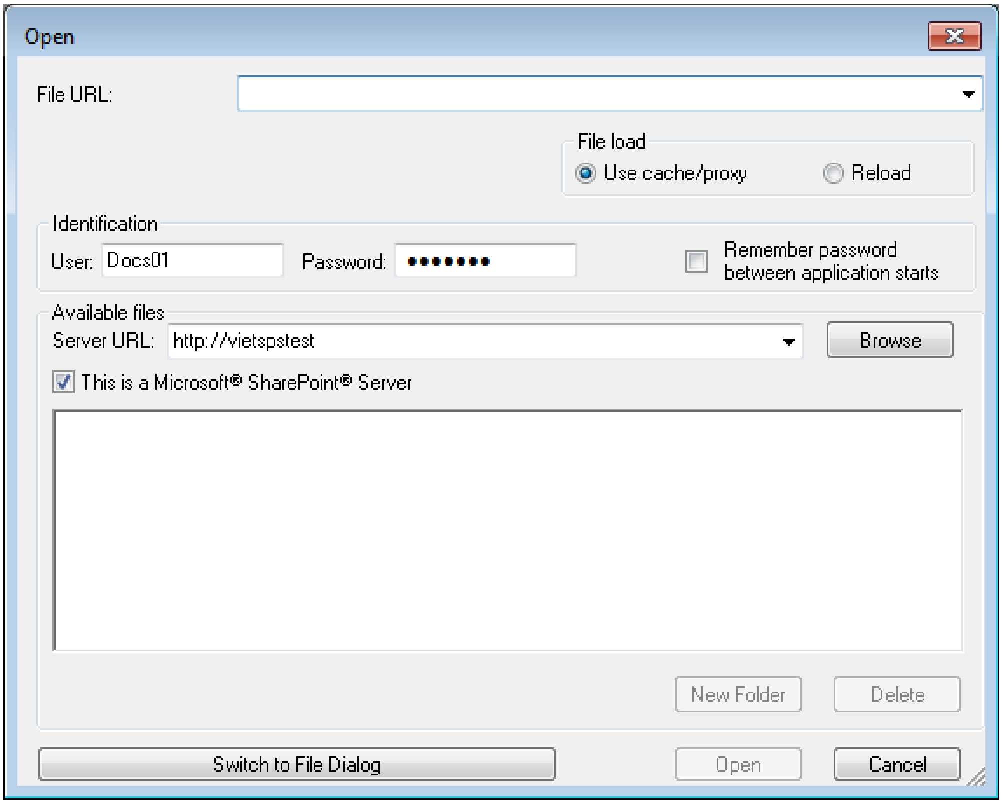
Name	Date modified	Type
AllControls.mtd	03/21/2014 2:07 AM	Mobile
Assertions.mtd	03/21/2014 2:07 AM	Mobile
Charts.mtd	03/21/2014 2:07 AM	Mobile
ControlActions.mtd	03/21/2014 2:07 AM	Mobile
DateTime.mtd	03/21/2014 2:07 AM	Mobile
EuroDollar.mtd	03/21/2014 2:07 AM	Mobile
EuroFXrates.mtd	03/21/2014 2:07 AM	Mobile
Images.mtd	03/21/2014 2:07 AM	Mobile
MortgageCalc.mtd	03/21/2014 2:07 AM	Mobile
MultiXML.mtd	03/21/2014 2:07 AM	Mobile
NDBC.mtd	03/21/2014 2:07 AM	Mobile
OfficeSales_DB.mtd	03/21/2014 2:07 AM	Mobile
Orientation.mtd	03/21/2014 2:07 AM	Mobile

File name: Charts.mtd

Files of type: MobileTogether Designs (*.mtd)

To select a file via a URL (either for opening or saving), do the following:

1. Click the **Switch to URL** command. This switches to the URL mode of the Open or Save dialog (*the screenshot below shows the Open dialog*).



File load: ● Use cache/proxy ○ Reload

User: Docs01 Password: ••••••• □ Remember password between application starts

Server URL: http://vietspstest

☑ This is a Microsoft® SharePoint® Server

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. The **Open** or **Save** button only becomes active at this point.
6. Click **Open** to load the file or **Save** to save it.

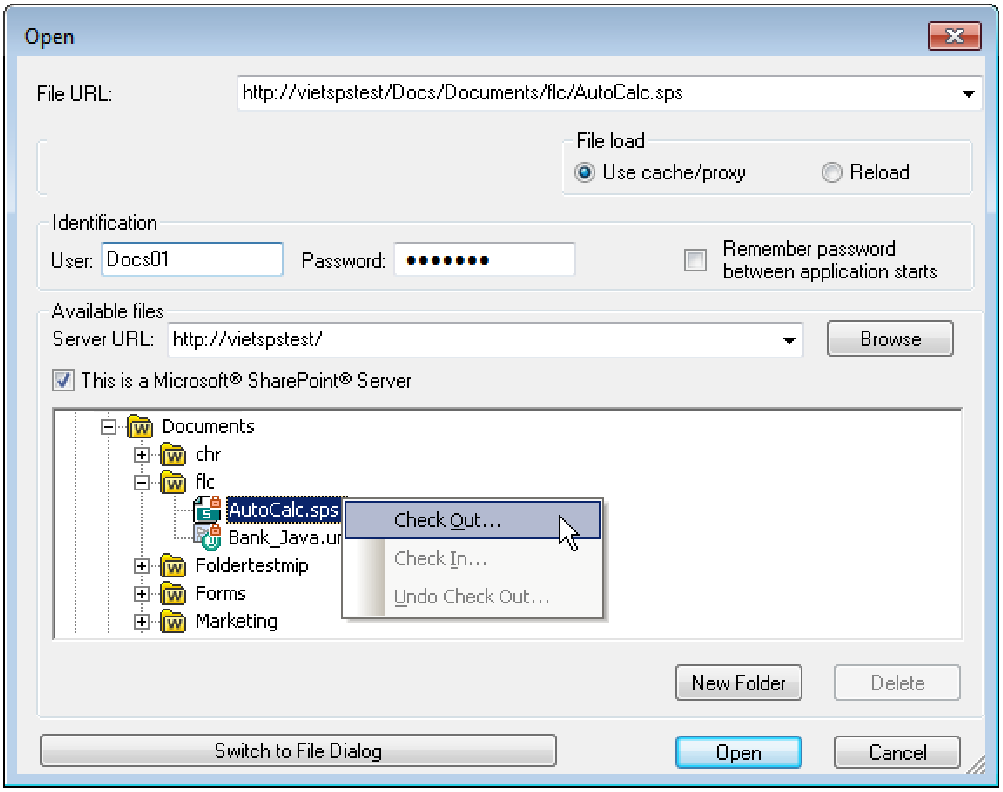
Note the following:

- The Browse function is only available on servers which support WebDAV and on Microsoft SharePoint Servers. The supported protocols are FTP, HTTP, and HTTPS.
- To give you more control over the loading process when opening a file, 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.



Open

File URL: http://vietspstest/Docs/Documents/flc/AutoCalc.sps

File load: ● Use cache/proxy ○ Reload

Identification

User: Docs01 Password: •••••••

□ Remember password between application starts

Available files

Server URL: http://vietspstest/

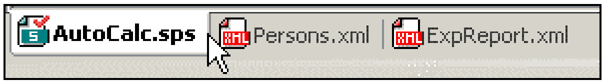
☑ This is a Microsoft® SharePoint® Server

Right-clicking a file pops up a context menu containing commands available for that file (*screenshot above*).

- The various file icons are shown below:

XML	Checked in. Available for check-out.
XML	Checked out by another user. Not available for check-out.
S	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 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**.

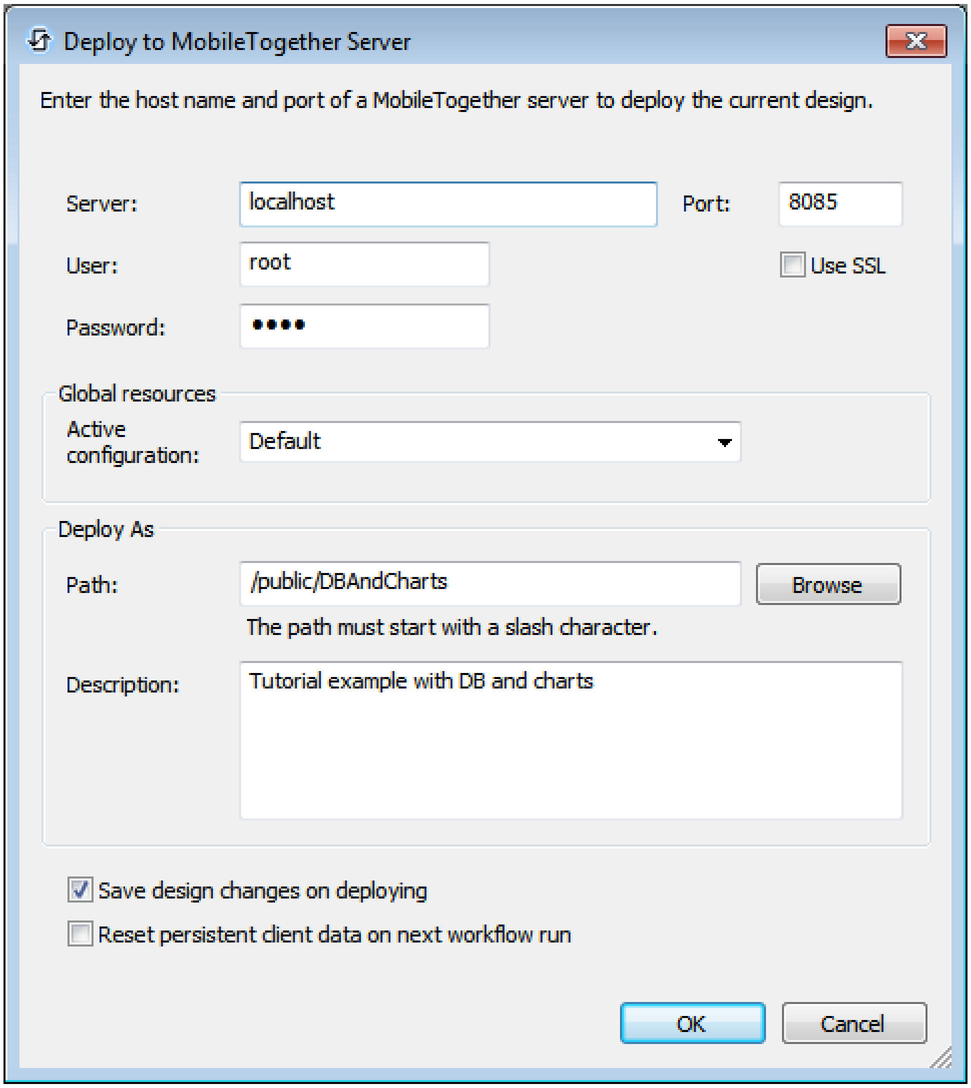
Deploy to MobileTogether Server

Deploy to MobileTogether Server

⊟ *Icon*

⊟ *Description*

Opens the Deploy to MobileTogether Server dialog, in which you specify: (i) server connection details, and (ii) deployment properties of the project on the server. On clicking **OK**, the currently active MobileTogether design file is deployed to MobileTogether Server.



Enter the host name and port of a MobileTogether server to deploy the current design.

Server: localhost
Port: 8085
User: root
☐ Use SSL
Password: ●●●●

Global resources
Active configuration: Default

Deploy As
Path: /public/DBAndCharts
The path must start with a slash character.
Description: Tutorial example with DB and charts

☑ Save design changes on deploying
☐ Reset persistent client data on next workflow run

The following details are required for deployment:

- *Server name and port:* The port refers to the MobileTogether Server administrator port and must match the *Administrator Ports setting* of MobileTogether Server. If you use SSL, make sure that you use the secure port of MobileTogether Server.
- *User name and password:* The user name and password of a user that has been

given the privilege *Save workflow from designer*. MobileTogether Server users and their privileges are specified in the *Users and Roles settings* of MobileTogether Server.

- *Active configuration of global resources:* Select the active configuration from the available configurations in the dropdown list of the combo box. The available configurations are those that are defined in the Global Resources Definitions file and are automatically obtained from there. See the section Altova Global Resources for details.
- *Deployment path and solution description:* The path and name of the deployed solution, and the description of the solution that will appear on the server.
- *Save design changes on deploying:* The project file is saved before deploying, so that the latest changes to the design are included.
- *Reset persistent client data on next workflow run:* Resets persistent client data when the solution is run the next time.

⊟ *Also see*

Location of Project Files
Files Pane
Deploying the Project
Data Storage on Servers.

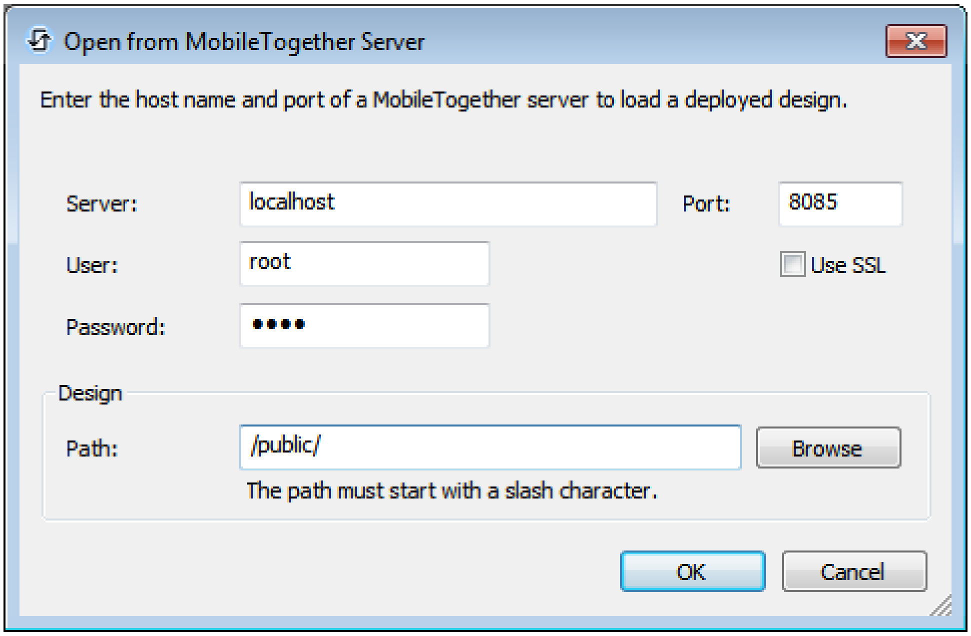
Open from MobileTogether Server

▼ Open from MobileTogether Server

⊟ *Icon*

⊟ *Description*

Opens a MobileTogether design file that has been deployed to MobileTogether Server server from its location on MobileTogether Server. Use the **Browse** button to select the file on the server that you want to open.



Open from MobileTogether Server

Enter the host name and port of a MobileTogether server to load a deployed design.

Server: localhost

Port: 8085

User: root

□ Use SSL

Password: ••••

Design

Path: /public/

The path must start with a slash character.

The following details are required for opening a design from MobileTogether Server:

- *Server name and port:* The port refers to the MobileTogether Server administrator port and must match the *Administrator Port setting* of MobileTogether Server. If you use SSL, make sure that you use the secure port of MobileTogether Server.
- *User name and password:* The user name and password of a user that has been given the privilege *Open workflow from designer*. MobileTogether Server users and their privileges are specified in the *Users and Roles settings* of MobileTogether Server.
- *Path and name of solution (design):* The path and name of the deployed solution. Click Browse to browse through all deployed solutions on MobileTogether Server.

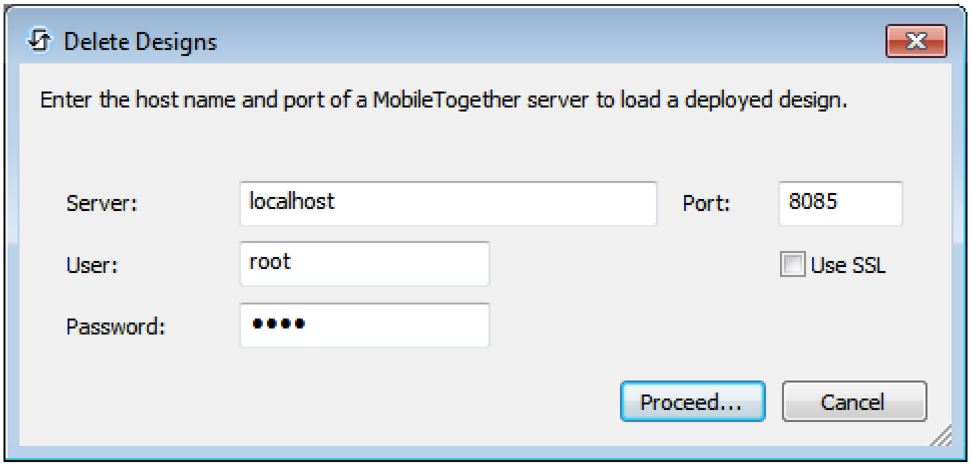
Delete from MobileTogether Server

Delete from MobileTogether Server

Icon

Description

Deletes a previously deployed design file from MobileTogether Server. Use the **Delete** button to select the file(s) on the server that you want to delete.



Enter the host name and port of a MobileTogether server to load a deployed design.

Server: localhost

Port: 8085

User: root

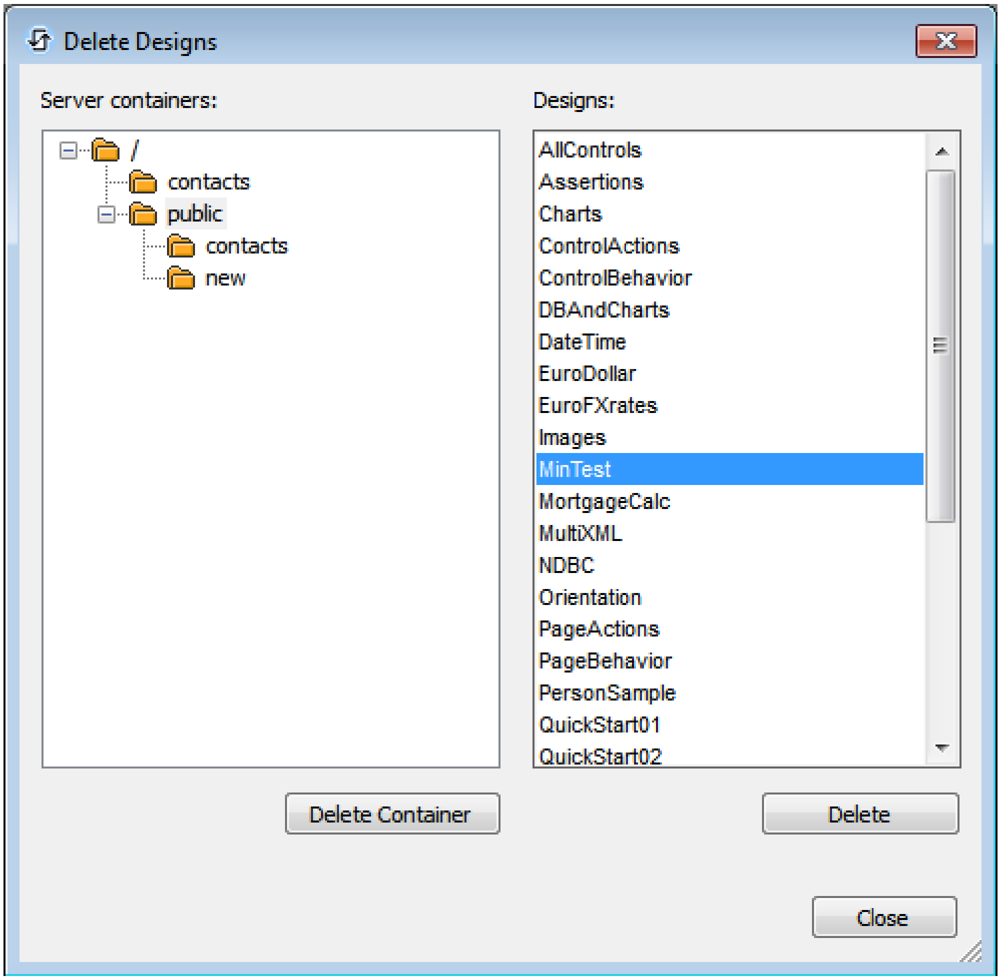
□ Use SSL

Password: ●●●●

The following details are required for connecting to the server:

- *Server name and port:* The port refers to the MobileTogether Server administrator port and must match the *Administrator Port setting* of MobileTogether Server. If you use SSL, make sure that you use the secure port of MobileTogether Server.
- *User name and password:* The user name and password of a user that has been given the privilege *Save workflow from designer*. MobileTogether Server users and their privileges are specified in the *Users and Roles settings* of MobileTogether Server.

On clicking Proceed, a window showing MobileTogether Server folders (containers) and their solutions (designs) is displayed (*screenshot below*). Browse for the container or design you wish to delete, select it, and click **Delete Container** or **Delete**.


Delete Designs
Server containers:
/
contacts
public
contacts
new
Designs:
AllControls
Assertions
Charts
ControlActions
ControlBehavior
DBAndCharts
DateTime
EuroDollar
EuroFXrates
Images
MinTest
MortgageCalc
MultiXML
NDBC
Orientation
PageActions
PageBehavior
PersonSample
QuickStart01
QuickStart02
Delete Container
Delete
Close

Send by Mail

▾ Send by Mail

⊟ *Icon*

⊟ *Description*

Sends the active MobileTogether Design document (MTD document) as an email attachment. On selecting the command, an email is opened with the active MTD document attached. Enter the name of the recipient/s in the *To:* field and subject information in the *Subject:* field of the email, then click the email application's **Send** command.

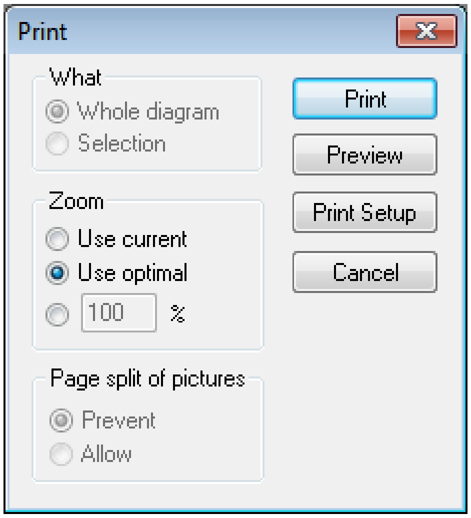
Print

▾ Print

⊟ *Icon*

⊟ *Shortcut*
Ctrl+P

⊟ *Description*
Opens the Print dialog box (*screenshot below*), in which you can select printing options for printing the currently active document.



Print

What
● Whole diagram
○ Selection

Zoom
○ Use current
● Use optimal
○ 100 %

Page split of pictures
● Prevent
○ Allow

Options that are applicable to the current printing job are enabled. The following options are available:

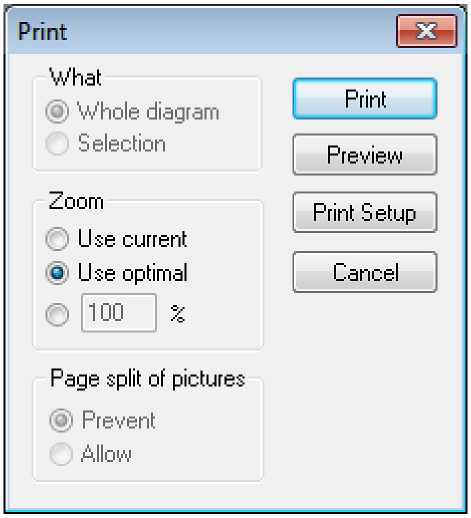
- *What:* Whether to print the entire design diagram or only the current selection.
- *Zoom:* The zoom level to use in th eprintout.
- *Page-split of pictures:* Whether images may be split (*Allow*) or not (*Prevent*).

Print Preview, Print Setup

Print Preview

Description

Opens the a print dialog box (*screenshot below*). Click **Preview** to see a preview of the currently active document according to the settings specified in the dialog.



Print

What
● Whole diagram
○ Selection

Zoom
○ Use current
● Use optimal
○ 100 %

Page split of pictures
● Prevent
○ Allow

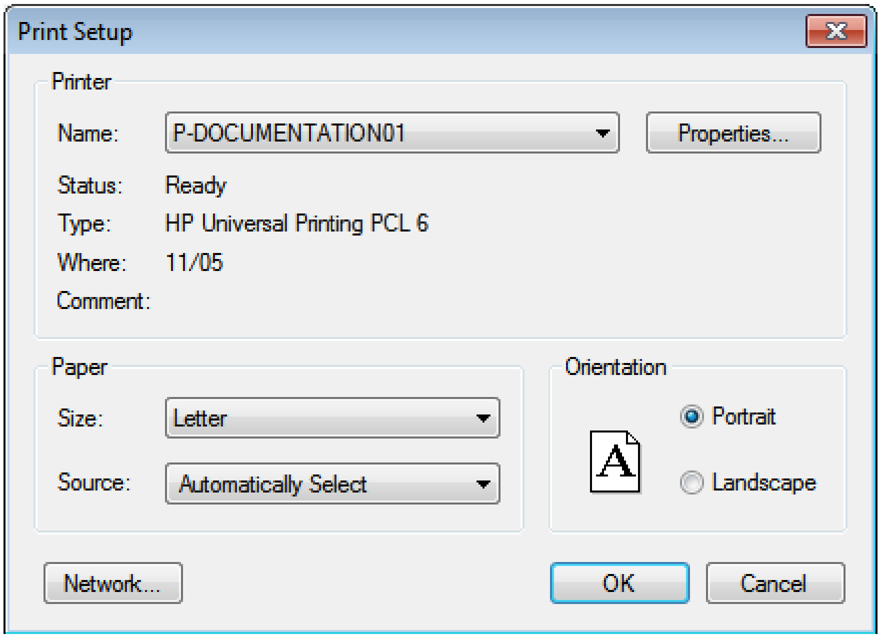
In Print Preview mode, the Print Preview toolbar at top left of the preview window provides print- and preview-related options. The preview can be magnified or miniaturized using the the **Zoom In** and **Zoom Out** buttons. When the page magnification is such that an entire page length fits in a preview 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.

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**.

Print Setup

Description

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.



Print Setup

Printer

Name: P-DOCUMENTATION01

Status: Ready

Type: HP Universal Printing PCL 6

Where: 11/05

Comment:

Paper

Size: Letter

Source: Automatically Select

Orientation

● Portrait

○ Landscape

Recent Files, Exit

▾ Recent Files

⊟ *Description*

At the bottom of the **File** menu is 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.

▾ Exit

⊟ *Description*

Quits MobileTogether Designer. If you have any open files with unsaved changes, you are prompted to save these changes. MobileTogether Designer also saves modifications to program settings and information about the most recently used files.

11.2 Edit

The **Edit** menu contains the following commands:

- Undo
- Redo
- Cut
- Copy
- Paste
- Delete
- Select All
- Find
- Find Next
- Find Previous

Undo, Redo

Undo

⊟ *Icon*

↶

⊟ *Shortcut*

Ctrl+Z

⊟ *Description*

Supports 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. You can step backwards and forwards through this history using the **Undo** and **Redo** commands (*see **Redo** command below*).

Redo

⊟ *Icon*

↷

⊟ *Shortcut*

Ctrl+Y

⊟ *Description*

Allows you to redo previously undone commands, thereby giving you a complete history of work completed. You can step backwards and forwards through this history using the **Undo** and **Redo** commands.

Cut, Copy, Paste, Delete

Cut

Icon

Shortcut
Ctrl+X or **Shift+Del**

Description
Copies the selected text or items to the clipboard, deleting the selection from its current location.

Copy

Icon

Shortcut
Ctrl+C

Description
Copies the selected text or items to the clipboard, *without* deleting the selection from its current location. The command can be used to duplicate data within MobileTogether Designer, or to move data to another application.

Paste

Icon

Shortcut
Ctrl+V

Description
Inserts the contents of the clipboard at the current cursor position.

Delete

Icon

⊟ *Shortcut*
Del

⊟ *Description*
Deletes the currently selected text or items without placing them in the clipboard.

Select All

▾ Select All

⊟ *Shortcut*
Ctrl+A

⊟ *Description*
Selects the contents of the entire document.

Find, Find Next, Find Previous

Find

Icon

Shortcut

Ctrl+F

Description

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 strings, and then specify the options for the search.

Find Next

Icon

Shortcut

F3

Description

Repeats the last **Find** command. It searches for the next occurrence of the input text.

Find Previous

Description

Repeats the last **Find** command, but in the reverse direction. It searches the document, upwards from the current position, for the next occurrence of the input text.

11.3 Project

The **Project** menu contains commands that apply to the entire project. It contains the following commands:

- Validate
- Reload Page Source Structures
- Simulate Workflow
- Trial Run on Client
- Use Server for Workflow Simulation
- Global Variables
- XPath/XQuery Functions
- Action Groups
- List Usages of All Action Groups
- Cache Overview
- Simulation Language
- Localization

Validate

Validate

Icon

Description

Validates the currently active project. If a project contains multiple pages, all the pages are validated. Validation results are reported in the Messages Pane in terms of the number of errors and warnings. If an error or warning is detected, a message about each is displayed.

Reload Page Source Structures

Reload Page Source Structures

Icon

Description

Reloads the page source data structures that have been defined in the Page Sources Pane. Use this command to refresh the page's source data while designing the page.

Simulate Workflow

▼ Simulate Workflow

⊟ *Icon*

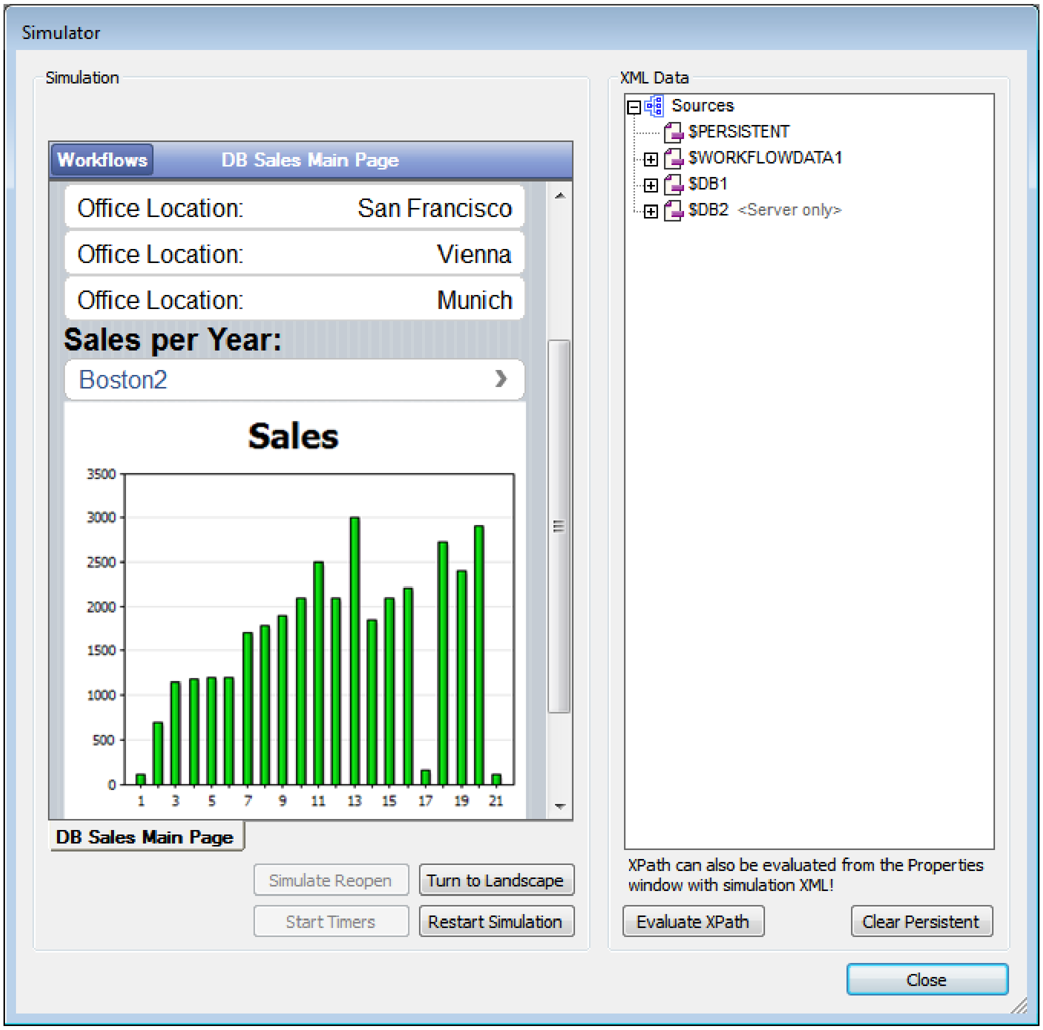
▶

⊟ *Shortcut*

F5

⊟ *Description*

Starts the local (MobileTogether Designer) simulator in a separate window for testing purposes (*see screenshot below*). The simulator goes step-by-step through the workflow of the currently active project. The mobile client that is currently selected in the Device Selector of the Page Settings toolbar will be simulated.



Office Location: San Francisco

Office Location: Vienna

Office Location: Munich

Sales per Year:

Boston2

XPath can also be evaluated from the Properties window with simulation XML!

Trial Run on Client

Trial Run on Client

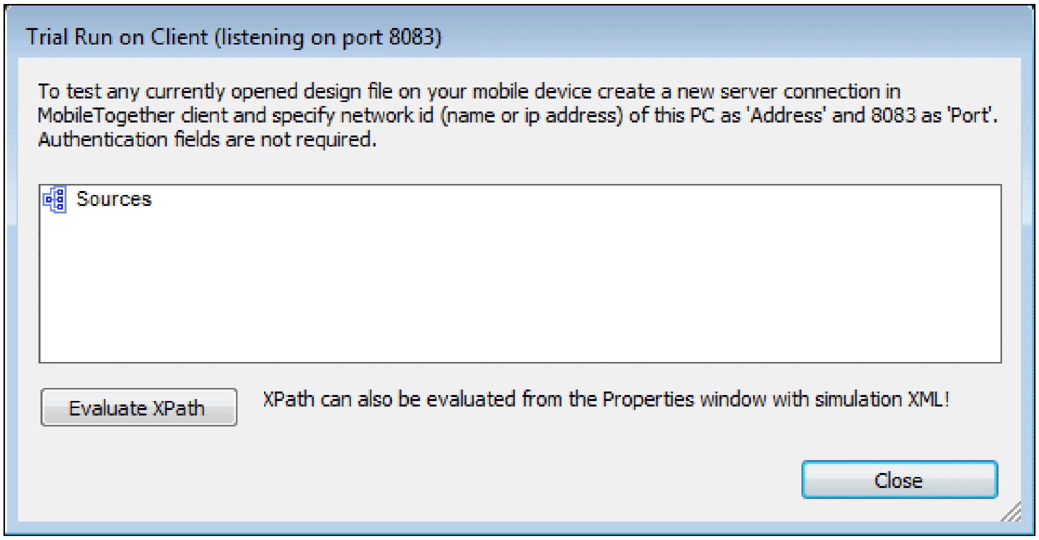
Icon

Shortcut

Shift+F5

Description

Tests the active MobileTogether design file on the specified client. MobileTogether Designer itself acts as the MobileTogether Server and serves the design and related data files directly to the client. In the MobileTogether Client app on your mobile device, you must set up a server connection to the local PC running MobileTogether Designer. Note that, by default, **`8083`** is the port on your local PC to which the client must connect. You can change this port in the Trial Run on Client tab of the Options dialog. After the connection between client and PC is established and the design is selected on the client, the Sources tree in the Trial Run on Client dialog (*screenshot below*) is populated and the trial run (simulation) begins.



Trial Run on Client (listening on port 8083)

To test any currently opened design file on your mobile device create a new server connection in MobileTogether client and specify network id (name or ip address) of this PC as 'Address' and 8083 as 'Port'. Authentication fields are not required.

XPath can also be evaluated from the Properties window with simulation XML!

Use Server for Workflow Simulation

Use Server for Workflow Simulation

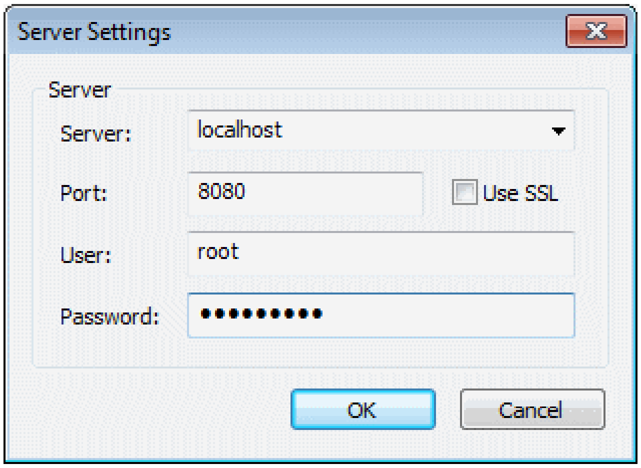
⊟ *Icon*

⊟ *Shortcut*

Ctrl+F5

⊟ *Description*

Displays the Server Settings dialog (*screenshot below*). Enter the connection and authentication datails of the MobileTogether Server on which you want to run the simulation. On clicking **OK**, the simulation starts in a separate window.



Server Settings

Server

Server: localhost

Port: 8080 □ Use SSL

User: root

Password: ••••••••••

Global Variables

Global Variables

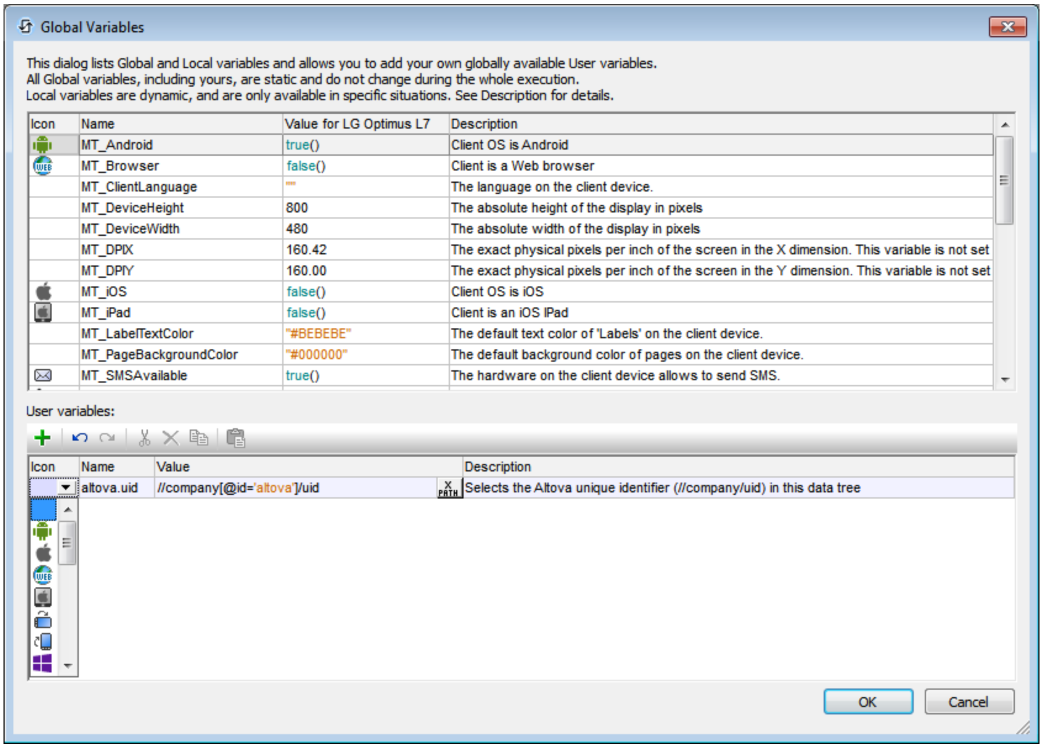
Icon

VAR

Description

Opens the Global Variables dialog (*screenshot below*). Global variables are available to the designer in programming contexts, such as XPath or XQuery, everywhere in the project. MobileTogether Designer provides a standard library of global variables, which are listed in the upper pane of the dialog. The values of the global variables are assigned by MobileTogether during simulation and when the app runs on the client device. Global variables are of three types, and the list of variables in the dialog is divided into three parts for these three types:

- Static-value variables (called *Global Variables* in the dialog): These variables contain information about the mobile device. Values of these variables do not change during the execution of the project. Notice that the *Value* column in the upper pane specifies the currently selected mobile device. The listed values are for that particular device. For simulations, the client device is considered to be the device selected in the Device Selector combo box. For example, the variable **`$MT_Android`** has a value of **`true`** when the mobile device being used is an Android. (Device information is sent by the device as part of standard mobile communication procedures.)
- Dynamic-value variables (called *Local Variables* in the dialog): These variables contain information related to the design page and its controls. Their values could change during execution. For example, the **`$MT_ControlNode`** variable has different values according to which node is the page source link of the current control at a given time during project execution.
- *User variables*: In addition to the standard library of global variables, you can add your own global variables (called *User Variables* in the dialog) in the lower pane of the dialog. You can give a user variable any value, and this value can then be used subsequently in any XPath/XQuery expression in the project.



This dialog lists Global and Local variables and allows you to add your own globally available User variables.
All Global variables, including yours, are static and do not change during the whole execution.
Local variables are dynamic, and are only available in specific situations. See Description for details.

Icon	Name	Value for LG Optimus L7	Description
	MT_Android	true()	Client OS is Android
	MT_Browser	false()	Client is a Web browser
	MT_ClientLanguage	""	The language on the client device.
	MT_DeviceHeight	800	The absolute height of the display in pixels
	MT_DeviceWidth	480	The absolute width of the display in pixels
	MT_DPIX	160.42	The exact physical pixels per inch of the screen in the X dimension. This variable is not set
	MT_DPIY	160.00	The exact physical pixels per inch of the screen in the Y dimension. This variable is not set
	MT_iOS	false()	Client OS is iOS
	MT_iPad	false()	Client is an iOS IPad
	MT_LabelTextColor	"#BEBEBE"	The default text color of 'Labels' on the client device.
	MT_PageBackgroundColor	"#000000"	The default background color of pages on the client device.
✉	MT_SMSAvailable	true()	The hardware on the client device allows to send SMS.

Icon	Name	Value	Description
	altova.uid	//company[@id='altova']/uid	Selects the Altova unique identifier (//company/uid) in this data tree

To add a user variable, in the lower pane, do the following:

1. Click the **Append** or **Insert** icons (located in the pane's toolbar) to add a line to the list.
2. Enter the name of your new variable (in the *Name* column) and give the variable a description (*Description* column). *See screenshot above*.
3. Click in the *Value* field to bring up the Edit XPath/XQuery Expression dialog, and enter the XPath expression that determines the value of this variable.
4. Select an icon to help identify the new variable as belonging to a particular group.
5. Click **OK** to finish. The variable is added as a global variable, and can be used in programming contexts.

See the section, Global Variables for a description of predefined global variables.

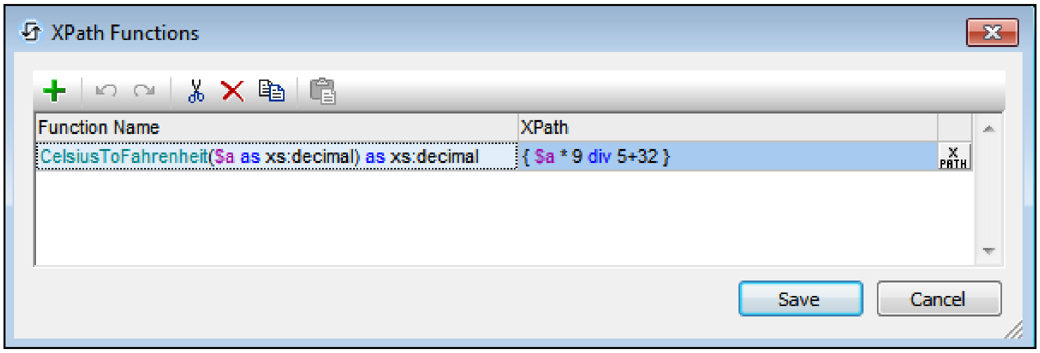
XPath/XQuery Functions

XPath/XQuery Functions

⊟ *Icon*

⊟ *Description*

Opens the XPath Functions dialog, which lists all the user-defined XPath functions in the project. These XPath functions can be used in all XPath expressions in the project. You can add and delete functions using the corresponding icons in the toolbar of the dialog. To edit the definition of a function, click the function's **Edit XPath Expression** button.

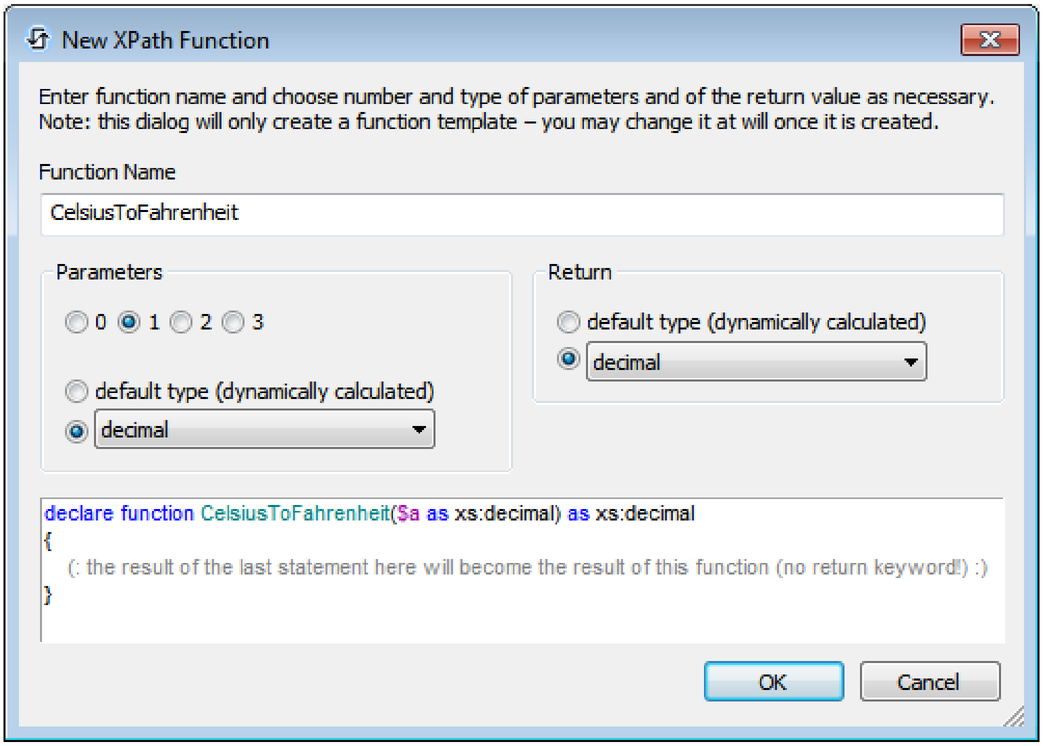


Function Name	XPath
CelsiusToFahrenheit($a as xs:decimal) as xs:decimal	{ $a * 9 div 5+32 }

Add a new user-defined XPath function

Adding a new user-defined function involves two steps: (i) declaring the function, and (ii) defining the function.

To add a new function, do the following, click **Add** in the toolbar of the dialog (*see screenshot above*). This displays the New XPath Function dialog (*screenshot below*).



Enter function name and choose number and type of parameters and of the return value as necessary.
Note: this dialog will only create a function template – you may change it at will once it is created.

Function Name: CelsiusToFahrenheit

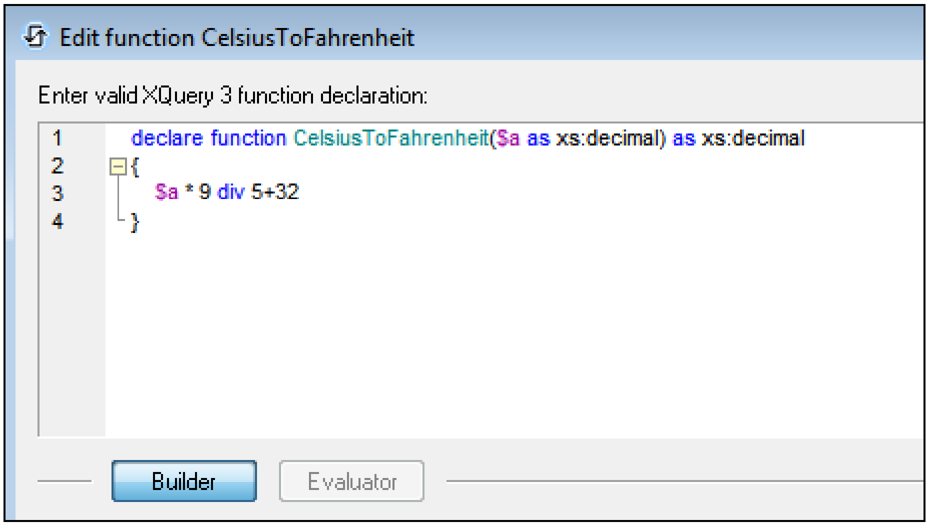
Parameters: ○ 0 ● 1 ○ 2 ○ 3

○ default type (dynamically calculated)
● decimal

Return:
○ default type (dynamically calculated)
● decimal

```
declare function CelsiusToFahrenheit($a as xs:decimal) as xs:decimal
{
    (: the result of the last statement here will become the result of this function (no return keyword!) :)
}
```

In this dialog, you can declare the name of the function, specify the number of function parameters (arguments) and their types, and specify the return type of the function. In the screenshot above we have declared a function to convert a decimal number from Celsius to Fahrenheit. The function takes one parameter, which is the input Celsius value as a decimal. It will output a decimal value, the Fahrenheit temperature. What the function does is defined in the next step. After declaring the function (*screenshot above*), click **OK**. This displays the Edit Function dialog (*screenshot below*), which contains the template of the newly declared function and in which you can now define the function.



Enter valid XQuery 3 function declaration:

```
declare function CelsiusToFahrenheit($a as xs:decimal) as xs:decimal
{
    $a * 9 div 5+32
}
```

Enter the definition of the function within the braces. In the definition shown in the screenshot above, **$a** is the input parameter. Click **OK**. when done. The function will be added to the list of user-defined functions in the XPath Functions dialog and can be used in all XPath expressions in the project.

List Usages of All User-Defined XPath/XQuery Functions

▾ List Usages of All User-Defined XPath/XQuery Functions

⊟ *Description*

Returns a list, in the Messages Pane, of all the user-defined XPath/XQuery functions used by pages in the project, together with the pages in which they occur.

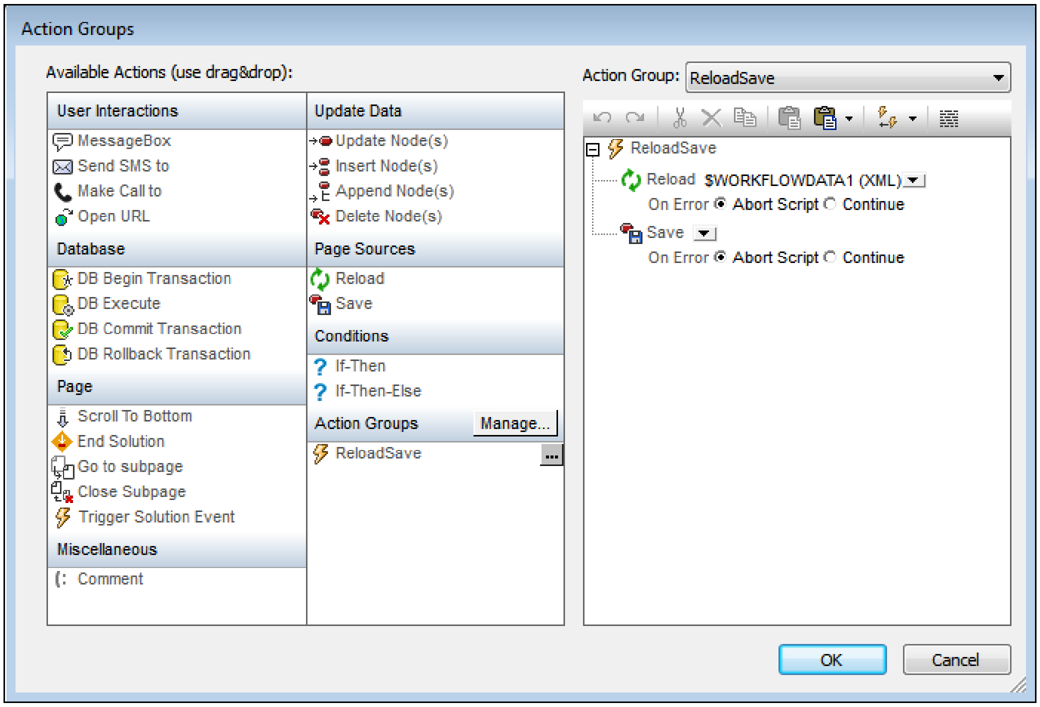
Action Groups

Action Groups

Icon

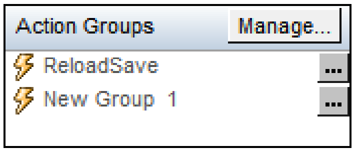
Description

Displays the Action Groups dialog (screenshot below), in which action groups can be created and managed.



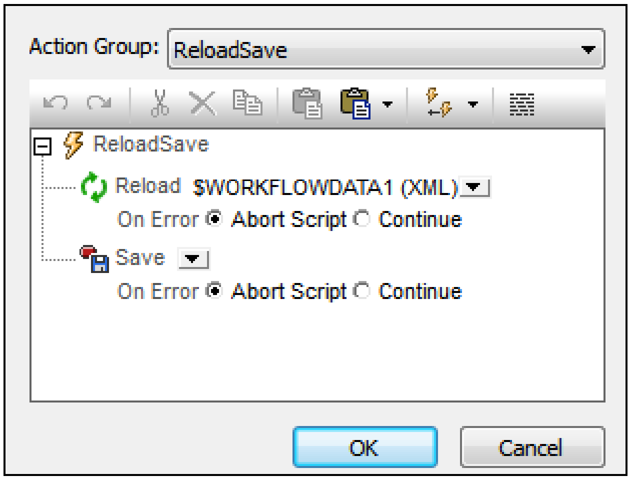
To create an Action Group, do the following:

1. Click **Manage** in the Action Groups pane (*see screenshot below*).



2. In the Manage Action Groups dialog that appears, click **Add** in the toolbar of the dialog. An Action Group with a default name is added to the list of Action Groups in this dialog.
3. Change the name of the Action Group to a suitable name, and click **OK**. The Manage Action Groups dialog closes, and the new Action Group is added to the list of groups in the Action Groups pane (*see screenshot above*).
4. In the Action Groups pane (*screenshot above*), click the **Edit** icon of the Action Group you want to edit. The group's contents are displayed in the pane on the right (*see screenshot below*). The details of an Action Group can also be

displayed by selecting the group in the Action Group combo box.



Action Group: ReloadSave

ReloadSave
Reload $WORKFLOWDATA1 (XML)
On Error ● Abort Script ○ Continue
Save
On Error ● Abort Script ○ Continue

5. To populate the active Action Group in the right-hand pane, drag and drop actions from the available actions in the Action Groups dialog.
6. Click **OK** to finish. The Action Group is available for use as a Page Action.

List Usages of All Action Groups

▼ List Usages of All Action Groups

⊟ *Description*

Returns a list, in the Messages Pane, of all the Action Groups used by pages in the project. The list is ordered by Action Group. Each Action Group is sub-divided into direct and indirect usages. Pages that use the Action Group are listed together with the event that triggers the Action Group.

Cache Overview

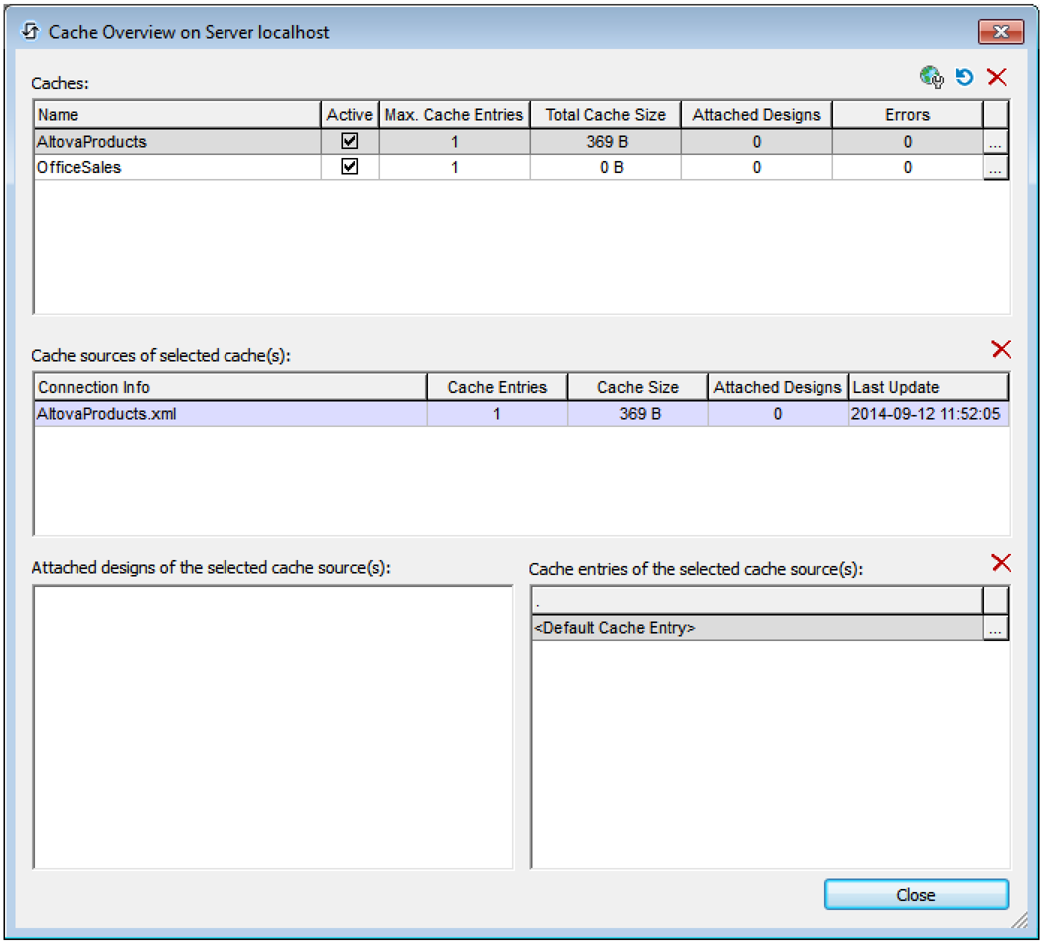
Cache Overview

Icon

Description

Enables connection to MobileTogether Server and an overview of the server cache: cache hits, cache size, purges, etc.

The Cache Overview dialog (*screenshot below*) is accessed with the menu command **Project | Cache Overview**.



Cache Overview on Server localhost

Caches:

Name	Active	Max. Cache Entries	Total Cache Size	Attached Designs	Errors	
AltovaProducts	☑	1	369 B	0	0	...
OfficeSales	☑	1	0 B	0	0	...

Cache sources of selected cache(s):

Connection Info	Cache Entries	Cache Size	Attached Designs	Last Update
AltovaProducts.xml	1	369 B	0	2014-09-12 11:52:05

Attached designs of the selected cache source(s):

Cache entries of the selected cache source(s):

.	
<Default Cache Entry>	...

The dialog provides an overview of all the caches on the server. In it, you can also do the following:

- Activate/deactivate caches.
- Delete caches.

See the section Caching for more information about caching.

Simulation Language

▾ Simulation Language

⊟ *Description*

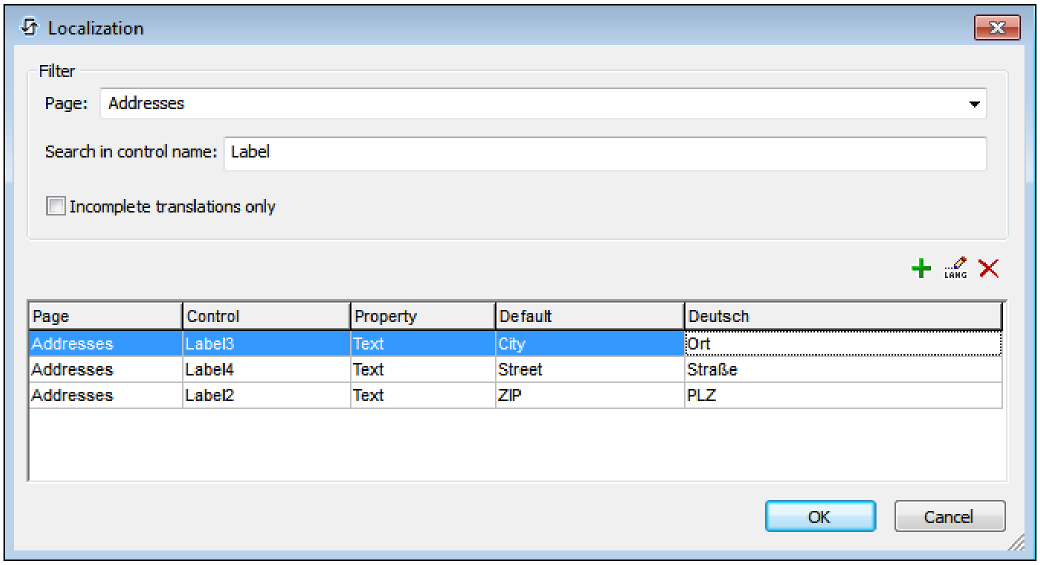
If MobileTogether Designer has been localized, this command lets you select the language used by the simulator. The command has a submenu that contains the localization languages as menu items.

Localization

Localization

Description

Displays the Localization dialog (*screenshot below*), in which you can localize (translate) strings that appear in various controls. These localized strings can the be used to specify the language in which the simulation should be carrried out. The dialog displays the strings of one project page or all project pages, together with their details.



Page: Addresses

Search in control name: Label

☐ Incomplete translations only

Page	Control	Property	Default	Deutsch
Addresses	Label3	Text	City	Ort
Addresses	Label4	Text	Street	Straße
Addresses	Label2	Text	ZIP	PLZ

- You can select the page to display in the Page combo box. If no page is selected in the Page combo box, then all pages are displayed.
- The *Search in Control Name* text box restricts the list to the results of only one control.
- If the Incomplete Translations check box is selected, only those strings are shown, for which no translation has been done.
- Click the **Add Language** icon to add a column for another language.
- Click the **Rename Language** icon to change the title of the languiage column and the entry in the sub-menu of the **Simulation Language** command.
- To delete a localization language column and its entries, place the cursor in that column and click the **Delete Language** icon.

11.4 Page

The **Page** menu contains commands that apply to the currently active page of a project. It contains the following commands:

- Page Actions
- Actions Overview
- Jump to Control

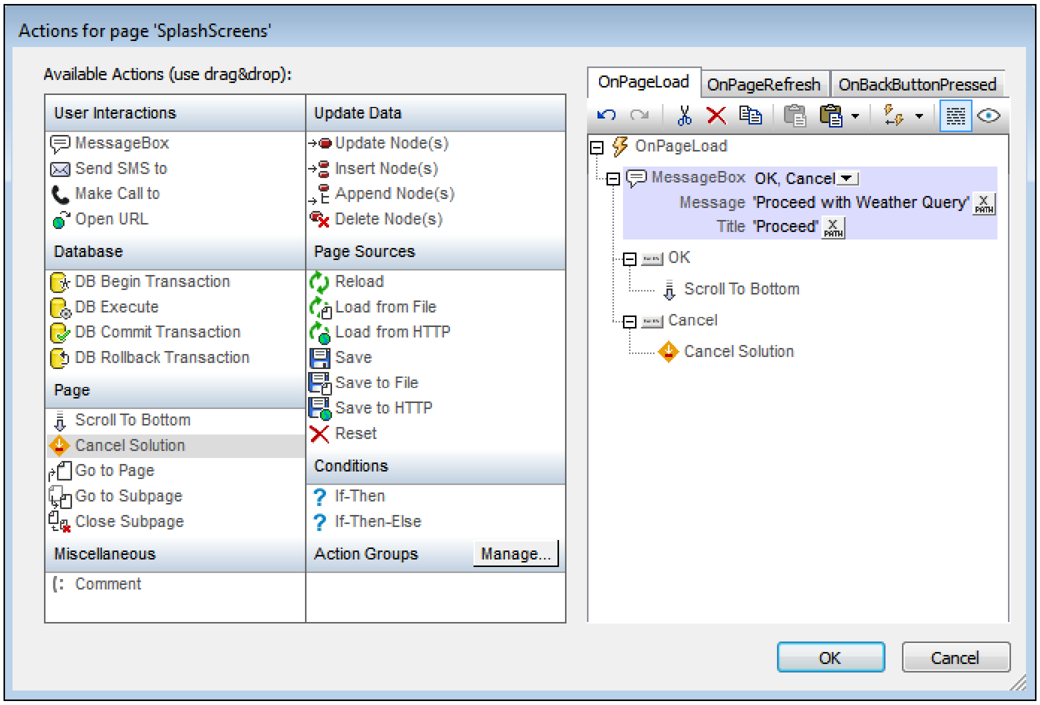
Note: **Page** menu commands are enabled only in Page Design View.

Page Actions

Page Actions

Description

Displays the Actions dialog of the currently active page (*see screenshot below*). The left-hand pane of the dialog contains the available actions, organized by functionality. The right-hand pane contains tabs of available events for that page. The events that are available depend on the role of that page in the project workflow. For instance, a page that cannot be returned to by pressing the **Back** button will not have the `OnBackButtonPressed` event tab.



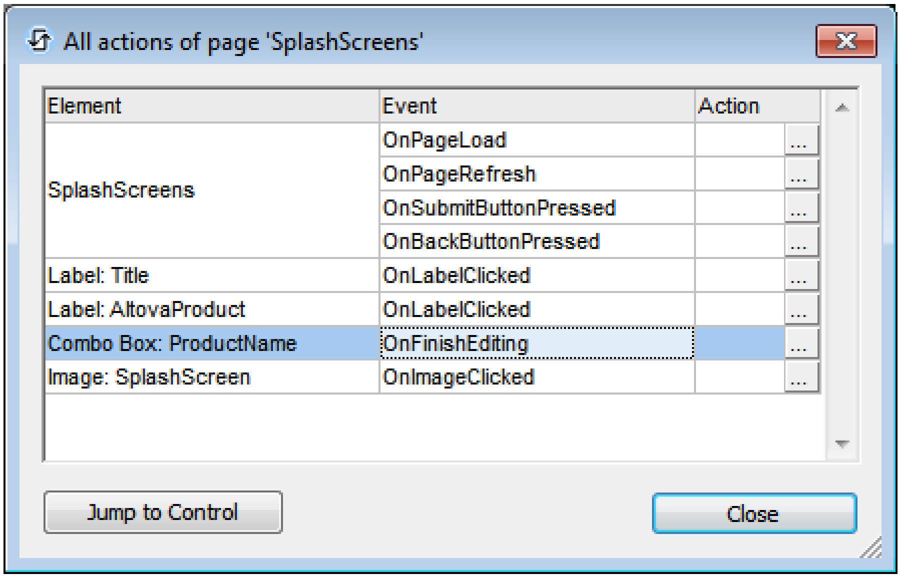
To specify that a particular action (from among the available actions) is carried out when an available event occurs, drag the action from the left-hand pane into the event's tab in the right-hand pane. Specify additional properties of the action as required. (For more information about individual page actions, see the section, Page Actions.) Note that you can also add any Action Group that might be defined for the project. Click **OK** to finish.

Actions Overview

Actions Overview

Description

Displays the Actions Overview dialog of the currently active page (*see screenshot below*). The dialog shows Control Actions and Page Actions. Each control in the page design is listed, together with its event/s and corresponding action/s; these are Control Actions. The dialog also shows available events for the active page; these are Page Actions. For instance, in the screenshot below, the `Income` item is a page item: it has page events that take page actions. All the other items are controls: they have control events that take control actions.



Element	Event	Action
SplashScreens	OnPageLoad	...
	OnPageRefresh	...
	OnSubmitButtonPressed	...
	OnBackButtonPressed	...
Label: Title	OnLabelClicked	...
Label: AltovaProduct	OnLabelClicked	...
Combo Box: ProductName	OnFinishEditing	...
Image: SplashScreen	OnImageClicked	...

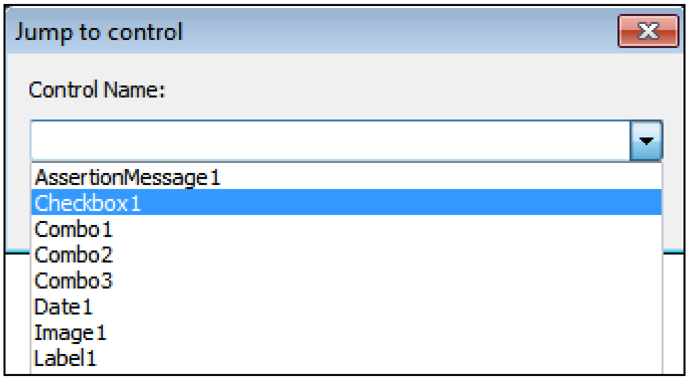
- To create an action for any event or to edit an existing action, click the the **Edit** icon of that event (*see screenshot above*). This takes you to the Actions dialog of that control or to the Actions dialog of the page.
- When a control is selected (not the page), then the **Jump to Control** button is enabled. Clicking it takes you to that control in the design.

Jump to Control

Jump to Control

Shortcut
Ctrl+J

Description
Displays the Jump to Control dialog (*screenshot below*). The dialog contains a combo box with a dropdown list that shows all the controls of the currently active page design. Page controls are listed alphabetically by the value of their `Name` properties.



Select a page control from the dropdown list or enter the control's name (auto-completion is available). The selected or entered page control gets the focus in the design. If the control is associated with a data node, that node in the Page Sources Pane gets the focus. Click **OK** to finish.

11.5 View

The **View** menu contains the following commands:

- Status Bar
- Pages
- Controls
- Page Sources
- Properties
- Messages
- All On/Off
- Zoom
- Zoom In
- Zoom Out
- Zoom Reset to 100%
- Zoom to Selection
- Fit to Window

Status Bar and Panes

The display of the Status Bar and various panes can be toggled on/off by clicking their commands in the View Menu:

- Status Bar
- Pages Pane
- Controls Pane
- Page Sources Pane
- Styles & Properties Pane
- Messages Pane

The **All On/Off** command toggles all the panes and the status bar together between being displayed and being hidden.

Zoom Levels

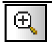
The zoom commands taken together provide considerable flexibility in changing the magnification level of the page design (between 10% and 100%) and the workflow diagram (between 10% and 200%). The zoom commands are enabled in Page Design View.

Zoom

Description
Opens the Zoom dialog, which contains a slid rule that enables you to change the magnification from 10% to 100% in Page Design View.

Zoom In

Icon



Description
Magnifies the diagram by 10%. In Page Design View, there might be an upper limit due to template size constraints.

Zoom Out

Icon

Description
Reduces the diagram by 10% in size.

Zoom Reset to 100%

Description
Resets the zoom factor to 100%. This is a quick way to regain the original size.

11.6 Tools

The **Tools** menu contains the following commands:

- Global Resources
- Active Configuration
- Customize
- Restore Toolbars and Windows
- Options

Global Resources

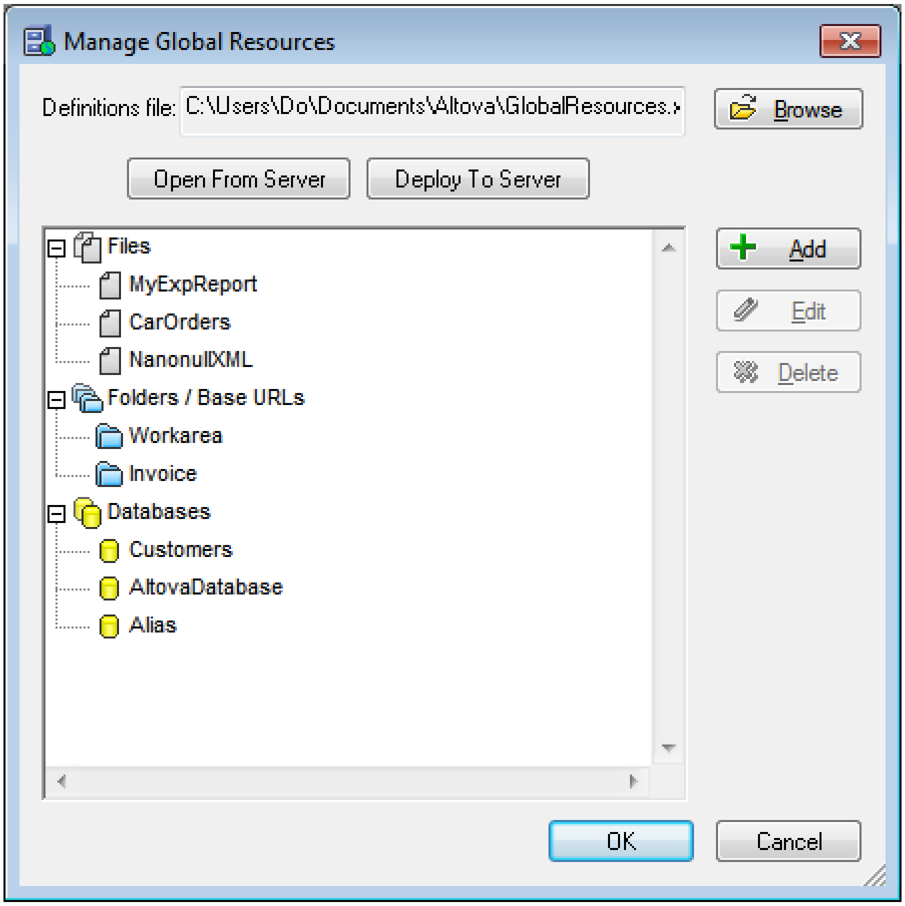
Global Resources

⊟ *Icon*

⊟ *Description*

Displays the Global Resources dialog (*screenshot below*), in which you can:

- Specify the Global Resources XML File to use for global resources (*Definitions file*).
- Add file, folder, and database global resources (or aliases)
- Specify various configurations for each global resource (alias). Each configuration maps to a specific resource. (Edit a a global resource to do this.)
- **Open From Server** and **Deploy to Server**, respectively, enables you to open a global resource from and deploy a global resource to a MobileTogether Server..



Definitions file: C:\Users\Do\Documents\Altova\GlobalResources.›

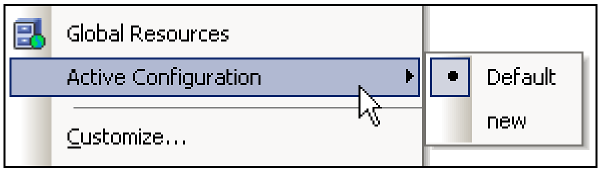
How to define global resources is described in detail in the section, Defining Global Resources.

Active Configuration

Active Configuration

Description

Placing the mouse over the command rolls out a submenu containing all the configurations defined in the currently active Global Resources XML File.



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.

Customize

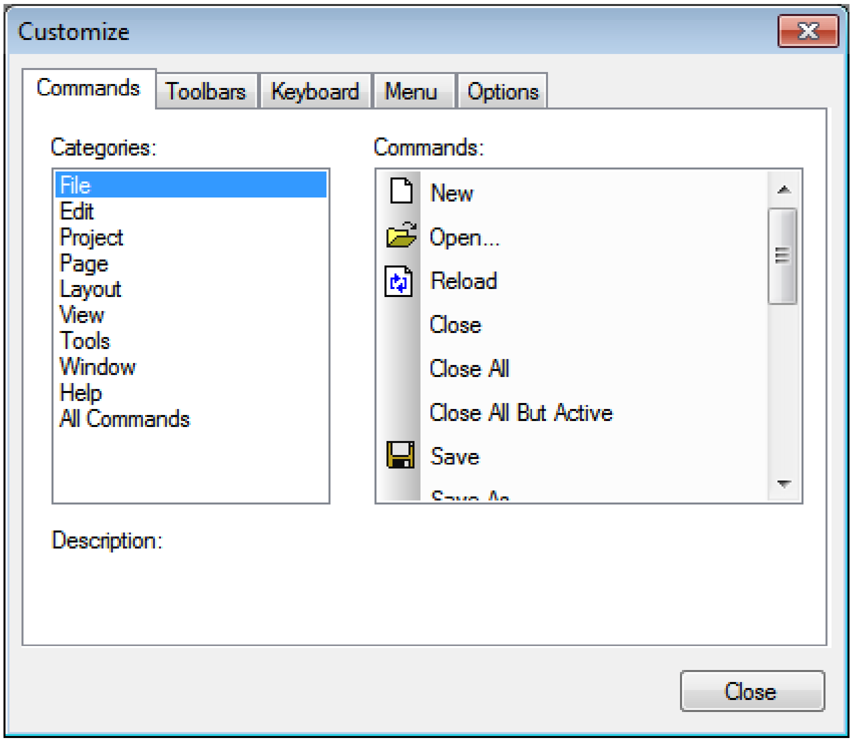
The customize command lets you customize MobileTogether Designer to suit your personal needs.

- Commands
- Toolbars
- Keyboard
- Menu
- Options

Commands

Commands

The Commands tab enables you to add application commands to menus and toolbars according to your preference. Note that you cannot create new application commands or menus yourself.



To add a command to a toolbar or menu, do this:

1. Select the **All Commands** category in the *Categories* list box. The available commands appear in the *Commands* list box.
2. 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.
3. Release the mouse button at the position you want to insert the command.

Note the following:

- When you drag a command, a small button appears at the tip of mouse pointer: This indicates that the command is currently being dragged.
- An "**x**" below the pointer indicates that the command cannot be dropped at the current cursor position.
- If the cursor is moved to a position at which the command can be dropped (a toolbar or menu), the "**x**" disappears and an **I**-beam indicates the valid position.
- Commands can be placed in menus or toolbars. If you have created you own toolbar, you can use this customization mechanism to populate the toolbar.
- Moving the cursor over a closed menu, opens that menu, allowing you to insert the

command anywhere in that menu.

To add a command to a context menu, do this:

1. In the Customize dialog, click the **Menu** tab.
2. In the Context Menu pane, select a context menu from the combo box. The selected context menu appears.
3. In the Customize dialog, switch back to the Commands tab.
4. Drag the command you wish to create from the *Commands* list box and drop it onto the desired location in the context menu.

To delete a command from a menu, context menu, or toolbar, or to delete an entire menu, do this.

1. With the Customize dialog open (and any tab selected), right-click a menu or a menu command.
2. Select **Delete** from the context menu that appears. Alternatively, drag the menu or menu command till an "**x**" icon appears below the mouse pointer, and then drop the menu or menu command.

To reinstate deleted menu commands, use the mechanisms described in this section. To reinstate a deleted menu, go to **Tools | Customize | Menu**, and click the **Reset** button in the *Application Frame Menus* pane. Alternatively, go to **Tools | Customize | Toolbars**, select Menu Bar, and click the **Reset** button.

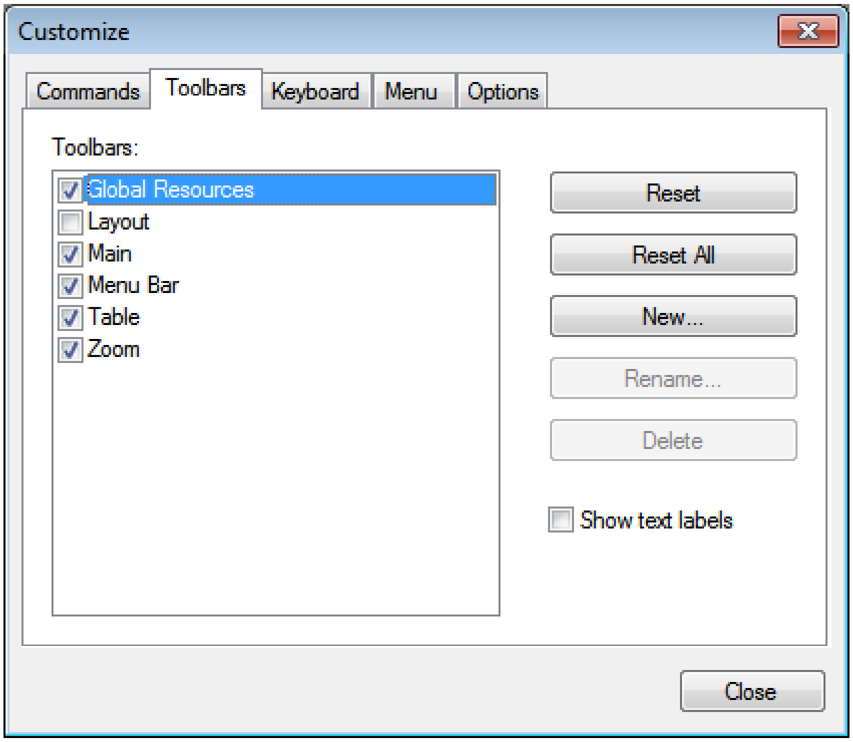
Toolbars

Toolbars

Application toolbars contain icons for the most frequently used menu commands. Information about each icon is displayed in a tooltip and in the Status Bar when the cursor is placed over the icon. You can drag a toolbar to any location on the screen, where it will appear as a floating window.

The Toolbars tab enables you to do the following:

- Activate or deactivate specific toolbars (that is, to decide which ones to display in the interface)
- Set what icons are displayed in each toolbar
- Create your own specialized toolbars



Toolbars:

☑ Global Resources
☐ Layout
☑ Main
☑ Menu Bar
☑ Table
☑ Zoom

☐ Show text labels

The following functionality is available:

- *To activate or deactivate a toolbar:* Click its check box in the *Toolbars* list box.
- *To apply changes to all views:* Click the check box at the bottom of the dialog. Otherwise, changes are applied only to the active view. Note that only changes made **after** clicking the *All Views* check box will apply to all views.
- *To add a new toolbar*: Click the **New** button and give the toolbar a name in the Toolbar Name dialog that pops up. From the Commands tab drag commands into the new toolbar.
- *To change the name of an added toolbar:* Select the added toolbar in the Toolbars pane, click the **Rename** button, and edit the name in the Toolbar Name dialog that pops up.

- *To reset the Menu bar*: Select the *Menu Bar* item in the Toolbars pane, and then click **Reset**. This resets the Menu bar to the state it was in when the application was installed.
- *To reset all toolbar and menu commands:* Click the **Reset All** button. This resets all toolbars and menus to the states they were in when the application was installed.
- *To delete a toolbar:* Select the toolbar you wish to delete in the Toolbars pane and click **Delete**.
- *To show text labels of commands in a particular toolbar*: Select that toolbar and click the *Show Text Labels* check box. Note that text labels have to be activated for each toolbar separately.

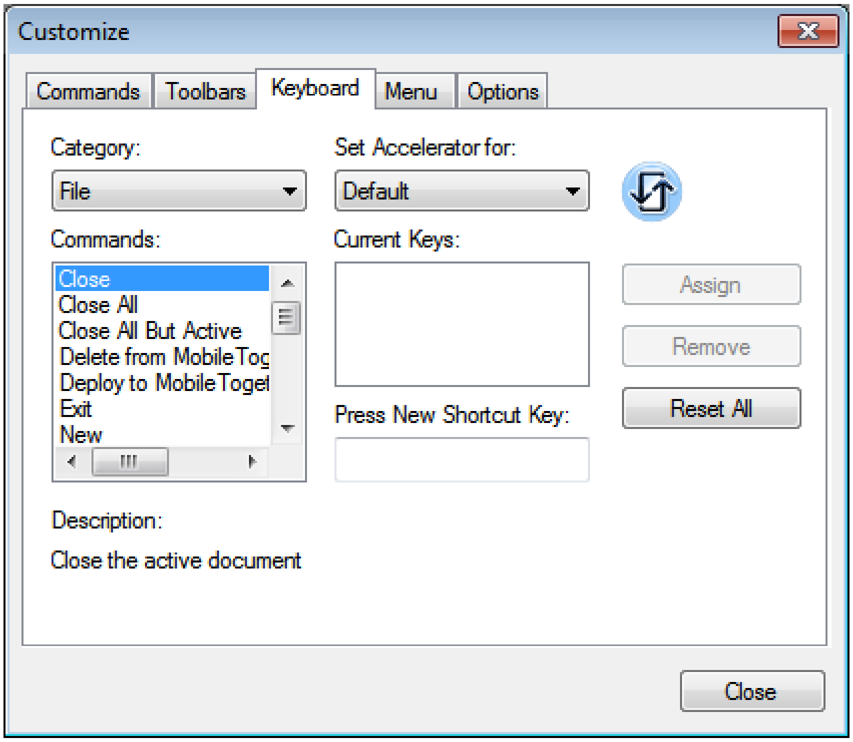
Note: To add a command to a toolbar, drag the command you want from the *Commands* list box in the Commands tab to the toolbar. To delete a command from a toolbar, open the Customize dialog, and with any tab selected, drag the command out of the toolbar (see Commands for more details).

Note: Toolbar settings defined in a particular view are, by default, valid for that view only. To make the settings apply to all views, click the check box at the bottom of the dialog

Keyboard

Keyboard

The Keyboard tab enables you to create new keyboard shortcuts for any command.



Assign a shortcut

To assign a shortcut to a command, do the following.

1. Select the *All Commands* category in the *Category* combo box.
2. In the *Commands* list box, select the command to which you wish to assign a new shortcut or select the command the shortcut of which you wish to change.
3. Click in the *Press New Shortcut Key* text box, and press the shortcut you wish to assign to that command. The shortcut appears in the *Press New Shortcut Key* text box. If the shortcut has not yet been assigned to any command, the **Assign** button is enabled. If the shortcut has already been assigned to a command, then that command is displayed below the text box and the **Assign** button is disabled. (To clear the *Press New Shortcut Key* text box, press any of the control keys, **Ctrl**, **Alt** or **Shift**).
4. Click the **Assign** button to assign the shortcut. The shortcut now appears in the *Current Keys* list box. You can assign multiple shortcuts to a single command.
5. Click the **Close** button to confirm.

Delete a shortcut

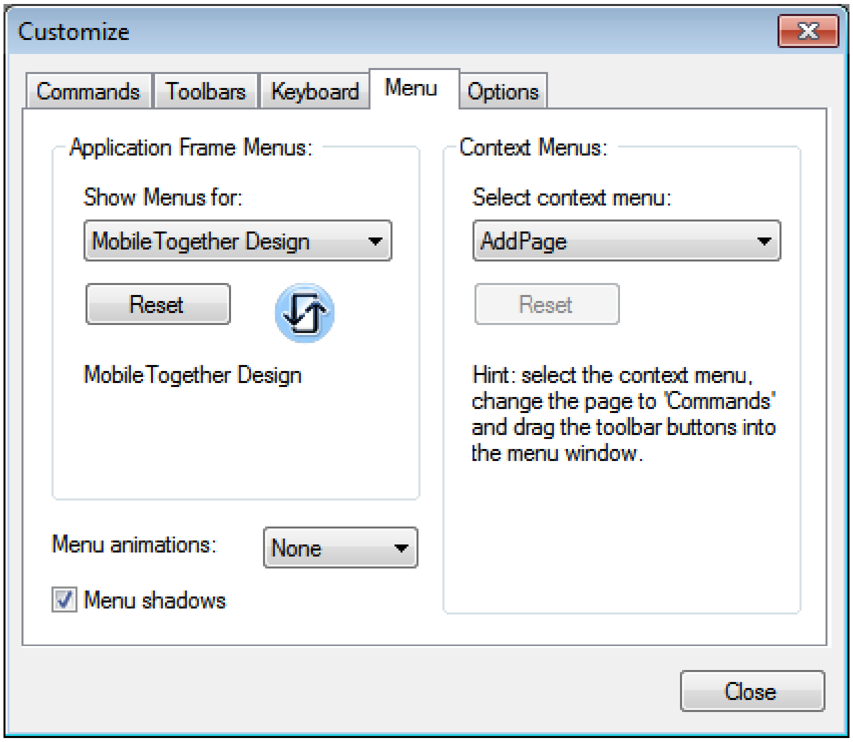
A shortcut cannot be assigned to multiple commands. If you wish to delete a shortcut, click it in the Current Keys list box and then click the **Remove** button. Click **Close**.

Set accelerator for
Currently no function is available.

Menu

Menu

The Menu tab allows you to customize the two main menu bars (default and application menu bars) as well as context menus.



MobileTogether Design

AddPage

MobileTogether Design

Hint: select the context menu, change the page to 'Commands' and drag the toolbar buttons into the menu window.

Menu animations: None

☑ Menu shadows

Customizing the default menu bar and application menu bar
The default menu bar is the menu bar that is displayed when no document is open in the main window. The application menu bar is the menu bar that is displayed when one or more documents are open in the main window. Each menu bar can be customized separately, and customization changes made to one do not affect the other. To customize a menu bar, select it in the *Show Menus For* combo box (*see screenshot above*). Then switch to the Commands tab of the Customize dialog and drag commands from the Commands list box to the menu bar or into any of the menus.

Deleting commands from menus and resetting the menu bars
To delete an entire menu or a command inside a menu, select that menu or menu command, and then either (i) right-click and select **Delete**, or (ii) drag away from the menu bar or menu, respectively. You can reset each of these two menu bars (default and application menu bars) to its original installation state by selecting the menu in the *Show Menus For* combo box and then clicking the **Reset** button below the combo box.

Customizing the application's context menus

Context menus are the menus that appear when you right-click certain objects in the application's interface. Each of these context menus can be customized by doing the following:

1. Select the context menu you want in the *Select Context Menu* combo box. This pops up the context menu.
2. Switching to the Commands tab of the Customize dialog.
3. Drag a command from the *Commands* list box into the context menu.
4. If you wish to delete a command from the context menu, right-click that command in the context menu, and click **Delete**. Alternatively, you can drag the command you want to delete out of the context menu.

You can reset any context menu to its original installation state by selecting it in the *Select Context Menu* combo box and then clicking the **Reset** button below the combo box.

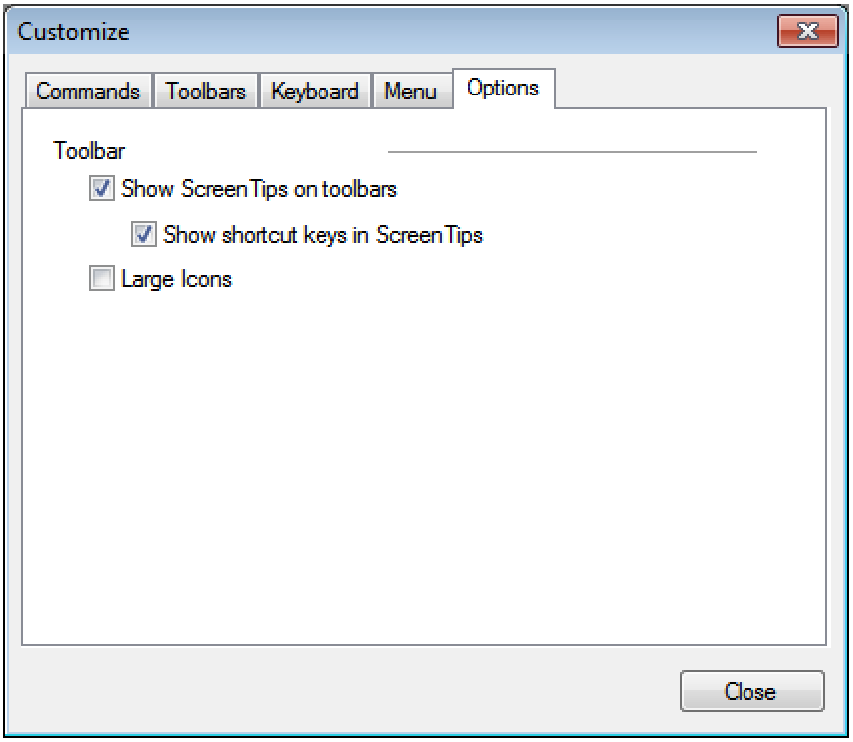
Menu shadows

Select the *Menu shadows* check box to give all menus shadows.

Options

Options

The Options tab allows you to define general environment settings.



Toolbar

☑ Show ScreenTips on toolbars

☑ Show shortcut keys in ScreenTips

□ Large Icons

Select the check boxes to toggle on the following options:

- *Show ScreenTips on toolbar:* Displays a popup when the mouse pointer is placed over an icon in any toolbar. The popup contains a short description of the icon function, as well as the associated keyboard shortcut, if one has been assigned and if the *Show shortcut keys* option has been checked .
- *Show shortcut keys in Screen Tips:* Defines whether shortcut information will be shown in screen tips.
- *Large icons:* Toggles the size of toolbar icons between standard and large.

Restore Toolbars and Windows

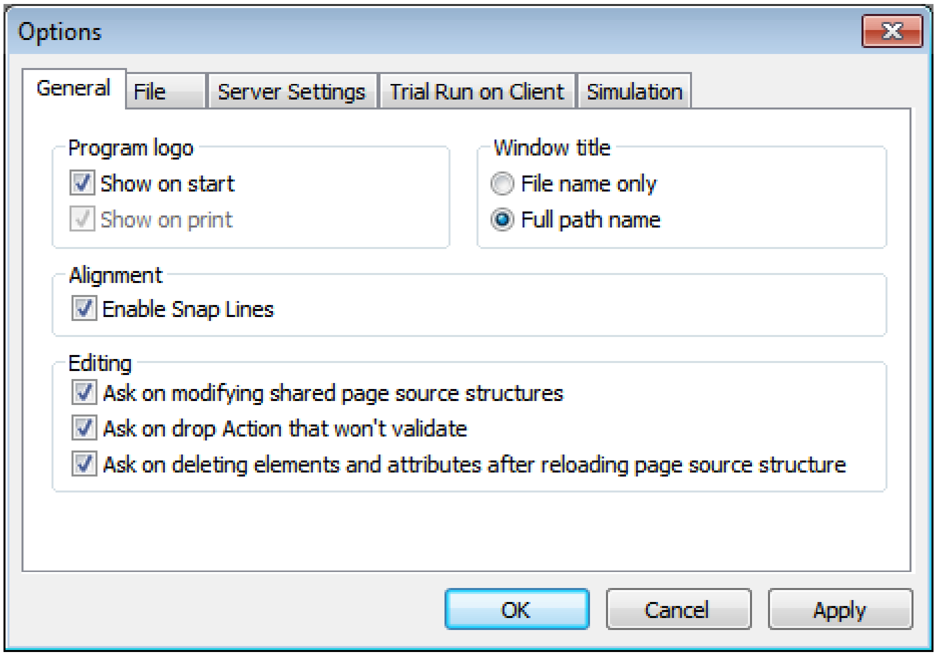
▼ Restore Toolbars and Windows

⊟ *Description*

Shuts down MobileTogether Designer and restarts it with all toolbars and windows reset to the original state in which they were at installation.

Options

The **Options** command displays the Options dialog (*screenshot below*). The settings available in the various tabs are described below.



Program logo
☑ Show on start
☑ Show on print

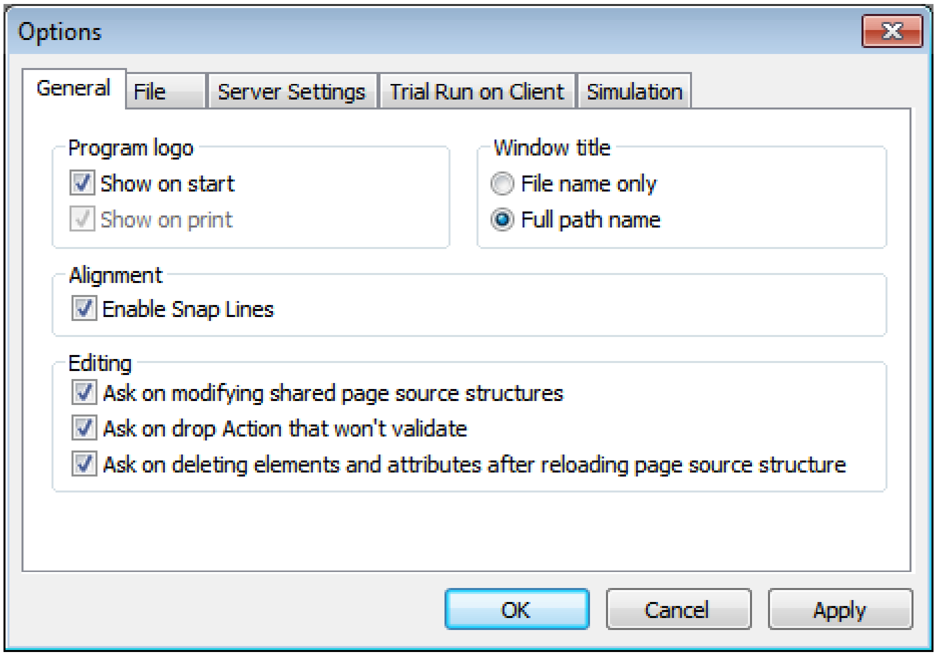
Window title
○ File name only
● Full path name

Alignment
☑ Enable Snap Lines

Editing
☑ Ask on modifying shared page source structures
☑ Ask on drop Action that won't validate
☑ Ask on deleting elements and attributes after reloading page source structure

General

In the General tab (*screenshot below*) you can define the settings shown in the screenshot.



Program logo

☑ Show on start

☑ Show on print

Window title

○ File name only

● Full path name

Alignment

☑ Enable Snap Lines

Editing

☑ Ask on modifying shared page source structures

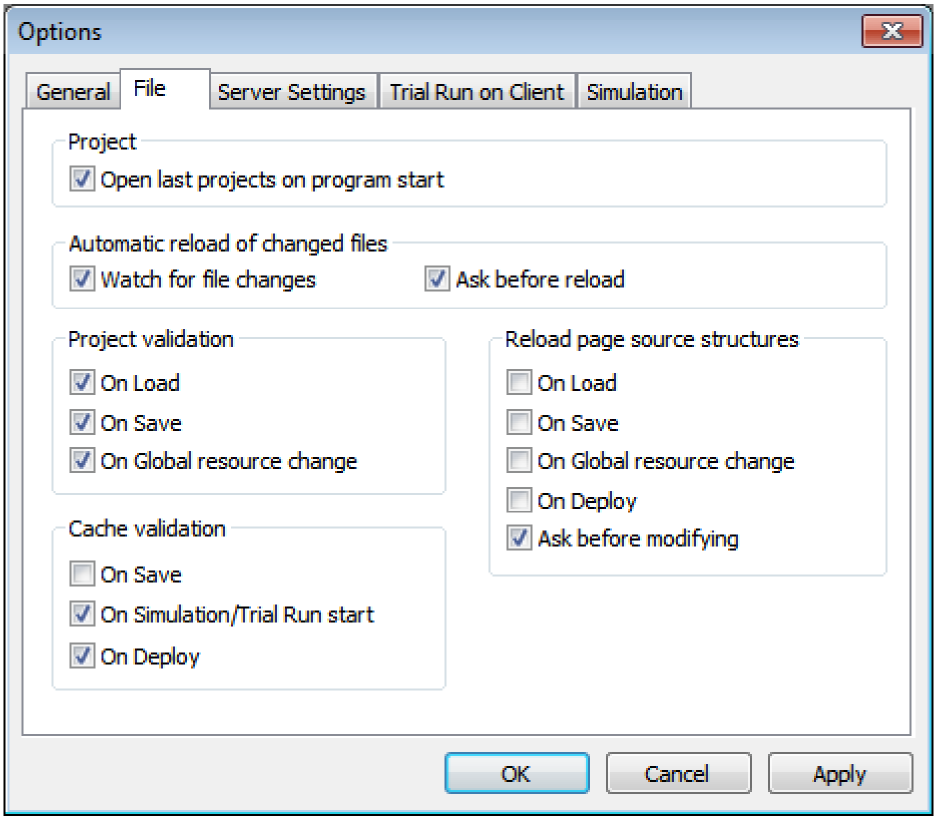
☑ Ask on drop Action that won't validate

☑ Ask on deleting elements and attributes after reloading page source structure

- *Program logo:* Can be shown at program start and in print output.
- *Window title:* The application window can display either the file name only, or the full file path and file name.
- *Alignment* | *Snap lines:* When snap lines are enabled, page controls that are dropped onto the design snap into place in a grid and relative to other components of the design. When snap lines are disabled, components can be placed freely in the design and are not constrained by the design grid.
- *Editing* | *Ask:* In situations where designer input is required, you are prompted about whether to go ahead with the action or not. For example, when a shared page resource is modified.

▾ File

In the File tab (*screenshot below*) you can define the settings shown in the screenshot.



Project

☑ Open last projects on program start

Automatic reload of changed files

☑ Watch for file changes ☑ Ask before reload

Project validation

☑ On Load
☑ On Save
☑ On Global resource change

Reload page source structures

□ On Load
□ On Save
□ On Global resource change
□ On Deploy
☑ Ask before modifying

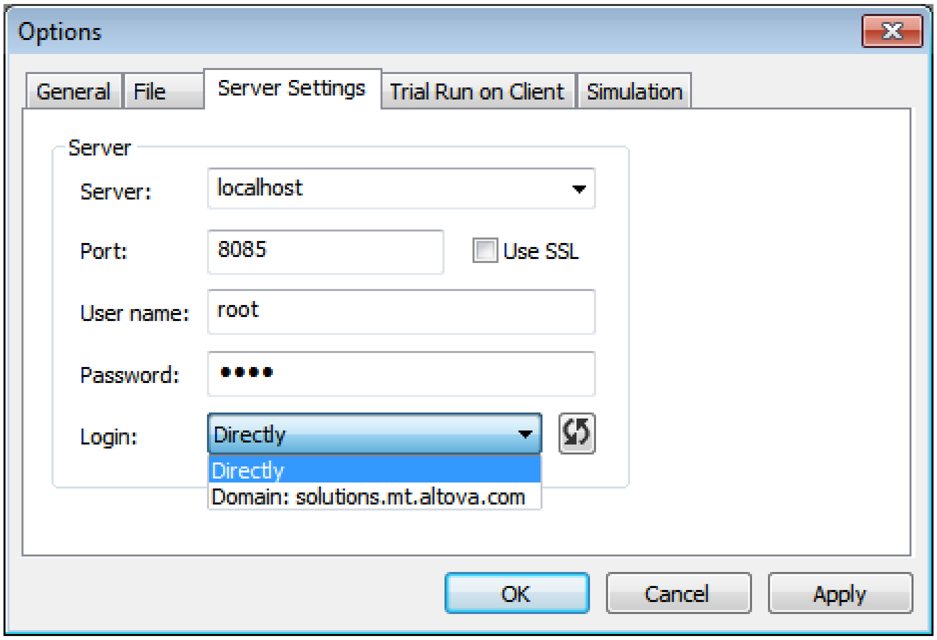
Cache validation

□ On Save
☑ On Simulation/Trial Run start
☑ On Deploy

- *Project:* If selected, then on program start, all the projects are opened that were open when the program was last closed.
- *Reload of changed files:* Options to watch for changes made by another user and to ask whether to reload or not. If a file is reloaded, your own changes from the time of the last save will be lost.
- *Project validation:* When to carry out project validation. Select the options you want.
- *Reload page source structures:* When to reload page source structures. Select the options you want.
- *Cache validation:* When the cache is validated. Select the options you want.

Server Settings

In the Server Settings tab (*screenshot below*) you can define the connection and authentication settings of the MobileTogether Server to which you wish to connect MobileTogether Designer. These settings will be used when solutions are deployed to the server and when the server is used for workflow simulation. The user must have the corresponding MobileTogether Server rights: *Save workflow from designer* and *Run server simulation*. The access rights of users of MobileTogether Server are defined in the Web UI of MobileTogether Server. See the MobileTogether Server user manual for information about how to do this.



Server: localhost

Port: 8085 □ Use SSL

User name: root

Password: ••••

Login: Directly

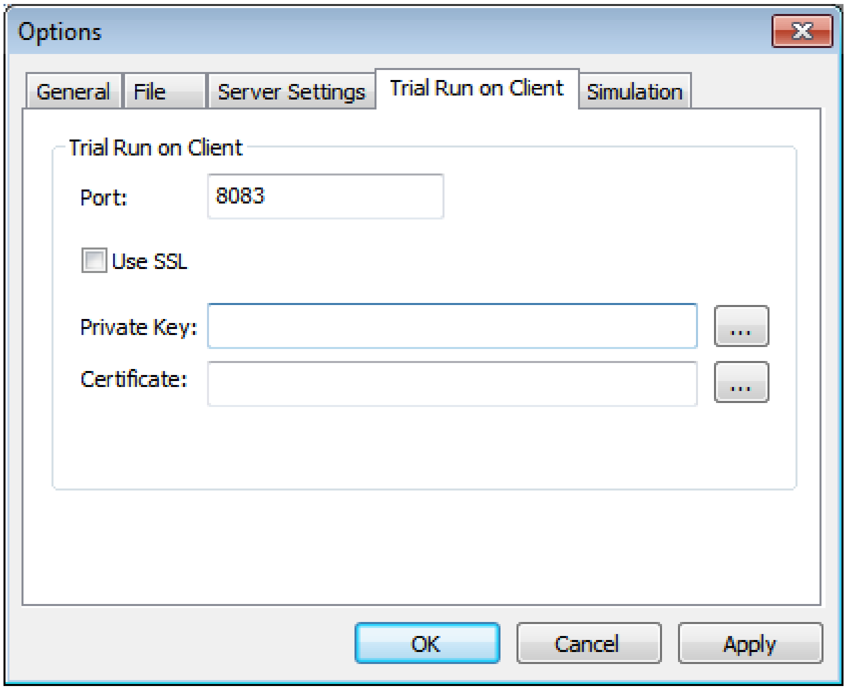
Directly
Domain: solutions.mt.altova.com

If the Active Directory login feature of MobileTogether Server has been enabled for you as a domain user, then the login information you enter for connecting MobileTogether Designer to MobileTogether Server can be your domain authentication info. For example, if your Windows user name and password on your office network domain has been enabled for use as MobileTogether Server authentication, then you can enter your domain-specific user name and password.

To select whether user credentials directly specified in MobileTogether Server or domain-specific user credentials are to be used, select the appropriate option from the *Login* combo box (*see screenshot above*). The button next to the combo box is for updating the connection with MobileTogether Server.

Trial Run on Client

In the Trial Run on Client tab (*screenshot below*) you can define the local port via which MobileTogether Designer connects with the client. For the Trial Run on Client feature, MobileTogether Designer itself acts as the MobileTogether Server and serves the design and related data files directly to the client.



Trial Run on Client

Port: 8083

□ Use SSL

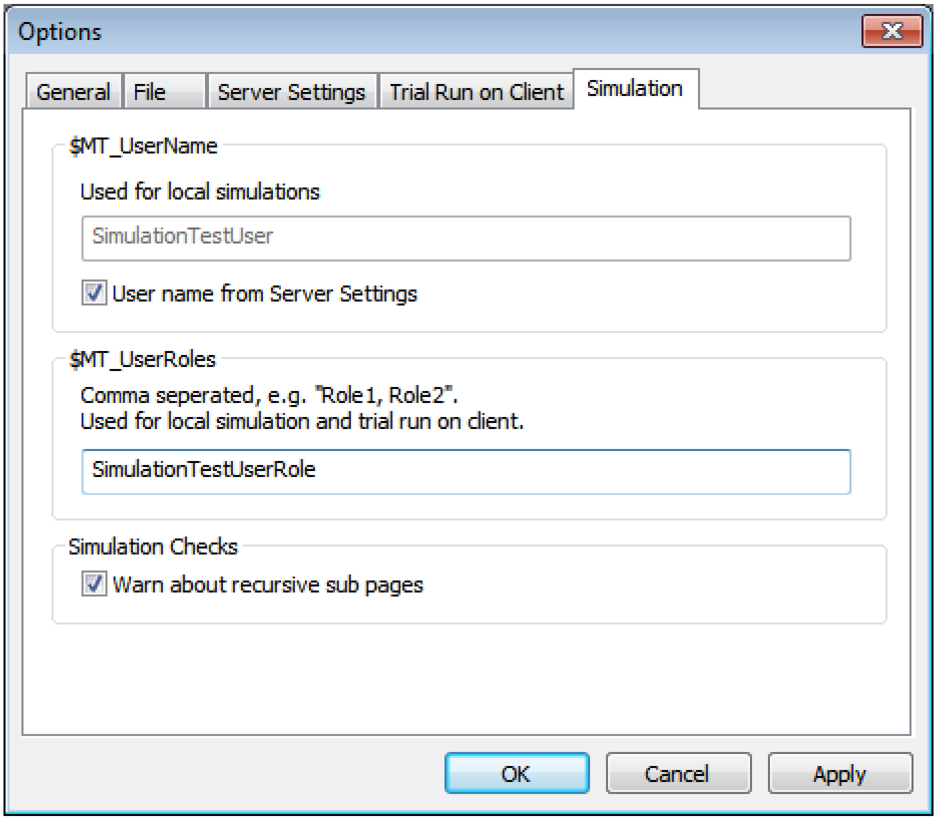
Private Key:

Certificate:

- *SSL:* Whether SSL is used.
- *Private key, certificate:* Browse for the SSL private key and certificate (if SSL is used).

Simulation

In the Simulation tab (*screenshot below*), you can specify the user name and roles to be used for simulations, and whether warnings should be issued about recursive sub pages.



$MT_UserName

Used for local simulations

SimulationTestUser

☑ User name from Server Settings

$MT_UserRoles

Comma seperated, e.g. "Role1, Role2".
Used for local simulation and trial run on client.

SimulationTestUserRole

Simulation Checks

☑ Warn about recursive sub pages

11.7 Window

The **Window** menu contains the following commands:

- Cascade
- Tile Horizontally
- Tile Vertically
- Currently Open Windows List

Cascade and Tile

The **Window** menu has commands to specify how MobileTogether Designer windows should be displayed in the GUI (cascaded, tiled, or maximized). To maximize a window, click the **Maximize** button of that window.

Currently Open Window List

All currently open document windows are listed at the bottom of the **Window** menu by document name, with the active window being checked. To make another window active, click the name of the window you wish to make active. This list shows all currently open windows, and lets you quickly switch between them. You can also use **CTRL+TAB** or **CTRL F6** keyboard shortcuts to cycle through the open windows.

Windows dialog
At the very bottom of the list of open windows is an entry for the **Windows** dialog. Clicking this entry opens the Windows dialog, which displays a list of all open windows and provides commands that can be applied to the selected window/s. (A window is selected by clicking on its name.)

Warning: To exit the Windows dialog, click **OK**. Do **not** click the **Close Window(s)** button. The **Close Window(s)** button closes the window/s currently selected in the Windows dialog.

11.8 Help

The **Help** menu contains commands required to get help or more information about MobileTogether Designer, as well as links to information and support pages on the Altova web server. The **Help** menu also contains the Registration dialog, which lets you enter your license key-code once you have purchased the product.

The **Help** menu contains the following commands:

- Table of Contents
- Index
- Search
- Software Activation
- Order Form
- Registration
- Check for Updates
- Support Center
- FAQ on the Web
- Download Components and Free Tools
- MobileTogether Designer on the Internet
- About MobileTogether Designer

Table of Contents, Index, Search

▼ Table of Contents

⊟ *Description*

Opens the onscreen help manual of MobileTogether Designer with the Table of Contents displayed in the left-hand-side pane of the Help window. The Table of Contents provides an overview of the entire Help document. Clicking an entry in the Table of Contents takes you to that topic.

▼ Index

⊟ *Description*

Opens the onscreen help manual of MobileTogether Designer with the Keyword Index displayed in the left-hand-side pane of the Help window. The index lists keywords and lets you navigate to a topic by double-clicking the keyword. If a keyword is linked to more than one topic, a list of these topics is displayed.

▼ Search

⊟ *Description*

Opens the onscreen help manual of MobileTogether Designer with the Search dialog displayed in the left-hand-side pane of the Help window. To search for a term, enter the term in the input field, and press **Return**. The Help system performs a full-text search on the entire Help documentation and returns a list of hits. Double-click any item to display that item.

Activation, Order Form, Registration, Updates

▼ Software Activation

⊟ *Description*

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

⊟ *Description*

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

⊟ *Description*

Opens the Altova Product Registration page in a tab of your browser. Registering your

Altova software will help ensure that you are always kept up to date with the latest product information.

Check for Updates

⊟ *Description*

Checks with the Altova server whether a newer version than yours is currently available and displays a message accordingly.

Other Commands

▾ Support Center

⊟ *Description*
A link to the Altova Support Center on the Internet. The Support Center provides FAQs, discussion forums where problems are discussed, and access to Altova's technical support staff.

▾ FAQ on the Web

⊟ *Description*
A link to Altova's FAQ database on the Internet. The FAQ database is constantly updated as Altova support staff encounter new issues raised by customers.

▾ Components Download

⊟ *Description*
A link to Altova's Component Download Center on the Internet. From here you can download a variety of companion software to use with Altova products. Such software ranges from XSLT and XSL-FO processors to Application Server Platforms. The software available at the Component Download Center is typically free of charge.

▾ MobileTogether Designer on the Internet

⊟ *Description*
A link to the Altova website on the Internet. You can learn more about MobileTogether Designer and related technologies and products at the Altova website.

▾ About MobileTogether Designer

⊟ *Description*
Displays the splash window and version number of your product.

Chapter 12

Appendices

12 Appendices

Information included in this section:

- XSLT and XQuery Engine Information
- XSLT and XPath/XQuery Functions
- License Information

12.1 XSLT and XPath/XQuery Functions

This section lists Altova extension functions and other extension functions that can be used in XPath and/or XQuery expressions. Altova extension functions can be used with Altova's XSLT and XQuery engines, and provide functionality additional to that available in the function libraries defined in the W3C standards.

General points
The following general points should be noted:

- Functions from the core function libraries defined in the W3C specifications can be called without a prefix. That's because the XSLT and XQuery engines read non-prefixed functions as belonging to a default functions namespace which is that specified in the XPath/XQuery functions specifications`http://www.w3.org/2005/xpath-functions`. If this namespace is explicitly declared in an XSLT or XQuery document, the prefix used in the namespace declaration can also optionally be used on function names.
- In general, if 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`.

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 `implicit-timezone()` function.

Collations
The default collation is the Unicode codepoint collation, which compares strings on the basis of their Unicode codepoint. Other supported collations are the ICU collations listed below. To use a specific collation, supply its URI as given in the list of supported collations (*table below*). Any string comparisons, including for the `max` and `min` functions, will be made according to the specified collation. If the collation option is not specified, the default Unicode-codepoint collation is used.

Language	URIs
`da`: Danish	`da_DK`
`de`: German	`de_AT, de_BE, de_CH, de_DE, de_LI, de_LU`

en: English	en_AS, en_AU, en_BB, en_BE, en_BM, en_BW, en_BZ, en_CA, en_GB, en_GU, en_HK, en_IE, en_IN, en_JM, en_MH, en_MP, en_MT, en_MU, en_NA, en_NZ, en_PH, en_PK, en_SG, en_TT, en_UM, en_US, en_VI, en_ZA, en_ZW
es: Spanish	es_419, es_AR, es_BO, es_CL, es_CO, es_CR, es_DO, es_EC, es_ES, es_GQ, es_GT, es_HN, es_MX, es_NI, es_PA, es_PE, es_PR, es_PY, es_SV, es_US, es_UY, es_VE
fr: French	fr_BE, fr_BF, fr_BI, fr_BJ, fr_BL, fr_CA, fr_CD, fr_CF, fr_CG, fr_CH, fr_CI, fr_CM, fr_DJ, fr_FR, fr_GA, fr_GN, fr_GP, fr_GQ, fr_KM, fr_LU, fr_MC, fr_MF, fr_MG, fr_ML, fr_MQ, fr_NE, fr_RE, fr_RW, fr_SN, fr_TD, fr_TG
it: Italian	it_CH, it_IT
ja: Japanese	ja_JP
nb: Norwegian Bokmal	nb_NO
nl: Dutch	nl_AW, nl_BE, nl_NL
nn: Nynorsk	nn_NO
pt: Portuguese	pt_AO, pt_BR, pt_GW, pt_MZ, pt_PT, pt_ST
ru: Russian	ru_MD, ru_RU, ru_UA
sv: Swedish	sv_FI, sv_SE

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 `in-scope-prefixes()`, `namespace-uri()` and `namespace-uri-for-prefix()` functions.

Altova Extension Functions

Altova extension functions can be used in XPath/XQuery expressions. They provide additional functionality to the functionality that is available in the standard library of XPath, XQuery, and XSLT functions.

- In order to distinguish Altova extension functions from functions in the standard library, Altova extension functions are named in this documentation with a suffix of **`[altova:]`**. For example **`add-years-to-date [altova:]`**.
- When using Altova extension functions in your XPath/XQuery expressions, however, you must use the function without any prefix or suffix, like this: **`add-years-to-date`**.

XPath functions (used in XPath expressions in XSLT):	XP1 XP2 XP3
XSLT functions (used in XPath expressions in XSLT):	XSLT1 XSLT2 XSLT3
XQuery functions (used in XQuery expressions in XQuery):	XQ1 XQ3

XSLT functions

XSLT functions can only be used in XPath expressions in an XSLT context (similarly to XSLT 2.0's `current-group()` or `key()` functions). These functions are not intended for, and will not work in, a non-XSLT context (for instance, in an XQuery context). Note that XSLT functions for XBRL can be used only with editions of Altova products that have XBRL support.

XPath/XQuery functions

XPath/XQuery functions (general, date/time, and string) can be used both in XPath expressions in XSLT contexts as well as in XQuery expressions.

XSLT Functions

XSLT extension functions can be used in XPath expressions in an XSLT context. They will not work in a non-XSLT context (for instance, in an XQuery context).

Note about naming of functions and language applicability
Altova extension functions can be used in XPath/XQuery expressions. They provide additional functionality to the functionality that is available in the standard library of XPath, XQuery, and XSLT functions.

- In order to distinguish Altova extension functions from functions in the standard library, Altova extension functions are named in this documentation with a suffix of **`[altova:]`**. For example **`add-years-to-date [altova:]`**.
- When using Altova extension functions in your XPath/XQuery expressions, however, you must use the function without any prefix or suffix, like this: **`add-years-to-date`**.

XPath functions (used in XPath expressions in XSLT):	XP1 XP2 XP3
XSLT functions (used in XPath expressions in XSLT):	XSLT1 XSLT2 XSLT3
XQuery functions (used in XQuery expressions in XQuery):	XQ1 XQ3

Standard functions

`distinct-nodes [altova:]`

`distinct-nodes(`*`node()*`***`)`** as **`node()*`** XSLT1 XSLT2 XSLT3

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`.

Examples

- **`distinct-nodes`**`(country)` returns all child `country` nodes less those having duplicate values.

`evaluate [altova:]`

`evaluate(XPathExpression` *`as xs:string`***`[, ValueOf$p1, ... ValueOf$pN])`** XSLT1 XSLT2 XSLT3

Takes an XPath expression, passed as a string, as its mandatory argument. It returns the output of the evaluated expression. For example: **`evaluate('//Name[1]')`** returns the contents of the first `Name` element in the document. Note that the expression `//Name[1]` is passed as a string by enclosing it in single quotes.

The `evaluate` function can optionally take additional arguments. These arguments are the values of in-scope variables that have the names `p1`, `p2`, `p3`... `pN`. Note the following points about usage: (i) The variables must be defined with names of the form `pX`, where `X` is an integer; (ii) the `evaluate` function's arguments (*see signature above*), from the second argument onwards, provide the values of the variables, with the sequence of the arguments corresponding to the numerically ordered sequence of variables: `p1` to `pN`: The second argument will be the value of the variable `p1`, the third argument that of the variable `p2`, and so on; (iii) The variable values must be of type `item*`.

⊟ *Example*

```
<xsl:variable name="xpath" select="'$p3, $p2, $p1'" />
<xsl:value-of select="evaluate($xpath, 10, 20, 'hi')" />
```

outputs `"hi 20 10"`

In the listing above, notice the following:

- The second argument of the `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.

⊟ *Examples to further illustrate the use of variables*

- `<xsl:variable name="xpath" select="'$p1'" />`
  `<xsl:value-of select="evaluate($xpath, //Name[1])" />`
  Outputs value of the first `Name` *element.*

- `<xsl:variable name="xpath" select="'$p1'" />`
  `<xsl:value-of select="evaluate($xpath, '//Name[1]')" />`
  Outputs `"//Name[1]"`

The `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="`**`evaluate(../UserReq/@sortkey)`**`" order="ascending"/>`. The `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 `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`).

Note: The static context includes namespaces, types, and functions—but not variables—from the calling environment. The base URI and default namespace are inherited.

⊟ *More examples*

- 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, $p1'" />`
  `<xsl:value-of select="evaluate($xpath, 10, 20, 30)" />`
  Outputs `"30 20 10"`

- Dynamic XPath expression with no dynamic variable:
  ```
  <xsl:variable name="xpath" select="'$p3, $p2, $p1'" />
  <xsl:value-of select="evaluate($xpath)" />
  ```
  Outputs error: No variable defined for $p3.

encode-for-rtf [altova:]

encode-for-rtf(input *as xs:string*, **preserveallwhitespace** *as xs:boolean*, **preservenewlines** *as xs:boolean*) as **xs:string** XSLT2 XSLT3

Converts the input string into code for RTF. Whitespace and new lines will be preserved according to the boolean value specified for their respective arguments.

[Top]

XBRL functions

Altova XBRL functions can be used only with editions of Altova products that have XBRL support.

xbrl-footnotes [altova:]

xbrl-footnotes(*node()***)** as **node()*** XSLT2 XSLT3

Takes a node as its input argument and returns the set of XBRL footnote nodes referenced by the input node.

xbrl-labels [altova:]

xbrl-labels(*xs:QName*, *xs:string***)** as **node()*** XSLT2 XSLT3

Takes two input arguments: a node name and the taxonomy file location containing the node. The function returns the XBRL label nodes associated with the input node.

[Top]

XPath/XQuery Functions: Date and Time

Altova's date/time extension functions can be used in XPath and XQuery expressions and provide additional functionality for the processing of data held as XML Schema's various date and time datatypes. The functions in this section can be used with Altova's **XPath 3.0** and **XQuery 3.0** engines. They are available in XPath/XQuery contexts.

Note about naming of functions and language applicability
Altova extension functions can be used in XPath/XQuery expressions. They provide additional functionality to the functionality that is available in the standard library of XPath, XQuery, and XSLT functions.

- In order to distinguish Altova extension functions from functions in the standard library, Altova extension functions are named in this documentation with a suffix of **`[altova:]`**. For example **`add-years-to-date [altova:]`**.
- When using Altova extension functions in your XPath/XQuery expressions, however, you must use the function without any prefix or suffix, like this: **`add-years-to-date`**.

XPath functions (used in XPath expressions in XSLT):	XP1 XP2 XP3
XSLT functions (used in XPath expressions in XSLT):	XSLT1 XSLT2 XSLT3
XQuery functions (used in XQuery expressions in XQuery):	XQ1 XQ3

▼ Grouped by functionality

- Add duration to xs:dateTime and return xs:dateTime
- Add a duration to xs:date and return xs:date
- Add a duration to xs:time and return xs:time
- Remove timezone from functions that generate current date/time
- Return weekday as integer from date
- Return week number as integer from date
- Build date, time, or duration type from lexical components of each type
- Construct date, dateTime, or time type from string input
- Age-related functions

▼ Grouped alphabetically

`altova:add-days-to-date`
`altova:add-days-to-dateTime`
`altova:add-hours-to-dateTime`
`altova:add-hours-to-time`
`altova:add-minutes-to-dateTime`
`altova:add-minutes-to-time`
`altova:add-months-to-date`
`altova:add-months-to-dateTime`
`altova:add-seconds-to-dateTime`
`altova:add-seconds-to-time`
`altova:add-years-to-date`
`altova:add-years-to-dateTime`
`altova:age`
`altova:age-details`
`altova:build-date`
`altova:build-duration`
`altova:build-time`
`altova:current-dateTime-no-TZ`
`altova:current-date-no-TZ`

altova:current-time-no-TZ
altova:parse-date
altova:parse-dateTime
altova:parse-time
altova:weekday-from-date
altova:weekday-from-dateTime
altova:weeknumber-from-date
altova:weeknumber-from-dateTime

[Top]

Add a duration to `xs:dateTime` XP3 XQ3

These functions add a duration to **`xs:dateTime`** and return **`xs:dateTime`**. The `xs:dateTime` type has a format of `CCYY-MM-DDThh:mm:ss.sss`. This is a concatenation of the `xs:date` and `xs:time` formats separated by the letter `T`. A timezone suffix`+01:00 (for example)` is optional.

▼ add-years-to-dateTime [altova:]

`add-years-to-dateTime(DateTime` *`as xs:dateTime`*, **`Years`** *`as xs:integer`*`)` as **`xs:dateTime`** XP3 XQ3

Adds a duration in years to an `xs:dateTime` (*see examples below*). The second argument is the number of years to be added to the `xs:dateTime` supplied as the first argument. The result is of type `xs:dateTime`.

⊟ *Examples*

- **`add-years-to-dateTime`**`(xs:dateTime("2014-01-15T14:00:00"), 10)` returns `2024-01-15T14:00:00`
- **`add-years-to-dateTime`**`(xs:dateTime("2014-01-15T14:00:00"), -4)` returns `2010-01-15T14:00:00`

▼ add-months-to-dateTime [altova:]

`add-months-to-dateTime(DateTime` *`as xs:dateTime`*, **`Months`** *`as xs:integer`*`)` as **`xs:dateTime`** XP3 XQ3

Adds a duration in months to an `xs:dateTime` (*see examples below*). The second argument is the number of months to be added to the `xs:dateTime` supplied as the first argument. The result is of type `xs:dateTime`.

⊟ *Examples*

- **`add-months-to-dateTime`**`(xs:dateTime("2014-01-15T14:00:00"), 10)` returns `2014-11-15T14:00:00`
- **`add-months-to-dateTime`**`(xs:dateTime("2014-01-15T14:00:00"), -2)` returns `2013-11-15T14:00:00`

▼ add-days-to-dateTime [altova:]

`add-days-to-dateTime(DateTime` *`as xs:dateTime`*, **`Days`** *`as xs:integer`*`)` as **`xs:dateTime`** XP3 XQ3

Adds a duration in days to an `xs:dateTime` (*see examples below*). The second argument is the number of days to be added to the `xs:dateTime` supplied as the first argument. The result is of type `xs:dateTime`.

⊟ *Examples*

- `add-days-to-dateTime(xs:dateTime("2014-01-15T14:00:00"), 10)` returns `2014-01-25T14:00:00`
- `add-days-to-dateTime(xs:dateTime("2014-01-15T14:00:00"), -8)` returns `2014-01-07T14:00:00`

add-hours-to-dateTime [altova:]

`add-hours-to-dateTime(DateTime as xs:dateTime, Hours as xs:integer)` as `xs:dateTime` XP3 XQ3

Adds a duration in hours to an `xs:dateTime` (*see examples below*). The second argument is the number of hours to be added to the `xs:dateTime` supplied as the first argument. The result is of type `xs:dateTime`.

⊟ *Examples*

- `add-hours-to-dateTime(xs:dateTime("2014-01-15T13:00:00"), 10)` returns `2014-01-15T23:00:00`
- `add-hours-to-dateTime(xs:dateTime("2014-01-15T13:00:00"), -8)` returns `2014-01-15T05:00:00`

add-minutes-to-dateTime [altova:]

`add-minutes-to-dateTime(DateTime as xs:dateTime, Minutes as xs:integer)` as `xs:dateTime` XP3 XQ3

Adds a duration in minutes to an `xs:dateTime` (*see examples below*). The second argument is the number of minutes to be added to the `xs:dateTime` supplied as the first argument. The result is of type `xs:dateTime`.

⊟ *Examples*

- `add-minutes-to-dateTime(xs:dateTime("2014-01-15T14:10:00"), 45)` returns `2014-01-15T14:55:00`
- `add-minutes-to-dateTime(xs:dateTime("2014-01-15T14:10:00"), -5)` returns `2014-01-15T14:05:00`

add-seconds-to-dateTime [altova:]

`add-seconds-to-dateTime(DateTime as xs:dateTime, Seconds as xs:integer)` as `xs:dateTime` XP3 XQ3

Adds a duration in seconds to an `xs:dateTime` (*see examples below*). The second argument is the number of seconds to be added to the `xs:dateTime` supplied as the first argument. The result is of type `xs:dateTime`.

⊟ *Examples*

- `add-seconds-to-dateTime(xs:dateTime("2014-01-15T14:00:10"), 20)` returns `2014-01-15T14:00:30`
- `add-seconds-to-dateTime(xs:dateTime("2014-01-15T14:00:10"), -5)` returns `2014-01-15T14:00:05`

[Top]

Add a duration to `xs:date` XP3 XQ3

These functions add a duration to **`xs:date`** and return **`xs:date`**. The `xs:date` type has a format of `CCYY-MM-DD`.

`add-years-to-date [altova:]`

`add-years-to-date(Date` *`as xs:date`*, **`Years`** *`as xs:integer`***`)`** as **`xs:date`** **XP3** **XQ3**

Adds a duration in years to a date. The second argument is the number of years to be added to the `xs:date` supplied as the first argument. The result is of type `xs:date`.

⊟ *Examples*

- **`add-years-to-date`**`(xs:date("2014-01-15"), 10)` returns `2024-01-15`
- **`add-years-to-date`**`(xs:date("2014-01-15"), -4)` returns `2010-01-15`

`add-months-to-date [altova:]`

`add-months-to-date(Date` *`as xs:date`*, **`Months`** *`as xs:integer`***`)`** as **`xs:date`** **XP3** **XQ3**

Adds a duration in months to a date. The second argument is the number of months to be added to the `xs:date` supplied as the first argument. The result is of type `xs:date`.

⊟ *Examples*

- **`add-months-to-date`**`(xs:date("2014-01-15"), 10)` returns `2014-11-15`
- **`add-months-to-date`**`(xs:date("2014-01-15"), -2)` returns `2013-11-15`

`add-days-to-date [altova:]`

`add-days-to-date(Date` *`as xs:date`*, **`Days`** *`as xs:integer`***`)`** as **`xs:date`** **XP3** **XQ3**

Adds a duration in days to a date. The second argument is the number of days to be added to the `xs:date` supplied as the first argument. The result is of type `xs:date`.

⊟ *Examples*

- **`add-days-to-date`**`(xs:date("2014-01-15"), 10)` returns `2014-01-25`
- **`add-days-to-date`**`(xs:date("2014-01-15"), -8)` returns `2014-01-07`

[Top]

Add a duration to `xs:time` **XP3** **XQ3**

These functions add a duration to **`xs:time`** and return **`xs:time`**. The `xs:time` type has a lexical form of `hh:mm:ss.sss`. An optional time zone may be suffixed. The letter `Z` indicates Coordinated Universal Time (UTC). All other time zones are represented by their difference from UTC in the format `+hh:mm`, or `-hh:mm`. If no time zone value is present, it is considered unknown; it is not assumed to be UTC.

`add-hours-to-time [altova:]`

`add-hours-to-time(Time` *`as xs:time`*, **`Hours`** *`as xs:integer`***`)`** as **`xs:time`** **XP3** **XQ3**

Adds a duration in hours to a time. The second argument is the number of hours to be added to the `xs:time` supplied as the first argument. The result is of type `xs:time`.

⊟ *Examples*

- **`add-hours-to-time`**`(xs:time("11:00:00"), 10)` returns `21:00:00`
- **`add-hours-to-time`**`(xs:time("11:00:00"), -7)` returns `04:00:00`

add-minutes-to-time [altova:]

add-minutes-to-time(Time *as xs:time*, **Minutes** *as xs:integer***)** as **xs:time** XP3 XQ3

Adds a duration in minutes to a time. The second argument is the number of minutes to be added to the `xs:time` supplied as the first argument. The result is of type `xs:time`.

Examples

- `add-minutes-to-time(xs:time("14:10:00"), 45)` returns `14:55:00`
- `add-minutes-to-time(xs:time("14:10:00"), -5)` returns `14:05:00`

add-seconds-to-time [altova:]

add-seconds-to-time(Time *as xs:time*, **Minutes** *as xs:integer***)** as **xs:time** XP3 XQ3

Adds a duration in seconds to a time. The second argument is the number of seconds to be added to the `xs:time` supplied as the first argument. The result is of type `xs:time`. The Seconds component can be in the range of `0` to `59.999`.

Examples

- `add-seconds-to-time(xs:time("14:00:00"), 20)` returns `14:00:20`
- `add-seconds-to-time(xs:time("14:00:00"), 20.895)` returns `14:00:20.895`

[Top]

Remove the timezone part from date/time datatypes XP3 XQ3

These functions remove the timezone from the current **`xs:dateTime`**, **`xs:date`**, or **`xs:time`** values, respectively. Note that the difference between `xs:dateTime` and `xs:dateTimeStamp` is that in the case of the latter the timezone part is required (while it is optional in the case of the former). So the format of an `xs:dateTimeStamp` value is: `CCYY-MM-DDThh:mm:ss.sss±hh:mm`. or `CCYY-MM-DDThh:mm:ss.sssZ`. If the date and time is read from the system clock as `xs:dateTimeStamp`, the `current-dateTime-no-TZ()` function can be used to remove the timezone if so required.

current-dateTime-no-TZ [altova:]

current-dateTime-no-TZ() as **xs:dateTime** XP3 XQ3

This function takes no argument. It removes the timezone part of `current-dateTime()` (which is the current date-and-time according to the system clock) and returns an `xs:dateTime` value.

Examples

If the current dateTime is **`2014-01-15T14:00:00+01:00`**:

- `current-dateTime-no-TZ()` returns `2014-01-15T14:00:00`

current-date-no-TZ [altova:]

current-date-no-TZ() as **xs:date** XP3 XQ3

This function takes no argument. It removes the timezone part of `current-date()` (which is the current date according to the system clock) and returns an `xs:date` value.

Examples

If the current date is **`2014-01-15+01:00`**:

- `current-date-no-TZ()` returns `2014-01-15`

▼ **`current-time-no-TZ [altova:]`**

`current-time-no-TZ()` as **`xs:time`** XP3 XQ3

This function takes no argument. It removes the timezone part of `current-time()` (which is the current time according to the system clock) and returns an `xs:time` value.

⊟ *Examples*

If the current time is **`14:00:00+01:00`**:

- `current-time-no-TZ()` returns `14:00:00`

[Top]

Return the weekday from `xs:dateTime` or `xs:date` XP3 XQ3

These functions return the weekday (as an integer) from `xs:dateTime` or `xs:date`. The days of the week are numbered (using the American format) from `1` to `7`, with `Sunday=1`. In the European format, the week starts with Monday (=`1`). The American format, where `Sunday=1`, can be set by using the integer `0` where an integer is accepted to indicate the format.

▼ **`weekday-from-dateTime [altova:]`**

`weekday-from-dateTime(DateTime` *`as xs:dateTime`***`)`** as **`xs:integer`** XP3 XQ3

Takes a date-with-time as its single argument and returns the day of the week of this date as an integer. The weekdays are numbered starting with `Sunday=1`. If the European format is required (where `Monday=1`), use the other signature of this function (*see next signature below*).

⊟ *Examples*

- `weekday-from-dateTime(xs:dateTime("2014-02-03T09:00:00"))` returns `2`, which would indicate a Monday.

`weekday-from-dateTime(DateTime` *`as xs:dateTime`***`, Format`** *`as xs:integer`***`)`** as **`xs:integer`** XP3 XQ3

Takes a date-with-time as its first argument and returns the day of the week of this date as an integer. The weekdays are numbered starting with `Monday=1`. If the second (integer) argument is `0`, then the weekdays are numbered `1` to `7` starting with `Sunday=1`. If the second argument is an integer other than `0`, then `Monday=1`. If there is no second argument, the function is read as having the other signature of this function (*see previous signature*).

⊟ *Examples*

- `weekday-from-dateTime(xs:dateTime("2014-02-03T09:00:00"), 1)` returns `1`, which would indicate a Monday
- `weekday-from-dateTime(xs:dateTime("2014-02-03T09:00:00"), 4)` returns `1`, which would indicate a Monday
- `weekday-from-dateTime(xs:dateTime("2014-02-03T09:00:00"), 0)` returns `2`, which would indicate a Monday.

▼ **`weekday-from-date [altova:]`**

`weekday-from-date(Date` *`as xs:date`***`)`** as **`xs:integer`** XP3 XQ3

Takes a date as its single argument and returns the day of the week of this date as an integer. The weekdays are numbered starting with `Sunday=1`. If the European format is required (where `Monday=1`), use the other signature of this function (*see next signature below*).

⊟ *Examples*

- `weekday-from-date(xs:date("2014-02-03+01:00"))` returns `2`, which would indicate a Monday.

`weekday-from-date(Date` *`as xs:date`*, **`Format`** *`as xs:integer`*) as **`xs:integer`** **XP3 XQ3**
Takes a date as its first argument and returns the day of the week of this date as an integer. The weekdays are numbered starting with `Monday=1`. If the second (`Format`) argument is `0`, then the weekdays are numbered `1` to `7` starting with `Sunday=1`. If the second argument is an integer other than `0`, then `Monday=1`. If there is no second argument, the function is read as having the other signature of this function (*see previous signature*).

⊟ *Examples*

- `weekday-from-date(xs:date("2014-02-03"), 1)` returns `1`, which would indicate a Monday
- `weekday-from-date(xs:date("2014-02-03"), 4)` returns `1`, which would indicate a Monday
- `weekday-from-date(xs:date("2014-02-03"), 0)` returns `2`, which would indicate a Monday.

[Top]

Return the week number from `xs:dateTime` or `xs:date` XP2 XQ1 XP3 XQ3

These functions return the week number (as an integer) from `xs:dateTime` or `xs:date`. Week-numbering is available in the US, ISO/European, and Islamic calendar formats. Week-numbering is different in these calendar formats because the week is considered to start on different days (on Sunday in the US format, Monday in the ISO/European format, and Saturday in the Islamic format).

▼ **`weeknumber-from-date [altova:]`**

`weeknumber-from-date(Date` *`as xs:date`*, **`Calendar`** *`as xs:integer`*) as **`xs:integer`**
XP2 XQ1 XP3 XQ3
Returns the week number of the submitted **`Date`** argument as an integer. The second argument (**`Calendar`**) specifies the calendar system to follow.
Supported **`Calendar`** values are:

- **`0 = US calendar`** (*week starts Sunday*)
- **`1 = ISO standard, European calendar`** (*week starts Monday*)
- **`2 = Islamic calendar`** (*week starts Saturday*)

Default is **`0`**.

⊟ *Examples*

- `weeknumber-from-date(xs:date("2014-03-23"), 0)` returns `13`
- `weeknumber-from-date(xs:date("2014-03-23"), 1)` returns `12`
- `weeknumber-from-date(xs:date("2014-03-23"), 2)` returns `13`
- `weeknumber-from-date(xs:date("2014-03-23")   )` returns `13`

The day of the date in the examples above (`2014-03-23`) is Sunday. So the US and Islamic calendars are one week ahead of the European calendar on this day.

▼ **`weeknumber-from-dateTime [altova:]`**

`weeknumber-from-dateTime(DateTime` *`as xs:dateTime`*, **`Calendar`** *`as xs:integer`*) as **`xs:integer`** **XP2 XQ1 XP3 XQ3**

Returns the week number of the submitted **`DateTime`** argument as an integer. The second argument (**`Calendar`**) specifies the calendar system to follow.

Supported **`Calendar`** values are:

- **`0 = US calendar`** (*week starts Sunday*)
- **`1 = ISO standard, European calendar`** (*week starts Monday*)
- **`2 = Islamic calendar`** (*week starts Saturday*)

Default is **`0`**.

⊟ *Examples*

- **`weeknumber-from-dateTime`**`(xs:dateTime("2014-03-23T00:00:00"), 0)` returns `13`
- **`weeknumber-from-dateTime`**`(xs:dateTime("2014-03-23T00:00:00"), 1)` returns `12`
- **`weeknumber-from-dateTime`**`(xs:dateTime("2014-03-23T00:00:00"), 2)` returns `13`
- **`weeknumber-from-dateTime`**`(xs:dateTime("2014-03-23T00:00:00")    )` returns `13`

The day of the dateTime in the examples above (`2014-03-23T00:00:00`) is Sunday. So the US and Islamic calendars are one week ahead of the European calendar on this day.

[Top]

Build date, time, and duration datatypes from their lexical components **XP3 XQ3**

The functions take the lexical components of the `xs:date`, `xs:time`, or `xs:duration` datatype as input arguments and combine them to build the respective datatype.

▼ **`build-date [altova:]`**

`build-date(Year` *`as xs:integer`*, **`Month`** *`as xs:integer`*, **`Date`** *`as xs:integer`*) as **`xs:date`** **XP3 XQ3**

The first, second, and third arguments are, respectively, the year, month, and date. They are combined to build a value of `xs:date` type. The values of the integers must be within the correct range of that particular date part. For example, the second argument (for the month part) should not be greater than `12`.

⊟ *Examples*

- **`build-date`**`(2014, 2, 03)` returns `2014-02-03`

▼ **`build-time [altova:]`**

`build-time(Hours` *`as xs:integer`*, **`Minutes`** *`as xs:integer`*, **`Seconds`** *`as xs:integer`*) as **`xs:time`** **XP3 XQ3**

The first, second, and third arguments are, respectively, the hour (`0` to `23`), minutes (`0` to `59`), and seconds (0 to 59) values. They are combined to build a value of xs:time type. The values of the integers must be within the correct range of that particular time part. For example, the second (Minutes) argument should not be greater than 59. To add a timezone part to the value, use the other signature of this function (*see next signature*).

⊟ *Examples*

- **build-time**(23, 4, 57) returns 23:04:57

build-time(Hours *as xs:integer*, **Minutes** *as xs:integer*, **Seconds** *as xs:integer*, **TimeZone** *as xs:string*) as **xs:time** **XP3** **XQ3**

The first, second, and third arguments are, respectively, the hour (0 to 23), minutes (0 to 59), and seconds (0 to 59) values. The fourth argument is a string that provides the timezone part of the value. The four arguments are combined to build a value of xs:time type. The values of the integers must be within the correct range of that particular time part. For example, the second (Minutes) argument should not be greater than 59.

⊟ *Examples*

- **build-time**(23, 4, 57, '+1') returns 23:04:57+01:00

▼ build-duration [altova:]

build-duration(Years *as xs:integer*, **Months** *as xs:integer*) as **xs:yearMonthDuration** **XP3** **XQ3**

Takes two arguments to build a value of type xs:yearMonthDuration. The first arguments provides the Years part of the duration value, while the second argument provides the Months part. If the second (Months) argument is greater than or equal to 12, then the integer is divided by 12; the quotient is added to the first argument to provide the Years part of the duration value while the remainder (of the division) provides the Months part. To build a duration of type xs:dayTimeDuration., see the next signature.

⊟ *Examples*

- **build-duration**(2, 10) returns P2Y10M
- **build-duration**(14, 27) returns P16Y3M
- **build-duration**(2, 24) returns P4Y

build-duration(Days *as xs:integer*, **Hours** *as xs:integer*, **Minutes** *as xs:integer*, **Seconds** *as xs:integer*) as **xs:dayTimeDuration** **XP3** **XQ3**

Takes four arguments and combines them to build a value of type xs:dayTimeDuration. The first argument provides the Days part of the duration value, the second, third, and fourth arguments provide, respectively, the Hours, Minutes, and Seconds parts of the duration value. Each of the three Time arguments is converted to an equivalent value in terms of the next higher unit and the result is used for calculation of the total duration value. For example, 72 seconds is converted to 1M+12S (1 minute and 12 seconds), and this value is used for calculation of the total duration value. To build a duration of type xs:yearMonthDuration., see the previous signature.

⊟ *Examples*

- **build-duration**(2, 10, 3, 56) returns P2DT10H3M56S
- **build-duration**(1, 0, 100, 0) returns P1DT1H40M
- **build-duration**(1, 0, 0, 3600) returns P1DT1H

[Top]

Construct date, dateTime, and time datatypes from string input XP2 XQ1 XP3 XQ3

These functions take strings as arguments and construct `xs:date`, `xs:dateTime`, or `xs:time` datatypes. The string is analyzed for components of the datatype based on a submitted pattern argument.

▼ `parse-date [altova:]`

`parse-date(Date` *`as xs:string`*`, DatePattern` *`as xs:string`*`)` as **`xs:date`** XP2 XQ1 XP3 XQ3

Returns the input string **`Date`** as an **`xs:date`** value. The second argument **`DatePattern`** specifies the pattern (sequence of components) of the input string. **`DatePattern`** is described with the component specifiers listed below and with component separators that can be any character. See the examples below.

D	Date
M	Month
Y	Year

The pattern in **`DatePattern`** must match the pattern in **`Date`**. Since the output is of type **`xs:date`**, the output will always have the lexical format **`YYYY-MM-DD`**.

⊟ *Examples*

- `parse-date(xs:string("06-03-2014"), "[D]-[M]-[Y]")` returns `2014-03-06`
- `parse-date(xs:string("06-03-2014"), "[M]-[D]-[Y]")` returns `2014-06-03`
- `parse-date("06/03/2014", "[M]/[D]/[Y]")` returns `2014-06-03`
- `parse-date("06 03 2014", "[M] [D] [Y]")` returns `2014-06-03`
- `parse-date("6 3 2014", "[M] [D] [Y]")` returns `2014-06-03`

▼ `parse-dateTime [altova:]`

`parse-dateTime(DateTime` *`as xs:string`*`, DateTimePattern` *`as xs:string`*`)` as **`xs:dateTime`** XP2 XQ1 XP3 XQ3

Returns the input string **`DateTime`** as an **`xs:dateTime`** value.The second argument **`DateTimePattern`** specifies the pattern (sequence of components) of the input string. **`DateTimePattern`** is described with the component specifiers listed below and with component separators that can be any character. See the examples below.

D	Date
M	Month
Y	Year
H	Hour
m	minutes
s	seconds

The pattern in **`DateTimePattern`** must match the pattern in **`DateTime`**. Since the output is of type **`xs:dateTime`**, the output will always have the lexical format **`YYYY-MM-DDTHH:mm:ss`**.

⊟ *Examples*

- `parse-dateTime(xs:string("06-03-2014 13:56:24"), "[D]-[M]-[Y] [H]:[m]:[s]")` returns `2014-03-06T13:56:24`

- **parse-dateTime**("time=13:56:24; date=06-03-2014", "time=[H]:[m]:[s]; date=[D]-[M]-[Y]") returns 2014-03-06T13:56:24

▼ parse-time [altova:]

parse-time(Time *as xs:string*, **TimePattern** *as xs:string***)** as **xs:time** XP2 XQ1 XP3 XQ3

Returns the input string **Time** as an **xs:time** value.The second argument **TimePattern** specifies the pattern (sequence of components) of the input string. **TimePattern** is described with the component specifiers listed below and with component separators that can be any character. See the examples below.

H	Hour
m	minutes
s	seconds

The pattern in **TimePattern** must match the pattern in **Time**. Since the output is of type **xs:time**, the output will always have the lexical format **HH:mm:ss**.

⊟ *Examples*

- **parse-time**(xs:string("13:56:24"), "[H]:[m]:[s]") returns 13:56:24
- **parse-time**("13-56-24", "[H]-[m]") returns 13:56:00
- **parse-time**("time=13h56m24s", "time=[H]h[m]m[s]s") returns 13:56:24
- **parse-time**("time=24s56m13h", "time=[s]s[m]m[H]h") returns 13:56:24

[Top]

Age-related functions XP3 XQ3

These functions return the age as calculated (i) between one input argument date and the current date, or (ii) between two input argument dates. The **altova:age** function returns the age in terms of years, the **altova:age-details** function returns the age as a sequence of three integers giving the years, months, and days of the age.

▼ age [altova:]

age(StartDate *as xs:date***)** as **xs:integer** XP3 XQ3

Returns an integer that is the age *in years* of some object, counting from a start-date submitted as the argument and ending with the current date (taken from the system clock). If the input argument is a date anything greater than or equal to one year in the future, the return value will be negative.

⊟ *Examples*

If the current date is **2014-01-15**:

- **age**(xs:date("2013-01-15")) returns 1
- **age**(xs:date("2013-01-16")) returns 0
- **age**(xs:date("2015-01-15")) returns -1
- **age**(xs:date("2015-01-14")) returns 0

age(StartDate *as xs:date*, **EndDate** *as xs:date***)** as **xs:integer** XP3 XQ3

Returns an integer that is the age *in years* of some object, counting from a start-date that is submitted as the first argument up to an end-date that is the second argument. The return value will be negative if the first argument is one year or more later than the second argument.

⊟ *Examples*

If the current date is **2014-01-15**:

- **age**(xs:date("2000-01-15"), xs:date("2010-01-15")) returns 10
- **age**(xs:date("2000-01-15"), current-date()) returns 14 if the current date is 2014-01-15
- **age**(xs:date("2014-01-15"), xs:date("2010-01-15")) returns -4

age-details [altova:]

age-details(InputDate *as xs:date*) as (**xs:integer**)* XP3 XQ3

Returns three integers that are, respectively, the years, months, and days between the date that is submitted as the argument and the current date (taken from the system clock). The sum of the returned `years+months+days` together gives the total time difference between the two dates (the input date and the current date). The input date may have a value earlier or later than the current date, but whether the input date is earlier or later is not indicated by the sign of the return values; the return values are always positive.

⊟ *Examples*

If the current date is **2014-01-15**:

- **age-details**(xs:date("2014-01-16")) returns (0 0 1)
- **age-details**(xs:date("2014-01-14")) returns (0 0 1)
- **age-details**(xs:date("2013-01-16")) returns (1 0 1)
- **age-details**(current-date()) returns (0 0 0)

age-details(Date-1 *as xs:date*, **Date-2** *as xs:date*) as (**xs:integer**)* XP3 XQ3

Returns three integers that are, respectively, the years, months, and days between the two argument dates. The sum of the returned `years+months+days` together gives the total time difference between the two input dates; it does not matter whether the earlier or later of the two dates is submitted as the first argument. The return values do not indicate whether the input date occurs earlier or later than the current date. Return values are always positive.

⊟ *Examples*

- **age-details**(xs:date("2014-01-16"), xs:date("2014-01-15")) returns (0 0 1)
- **age-details**(xs:date("2014-01-15"), xs:date("2014-01-16")) returns (0 0 1)

[Top]

XPath/XQuery Functions: String

The following general-purpose XPath/XQuery extension functions are supported in the current version of your Altova product and can be used in (i) XPath expressions in an XSLT context, or (ii) XQuery expressions in an XQuery document.

Note about naming of functions and language applicability
Altova extension functions can be used in XPath/XQuery expressions. They provide additional functionality to the functionality that is available in the standard library of XPath, XQuery, and XSLT functions.

- In order to distinguish Altova extension functions from functions in the standard library, Altova extension functions are named in this documentation with a suffix of **`[altova:]`**. For example **`add-years-to-date [altova:]`**.
- When using Altova extension functions in your XPath/XQuery expressions, however, you must use the function without any prefix or suffix, like this: **`add-years-to-date`**.

XPath functions (used in XPath expressions in XSLT):	XP1 XP2 XP3
XSLT functions (used in XPath expressions in XSLT):	XSLT1 XSLT2 XSLT3
XQuery functions (used in XQuery expressions in XQuery):	XQ1 XQ3

camel-case [altova:]

`camel-case(InputString` *`as xs:string`***`)`** as **`xs:string`** XP3 XQ3

Returns the input string **`InputString`** in CamelCase. The string is analyzed using the regular expression **`'\s'`** (which is a shortcut for the whitespace character). The first non-whitespace character after a whitespace or sequence of consecutive whitespaces is capitalized. The first character in the output string is capitalized.

⊟ *Examples*

- **`camel-case`**`("max")` returns `Max`
- **`camel-case`**`("max max")` returns `Max Max`
- **`camel-case`**`("file01.xml")` returns `File01.xml`
- **`camel-case`**`("file01.xml file02.xml")` returns `File01.xml File02.xml`
- **`camel-case`**`("file01.xml   file02.xml")` returns `File01.xml   File02.xml`
- **`camel-case`**`("file01.xml -file02.xml")` returns `File01.xml -file02.xml`

`camel-case(InputString` *`as xs:string`***`, SplitChars`** *`as xs:string`***`, IsRegex`** *`as xs:boolean`***`)`** as **`xs:string`** XP3 XQ3

Converts the input string **`InputString`** to camel case by using **`SplitChars`** to determine the character/s that trigger the next capitalization. **`SplitChars`** is used as a regular expression when **`IsRegex = true()`**, or as plain characters when **`IsRegex = false()`**. The first character in the output string is capitalized.

⊟ *Examples*

- **`camel-case`**`("setname getname", "set|get", true())` returns `setName getName`
- **`camel-case`**`("altova\documents\testcases", "\", false())` returns `Altova\Documents\Testcases`

char [altova:]

`char(Position` *`as xs:integer`***`)`** as **`xs:string`** XP3 XQ3

Returns a string containing the character at the position specified by the `Position` argument, in the string obtained by converting the value of the context item to `xs:string`. The result string will be empty if no character exists at the index submitted by the `Position` argument.

⊟ *Examples*

If the context item is **`1234ABCD`**:

- `char(2)` returns `2`
- `char(5)` returns `A`
- `char(9)` returns the empty string.
- `char(-2)` returns the empty string.

`char(InputString` *`as xs:string`*`, Position` *`as xs:integer`*`)` as **`xs:string`** **XP3 XQ3**

Returns a string containing the character at the position specified by the `Position` argument, in the string submitted as the `InputString` argument. The result string will be empty if no character exists at the index submitted by the `Position` argument.

⊟ *Examples*

- `char("2014-01-15", 5)` returns `-`
- `char("USA", 1)` returns `U`
- `char("USA", 10)` returns the empty string.
- `char("USA", -2)` returns the empty string.

▼ first-chars [altova:]

`first-chars(X-Number` *`as xs:integer`*`)` as **`xs:string`** **XP3 XQ3**

Returns a string containing the first `X-Number` of characters of the string obtained by converting the value of the context item to `xs:string`.

⊟ *Examples*

If the context item is **`1234ABCD`**:

- `first-chars(2)` returns `12`
- `first-chars(5)` returns `1234A`
- `first-chars(9)` returns `1234ABCD`

`first-chars(InputString` *`as xs:string`*`, X-Number` *`as xs:integer`*`)` as **`xs:string`** **XP3 XQ3**

Returns a string containing the first `X-Number` of characters of the string submitted as the `InputString` argument.

⊟ *Examples*

- `first-chars("2014-01-15", 5)` returns `2014-`
- `first-chars("USA", 1)` returns `U`

▼ last-chars [altova:]

`last-chars(X-Number` *`as xs:integer`*`)` as **`xs:string`** **XP3 XQ3**

Returns a string containing the last `X-Number` of characters of the string obtained by converting the value of the context item to `xs:string`.

⊟ *Examples*

If the context item is **`1234ABCD`**:

- `last-chars(2)` returns `CD`
- `last-chars(5)` returns `4ABCD`
- `last-chars(9)` returns `1234ABCD`

last-chars(InputString *as xs:string*, **X-Number** *as xs:integer***)** as **xs:string**
XP3 XQ3

Returns a string containing the last `X-Number` of characters of the string submitted as the `InputString` argument.

⊟ *Examples*

- **last-chars**("2014-01-15", 5) returns 01-15
- **last-chars**("USA", 10) returns USA

pad-string-left [altova:]

pad-string-left(StringToPad *as xs:string*, **Repeats** *as xs:integer*, **PadCharacter** *as xs:string***)** as **xs:string** **XP3 XQ3**

The `PadCharacter` argument is a single character that is padded to the left of the string submitted as the `StringToPad` argument. The `Repeats` argument gives the number of times the pad-character is to be repeated at the left of `StringToPad`.

⊟ *Examples*

- **pad-string-left**('AP', 1, 'Z') returns 'ZAP'
- **pad-string-left**('AP', 3, 'Z') returns 'ZZZAP'
- **pad-string-left**('AP', 0, 'Z') returns 'AP'
- **pad-string-left**('AP', 3, 'YZ') returns a pad-character-too-long error

pad-string-right [altova:]

pad-string-right(StringToPad *as xs:string*, **Repeats** *as xs:integer*, **PadCharacter** *as xs:string***)** as **xs:string** **XP3 XQ3**

The `PadCharacter` argument is a single character that is padded to the right of the string submitted as the `StringToPad` argument. The `Repeats` argument gives the number of times the pad-character is to be repeated at the right of `StringToPad`.

⊟ *Examples*

- **pad-string-right**('AP', 1, 'Z') returns 'APZ'
- **pad-string-right**('AP', 3, 'Z') returns 'APZZZ'
- **pad-string-right**('AP', 0, 'Z') returns 'AP'
- **pad-string-right**('AP', 3, 'YZ') returns a pad-character-too-long error

repeat-string [altova:]

repeat-string(InputString *as xs:string*, **Repeats** *as xs:integer***)** as **xs:string**
XP2 XQ1 XP3 XQ3

Generates a string that is composed of the first `InputString` argument repeated `Repeats` number of times.

⊟ *Examples*

- **repeat-string**("Altova #", 3) returns "Altova #Altova #Altova #"

substring-after-last [altova:]

substring-after-last(MainString *as xs:string*, **CheckString** *as xs:string***)** as **xs:string** **XP3 XQ3**

If `CheckString` is found in `MainString`, then the substring that occurs after `CheckString` in

MainString is returned. If CheckString is not found in MainString, then the empty string is returned. If CheckString is an empty string, then MainString is returned in its entirety. If there is more than one occurrence of CheckString in MainString, then the substring after the last occurrence of CheckString is returned.

⊟ *Examples*

- **substring-after-last**('ABCDEFGH', 'B') returns 'CDEFGH'
- **substring-after-last**('ABCDEFGH', 'BC') returns 'DEFGH'
- **substring-after-last**('ABCDEFGH', 'BD') returns ''
- **substring-after-last**('ABCDEFGH', 'Z') returns ''
- **substring-after-last**('ABCDEFGH', '') returns 'ABCDEFGH'
- **substring-after-last**('ABCD-ABCD', 'B') returns 'CD'
- **substring-after-last**('ABCD-ABCD-ABCD', 'BCD') returns ''

▼ substring-before-last [altova:]

substring-before-last(MainString *as xs:string*, **CheckString** *as xs:string*) as **xs:string** **XP3** **XQ3**

If CheckString is found in MainString, then the substring that occurs before CheckString in MainString is returned. If CheckString is not found in MainString, or if CheckString is an empty string, then the empty string is returned. If there is more than one occurrence of CheckString in MainString, then the substring before the last occurrence of CheckString is returned.

⊟ *Examples*

- **substring-before-last**('ABCDEFGH', 'B') returns 'A'
- **substring-before-last**('ABCDEFGH', 'BC') returns 'A'
- **substring-before-last**('ABCDEFGH', 'BD') returns ''
- **substring-before-last**('ABCDEFGH', 'Z') returns ''
- **substring-before-last**('ABCDEFGH', '') returns ''
- **substring-before-last**('ABCD-ABCD', 'B') returns 'ABCD-A'
- **substring-before-last**('ABCD-ABCD-ABCD', 'ABCD') returns 'ABCD-ABCD-'

▼ substring-pos [altova:]

substring-pos(StringToCheck *as xs:string*, **StringToFind** *as xs:string*) as **xs:integer** **XP3** **XQ3**

Returns the character position of the first occurrence of StringToFind in the string StringToCheck. The character position is returned as an integer. The first character of StringToCheck has the position 1. If StringToFind does not occur within StringToCheck, the integer 0 is returned. To check for the second or a later occurrence of StringToCheck, use the next signature of this function.

⊟ *Examples*

- **substring-pos**('Altova', 'to') returns 3
- **substring-pos**('Altova', 'tov') returns 3
- **substring-pos**('Altova', 'tv') returns 0
- **substring-pos**('AltovaAltova', 'to') returns 3

substring-pos(StringToCheck *as xs:string*, **StringToFind** *as xs:string*, **Integer** *as xs:integer*) as **xs:integer** **XP3** **XQ3**

Returns the character position of StringToFind in the string, StringToCheck. The search for StringToFind starts from the character position given by the Integer argument; the character substring before this position is not searched. The returned integer, however, is the position of the found string within the *entire* string, `StringToCheck`. This signature is useful for finding the second or a later position of a string that occurs multiple times with the `StringToCheck`. If `StringToFind` does not occur within `StringToCheck`, the integer `0` is returned.

⊟ *Examples*

- `substring-pos('Altova', 'to', 1)` returns `3`
- `substring-pos('Altova', 'to', 3)` returns `3`
- `substring-pos('Altova', 'to', 4)` returns `0`
- `substring-pos('Altova-Altova', 'to', 0)` returns `3`
- `substring-pos('Altova-Altova', 'to', 4)` returns `10`

trim-string [altova:]

`trim-string(InputString` *`as xs:string`*`)` as **`xs:string`** **XP3** **XQ3**

This function takes an `xs:string` argument, removes any leading and trailing whitespace, and returns a "trimmed" `xs:string`.

⊟ *Examples*

- `trim-string("   Hello World   "))` returns `"Hello World"`
- `trim-string("Hello World   "))` returns `"Hello World"`
- `trim-string("   Hello World"))` returns `"Hello World"`
- `trim-string("Hello World"))` returns `"Hello World"`
- `trim-string("Hello   World"))` returns `"Hello   World"`

trim-string-left [altova:]

`trim-string-left(InputString` *`as xs:string`*`)` as **`xs:string`** **XP3** **XQ3**

This function takes an `xs:string` argument, removes any leading whitespace, and returns a left-trimmed `xs:string`.

⊟ *Examples*

- `trim-string-left("   Hello World   "))` returns `"Hello World   "`
- `trim-string-left("Hello World   "))` returns `"Hello World   "`
- `trim-string-left("   Hello World"))` returns `"Hello World"`
- `trim-string-left("Hello World"))` returns `"Hello World"`
- `trim-string-left("Hello   World"))` returns `"Hello   World"`

trim-string-right [altova:]

`trim-string-right(InputString` *`as xs:string`*`)` as **`xs:string`** **XP3** **XQ3**

This function takes an `xs:string` argument, removes any trailing whitespace, and returns a right-trimmed `xs:string`.

⊟ *Examples*

- `trim-string-right("   Hello World   "))` returns `"   Hello World"`
- `trim-string-right("Hello World   "))` returns `"Hello World"`
- `trim-string-right("   Hello World"))` returns `"   Hello World"`
- `trim-string-right("Hello World"))` returns `"Hello World"`
- `trim-string-right("Hello   World"))` returns `"Hello   World"`

XPath/XQuery Functions: Miscellaneous

The following general-purpose XPath/XQuery extension functions are supported in the current version of your Altova product and can be used in (i) XPath expressions in an XSLT context, or (ii) XQuery expressions in an XQuery document.

Note about naming of functions and language applicability
Altova extension functions can be used in XPath/XQuery expressions. They provide additional functionality to the functionality that is available in the standard library of XPath, XQuery, and XSLT functions.

- In order to distinguish Altova extension functions from functions in the standard library, Altova extension functions are named in this documentation with a suffix of **`[altova:]`**. For example **`add-years-to-date [altova:]`**.
- When using Altova extension functions in your XPath/XQuery expressions, however, you must use the function without any prefix or suffix, like this: **`add-years-to-date`**.

XPath functions (used in XPath expressions in XSLT):	XP1 XP2 XP3
XSLT functions (used in XPath expressions in XSLT):	XSLT1 XSLT2 XSLT3
XQuery functions (used in XQuery expressions in XQuery):	XQ1 XQ3

Auto-numbering functions

▼ **`generate-auto-number [altova:]`**

`generate-auto-number(ID` *`as xs:string`*, **`StartsWith`** *`as xs:double`*, **`Increment`** *`as xs:double`*, **`ResetOnChange`** *`as xs:string`***`)`** as `xs:integer` XP1 XP2 XQ1 XP3 XQ3

Generates a number each time the function is called. The first number, which is generated the first time the function is called, is specified by the `StartsWith` argument. Each subsequent call to the function generates a new number, this number being incremented over the previously generated number by the value specified in the `Increment` argument. In effect, the `generate-auto-number` function creates a counter having a name specified by the `ID` argument, with this counter being incremented each time the function is called. If the value of the `ResetOnChange` argument changes from that of the previous function call, then the value of the number to be generated is reset to the `StartsWith` value. Auto-numbering can also be reset by using the `reset-auto-number` function.

⊟ *Examples*

- **`generate-auto-number`**`("ChapterNumber", 1, 1, "SomeString")` will return one number each time the function is called, starting with `1`, and incrementing by `1` with each call to the function. As long as the fourth argument remains `"SomeString"` in each subsequent call, the incrementing will continue. When the value of the fourth argument changes, the counter (called `ChapterNumber`) will reset to 1. The value of `ChapterNumber` can also be reset by a call to the `reset-auto-number` function, like this: `reset-auto-number("ChapterNumber")`.

▼ **`reset-auto-number [altova:]`**

`reset-auto-number(ID` *`as xs:string`***`)`** XP1 XP2 XQ1 XP3 XQ3

This function resets the number of the auto-numbering counter named in the `ID` argument.

The number is reset to the number specified by the `StartsWith` argument of the `generate-auto-number` function that created the counter named in the `ID` argument.

⊟ *Examples*

- `reset-auto-number("ChapterNumber")` resets the number of the auto-numbering counter named `ChapterNumber` that was created by the `generate-auto-number` function. The number is reset to the value of the `StartsWith` argument of the `generate-auto-number` function that created `ChapterNumber`.

[Top]

Numeric functions

▼ **hex-string-to-integer [altova:]**

hex-string-to-integer(HexString *as xs:string***)** as **xs:integer** **XP3 XQ3**

Takes a string argument that is the Base-16 equivalent of an integer in the decimal system (Base-10), and returns the decimal integer.

⊟ *Examples*

- `hex-string-to-integer('1')` returns `1`
- `hex-string-to-integer('9')` returns `9`
- `hex-string-to-integer('A')` returns `10`
- `hex-string-to-integer('B')` returns `11`
- `hex-string-to-integer('F')` returns `15`
- `hex-string-to-integer('G')` returns an error
- `hex-string-to-integer('10')` returns `16`
- `hex-string-to-integer('01')` returns `1`
- `hex-string-to-integer('20')` returns `32`
- `hex-string-to-integer('21')` returns `33`
- `hex-string-to-integer('5A')` returns `90`
- `hex-string-to-integer('USA')` returns an error

▼ **integer-to-hex-string [altova:]**

integer-to-hex-string(Integer *as xs:integer***)** as **xs:string** **XP3 XQ3**

Takes an integer argument and returns its Base-16 equivalent as a string.

⊟ *Examples*

- `integer-to-hex-string(1)` returns `'1'`
- `integer-to-hex-string(9)` returns `'9'`
- `integer-to-hex-string(10)` returns `'A'`
- `integer-to-hex-string(11)` returns `'B'`
- `integer-to-hex-string(15)` returns `'F'`
- `integer-to-hex-string(16)` returns `'10'`
- `integer-to-hex-string(32)` returns `'20'`
- `integer-to-hex-string(33)` returns `'21'`
- `integer-to-hex-string(90)` returns `'5A'`

[Top]

Sequence functions

▼ **attributes [altova:]**

attributes(AttributeName *as xs:string*) as **attribute()*** **XP3** **XQ3**

Returns all attributes that have a local name which is the same as the name supplied in the input argument, `AttributeName`. The search is case-sensitive and conducted along the `attribute::` axis.

⊟ *Examples*

- **attributes**("MyAttribute") returns `MyAttribute()*`

attributes(AttributeName *as xs:string*, **SearchOptions** *as xs:string*) as **attribute()*** **XP3** **XQ3**

Returns all attributes that have a local name which is the same as the name supplied in the input argument, `AttributeName`. The search is case-sensitive and conducted along the `attribute::` axis. The second argument is a string containing option flags. Available flags are:

r = switches to a regular-expression search; `AttributeName` must then be a regular-expression search string;
i = switches to a case-insensitive search;
p = includes the namespace prefix in the search; `AttributeName` should then contain the namespace prefix, for example: `altova:MyAttribute`.
The flags can be written in any order. Invalid flags will generate errors. One or more flags can be omitted. The empty string is allowed, and will produce the same effect as the function having only one argument (*previous signature*). However, an empty sequence is not allowed.

⊟ *Examples*

- **attributes**("MyAttribute", "rip") returns `MyAttribute()*`
- **attributes**("MyAttribute", "pri") returns `MyAttribute()*`
- **attributes**("MyAttribute", "") returns `MyAttribute()*`
- **attributes**("MyAttribute", "Rip") returns an unrecognized-flag error.
- **attributes**("MyAttribute",) returns a missing-second-argument error.

▼ **elements [altova:]**

elements(ElementName *as xs:string*) as **element()*** **XP3** **XQ3**

Returns all elements that have a local name which is the same as the name supplied in the input argument, `ElementName`. The search is case-sensitive and conducted along the `child::` axis.

⊟ *Examples*

- **elements**("MyElement") returns `MyElement()*`

elements(ElementName *as xs:string*, **SearchOptions** *as xs:string*) as **element()*** **XP3** **XQ3**

Returns all elements that have a local name which is the same as the name supplied in the input argument, `ElementName`. The search is case-sensitive and conducted along the `child::` axis. The second argument is a string containing option flags. Available flags are:

r = switches to a regular-expression search; `ElementName` must then be a regular-expression search string;
i = switches to a case-insensitive search;

p = includes the namespace prefix in the search; `ElementName` should then contain the namespace prefix, for example: `altova:MyElement`.
The flags can be written in any order. Invalid flags will generate errors. One or more flags can be omitted. The empty string is allowed, and will produce the same effect as the function having only one argument (*previous signature*). However, an empty sequence is not allowed.

⊟ *Examples*

- `elements("MyElement", "rip")` returns `MyElement()*`
- `elements("MyElement", "pri")` returns `MyElement()*`
- `elements("MyElement", "")` returns `MyElement()*`
- `elements("MyElement", "Rip")` returns an unrecognized-flag error.
- `elements("MyElement", )` returns a missing-second-argument error.

▼ find-first [altova:]

`find-first((Sequence as item()*), (Condition( Sequence-Item as xs:boolean))`
as `item()?` **XP3 XQ3**

This function takes two arguments. The first argument is a sequence of one or more items of any datatype. The second argument, `Condition`, is a reference to an XPath function that takes one argument (has an arity of **`1`**) and returns a boolean. Each item of **`Sequence`** is submitted, in turn, to the function referenced in `Condition`. (*Remember:* This function takes a single argument.) The first **`Sequence`** item that causes the function in **`Condition`** to evaluate to **`true()`** is returned as the result of **`find-first`**, and the iteration stops.

⊟ *Examples*

- `find-first(5 to 10, function($a) {$a mod 2 = 0})` returns `xs:integer 6`

  The **`Condition`** argument references the XPath 3.0 inline function, **`function()`**, which declares an inline function named **`$a`** and then defines it. Each item in the **`Sequence`** argument of **`find-first`** is passed, in turn, to **`$a`** as its input value. The input value is tested on the condition in the function definition (`$a mod 2 = 0`). The first input value to satisfy this condition is returned as the result of **`find-first`** (in this case **`6`**).

- `find-first((1 to 10), (function($a) {$a+3=7}))` returns `xs:integer 4`

Further examples

If the file **`C:\Temp\Customers.xml`** exists:

- `find-first( ("C:\Temp\Customers.xml", "http://www.altova.com/index.html"), (doc-available#1) )` returns `xs:string C:\Temp\Customers.xml`

If the file **`C:\Temp\Customers.xml`** does not exist, and **`http://www.altova.com/index.html`** exists:

- `find-first( ("C:\Temp\Customers.xml", "http://www.altova.com/index.html"), (doc-available#1) )` returns `xs:string http://www.altova.com/index.html`

If the file **`C:\Temp\Customers.xml`** does not exist, and **`http://www.altova.com/index.html`** also does not exist:

- `find-first( ("C:\Temp\Customers.xml", "http://www.altova.com/index.html"), (doc-available#1) )` returns no result

Notes about the examples given above

- The XPath 3.0 function, `doc-available`, takes a single string argument, which is used as a URI, and returns `true` if a document node is found at the submitted URI. (The document at the submitted URI must therefore be an XML document.)
- The `doc-available` function can be used for **`Condition`**, the second argument of `find-first`, because it takes only one argument (arity=1), because it takes an `item()` as input (a string which is used as a URI), and returns a boolean value.
- Notice that the `doc-available` function is only referenced, not called. The `#1` suffix that is attached to it indicates a function with an arity of 1. In its entirety `doc-available#1` simply means: *Use the doc-availabe() function that has arity=1, passing to it as its single argument, in turn, each of the items in the first sequence*. As a result, each of the two strings will be passed to **`doc-available()`**, which uses the string as a URI and tests whether a document node exists at the URI. If one does, the **`doc-available()`** evaluates to `true()` and that string is returned as the result of the **`find-first`** function. *Note about the doc-available() function: Relative paths are resolved relative to the the current base URI, which is by default the URI of the XML document from which the function is loaded.*

▾ `find-first-combination [altova:]`

`find-first-combination((Seq-01` *`as item()*`*`),` **`(Seq-02`** *`as item()*`*`),` **`(Condition( Seq-01-Item, Seq-02-Item`** *`as xs:boolean`*`))` as **`item()*`** **XP3 XQ3**

This function takes three arguments:

- The first two arguments, **`Seq-01`** and **`Seq-02`**, are sequences of one or more items of any datatype.
- The third argument, **`Condition`**, is a reference to an XPath function that takes two arguments (has an arity of **2**) and returns a boolean.

The items of **`Seq-01`** and **`Seq-02`** are passed in ordered pairs (one item from each sequence making up a pair) as the arguments of the function in **`Condition`**. The pairs are ordered as follows.

```
If   Seq-01 = X1, X2, X3 ... Xn
And  Seq-02 = Y1, Y2, Y3 ... Yn
Then (X1 Y1), (X1 Y2), (X1 Y3) ... (X1 Yn), (X2 Y1), (X2 Y2) ... (Xn Yn)
```

The first ordered pair that causes the **`Condition`** function to evaluate to **`true()`** is returned as the result of **`find-first-combination`**. Note that: (i) If the **`Condition`** function iterates through the submitted argument pairs and does not once evaluate to **`true()`**, then **`find-first-combination`** returns *No results*; (ii) The result of **`find-first-combination`** will always be a pair of items (of any datatype) or no item at all.

⊟ *Examples*

-

▾ `find-first-pair [altova:]`

`find-first-pair((Seq-01` *`as item()*`*`),` **`(Seq-02`** *`as item()*`*`),` **`(Condition( Seq-01-Item, Seq-02-Item`** *`as xs:boolean`*`))` as **`item()*`** **XP3 XQ3**

This function takes three arguments:

- The first two arguments, **`Seq-01`** and **`Seq-02`**, are sequences of one or more items of

any datatype.
- The third argument, **`Condition`**, is a reference to an XPath function that takes two arguments (has an arity of **2**) and returns a boolean.

The items of **`Seq-01`** and **`Seq-02`** are passed in ordered pairs as the arguments of the function in **`Condition`**. The pairs are ordered as follows.

```
If   Seq-01 = X1, X2, X3 ... Xn
And  Seq-02 = Y1, Y2, Y3 ... Yn
Then (X1 Y1), (X2 Y2), (X3 Y3) ... (Xn Yn)
```

The first ordered pair that causes the **`Condition`** function to evaluate to **`true()`** is returned as the result of **`find-first-pair`**. Note that: (i) If the **`Condition`** function iterates through the submitted argument pairs and does not once evaluate to **`true()`**, then **`find-first-pair`** returns *No results*; (ii) The result of **`find-first-pair`** will always be a pair of items (of any datatype) or no item at all.

⊟ *Examples*
- **`find-first-pair`**`(11 to 20, 21 to 30, function($a, $b) {$a+$b = 32})` returns the sequence of `xs:integers (11, 21)`
- **`find-first-pair`**`(11 to 20, 21 to 30, function($a, $b) {$a+$b = 33})` returns *No results*

Notice from the two examples above that the ordering of the pairs is: `(11, 21) (12, 22) (13, 23)...(20, 30)`. This is why the second example returns *No results* (because no ordered pair gives a sum of `33`).

▼ find-first-pair-pos [altova:]

`find-first-pair-pos((Seq-01` *`as item()*`***`), (Seq-02`** *`as item()*`***`), (Condition( Seq-01-Item, Seq-02-Item`** *`as xs:boolean`***`))`** as **`xs:integer`** **XP3** **XQ3**

This function takes three arguments:
- The first two arguments, **`Seq-01`** and **`Seq-02`**, are sequences of one or more items of any datatype.
- The third argument, **`Condition`**, is a reference to an XPath function that takes two arguments (has an arity of **2**) and returns a boolean.

The items of **`Seq-01`** and **`Seq-02`** are passed in ordered pairs as the arguments of the function in **`Condition`**. The pairs are ordered as follows.

```
If   Seq-01 = X1, X2, X3 ... Xn
And  Seq-02 = Y1, Y2, Y3 ... Yn
Then (X1 Y1), (X2 Y2), (X3 Y3) ... (Xn Yn)
```

The index position of the first ordered pair that causes the **`Condition`** function to evaluate to **`true()`** is returned as the result of **`find-first-pair-pos`**. Note that if the **`Condition`** function iterates through the submitted argument pairs and does not once evaluate to **`true()`**, then **`find-first-pair-pos`** returns *No results*.

⊟ *Examples*
- **`find-first-pair`**`(11 to 20, 21 to 30, function($a, $b) {$a+$b = 32})` returns `1`

- **find-first-pair**(11 to 20, 21 to 30, function($a, $b) {$a+$b = 33}) returns *No results*

Notice from the two examples above that the ordering of the pairs is: (11, 21) (12, 22) (13, 23)...(20, 30). In the first example, the first pair causes the **Condition** function to evaluate to **true()**, and so its index position in the sequence, **1**, is returned. The second example returns *No results* because no pair gives a sum of 33.

find-first-pos [altova:]

find-first-pos((Sequence *as item()**), **(Condition(Sequence-Item** *as xs:boolean*)**)** as **xs:integer** **XP3** **XQ3**

This function takes two arguments. The first argument is a sequence of one or more items of any datatype. The second argument, Condition, is a reference to an XPath function that takes one argument (has an arity of **1**) and returns a boolean. Each item of **Sequence** is submitted, in turn, to the function referenced in Condition. (*Remember:* This function takes a single argument.) The first **Sequence** item that causes the function in **Condition** to evaluate to **true()** has its index position in **Sequence** returned as the result of **find-first-pos**, and the iteration stops.

⊟ *Examples*

- **find-first-pos**(5 to 10, function($a) {$a mod 2 = 0}) returns xs:integer 2

The **Condition** argument references the XPath 3.0 inline function, **function()**, which declares an inline function named **$a** and then defines it. Each item in the **Sequence** argument of **find-first-pos** is passed, in turn, to **$a** as its input value. The input value is tested on the condition in the function definition ($a mod 2 = 0). The index position in the sequence of the first input value to satisfy this condition is returned as the result of **find-first-pos** (in this case **2**, since **6**, the first value (in the sequence) to satisfy the condition, is at index position **2** in the sequence).

- **find-first-pos**((2 to 10), (function($a) {$a+3=7})) returns xs:integer 3

Further examples

If the file **C:\Temp\Customers.xml** exists:

- **find-first-pos**(("C:\Temp\Customers.xml", "http://www.altova.com/index.html"), (doc-available#1)) returns 1

If the file **C:\Temp\Customers.xml** does not exist, and **http://www.altova.com/index.html** exists:

- **find-first-pos**(("C:\Temp\Customers.xml", "http://www.altova.com/index.html"), (doc-available#1)) returns 2

If the file **C:\Temp\Customers.xml** does not exist, and **http://www.altova.com/index.html** also does not exist:

- **find-first-pos**(("C:\Temp\Customers.xml", "http://www.altova.com/index.html"), (doc-available#1)) returns no result

Notes about the examples given above

- The XPath 3.0 function, doc-available, takes a single string argument, which is used as a URI, and returns `true` if a document node is found at the submitted URI. (The document at the submitted URI must therefore be an XML document.)

- The `doc-available` function can be used for **`Condition`**, the second argument of `find-first-pos`, because it takes only one argument (arity=1), because it takes an `item()` as input (a string which is used as a URI), and returns a boolean value.
- Notice that the `doc-available` function is only referenced, not called. The `#1` suffix that is attached to it indicates a function with an arity of 1. In its entirety `doc-available#1` simply means: *Use the doc-availabe() function that has arity=1, passing to it as its single argument, in turn, each of the items in the first sequence*. As a result, each of the two strings will be passed to **`doc-available()`**, which uses the string as a URI and tests whether a document node exists at the URI. If one does, the **`doc-available()`** function evaluates to `true()` and the index position of that string in the sequence is returned as the result of the **`find-first-pos`** function. *Note about the doc-available() function: Relative paths are resolved relative to the the current base URI, which is by default the URI of the XML document from which the function is loaded.*

▼ **`substitute-empty [altova:]`**

`substitute-empty(FirstSequence` *`as item()*`*, **`SecondSequence`** *`as item()`*) as **`item()*`** **XP3 XQ3**

If `FirstSequence` is empty, returns `SecondSequence`. If `FirstSequence` is not empty, returns `FirstSequence`.

⊟ *Examples*

- **`substitute-empty`**`( (1,2,3), (4,5,6) )` returns `(1,2,3)`
- **`substitute-empty`**`( (), (4,5,6) )` returns `(4,5,6)`

[Top]

URI functions

▼ **`get-temp-folder [altova:]`**

`get-temp-folder()` as `xs:string` **XP2 XQ1 XP3 XQ3**

This function takes no argument. It returns the path to the temporary folder of the current user.

⊟ *Examples*

- **`get-temp-folder`**`()` would return, on a Windows machine, something like `C:\Users\<UserName>\AppData\Local\Temp\` as an `xs:string`.

[Top]

Miscellaneous Extension Functions

There are several ready-made functions in programming languages such as Java and C# that are not available as XQuery/XPath functions or as XSLT functions. A good example would be the math functions available in Java, such as `sin()` and `cos()`. If these functions were available to the designers of XSLT stylesheets and XQuery queries, it would increase the application area of stylesheets and queries and greatly simplify the tasks of stylesheet creators. The XSLT and XQuery engines used in a number of Altova products support the use of extension functions in Java and .NET, as well as MSXSL scripts for XSLT. This section describes how to use extension functions and MSXSL scripts in your XSLT stylesheets and XQuery documents. The available extension functions are organized into the following sections:

- Java Extension Functions
- .NET Extension Functions

The two main issues considered in the descriptions are: (i) how functions in the respective libraries are called; and (ii) what rules are followed for converting arguments in a function call to the required input format of the function, and what rules are followed for the return conversion (function result to XSLT/XQuery data object).

Requirements
For extension functions support, a Java Runtime Environment (for access to Java functions) and .NET Framework 2.0 (minimum, for access to .NET functions) must be installed on the machine running the XSLT transformation or XQuery execution, or must be accessible for the transformations.

Java Extension Functions

A Java extension function can be used within an XPath or XQuery expression to invoke a Java constructor or call a Java method (static or instance).

A field in a Java class is considered to be a method without any argument. A field can be static or instance. How to access fields is described in the respective sub-sections, static and instance.

This section is organized into the following sub-sections:

- Java: Constructors
- Java: Static Methods and Static Fields
- Java: Instance Methods and Instance Fields
- Datatypes: XPath/XQuery to Java
- Datatypes: Java to XPath/XQuery

Form of the extension function

The extension function in the XPath/XQuery expression must have the form `prefix:fname()`.

- The `prefix:` part identifies the extension function as a Java function. It does so by associating the extension function with an in-scope namespace declaration, the URI of which must begin with `java:` (*see below for examples*). The namespace declaration should identify a Java class, for example: `xmlns:myns="java:java.lang.Math"`. However, it could also simply be: `xmlns:myns="java"` (without a colon), with the identification of the Java class being left to the `fname()` part of the extension function.
- The `fname()` part identifies the Java method being called, and supplies the arguments for the method (*see below for examples*). However, if the namespace URI identified by the `prefix:` part does not identify a Java class (*see preceding point*), then the Java class should be identified in the `fname()` part, before the class and separated from the class by a period (*see the second XSLT example below*).

Note: The class being called must be on the classpath of the machine.

XSLT example

Here are two examples of how a static method can be called. In the first example, the class name (`java.lang.Math`) is included in the namespace URI and, therefore, must not be in the `fname()` part. In the second example, the `prefix:` part supplies the prefix `java:` while the `fname()` part identifies the class as well as the method.

```
<xsl:value-of xmlns:jMath="java:java.lang.Math"
              select="jMath:cos(3.14)" />

<xsl:value-of xmlns:jmath="java"
              select="jmath:java.lang.Math.cos(3.14)" />
```

The method named in the extension function (`cos()` in the example above) must match the name of a public static method in the named Java class (`java.lang.Math` in the example above).

XQuery example

Here is an XQuery example similar to the XSLT example above:

```
<cosine xmlns:jMath="java:java.lang.Math">
   {jMath:cos(3.14)}
</cosine>
```

User-defined Java classes

If you have created your own Java classes, methods in these classes are called differently according to: (i) whether the classes are accessed via a JAR file or a class file, and (ii) whether these files (JAR or class) are located in the current directory (the same directory as the XSLT or XQuery document) or not. How to locate these files is described in the sections User-Defined Class Files and User-Defined Jar Files. Note that paths to class files not in the current directory and to all JAR files must be specified.

User-Defined Class Files

If access is via a class file, then there are two possibilities:

- The class file is in a package. The XSLT or XQuery file is in the same folder as the Java package. (*See example below*.)
- The class file is not packaged. The XSLT or XQuery file is in the same folder as the class file. (*See example below*.)
- The class file is in a package. The XSLT or XQuery file is at some random location. (*See example below*.)
- The class file is not packaged. The XSLT or XQuery file is at some random location. (*See example below*.)

Consider the case where the class file is not packaged and is in the same folder as the XSLT or XQuery document. In this case, since all classes in the folder are found, the file location does not need to be specified. The syntax to identify a class is:

```
java:classname
```

where

`java:` indicates that a user-defined Java function is being called; (Java classes in the current directory will be loaded by default)
`classname` is the name of the required method's class

The class is identified in a namespace URI, and the namespace is used to prefix a method call.

Class file packaged, XSLT/XQuery file in same folder as Java package

The example below calls the `getVehicleType()`method of the `Car` class of the `com.altova.extfunc` package. The `com.altova.extfunc` package is in the folder `JavaProject`. The XSLT file is also in the folder `JavaProject`.

```
<xsl:stylesheet version="2.0"
       xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
       xmlns:xs="http://www.w3.org/2001/XMLSchema"
       xmlns:fn="http://www.w3.org/2005/xpath-functions"
       xmlns:car="java:com.altova.extfunc.Car" >
<xsl:output exclude-result-prefixes="fn car xsl fo xs"/>

<xsl:template match="/">
   <a>
   <xsl:value-of select="car:getVehicleType()"/>
   </a>
</xsl:template>

</xsl:stylesheet>
```

Class file not packaged, XSLT/XQuery file in same folder as class file
The example below calls the `getVehicleType()`method of the `Car` class of the `com.altova.extfunc` package. The `Car` class file is in the following folder location: `JavaProject/com/altova/extfunc`. The XSLT file is also in the folder `JavaProject/com/altova/extfunc`.

```
<xsl:stylesheet version="2.0"
       xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
       xmlns:xs="http://www.w3.org/2001/XMLSchema"
       xmlns:fn="http://www.w3.org/2005/xpath-functions"
       xmlns:car="java:Car" >
<xsl:output exclude-result-prefixes="fn car xsl fo xs"/>

<xsl:template match="/">
   <a>
   <xsl:value-of select="car:getVehicleType()"/>
   </a>
</xsl:template>

</xsl:stylesheet>
```

Class file packaged, XSLT/XQuery file at any location
The example below calls the `getCarColor()`method of the `Car` class of the `com.altova.extfunc` package. The `com.altova.extfunc` package is in the folder `JavaProject`. The XSLT file is at any location. In this case, the location of the package must be specified within the URI as a query string. The syntax is:

```
java:classname[?path=uri-of-package]
```

where

`java:` indicates that a user-defined Java function is being called
`uri-of-package` is the URI of the Java package
`classname` is the name of the required method's class

The class is identified in a namespace URI, and the namespace is used to prefix a method call. The example below shows how to access a class file that is located in another directory than the current directory.

```
<xsl:stylesheet version="2.0"
       xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
       xmlns:xs="http://www.w3.org/2001/XMLSchema"
       xmlns:fn="http://www.w3.org/2005/xpath-functions"
       xmlns:car="java:com.altova.extfunc.Car?path=file:///C:/
JavaProject/" >

<xsl:output exclude-result-prefixes="fn car xsl xs"/>

<xsl:template match="/">
   <xsl:variable name="myCar" select="car:new('red')" />
   <a><xsl:value-of select="car:getCarColor($myCar)"/></a>
</xsl:template>

</xsl:stylesheet>
```

Class file not packaged, XSLT/XQuery file at any location
The example below calls the `getCarColor()`method of the `Car` class of the `com.altova.extfunc` package. The `com.altova.extfunc` package is in the folder `JavaProject`. The XSLT file is at any location. The location of the class file is specified within the namespace URI as a query string. The syntax is:

```
java:classname[?path=uri-of-classfile]
```

where

`java:` indicates that a user-defined Java function is being called
`uri-of-classfile` is the URI of the folder containing the class file
`classname` is the name of the required method's class

The class is identified in a namespace URI, and the namespace is used to prefix a method call. The example below shows how to access a class file that is located in another directory than the current directory.

```
<xsl:stylesheet version="2.0"
       xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
       xmlns:xs="http://www.w3.org/2001/XMLSchema"
```

```
              xmlns:fn="http://www.w3.org/2005/xpath-functions"
              xmlns:car="java:Car?path=file:///C:/JavaProject/com/altova/
extfunc/" >

       <xsl:output exclude-result-prefixes="fn car xsl xs"/>

       <xsl:template match="/">
          <xsl:variable name="myCar" select="car:new('red')" />
          <a><xsl:value-of select="car:getCarColor($myCar)"/></a>
       </xsl:template>

       </xsl:stylesheet>
```

Note: When a path is supplied via the extension function, the path is added to the ClassLoader.

User-Defined Jar Files

If access is via a JAR file, the URI of the JAR file must be specified using the following syntax:

```
xmlns:classNS="java:classname?path=jar:uri-of-jarfile!/"
```

The method is then called by using the prefix of the namespace URI that identifies the class: `classNS:method()`

In the above:

`java:` indicates that a Java function is being called
`classname` is the name of the user-defined class
`?` is the separator between the classname and the path
`path=jar:` indicates that a path to a JAR file is being given
`uri-of-jarfile` is the URI of the jar file
`!/` is the end delimiter of the path
`classNS:method()` is the call to the method

Alternatively, the classname can be given with the method call. Here are two examples of the syntax:

```
       xmlns:ns1="java:docx.layout.pages?path=jar:file:///c:/projects/docs/
docx.jar!/"
       ns1:main()

       xmlns:ns2="java?path=jar:file:///c:/projects/docs/docx.jar!/"
       ns2:docx.layout.pages.main()
```

Here is a complete XSLT example that uses a JAR file to call a Java extension function:

```
       <xsl:stylesheet version="2.0"
              xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
              xmlns:xs="http://www.w3.org/2001/XMLSchema"
              xmlns:fn="http://www.w3.org/2005/xpath-functions"
              xmlns:car="java?path=jar:file:///C:/test/Car1.jar!/" >
```

```
<xsl:output exclude-result-prefixes="fn car xsl xs"/>

<xsl:template match="/">
      <xsl:variable name="myCar" select="car:Car1.new('red')" />
      <a><xsl:value-of select="car:Car1.getCarColor($myCar)"/></a>
</xsl:template>

<xsl:template match="car"/>

</xsl:stylesheet>
```

Note: When a path is supplied via the extension function, the path is added to the ClassLoader.

Java: Constructors

An extension function can be used to call a Java constructor. All constructors are called with the pseudo-function `new()`.

If the result of a Java constructor call can be implicitly converted to XPath/XQuery datatypes, then the Java extension function will return a sequence that is an XPath/XQuery datatype. If the result of a Java constructor call cannot be converted to a suitable XPath/XQuery datatype, then the constructor creates a wrapped Java object with a type that is the name of the class returning that Java object. For example, if a constructor for the class `java.util.Date` is called (`java.util.Date.new()`), then an object having a type `java.util.Date` is returned. The lexical format of the returned object may not match the lexical format of an XPath datatype and the value would therefore need to be converted to the lexical format of the required XPath datatype and then to the required XPath datatype.

There are two things that can be done with a Java object created by a constructor:

- It can be assigned to a variable:
  `<xsl:variable name="currentdate" select="date:new()" xmlns:date="java:java.util.Date" />`
- It can be passed to an extension function (*see* *Instance Method and Instance Fields*):
  `<xsl:value-of select="date:toString(date:new())" xmlns:date="java:java.util.Date" />`

Java: Static Methods and Static Fields

A static method is called directly by its Java name and by supplying the arguments for the method. Static fields (methods that take no arguments), such as the constant-value fields `E` and `PI`, are accessed without specifying any argument.

XSLT examples

Here are some examples of how static methods and fields can be called:

```
<xsl:value-of xmlns:jMath="java:java.lang.Math"
              select="jMath:cos(3.14)" />
```

```
<xsl:value-of xmlns:jMath="java:java.lang.Math"
              select="jMath:cos( jMath:PI() )" />

<xsl:value-of xmlns:jMath="java:java.lang.Math"
              select="jMath:E() * jMath:cos(3.14)" />
```

Notice that the extension functions above have the form `prefix:fname()`. The prefix in all three cases is `jMath:`, which is associated with the namespace URI `java:java.lang.Math`. (The namespace URI must begin with `java:`. In the examples above it is extended to contain the class name (`java.lang.Math`).) The `fname()` part of the extension functions must match the name of a public class (e.g. `java.lang.Math`) followed by the name of a public static method with its argument/s (such as `cos(3.14)`) or a public static field (such as `PI()`).

In the examples above, the class name has been included in the namespace URI. If it were not contained in the namespace URI, then it would have to be included in the `fname()` part of the extension function. For example:

```
<xsl:value-of xmlns:java="java:"
              select="java:java.lang.Math.cos(3.14)" />
```

XQuery example

A similar example in XQuery would be:

```
<cosine xmlns:jMath="java:java.lang.Math">
   {jMath:cos(3.14)}
</cosine>
```

Java: Instance Methods and Instance Fields

An instance method has a Java object passed to it as the first argument of the method call. Such a Java object typically would be created by using an extension function (for example a constructor call) or a stylesheet parameter/variable. An XSLT example of this kind would be:

```
<xsl:stylesheet version="1.0" exclude-result-prefixes="date"
  xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
  xmlns:date="java:java.util.Date"
  xmlns:jlang="java:java.lang">
  <xsl:param name="CurrentDate" select="date:new()"/>
  <xsl:template match="/">
      <enrollment institution-id="Altova School"
                date="{date:toString($CurrentDate)}"
                type="
{jlang:Object.toString(jlang:Object.getClass( date:new() ))}">
      </enrollment>
  </xsl:template>
</xsl:stylesheet>
```

In the example above, the value of the node `enrollment/@type` is created as follows:

1. An object is created with a constructor for the class `java.util.Date` (with the `date:new()` constructor).
2. This Java object is passed as the argument of the `jlang.Object.getClass` method.

3. The object obtained by the `getClass` method is passed as the argument to the `jlang.Object.toString` method.

The result (the value of `@type`) will be a string having the value: `java.util.Date`.

An instance field is theoretically different from an instance method in that it is not a Java object per se that is passed as an argument to the instance field. Instead, a parameter or variable is passed as the argument. However, the parameter/variable may itself contain the value returned by a Java object. For example, the parameter `CurrentDate` takes the value returned by a constructor for the class `java.util.Date`. This value is then passed as an argument to the instance method `date:toString` in order to supply the value of `/enrollment/@date`.

Datatypes: XPath/XQuery to Java

When a Java function is called from within an XPath/XQuery expression, the datatype of the function's arguments is important in determining which of multiple Java classes having the same name is called.

In Java, the following rules are followed:

- If there is more than one Java method with the same name, but each has a different number of arguments than the other/s, then the Java method that best matches the number of arguments in the function call is selected.
- The XPath/XQuery string, number, and boolean datatypes (*see list below*) are implicitly converted to a corresponding Java datatype. If the supplied XPath/XQuery type can be converted to more than one Java type (for example, `xs:integer`), then that Java type is selected which is declared for the selected method. For example, if the Java method being called is `fx(decimal)` and the supplied XPath/XQuery datatype is `xs:integer`, then `xs:integer` will be converted to Java's `decimal` datatype.

The table below lists the implicit conversions of XPath/XQuery string, number, and boolean types to Java datatypes.

`xs:string`	`java.lang.String`
`xs:boolean`	`boolean` (primitive), `java.lang.Boolean`
`xs:integer`	int, long, short, byte, float, double, and the wrapper classes of these, such as `java.lang.Integer`
`xs:float`	`float` (primitive), `java.lang.Float`, `double` (primitive)
`xs:double`	`double` (primitive), `java.lang.Double`
`xs:decimal`	`float` (primitive), `java.lang.Float`, `double`(primitive), `java.lang.Double`

Subtypes of the XML Schema datatypes listed above (and which are used in XPath and XQuery) will also be converted to the Java type/s corresponding to that subtype's ancestor type.

In some cases, it might not be possible to select the correct Java method based on the supplied information. For example, consider the following case.

- The supplied argument is an `xs:untypedAtomic` value of 10 and it is intended for the method `mymethod(float)`.
- However, there is another method in the class which takes an argument of another datatype: `mymethod(double)`.
- Since the method names are the same and the supplied type (`xs:untypedAtomic`) could be converted correctly to either `float` or `double`, it is possible that `xs:untypedAtomic` is converted to `double` instead of `float`.
- Consequently the method selected will not be the required method and might not produce the expected result. To work around this, you can create a user-defined method with a different name and use this method.

Types that are not covered in the list above (for example `xs:date`) will not be converted and will generate an error. However, note that in some cases, it might be possible to create the required Java type by using a Java constructor.

Datatypes: Java to XPath/XQuery

When a Java method returns a value, the datatype of the value is a string, numeric or boolean type, then it is converted to the corresponding XPath/XQuery type. For example, Java's `java.lang.Boolean` and `boolean` datatypes are converted to `xsd:boolean`.

One-dimensional arrays returned by functions are expanded to a sequence. Multi-dimensional arrays will not be converted, and should therefore be wrapped.

When a wrapped Java object or a datatype other than string, numeric or boolean is returned, you can ensure conversion to the required XPath/XQuery type by first using a Java method (e.g `toString`) to convert the Java object to a string. In XPath/XQuery, the string can be modified to fit the lexical representation of the required type and then converted to the required type (for example, by using the `cast as` expression).

.NET Extension Functions

If you are working on the .NET platform on a Windows machine, you can use extension functions written in any of the .NET languages (for example, C#). A .NET extension function can be used within an XPath or XQuery expression to invoke a constructor, property, or method (static or instance) within a .NET class.

A property of a .NET class is called using the syntax `get_PropertyName()`.

This section is organized into the following sub-sections:

- .NET: Constructors
- .NET: Static Methods and Static Fields
- .NET: Instance Methods and Instance Fields
- Datatypes: XPath/XQuery to .NET
- Datatypes: .NET to XPath/XQuery

Form of the extension function
The extension function in the XPath/XQuery expression must have the form `prefix:fname()`.

- The `prefix:` part is associated with a URI that identifies the .NET class being addressed.
- The `fname()` part identifies the constructor, property, or method (static or instance) within the .NET class, and supplies any argument/s, if required.
- The URI must begin with `clitype:` (which identifies the function as being a .NET extension function).
- The `prefix:fname()` form of the extension function can be used with system classes and with classes in a loaded assembly. However, if a class needs to be loaded, additional parameters containing the required information will have to be supplied.

Parameters
To load an assembly, the following parameters are used:

`asm`	The name of the assembly to be loaded.
`ver`	The version number (maximum of four integers separated by periods).
`sn`	The key token of the assembly's strong name (16 hex digits).
`from`	A URI that gives the location of the assembly (DLL) to be loaded. If the URI is relative, it is relative to the XSLT or XQuery document. If this parameter is present, any other parameter is ignored.
`partialname`	The partial name of the assembly. It is supplied to `Assembly.LoadWith.PartialName()`, which will attempt to load the assembly. If `partialname` is present, any other parameter is ignored.
`loc`	The locale, for example, `en-US`. The default is `neutral`.

If the assembly is to be loaded from a DLL, use the `from` parameter and omit the `sn` parameter. If the assembly is to be loaded from the Global Assembly Cache (GAC), use the `sn` parameter and omit the `from` parameter.

A question mark must be inserted before the first parameter, and parameters must be separated by a semi-colon. The parameter name gives its value with an equals sign (*see example below*).

Examples of namespace declarations

An example of a namespace declaration in XSLT that identifies the system class `System.Environment`:

```
xmlns:myns="clitype:System.Environment"
```

An example of a namespace declaration in XSLT that identifies the class to be loaded as `Trade.Forward.Scrip`:

```
xmlns:myns="clitype:Trade.Forward.Scrip?asm=forward;version=10.6.2.1"
```

An example of a namespace declaration in XQuery that identifies the system class `MyManagedDLL.testClass`:. Two cases are distinguished:

1. When the assembly is loaded from the GAC:
   ```
   declare namespace cs="clitype:MyManagedDLL.testClass?asm=MyManagedDLL;
   ver=1.2.3.4;loc=neutral;sn=b9f091b72dccfba8";
   ```

2. When the assembly is loaded from the DLL (complete and partial references below):
   ```
          declare namespace cs="clitype:MyManagedDLL.testClass?from=file:///C:/
   Altova
          Projects/extFunctions/MyManagedDLL.dll;

          declare namespace cs="clitype:MyManagedDLL.testClass?
          from=MyManagedDLL.dll;
   ```

XSLT example

Here is a complete XSLT example that calls functions in system class `System.Math`:

```
<xsl:stylesheet version="2.0"
   xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
   xmlns:xs="http://www.w3.org/2001/XMLSchema"
   xmlns:fn="http://www.w3.org/2005/xpath-functions">
   <xsl:output method="xml" omit-xml-declaration="yes" />
   <xsl:template match="/">
      <math xmlns:math="clitype:System.Math">
         <sqrt><xsl:value-of select="math:Sqrt(9)"/></sqrt>
         <pi><xsl:value-of select="math:PI()"/></pi>
         <e><xsl:value-of select="math:E()"/></e>
         <pow><xsl:value-of select="math:Pow(math:PI(), math:E())"/></pow>
      </math>
```

```
   </xsl:template>
</xsl:stylesheet>
```

The namespace declaration on the element `math` associates the prefix `math:` with the URI `clitype:System.Math`. The `clitype:` beginning of the URI indicates that what follows identifies either a system class or a loaded class. The `math:` prefix in the XPath expressions associates the extension functions with the URI (and, by extension, the class) `System.Math`. The extension functions identify methods in the class `System.Math` and supply arguments where required.

XQuery example
Here is an XQuery example fragment similar to the XSLT example above:

```
<math xmlns:math="clitype:System.Math">
   {math:Sqrt(9)}
</math>
```

As with the XSLT example above, the namespace declaration identifies the .NET class, in this case a system class. The XQuery expression identifies the method to be called and supplies the argument.

.NET: Constructors

An extension function can be used to call a .NET constructor. All constructors are called with the pseudo-function `new()`. If there is more than one constructor for a class, then the constructor that most closely matches the number of arguments supplied is selected. If no constructor is deemed to match the supplied argument/s, then a `'No constructor found'` error is returned.

Constructors that return XPath/XQuery datatypes
If the result of a .NET constructor call can be implicitly converted to XPath/XQuery datatypes, then the .NET extension function will return a sequence that is an XPath/XQuery datatype.

Constructors that return .NET objects
If the result of a .NET constructor call cannot be converted to a suitable XPath/XQuery datatype, then the constructor creates a wrapped .NET object with a type that is the name of the class returning that object. For example, if a constructor for the class `System.DateTime` is called (with `System.DateTime.new()`), then an object having a type `System.DateTime` is returned.

The lexical format of the returned object may not match the lexical format of a required XPath datatype. In such cases, the returned value would need to be: (i) converted to the lexical format of the required XPath datatype; and (ii) cast to the required XPath datatype.

There are three things that can be done with a .NET object created by a constructor:

- It can be used within a variable:
  ```
  <xsl:variable name="currentdate" select="date:new(2008, 4, 29)"
  xmlns:date="clitype:System.DateTime" />
  ```
- It can be passed to an extension function (*see* *Instance Method and Instance Fields*):
  ```
  <xsl:value-of select="date:ToString(date:new(2008, 4, 29))"
  xmlns:date="clitype:System.DateTime" />
  ```
- It can be converted to a string, number, or boolean:

-
  ```
  <xsl:value-of select="xs:integer(data:get_Month(date:new(2008, 4, 29)))"
  xmlns:date="clitype:System.DateTime" />
  ```

.NET: Static Methods and Static Fields

A static method is called directly by its name and by supplying the arguments for the method. The name used in the call must exactly match a public static method in the class specified. If the method name and the number of arguments that were given in the function call matches more than one method in a class, then the types of the supplied arguments are evaluated for the best match. If a match cannot be found unambiguously, an error is reported.

Note: A field in a .NET class is considered to be a method without any argument. A property is called using the syntax `get_PropertyName()`.

Examples

An XSLT example showing a call to a method with one argument (`System.Math.Sin(arg)`):

```
<xsl:value-of select="math:Sin(30)" xmlns:math="clitype:System.Math"/>
```

An XSLT example showing a call to a field (considered a method with no argument) (`System.Double.MaxValue()`):

```
<xsl:value-of select="double:MaxValue()" xmlns:double="clitype:System.Double"/>
```

An XSLT example showing a call to a property (syntax is `get_PropertyName()`) (`System.String()`):

```
<xsl:value-of select="string:get_Length('my string')"
xmlns:string="clitype:System.String"/>
```

An XQuery example showing a call to a method with one argument (`System.Math.Sin(arg)`):

```
<sin xmlns:math="clitype:System.Math">
   { math:Sin(30) }
</sin>
```

.NET: Instance Methods and Instance Fields

An instance method has a .NET object passed to it as the first argument of the method call. This .NET object typically would be created by using an extension function (for example a constructor call) or a stylesheet parameter/variable. An XSLT example of this kind would be:

```
<xsl:stylesheet version="2.0"
   xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
   xmlns:xs="http://www.w3.org/2001/XMLSchema"
   xmlns:fn="http://www.w3.org/2005/xpath-functions">
   <xsl:output method="xml" omit-xml-declaration="yes"/>
   <xsl:template match="/">
      <xsl:variable name="releasedate"
         select="date:new(2008, 4, 29)"
         xmlns:date="clitype:System.DateTime"/>
      <doc>
         <date>
            <xsl:value-of select="date:ToString(date:new(2008, 4, 29))"
               xmlns:date="clitype:System.DateTime"/>
         </date>
         <date>
            <xsl:value-of select="date:ToString($releasedate)"
               xmlns:date="clitype:System.DateTime"/>
         </date>
      </doc>
   </xsl:template>
</xsl:stylesheet>
```

In the example above, a `System.DateTime` constructor (`new(2008, 4, 29)`) is used to create a .NET object of type `System.DateTime`. This object is created twice, once as the value of the variable `releasedate`, a second time as the first and only argument of the `System.DateTime.ToString()` method. The instance method `System.DateTime.ToString()` is called twice, both times with the `System.DateTime` constructor (`new(2008, 4, 29)`) as its first and only argument. In one of these instances, the variable `releasedate` is used to get the .NET object.

Instance methods and instance fields

The difference between an instance method and an instance field is theoretical. In an instance method, a .NET object is directly passed as an argument; in an instance field, a parameter or variable is passed instead—though the parameter or variable may itself contain a .NET object. For example, in the example above, the variable `releasedate` contains a .NET object, and it is this variable that is passed as the argument of `ToString()` in the second `date` element constructor. Therefore, the `ToString()` instance in the first `date` element is an instance method while the second is considered to be an instance field. The result produced in both instances, however, is the same.

Datatypes: XPath/XQuery to .NET

When a .NET extension function is used within an XPath/XQuery expression, the datatypes of the function's arguments are important for determining which one of multiple .NET methods having the same name is called.

In .NET, the following rules are followed:

- If there is more than one method with the same name in a class, then the methods available for selection are reduced to those that have the same number of arguments as the function call.

- The XPath/XQuery string, number, and boolean datatypes (*see list below*) are implicitly converted to a corresponding .NET datatype. If the supplied XPath/XQuery type can be converted to more than one .NET type (for example, `xs:integer`), then that .NET type is selected which is declared for the selected method. For example, if the .NET method being called is `fx(double)` and the supplied XPath/XQuery datatype is `xs:integer`, then `xs:integer` will be converted to .NET's `double` datatype.

The table below lists the implicit conversions of XPath/XQuery string, number, and boolean types to .NET datatypes.

`xs:string`	`StringValue, string`
`xs:boolean`	`BooleanValue, bool`
`xs:integer`	`IntegerValue, decimal, long, integer, short, byte, double, float`
`xs:float`	`FloatValue, float, double`
`xs:double`	`DoubleValue, double`
`xs:decimal`	`DecimalValue, decimal, double, float`

Subtypes of the XML Schema datatypes listed above (and which are used in XPath and XQuery) will also be converted to the .NET type/s corresponding to that subtype's ancestor type.

In some cases, it might not be possible to select the correct .NET method based on the supplied information. For example, consider the following case.

- The supplied argument is an `xs:untypedAtomic` value of 10 and it is intended for the method `mymethod(float)`.
- However, there is another method in the class which takes an argument of another datatype: `mymethod(double)`.
- Since the method names are the same and the supplied type (`xs:untypedAtomic`) could be converted correctly to either `float` or `double`, it is possible that `xs:untypedAtomic` is converted to `double` instead of `float`.
- Consequently the method selected will not be the required method and might not produce the expected result. To work around this, you can create a user-defined method with a different name and use this method.

Types that are not covered in the list above (for example `xs:date`) will not be converted and will generate an error.

Datatypes: .NET to XPath/XQuery

When a .NET method returns a value and the datatype of the value is a string, numeric or boolean type, then it is converted to the corresponding XPath/XQuery type. For example, .NET's `decimal` datatype is converted to `xsd:decimal`.

When a .NET object or a datatype other than string, numeric or boolean is returned, you can ensure conversion to the required XPath/XQuery type by first using a .NET method (for example

`System.DateTime.ToString()`) to convert the .NET object to a string. In XPath/XQuery, the string can be modified to fit the lexical representation of the required type and then converted to the required type (for example, by using the `cast as` expression).

12.2 License Information

This section contains:

- Information about the distribution of this software product
- Information about software activation and license metering
- 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.

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.

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* the IANA website (http://www.iana.org/) *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.

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.

Altova MobileTogether Designer End User License Agreement

THIS IS A LEGAL DOCUMENT -- RETAIN FOR YOUR RECORDS

ALTOVA® MOBILETOGETHER DESIGNER 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 ("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 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 Agreement as part of the installation process at the time of acceptance. Alternatively, a copy of this Agreement may be found at http://www.altova.com/legal.html and a copy of the Privacy Policy may be found at http://www.altova.com/privacy.

1. SOFTWARE LICENSE

(a) License Grant.
(i) Upon your acceptance of this 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.

(ii) You may not use the Software to develop and distribute other software programs that directly compete with any Altova software or service without prior written permission. Altova reserves all other rights in and to the Software.

(b) 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.

(c) 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 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 Agreement, are your property.

(d) 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.

(e) Other Restrictions. You may not loan, rent, lease, sublicense, distribute or otherwise transfer all or any portion of the Software to third parties except as otherwise expressly provided. You may not copy the Software except as expressly set forth herein, and any copies that you are permitted to make pursuant to this 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 Agreement. You may not permit any use of or access to the Software by any third party in connection with a commercial service offering, such as for a cloud-based or web-based SaaS offering.

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 Agreement and to ensure their compliance with these restrictions.

(f) NO GUARANTEE. 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 THIRD 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

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. Altova®, XMLSpy®, Authentic®, StyleVision®, MapForce®, UModel®, DatabaseSpy®, DiffDog®, SchemaAgent®, SemanticWorks®, MissionKit®, Markup Your Mind®, Nanonull™, MobileTogether™, MobileTogether Server™, MobileTogether Designer™, MobileTogether Mobile App™, RaptorXML™, RaptorXML Server™, RaptorXML +XBRL Server™, Powered By RaptorXML™, FlowForce Server™, StyleVision Server™, and MapForce Server™ are trademarks of Altova GmbH. (pending or registered in numerous countries).Unicode and the Unicode Logo are trademarks of Unicode, Inc. Windows, Windows XP, Windows Vista, Windows 7, and Windows 8 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 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 WARRANTY AND LIMITATION OF LIABILITY

(a) Limited Warranty and Customer Remedies. YOU ACKNOWLEDGE THAT THE 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 THE 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.

(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. 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 Agreement between Altova and you.

(d) Waiver and Indemnity. BY USING THE SOFTWARE, YOU AGREE, TO THE EXTENT PERMITTED BY LAW, TO SAVE AND PROTECT, HOLD HARMLESS, INDEMNIFY AND DEFEND ALTOVA, ITS DIRECTORS, OFFICERS, EMPLOYEES, AFFILIATES, AGENTS, CONTRACTORS, AND LICENSORS AGAINST ANY AND ALL LIABILITY, CAUSES OF ACTION, CLAIMS, LOSS DAMAGE OR COST AND EXPENSE ARISING FROM, ALLEGEDLY ARISING FROM, OR RESULTING DIRECTLY OR INDIRECTLY FROM ANY ACTS OF THE LICENSEE OR ANY OF ITS OFFICERS, EMPLOYEES, INDEPENDENT CONTRACTORS OR AGENTS DONE IN THE PERFORMANCE, OPERATION, OR USE OF THE SOFTWARE, OR ANY ACT DONE UNDER PRETENDED AUTHORITY OF THIS LICENSE. THIS AGREEMENT TO INDEMNIFY AND HOLD ALTOVA HARMLESS SHALL INCLUDE ANY COSTS INCURRED BY ALTOVA IN DEFENDING ANY ACTION INVOLVING AN ACT BY THE LICENSEE OR ANY OF ITS OFFICERS, EMPLOYEES, INDEPENDENT CONTRACTORS OR AGENTS, AND SHALL INCLUDE ANY ATTORNEY'S FEES INCURRED BY ALTOVA.

4. SUPPORT AND MAINTENANCE

Altova offers "Support & Maintenance Package(s)" ("SMP") for the Software only if you have obtained a valid license for MobileTogether Server, which may be obtained from Altova at www.altova.com. The Support Period shall coincide with the MobileTogether Server Software license term. The terms of SMP are set forth in the Altova MobileTogether Server Software License Agreement located at http://www.altova.com/license_agreements.html.

5. SOFTWARE ACTIVATION AND UPDATES

(a) License Metering. The Software includes a built-in license metering module that is designed to assist you with monitoring license compliance in small local networks. The metering module attempts to communicate with other machines on your local area network. You permit Altova to use your internal network for license monitoring for this purpose. This license metering module may be used to assist with your license compliance but should not be the sole method. Should your firewall settings block said communications, you must deploy an accurate means of monitoring usage by the end user and preventing users from using the Software more than the Permitted Number.

(b) License Compliance Monitoring. You are required to utilize a process or tool to ensure that the Permitted Number is not exceeded. Without prejudice or waiver of any potential violations of the Agreement, Altova may provide you with additional compliance tools should you be unable to accurately account for license usage within your organization. If provided with such a tool by Altova, you (a) are required to use it in order to comply with the terms of this Agreement and (b) permit Altova to use your internal network for license monitoring and metering and to generate compliance reports that are communicated to Altova from time to time.

(c) 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 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 Agreement. You agree that efforts to circumvent or disable Altova's copyright protection mechanisms or License Server violate Altova's intellectual property rights as well as the terms of this 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.

(d) 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.

(e) 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 Agreement. By your acceptance of the terms of this 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 Agreement and/or the Privacy Policy. Altova has the right in its sole discretion to amend this provision of the 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.

(f) Audit Rights. You agree that Altova may audit your use of the Software for compliance with the terms of this Agreement at any time, upon reasonable notice. In the event that such audit reveals any use of the Software by you other than in full compliance with the terms of this Agreement, you shall reimburse Altova for all reasonable expenses related to such audit in addition to any other liabilities you may incur as a result of such non-compliance.

(g) Notice to European Users. Please note that the information as described in paragraph 5(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 Agreement, you consent to the transfer of all such information to the United States and the processing of that information as described in this Agreement and the Privacy Policy.

6. TERM AND TERMINATION

This Agreement may be terminated (a) by your giving Altova written notice of termination; or (b) by Altova, at any time without prior notice. Upon any termination of the Agreement, you must cease all use of the Software that this 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(c), 1(d), 1(e), 1(l), 2, 3, 5, 7, 8, and 9 survive termination as applicable.

7. 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 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.

8. U.S. GOVERNMENT ENTITIES

Notwithstanding the foregoing, if you are an agency, instrumentality or department of the federal government of the United States, then this Agreement shall be governed in accordance with the laws of the United States of America, and in the absence of applicable federal law, the laws of the Commonwealth of Massachusetts will apply. Further, and notwithstanding anything to the contrary in this Agreement (including but not limited to Section 5 (Indemnification)), all claims, demands, complaints and disputes will be subject to the Contract Disputes Act (41 U.S.C. §§7101 *et seq.*), the Tucker Act (28 U.S.C. §1346(a) and §1491), or the Federal Tort Claims Act (28 U.S.C. §§1346(b), 2401-2402, 2671-2672, 2674-2680), FAR 1.601(a) and 43.102 (Contract Modifications); FAR 12.302(b), as applicable, or other applicable governing authority. For the avoidance of doubt, if you are an agency, instrumentality, or department of the federal, state or local government of the U.S. or a U.S. public and accredited educational institution, then your indemnification obligations are only applicable to the extent they would not cause you to violate any applicable law (e.g., the Anti-Deficiency Act), and you have any legally required authorization or authorizing statute.

9. 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.

10. 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 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 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 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 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 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 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 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 Agreement. This 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 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 Agreement. If, for any reason, any provision of this Agreement is held invalid or otherwise unenforceable, such invalidity or unenforceability shall not affect the remainder of this Agreement, and this 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: 2014-06-05

Index