Quest® One

Identity Manager 6.1

Web Designer Reference Guide

QUEST SOFTWARE®
Simplicity At Work®

If you have any questions regarding your potential use of this material, contact:

Dell Software Inc.
Attn: LEGAL Dept
5 Polaris Way
Aliso Viejo, CA 92656

Refer to our web site (www.software.dell.com) for regional and international office information.

Patents

This product is protected by U.S. Patent #8,639,827 and #8,601,539. Additional patents pending.

Trademarks

Dell, the Dell logo, and Quest One Identity Manager, Data Governance, ActiveRoles, Quick Connect, Password Manager and Webthority are trademarks of Dell Inc. and/or its affiliates. Other trademarks and trade names may be used in this document to refer to either the entities claiming the marks and names or their products. Dell disclaims any proprietary interest in the marks and names of others.

Third Party Contributions

This product contains some third-party components (listed below). Copies of their licenses may be found at referencing http://software.dell.com/legal/license-agreements.aspx. Source code for components marked with an asterisk (*) is available at http://opensource.dell.com.

List of Third-Party Contributions

COMPONENT	LICENSE OR ACKNOWLEDGEMENT
.Less 1.3.1	Apache License Version 2.0, January 2004 (http://www.apache.org/licenses). Apache 2.0 License.

List of Third-Party Contributions

COMPONENT	LICENSE OR ACKNOWLEDGEMENT
.Less 1.3.1	Copyright (c) 2009-2010 dotless project, http://www.dotlesscss.com. Apache License Version 2.0, Januar 2004 (http://www.apache.org/licenses). Apache 2.0 License.
.NET logging library 1.0	Copyright (c) 2004-2011 Jaroslaw Kowalski <jaak@jkowalski.net>. BSD 4.4 License.
Boost 1.34.1	Boost Software License - Version 1.0 - August 17th, 2003. Boost 1.0 License.
cherrypy 3.1.1	Copyright (c) 2004-2011, CherryPy Team (team@cherrypy.org) All rights reserved. BSD 4.4 License.
Dojo Toolkit 1.8.3	Copyright (c) 2005-2014, The Dojo Foundation All rights reserved. BSD Simple License.
Google APIs Auth Client Library 1.6.0.1	Apache License Version 2.0, Januar 2004 (http://www.apache.org/licenses). Apache 2.0 License.
Google APIs Auth MVC Extensions 1.6.0	Apache License Version 2.0, Januar 2004 (http://www.apache.org/licenses). Apache 2.0 License.
Google APIs Client Library for .NET 1.6.0 (Beta)	Apache License Version 2.0, Januar 2004 (http://www.apache.org/licenses). Apache 2.0 License.
Google Open Sans 1.0	Apache License Version 2.0, Januar 2004 (http://www.apache.org/licenses). Apache 2.0 License.
Google.Apis.Admin.Directory.directory_v1 Client Li 1.6.0.21	Apache License Version 2.0, Januar 2004 (http://www.apache.org/licenses). Apache 2.0 License.
jcrop 0.9.9	Copyright ©2008 Kelly Hallman and Deep Liquid Group. MIT License.
JQuery 1.7.1	Copyright (c) 2011, John Resig. MIT License.
JQuery UI 1.8.20	Copyright ©. MIT License.
Log4Net 1.2.11	Apache License Version 2.0, Januar 2004 (http://www.apache.org/licenses). Apache 2.0 License.
Log4Net 2.0.3	Apache License Version 2.0, Januar 2004 (http://www.apache.org/licenses). Apache 2.0 License.
Mono.Security 2.0.3600.1	Copyright (c) 2001, 2002, 2003 Ximian, Inc and the individuals listed on the ChangeLog entries. MIT License.
Newtonsoft.Json.dll 5.0.8	Copyright ©. MIT License.
Novell.Directory.LDAP 2.1.9.0	Copyright © 2011 Novell, Inc.All rights reserved. MIT License.
pyodbc 2.1.3	Copyright ©. MIT License.
Python 2.5.2	Python 2.5 license 2.5. Copyright 2001-2006 Python Software Foundation. All rights reserved. Copyright 1995-2001 Corporation for National Research Initiatives. All rights reserved. Copyright 1991-1995 Stichting Mathematisch Centrum Amsterdam, The Netherlands. All rights reserved.

List of Third-Party Contributions

COMPONENT	LICENSE OR ACKNOWLEDGEMENT
SharpZipLib 0.85.4.369	SharpZipLib License.
spin.js 1.2.2	Copyright ©. MIT License.
SQLAlchemy 0.5.0	Copyright (c) 2005-2014 Michael Bayer and contributors. SQLAlchemy is a trademark of Michael Bayer. mike(&)zzzcomputing.com All rights reserved. MIT License.
Windows Installer XML toolset (aka WIX) 3.6.3303.0	Microsoft Reciprocal License (MS-RL).
zlib 1.2.3	Copyright © 1995-2005 Jean-loup Gailly and Mark Adler. zlib 1.2.3 License.
zlib portable 1.9.2	Copyright (C) 1995-2012 Jean-loup Gailly and Mark Adler. zlib 1.2.7 License.
ZLib.NET 1.0.3	Copyright © 2006, ComponentAce (http://www.componentace.com). All rights reserved. ZLib.NET 1.0.3 License.

Quest One Identity Manager - Web Designer Reference Guide

Updated - August 2015

Software Version - 6.1.3

Content

1

About this Guide

- Quest® One Identity Manager
- Intended Audience
- Identity Manager Documentation

Quest® One Identity Manager

Quest One Identity Manager streamlines the process of managing user identities, access privileges and security enterprise wide. It empowers identity and access management to be driven by business needs, not IT capabilities. Quest One Identity Manager is based on an automation-optimized architecture that addresses major identity and access management challenges at a fraction of the complexity, time, or expense of “traditional” solutions.

Intended Audience

This guide explains the functionality of the Web Designer tool to you. You will discover how you can customize the operational processes from the Web Portal installation version and implement more operational processes. To do this, you will be introduced to the interface and each of the Web Designer functions.

This guide is intended for system administrators, consultants, analysts, and any other IT professionals using the product.

This guide describes the default user functionality of the Web Designer. It is possible that not all the functions described here are available to you. This depends on your system configuration and permissions.

Identity Manager Documentation

Identity Manager documentation includes the following manuals. They can be found on the installation media in the directory ...\Quest One Identity Manager\Documentation.

- *Getting Started Guide*
  This guide provides Identity Manager installation instructions and product walkthroughs.
- *Identity Management Guide*
  This guide provides Identity Manager in-depth administration information.
- *Process Orchestration Guide*
  This guide provides information about designing and implementing process automation with the Identity Manager.
- *Configuration Guide*
  This guide provides information about configuring and customizing theIdentity Manager.
- *IT Shop Guide*
  This guide provides in-depth administration information on managing the IT Shop.
- *Web Portal User Guide*
  This guide provides in-depth information about Web Portal handling.
- *Web Portal Installation Guide*
  This guide provides Web Portal installation instructions.
- *Web Designer Reference Guide*
  This guide provides information about configuring and customizing the Web Portal.
- *Identity Manager Web Installation Guide*
  This guide provides Identity Manager Web installation instructions.

2

Before Starting

- Glossary
- Files needed by Web Designer
- Files generated by Web Designer

Glossary

The Web Designer tool is the one that is used to configure and extend the Identity Manager standard web portal within the Quest One Identity Manager Suite.

The browser supported standard web portal makes it possible for users with the appropriate permissions to query data in the Identity Manager database and to save it there. These queries and saving data take place within the framework of the implemented operational processes and is set up in such a way that even inexperienced users cannot create any invalid data. The necessarily restrictive user guidance resulting from this approach can be customized in the main areas.

If the configuration options are not sufficient to customize the default processes as required to meet company requirements, custom operational processes have to be implemented.

To this end, Web Designer is based on Model-View-Controller (MVC) architecture, whose model layer contains the data that is to be visualized, whose view layer allows for the display of data from the model layer and the receipt of user interactions, and whose controller layer receives user actions, assesses them, and acts accordingly; it also decides on the basis of user actions in the view layer which model layer data needs to be modified. The controller layer contains additional mechanisms that restrict user interactions in the view layer.

Web Designer also integrates an extensive GUI where the web application functions can be "clicked together."

In the interest of helping you to better understand the sections that follow, you should first familiarize yourself with the terms and definitions that arise in working with Web Designer.

Glossary

TERM	DEFINITION
Web application	A web application is a productive web site that is published on a web server and that interacts with the Identity Manager database, thus making available functions such as processing and saving employee or order data. Each web application has a dedicated intranet or internet address.
Web project	A web project is a programming phase that precedes a web application, where each web application relates to a specific web project. Web projects are independent of each other in respect to their design and functions. A series of web projects can be created and worked on concurrently in the Identity Manager database.
Definition tree	Each web project has a dedicated node tree. This tree is edited and its nodes configured using the Web Designer GUI. While a specific web application is in use, the user is always located at a defined position on the definition tree.

Glossary

TERM	DEFINITION
Nodes	A definition tree is composed of nodes. Accordingly, each node has one superordinate node and an unlimited number of subordinate nodes. There are various types of nodes: • Data, display and action nodes correspond to the MVC architecture of Web Designer. • Helper nodes, which are needed in order to create an optimally clearly structured definition tree. All non-helper nodes are associated with a function such as displaying browser data, querying data in the SQL server and so on. The function that a node executes is determined by its type (e.g. label node, collection node and so on). In this context, it is necessary for each node to be configured exactly in accordance with its specific task. A series of properties are available for these configurations (see Node Editing, Tasks, Properties and Queries on page 40).
Control	"Control" in this document refers to a display node that is rendered by a browser.
Module	Modules are self contained elements of a web project (and thus part of a definition tree) that normally represent a workflow or an otherwise delimited work domain such as "processing employee data" or "home page". Subdividing a web project into various modules enhances transparency during the programming phase and allows for the reuse of specific modules in various web projects (n module can be part of a number of web projects at the same time). Modules can contain an unlimited number of web pages.
Session module	Each web project needs a dedicated module that is known as a session module, whose main distinguishing feature relative to other modules is its so-called lifecycle. Session module data can be called up at any time during a valid web server session. Data from other modules can be called up while a web page that is defined in the relevant module is being displayed.
Component	A component is an element of the definition tree that can be regarded as having been "dispatched" from the actual tree, whereas the tree itself contains a reference to the component. Components can be referenced from an unlimited number of positions on the definition tree. Thus a component that is referenced two or more times is to all intents and purposes reused. This reduces programming and support workloads, while at the same enhancing quality by virtue of the fact that "similar" positions function the same way.
Local components	Local components are essentially the same as components, except that local components are defined within a module and are only found there. Hence a reference to a local component can only be set in the module in which the local component has also been defined. This is an advantage in that it allows for simpler referencing of local components.

Glossary

TERM	DEFINITION
Component type	Inasmuch as each specific type of node only admits certain other types of node, component referencing must also follow this logic. Hence when a component is defined, it is necessary to indicate which type of component it is. Then, depending on the component's position in the definition tree, only specific types of components can be referenced.
Database object	A database object, an object stored in a database, can contain structural information such as files generated by Web Designer, as well as utility data such as personal objects.
"VI" prefix	This prefix is used to indicate that a module, component or other element is part of the standard delivery. **Note:** This prefix cannot be changed in the customer installation.
"Customer" prefix	This prefix is used to indicate that a module, component or other element is not part of our standard delivery. This prefix can be changed in the customer installation. "Customer" here is a placeholder for the customer prefix that is defined in a delivery.
Collection	A collection is a compilation of specific data from the Identity Manager database on the web server. This data is called up (loaded) from the database using a specific rule ("where clause"). A loaded collection contains data sets which may, but do not necessarily have to, represent a database object. **For example:** A "Database object" collection can only ever access one Identity Manager database table whose data sets thus also always represent database objects. A "Database SQL-query" collection can load data fields from a series of tables via various SQL commands such as "union" or "join". In such cases, the data sets do not represent a database object.
Property	"Property" in this document refers to a collection property such as all UID_Persons of the Collection Person. A property can, but does not necessarily have to, correspond to a database table column.
Web SQL	In some Web Designer domains it is necessary to enter textually formulated programming instructions ("expressions"). In order to meet all web programming requirements, a dedicated programming language known as Web SQL was developed and implemented for Web Designer. The syntax of Web SQL is largely based on MS SQL syntax. One of the great advantages of Web SQL is that it allows for the formulation of very concrete runtime expressions such as SQL statements that are executed by the SQL server.
Expression	There are two Web Designer domains that necessitate textually formulated programming instructions (expressions). One such domain selects or manipulates collection data, whereas the other relates to css rule entries. In both domains, expressions must be entered using Web SQL syntax. In the first domain, the Web SQL expression assessment results must be consistent with standard SQL syntax, and in the second domain they must be consistent with css syntax.

Glossary

TERM	DEFINITION
Condition	A condition is a special form of an SQL expression. In Web Designer there are visibility, execution and activity conditions for all of which the anticipated result of an expression is "True" or "False". If the condition of a visibility condition is met in a container, the display nodes that are defined under the container are visible in the browser. If the condition of an execution condition is met in an action node, the action defined under this node is implemented. If the condition of an activity condition is met at a button, the button changes to an active state.
Filter	A filter is a special form of an SQL expression. Filters select a defined number of data sets from among all the data sets in a collection.

Files needed by Web Designer

To start Web Designer, launch the WebDesigner.exe file, which is located in the binary directory once Identity Manager has successfully been installed.

In order for Web Designer to work properly, it needs a series of additional files, which are also located in the binary directory. Apart from these files, which are used by other Identity Manager applications as well, there are files that are used solely by Web Designer. These files are as follows:

- WebDesigner.Base.dll
- WebDesigner.ImageLibrary.dll
- AE.WebDesigner.Template.Standard.dll

When WebDesigner.exe is started, this must be notified to the database. For further information in this regard, see the Logging into Identity Manager tools in the *Getting Started Guide*.

Files generated by Web Designer

The xml files generated by Web Designer are stored in the DialogAEDS table in the Identity Manager database. Inasmuch as a published web application directly accesses the xml files in the Identity Manager database, the web server directory structure needs no updating. In order for changes that have been implemented and saved using Web Designer to also be available in the web application, these changes must be accepted (see Project Approval Dialog on page 64). This function allows you to save interim states (i.e. states that cannot be compiled in Web Designer) without interfering with the published web application.

For publication of web applications see *Web Portal Installation Guide*.

3

The Web Designer GUI

- Introduction
- Start Mask
- Title Bar
- Status Bar
- Menu Bar
- Toolbar
- Navigation Window
- Preview Mode, Project Browser and Object State
- Node Editing, Tasks, Properties and Queries
- Additional Functions
- Definition Tree Window

Introduction

Proceed as follows to start the Web Designer:

1. Open the directory that contains the WebDesigner.exe and open it by double clicking.
2. Select your preferred connection

   - OR -

   Click **Add a new connection** to create a new connection.
3. Select from the drop-down list **Login as** the authentication module for database connect.
4. Enter your user name in the **User** text box.
5. Enter your password in the **Password** text box.
6. Click **Connect**.
7. Select the project save type that you want to use with the Web Designer. You can choose between:
   a) Database project

      The project is saved in the database. All changes are immediately visible to all users.

   b) File based project

      The project and your changes are saved in your local file system. Changes must be imported manually, so that your changes are visible to other users.

   c) Standard project

      The project is saved in a local folder. The database objects cannot be reused.

Enable the option Use as default to save your preferred project save type. You can disable the display (see table Functions in the Global Settings Dialog on page 21).

8. Click **OK**.

   The Web Designer is opened.

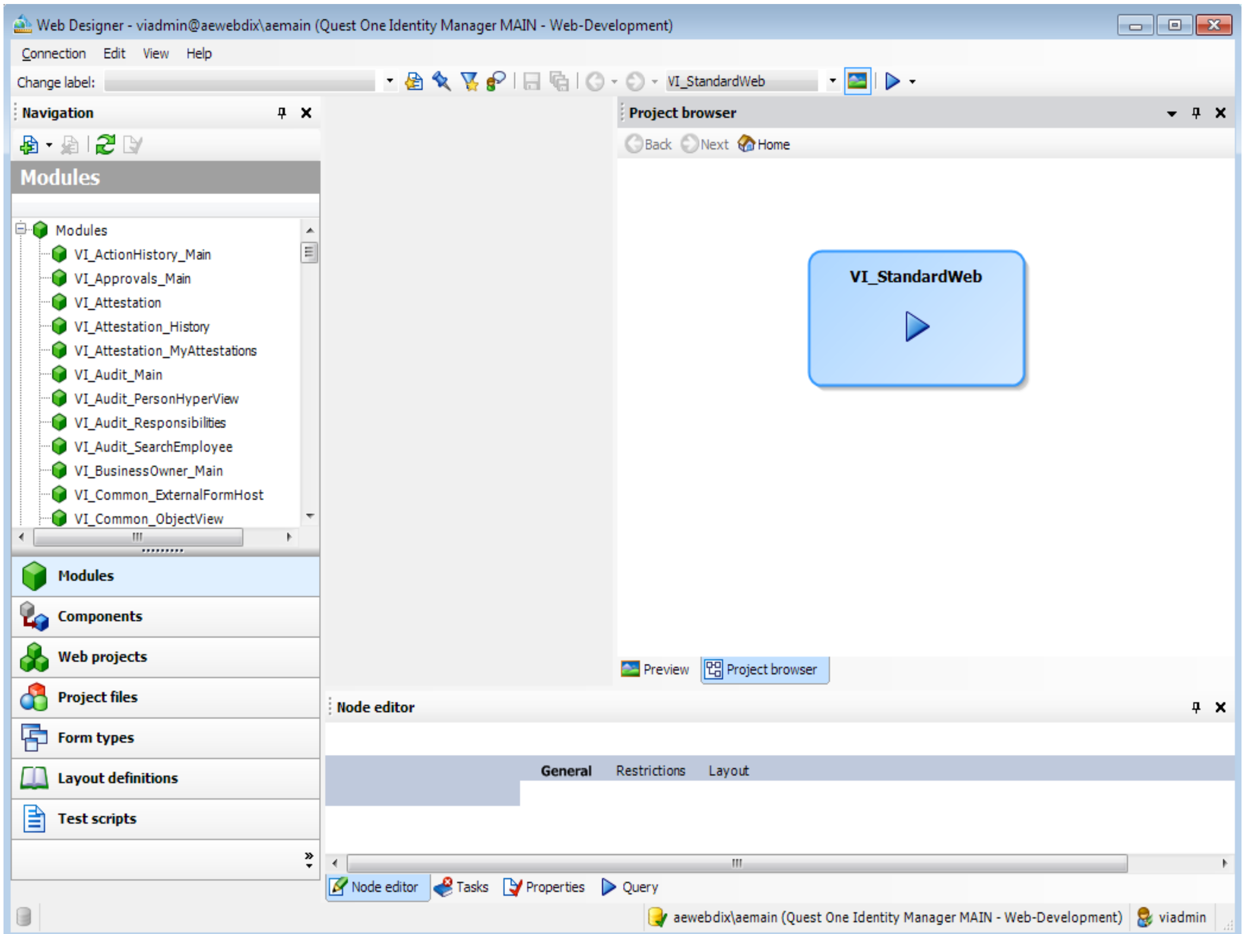
Start Mask

The Web Designer GUI can be navigated using a mouse and keyboard shortcuts. For optimum graphics display, a minimum screen resolution of 1280 x 1024 pixels with 16 colors is recommended.

As with all of Identity Manager tools, normal view can be customized by dragging, closing or hiding the various windows. To do this, use auto-hide mode through the Pin icon in the upper right hand corner of the window.

After you log on, a start mask, which is empty for the most part, will be displayed. It contains a title bar, status bar, menu bar, toolbar, and various editing areas. All of these elements are discussed in the sections that follow.

Web Designer Start Mask



Title Bar

The title bar contains the program icon, program name, and the name of the linked database. These elements are shown in the following form:

Web Designer - <user>@<database server>\<database (description)>.

Status Bar

The status bar shows the linked database in the following notation:

<server>\<database (description)> and the connected user or system user.

When Web Designer launches a database activity such as loading or saving a database object, the status icon is activated in the left corner of the status bar. A gray icon means "no activity".

Menu Bar

The items in the menu bar allow you to call up virtually all Web Designer functions.

Menu Bar

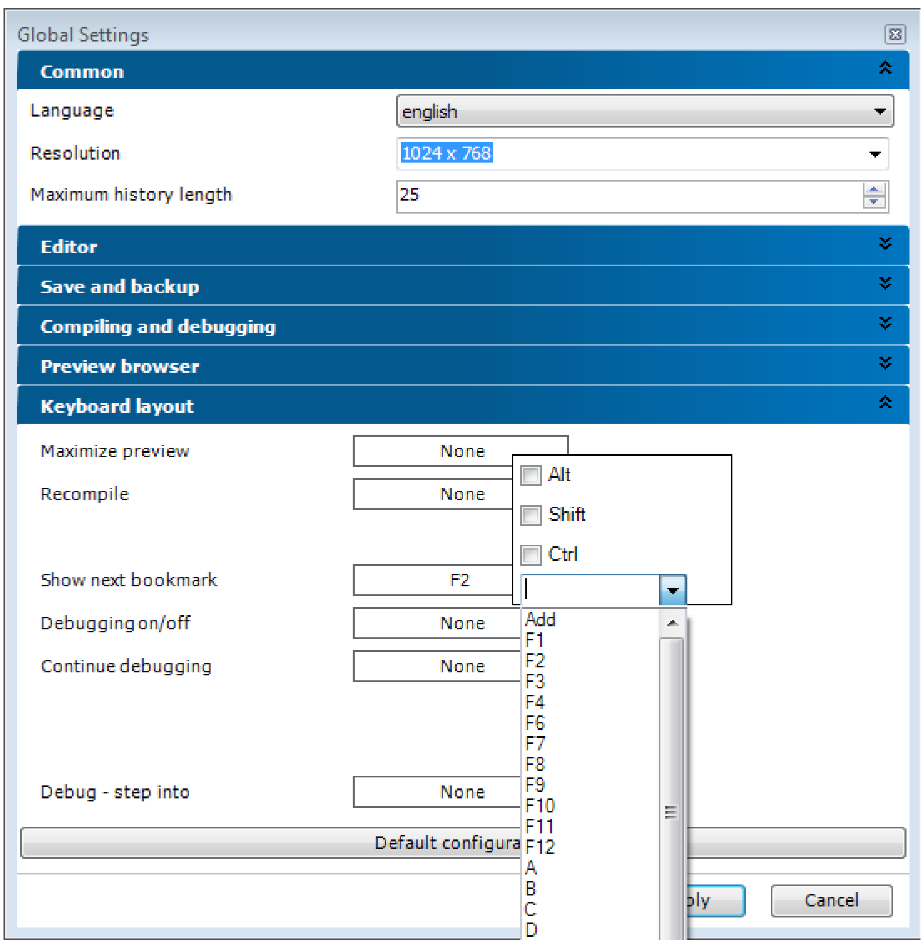
Connect

Functions of the Connect Tab on the Menu Bar

MENU ITEM	FUNCTION
Settings	Opens the global settings dialog.
Exit	Close the database connection and shuts down Web Designer.

The global settings dialog is divided into 6 frames. The functions of the frames Common, Editor, Save and backup, Compiling and debugging, and Preview browser are described in the following table. The Keyboard layout frame allows you to restore the standard keyboard layout or customize it.

Global Settings Dialog



Functions in the Global Settings Dialog

MENU ITEM	FUNCTION
Language	The language selection box enables you to change the Web Designer display language. To implement such a change, you need to restart Web Designer.
Resolution	The value shown here determines the size of the preview window after it has been undocked. Normally, web applications are developed for a customer-defined screen resolution. The Web Designer resolution function helps you to make optimum use of the software's display areas.
Maximum history length	With this option you can define the maximum history length of elements, which are shown of the function One node back and One node forward (Web Designer toolbar).
Use colors to distinguish data, display and action nodes	When this option is activated, Web Designer provides a color-coded display of the nodes in the definition tree window, as follows: data - red, display - green, actions - blue, other elements - black.
Shorten long property values	When this option is activated, expressions entered in the node editor window are displayed in an abbreviated form insofar as a specific length is exceeded. However, the full expression is always shown in the definition tree window.
Hide optional identifiers in definition tree	When this option is activated, any optional identifiers in the definition tree display are replaced by standard identifiers.
Display property categories	When this option is activated, the properties of the marked nodes are grouped according to category.
Show message after correcting XML consistency errors	When this option is activated, an error message is displayed if you enter incorrect expressions. Is the option deactivated, a tool tip with an error description is displayed of the red ! button.
Save and load Web Designer size within layout	When this option is activated, the currently set layout and size of the Web Designer is saved and after a restart the Web Designer will be opened with the saved settings.
Ask for save type after every start up	When you log on to the Web Designer, you have to select your preferred project save type. If you have enabled the option Use as default in the Please choose project save type window, you can disable the display of this window (see Introduction on page 18).
Ask for change label if none is selected	When this option is activated, before changes are saved, the program asks you whether you wish to select a name under which the changes are to be saved. If this is not the case, a message to this effect will be displayed.
Keep backup of unsaved objects on the local machine	When this option is activated, a backup copy of all modified and non-saved objects is saved on the local computer. This backup copy can be recovered if the program crashes.
Perform multi-language check	When this option is activated, error messages are displayed in case of missing texts (Inspection during compilation). For more information see Figure Captions Dialog on page 65 and Figure Clean Language-Dependent Text Dialog on page 67.
Suppress timed jobs	When this option is activated, time-controlled actions such as updating pending decisions in the menu bar are not executed in Web Designer. This is a particularly useful feature in Web Designer debug mode.

Functions in the Global Settings Dialog

MENU ITEM	FUNCTION
Profiling mode	When this option is activated, a detailed logging mechanism for all activities in the preview window is activated. This mode is useful for troubleshooting errors, but is very resource-intensive owing to the level of detail involved, and should only be used on request from Identity Manager support.
Show website without JavaScript	This option activates the JavaScript-free display mode in the preview window. This option allows you to display web pages as they would appear in a web browser without JavaScript.
Preview browser	Select your preferred Internet Explorer version. A change will take effect after a restart of the application.

Edit

Functions of the Edit Tab on the Menu Bar

<table>
<tr><th>MENU ITEM</th><th>FUNCTION</th></tr>
<tr><td>Compile project</td><td>Starts the compiler, which checks the current definition for errors. For this process, you need not to save the development state beforehand. For information concerning compiler error handling, see Tasks on page 41.</td></tr>
<tr><td>Publish project</td><td>Opens the publish project dialog, see Project Approval Dialog on page 64.</td></tr>
<tr><td>Save current project to file system</td><td>Opens the project export wizard.

To save the current project:
Select your preferred web project. Its shown in the Web Designer toolbar.
Click Save current project to file system.
Enter the file name in the File name text box or click the button besides the text box and choose a folder.
If necessary, activate the option Load created project after running this wizard.
Click Next.
The project is compiled and the objects are copied to the target folder.
Click Finish.</td></tr>
<tr><td>Set save type for projects</td><td>Opens the Please choose project type window, see Introduction on page 18 and table Functions in the Global Settings Dialog on page 21.</td></tr>
<tr><td>Configure project</td><td>Opens the project configuration window (also called configurator for short). See Project Configuration on page 55 for a description of configurator functions.</td></tr>
<tr><td>Copy objects</td><td>Opens the copy objects window. The process is realized by a wizard, see Copy Objects Wizard on page 75.</td></tr>
<tr><td>Create new module</td><td>Opens the create new module window. The process is realized by a wizard, see New Module Wizard on page 76.</td></tr>
<tr><td>Create new project</td><td>Opens the create new project window. The process is realized by a wizard, see New Project Wizard on page 72.</td></tr>
<tr><td>Migrations wizard</td><td>Opens the migration wizard window. The process is realized by a wizard, see Migration Wizard on page 74.</td></tr>
<tr><td>Search</td><td>Opens the find dialog, see Search and Replace Dialog on page 64.</td></tr>
<tr><td>Search next</td><td>Executes a search using the current search settings without opening the find mask.</td></tr>
<tr><td>Captions</td><td>Opens the captions dialog, where you can insert, delete and edit captions, see Captions Dialog on page 65.</td></tr>
<tr><td>Clean language-dependent text</td><td>Opens the clean language-dependent text dialog, where you can enter missing translations and delete unused captions, see Clean Language-Dependent Text Dialog on page 67.</td></tr>
</table>

Functions of the Edit Tab on the Menu Bar

MENU ITEM	FUNCTION
Import object	This function allows you to import complete modules or components to Web Designer using the standard Windows file import dialog. The function recognizes alone whether complete modules or components are being dealt with and imports them as such. In all other cases the selected file is loaded as a project file.
Related applications	Opens the related application dialog, where you can insert, delete and edit applications, which are shown in the web portal's menu bar. For any application you have to define a clear name, the display name, the description, the basis URL, the parent application (for a direct entry in subforms of one main application) and a picture (optional).
Start recording a test script	Starts recording a test script, see Test Scripts on page 31 and Scripting in the Web Designer on page 51.
Stop recording	Stops recording a test script.
Check model integrity	Checks references, function calls and other details of the current web project. Any errors found do not result in compiler errors but should be corrected for the sake of the web applications's stability.
Check accessibility	This function determines whether a web project meets the basic requirements for a readily accessible web application in accordance with the current standard. Examples of deficiencies in this regard include missing alternative texts for images or the use of mouse click actions that cannot be realized using a keyboard shortcut. Insofar as the web project contains the function for such items, the attendant list will appear in the output window along with the relevant warning. Breaches of accessibility rules do not engender compilation errors.

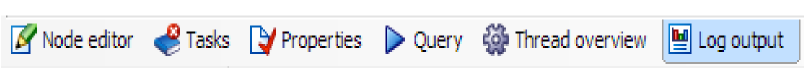
View

Functions of the View Tab on the Menu Bar.

MENU ITEM	FUNCTION
Restore standard layout	Restores Web Designer's initial window size and views.
Restore saved layout	Restores your own saved window size and views.
Save layout	Saves your own window size and views.
Object state	Brings up the object status window and places it in the foreground (see Object State on page 38).
Project browser	Brings up the project browser window and places it in the fore-ground (see Project Browser on page 34).
Preview	Brings up the preview window and places it in the foreground (see Preview Window on page 31).
Node editor	Brings up the node editor window and places it in the foreground (see Node Editor on page 40).
Tasks	Brings up the tasks window and places it in the foreground (see Tasks on page 41).
Properties	Brings up the properties window and places it in the foreground (see Properties on page 43).
Query	Brings up the query window and places it in the foreground (see Query on page 44).
Log output	Brings up the log output window and places it in the foreground (see Log Output on page 46).
Thread overview	Brings up the thread list window and places it in the foreground (see Thread Overview on page 47).
Call stack	Brings up the call stack window and places it in the foreground (see Call Stack on page 48).
Command list	Brings up the command list window and places it in the foreground (see Command List on page 49).
Compiled objects	Brings up the compiled objects window and places it in the fore-ground. All compiled web projects are shown in the window. A dou-ble click on a web project, component or module represents the selected object in the definition tree window and can be edited.
Bookmarks	Brings up the bookmarks window and places it in the foreground.
Navigation	Brings up the navigation window and places it in the foreground (see Navigation Window on page 27).

You can also toggle between windows using the tab bar tabs insofar as the various windows are dis-played and share the same frame.

Tab Bar



Toolbar

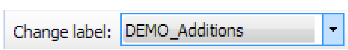
The Web Designer toolbar is currently non-configurable.

Web Designer Toolbar

Change Label

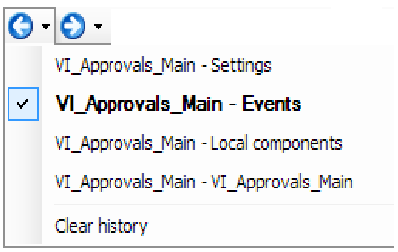
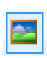
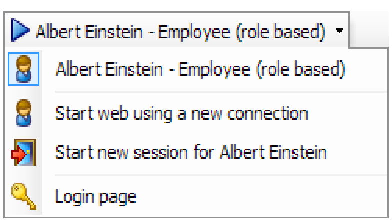
All changes that are effected in Web Designer can be saved under a change label, which makes it easy to export modified web projects to another database (for example from the development environment to the QS environment database). For information on how to use change labels, see Change Label for Web Project Dialog on page 69 and “Working With Change Labels” of the *Configuration Guide*. For further information concerning database exporter, see the tool Database Transporter, in the *Getting Started Guide*.

Change Label Functions in the Toolbar

ICON	FUNCTION
	Selection of a change label Changes in the Identity Manager database are stored under the change labels that are selected here.
	Change label administration This function opens the standard dialog for inserting, editing, or deleting change labels. For information on how to use change label, see “Working With Change Labels” of the *Configuration Guide*.
	This function sets the currently selected change label as a default label that will be selected automatically when Web Designer is restarted. The selection is client-specific and has no impact on other Identity Manager database users.
	Opens a dialog that allows you to assign a selectable change label to all current web project-referenced Web Designer and/or caption objects. This function also allows for the export of complete web projects, regardless of any previously assigned change label (see Change Label for Web Project Dialog on page 69).

Other Toolbar Functions

Other Toolbar Functions

ICON	FUNCTION
	Opens the captions dialog, where you can insert, delete and edit captions (see Captions Dialog on page 65).
	When you modify a database object, the Save button is activated. Changes are only saved for the database object that is shown in the currently active tab. If you wish to also save changes in other database objects, it is necessary to re-launch the saving process for each such object.
	The Save all button save any database objects with changes.
	One node back / One node forward Shows the course of the previously selected objects. You can configure the maximum length of elements in the global settings dialog.
	The drop-down list displays all web projects in the relevant Identity Manager database for selection purposes. Only the modules that are part of the web project selected here are compiled for the preview window.
	Opens the preview window. The preview will be calculated for the currently selected person.
	Clicking the Preview button opens a drop-down menu, where a person must first be selected. The permissions of the selected person form the basis for preview window rendering. To close the current session and launch a new one, click Start new session for... in the menu. This action has the same effect as the current user logging off and then logging on again. To close the current session and launch a new one, click Start web using a new connection. To configure the login page, click Login page.

Navigation Window

In normal view, the navigation window is on the left edge of the screen and is subdivided into the following areas:

- Modules
- Components
- Web projects
- Project files
- Form types
- Layout definitions
- Test scripts

The basic procedure here is to select in navigation window the database objects that you wish to work on using Web Designer. Various functions are available in the definition tree window, depending on the area selected.

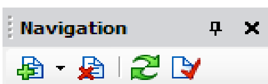
Database objects are displayed in a tree view that integrates three root nodes under which the Quest-defined database objects are grouped. These objects cannot be edited in an Identity Manager customer installation, but they can be copied using a wizard (see New Project Wizard on page 72). Each second root node has a custom prefix, under which all customer-specific objects are grouped.

Hence on initial installation of the program, these root nodes have no subordinate objects. Any modules or other elements that are added to a customer installation are automatically grouped under these root nodes.

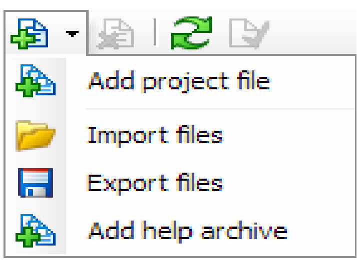
The objects you have already edited are displayed in the first root node with the prefix “Recent files”. This displays objects that you have changed but not saved.

You have the option to search for data objects by entering part of an object name in the search box under the navigation toolbar.

Navigation Window Toolbar



Functions in the Navigation Window Toolbar

ICON	FUNCTION
	Adds new modules, components, project files, form types or layout definitions. Clicking the arrow button opens a context-sensitive dialog. New web projects cannot be added using this function. The adding of new modules can be done using the relevant wizard (see New Module Wizard on page 76). Files can be exported to the database through the Import files option and can be exported from the database through the Export files option. These options open a standard Windows dialog to upload and save files. There is a possibility to create a help archive for a web project.
	Selecting a database object in the navigation window activates the Delete button. To delete a selected database object from the database, click the button and the confirmation prompt. Deletions can not be undone.
	Reloads database objects from the relevant area. This is a particularly useful function for situations where two or more developers are using the same database.
	Opens the object properties dialog, which displays key properties of the database object that is marked in the navigation window (see Object Properties Dialog on page 68).

Modules and Session Modules

A module can be regarded as a logically self-enclosed unit within a web project. Hence a module can only contain one web page such as a home page, with a brief introductory text, but a module can also integrate an extensive workflow that extends across a number of web pages (for example for product orders).

Well structured and carefully planned architecture of the desired functional scope of a web application in a module reduces the extent of any debugging that may be necessary later on and makes it easier to implement add-on functions.

Although there is no limit on module size or number, you need to define a session module for each web project. A module becomes a web project session module insofar as the relevant option is set in the module’s root node and the module is defined as a session module in the web project.

When a session is opened, a session module is loaded and the module and its data remain in existence throughout the session lifecycle. The session module must contain information concerning the logged on user. It is also crucial that a session module contain:

- Collections named User
- UserRuntimeModule
- UserMenu
- UserLanguage.

Proprietary user groups concerning elements such as a user’s affiliation with an organizational unit can be defined for the web application. This affiliation, which must be notified once to the web application in the session module after the user logs on, remains available throughout the session for toggling menu items or form sections on and off. Modules contain the definitions of the nodes that are visible in the browser, in addition to other important information. However, session module content is never rendered in the browser. Hence Form nodes are also hidden in cases where a module is marked as a session module.

Components

Components are module elements that have been exported from a module, which, to this end, contains a reference to a component at the relevant place. A component can be referenced at any place in a module as well as in an unlimited number of places across a series of modules. By the same token, a component can also be rendered iteratively in the browser. In the interest of ensuring that the various instances of an iteratively rendered component can be reliably differentiated from each other, each instance is given an automatically generated suffix as at runtime “R”. In such cases, all nodes of the same instance have the same suffix.

Thus components keep the definition tree free of redundancies. This in turn reduces programming work and ensures that the web application will exhibit a uniform appearance and functionality. A distinction is made between various types of components. Component position in the definition tree determines which type of component can be referenced at any given time.

The components are listed by the following criteria in the navigation view to give you a better overview:

- Object dependent references
- Table dependent references
- Column dependent references
- Other components

Web Projects

Multiple web projects can be defined in the Identity Manager database. To assign a web project to a web application that is on the web server, use the web application configuration file (see *Web Portal Installation Guide*).

It is necessary to manage/update a series of specific data in the definition tree window (see Definition Tree Window on page 50). New web projects cannot be added through the navigation window menu bar. Such projects are created through the Create New Project wizard (see New Project Wizard on page 72).

Project Files

Project files are any type of files that are needed for a web project and thus should also be created in the Identity Manager database (for example additional icons). All relevant files that have already been uploaded to the Identity Manager database are shown in the Project files area. You can, for example, replace images already loaded in the database by project files. For more information, see Ersetzen von Bildern aus Ressourcen-Dateien on page 134.

Form Types

Form types define the structure of the rendered HTML pages. In older versions of the web designer, the form types were a directly part of the HTML tree of a web project and could be edited. In the navigation tree, there is now a separate area Form types. The form-definitions should be kept as separate DialogAEDS objects in the database. The form types describe the available form parts and their positions in the HTML tree.

The standard system contains four different VI-defined objects as templates. In order to edit the form types, you have to create an object copy (see Copy Objects Wizard on page 75). Each form type consists of two form parts, title and main, and different output nodes. In the definition tree window you can add form parts and output nodes to the customer-specific form types (context menu). This can be edited in the node editor window.

Layout Definitions

Layout definitions describe how rendered display nodes are visualized in the browser (node of type view). There is at least one layout defined for each of the display node types that can be provided with layout information. Layouts with the ending “default” are used by the associated node type without having to be entered on individual nodes.

Every object in the layout definition list corresponds to a css class. Any number of new layout definitions can be set up. Each layout definition is linked to a node type. This layout definition is therefore available for each node of the corresponding type. If a default layout definition should replaced by a layout definition then a wizard is used (see Copy Objects Wizard on page 75).

Layout definitions are separated by type and group together to give you a better overview in the navigation view.

Test Scripts

Test scripts are displayed under the heading Test scripts in the navigation window. Test scripts are recorded using the Edit | Start recording test script option. Test scripts can be used for testing web projects after they have been changed or for load tests on the web server. It is necessary to use a particular tool for running load tests.

If a test script is going to be run repeatedly automatically, recording must be started from the web project home page. For further information see Scripting in the Web Designer on page 51.

Preview Mode, Project Browser and Object State

Normal view integrates a frame (on the right edge of the screen) containing the Preview and Project browser functions, whose dialogs you can toggle between using the tabs at the bottom of the frame. These two dialogs can only be displayed following the successful compilation of the selected web project.

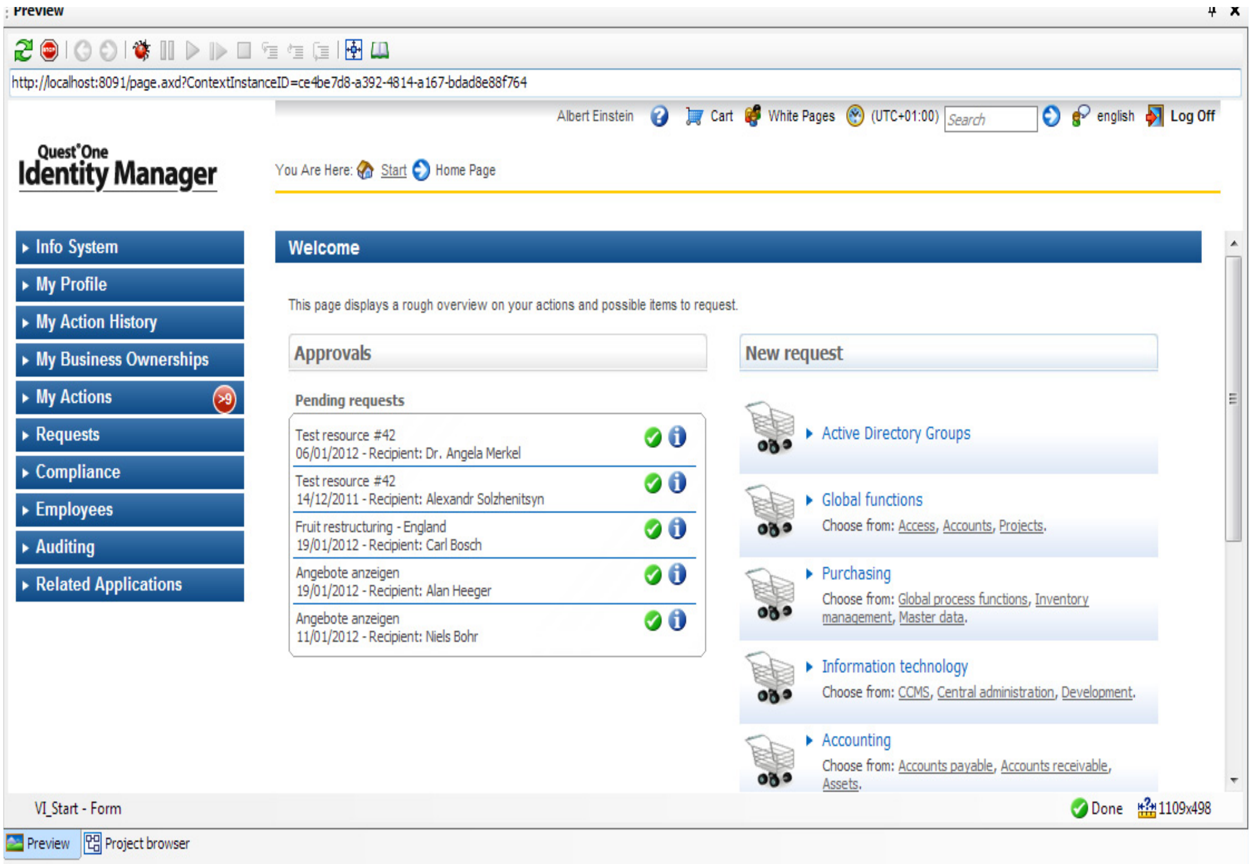
Clicking in the menu bar View | Object state displays the object state dialog in the aforementioned frame as well.

Preview Window

The preview window shows how a specific web application will look and respond in the browser. This means that the preview window can be used to test out the entire functionality of all buttons, links and so on. The displayed view is rendered using Internet Explorer, which is installed in the current client.

Note that other browsers or other versions of the same browser may generate views that differ from the preview. Hence it is advisable to perform an extensive layout test of the web application with the previously predefined browser versions.

Preview Window following successful Compilation

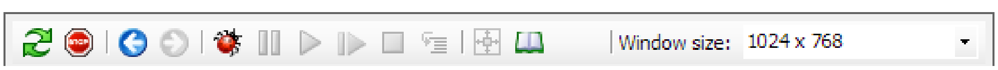


A preview window can only be generated insofar as (a) the user does not log on to Web Designer through a system account or (b) a person is selected following log on through a system account. This is necessary due to the fact that when a view is rendered, Web Designer assesses person-related dependencies (for example which person is allowed to view which menu item) and displays the results of this assessment. This means that a realistic function test can only be realized in the preview window using such a person-related display.

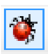
The preview function also writes data to the database or deletes database objects if this function is activated by the user. Hence caution should be exercised when testing this function.

The preview window integrates a toolbar of its own, most of whose functions are used to manage debugging mode. When the debugger is activated, program functions are realized step by step rather than dynamically, thus enabling you to see which program step is actually being realized and the results yielded by the step. In addition, with the debugger activated, you can navigate at will to the query or data schema window.

Preview Window Toolbar



Functions of the Toolbar in the Preview Window

ICON	FUNCTION
	Recompile This button, which is activated following successful compilation, allows for recompilation (for example after a change has been made in the definition tree window) without the need to select a person beforehand. The preview window is generated for the previously selected person. In this process, the Compiler factors in all module and component changes that have been effected thus far in connection with the selected web project. Such changes need not be saved beforehand.
	Suppress execution This function, which prevents any action from being executed in the preview window, is useful in cases where a specific definition tree node has been found in the properties window using the Show definition object function. When the function is activated, the object is underlined in blue.
	One node back / One node forward A detailed view of the course is not available.
	Enable debugger This field allows you to toggle the debugger on and off. “Debugger” is underlined in blue when the function is activated. All actions are executed in succession (for example one node at a time) when the debugger is deactivated. If the debugger status is deactivated in “Paused” state, click the Resume execution button to resume debugging.
	Single step The Single step button is only activated when the debugger is activated. When single step mode is activated (a state indicated by blue underlining), each time an action is realized the program stops at each executed step in the definition tree. When single step mode is deactivated, the action is realized until the next break point (break points are inserted in the definition tree at action nodes above the context menu).
	Resume execution This button is activated with the debugger deactivated insofar as any program steps remain to be carried out, for example no user entries are anticipated. Clicking this button launches realization of the next program step.
	“Slow Motion”-Debugging If this button is enabled, the program is automatically run through a step at a time. This mode can be switched on or off during execution. This saves the user from having to click so often.
	Abort execution This button is activated with the debugger deactivated insofar as any program steps remain to be carried out, for example no user entries are anticipated. Clicking this button cancels realization of the current program step.
	Debug - step into The debugger realizes all steps until the next action node.

Functions of the Toolbar in the Preview Window

ICON	FUNCTION
	Demo mode The preview window switches to full screen mode and all other Web Designer windows are covered. The resolution defined for the preview window is disregarded here. This button is only active with the window in a docked state.
	View html source This button, which is activated following successful compilation, opens a popup window containing the html source code generated by Web Designer. This function allows for debugging directly in the source code.

In addition to a toolbar, the preview window also has a foot bar. The currently displayed Web Designer form is indicated on the left side of the status bar.

Functions in the Preview Window Status Bar

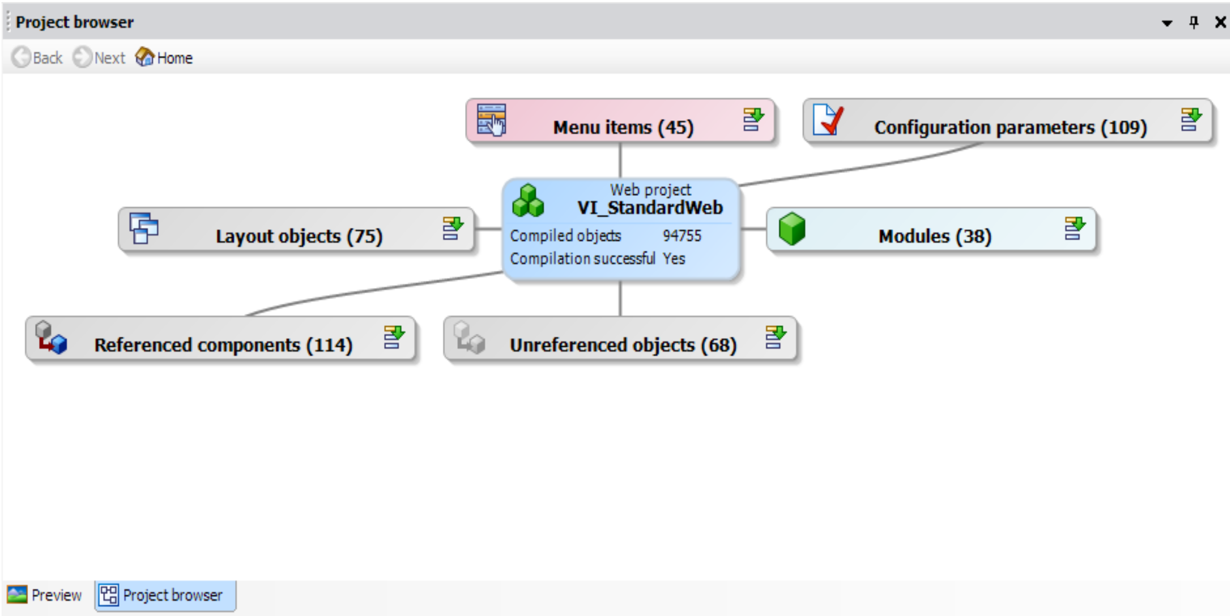
ICON	FUNCTION
	Status Done here means that compilation and rendering of the current project state have been completed and that entries can be made in the relevant pages.
	Pixel lines The pixel line count indicates the dimensions of the window area that are available for the display of rendered html code. This information obviates the need for scroll bars during programming in cases where the amount of available browser space is known. This in turn is affected by the screen resolution and the number of menu bars displayed in the browser.

Project Browser

The project browser window contains essential information concerning the interrelationships of the various database objects. The project browser view contains the selected object (for example the object whose information and dependencies are displayed) in the center of the window. One of the key project browser-related informational elements here is the reference list indicating, for example, which modules reference a specific component. After a module or component has been modified, this list allows you to find all places in the web application workflow that are affected by the change.

The display is realized through bubble sheets. The view in the project browser can be launched from a web project, module or component by (a) selecting a relevant object in the definition tree window (tab is activated) and (b) calling the function through View | Project browser in the menu bar or activating the button. Each bubble entry is rendered as a link. Depending on the type of object referenced, one of two things happens next: the object in question is marked and displayed in the definition tree window (for example for nodes with buttons or labels, or also for components) or the project browser navigates to the selected object (for example for collections of modules). The bubble lists can be opened or closed using the button in the upper right hand corner. The screenshot below shows characteristic object type results.

Specimen Project Browser View for a Web Project

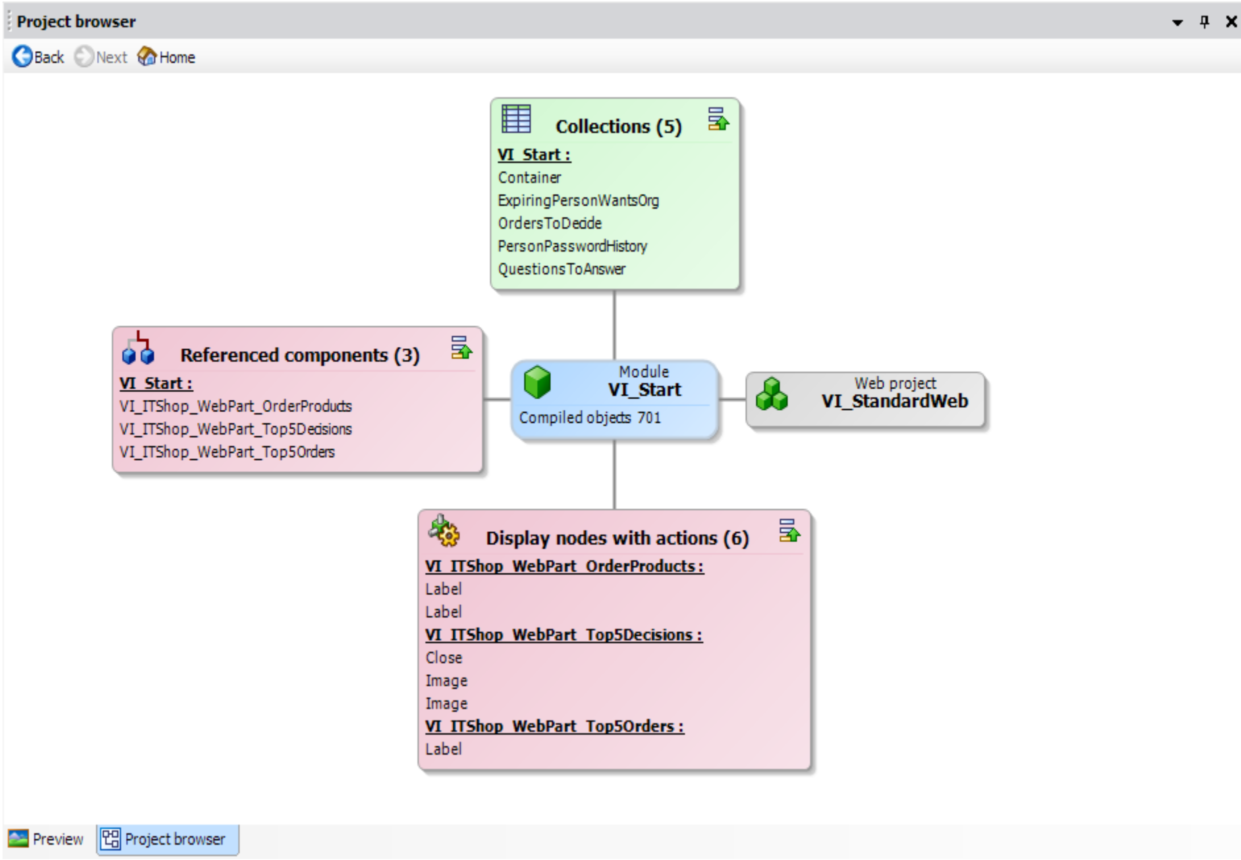


The project browser view contains the following web project information:

- Referenced components
  List of components that are being used for the web project, for example components that are referenced in any way.
- Modules
  List of modules that are being used for the web project, for example modules that are referenced in any way.
- Unreferenced objects
  List of modules and components that are not being used for the web project, for example that are not referenced.
- Menu items
  List of all web project menu items.
- Configuration parameters
  List of all configuration parameters that are being used for the web project.
- Layout objects

Lists the layout objects that are used in the web project.

Specimen Project Browser View for a Module

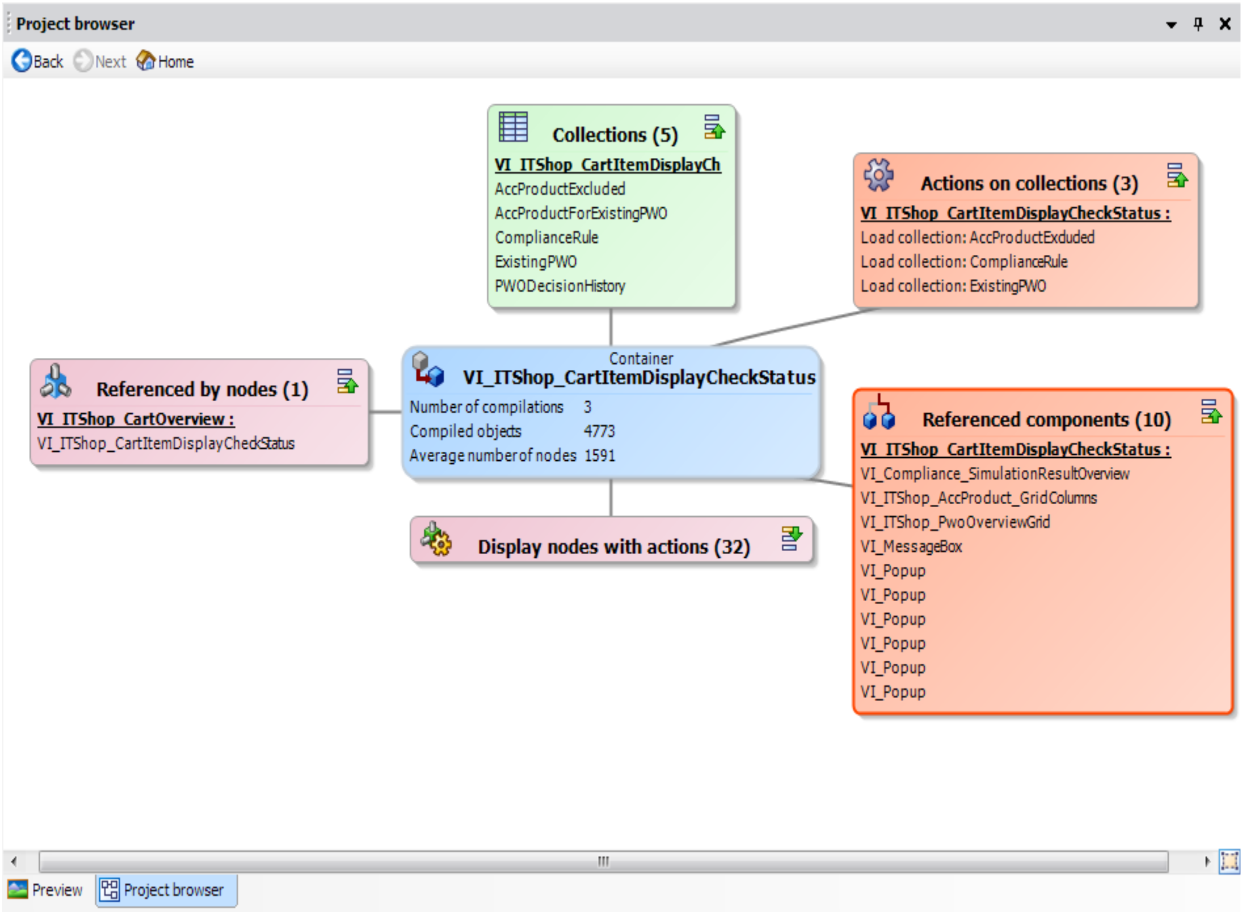


The following information is indicated for each module:

- Web project
  List of web projects that reference the module (a module can be used by more than one web project).
- Collections
  List of all collections that are defined in the module.
- Display nodes with actions
  List of all components that are defined in the module and the referenced components and that are associated with an executable action.
- Referenced components
  List of all components that are directly referenced by the module.
- Elements referenced by a node
  List of all menu items and redirecting elements that point to the module.
- Functions

List of all functions that are defined in the module.

Specimen Project Browser View for a Component

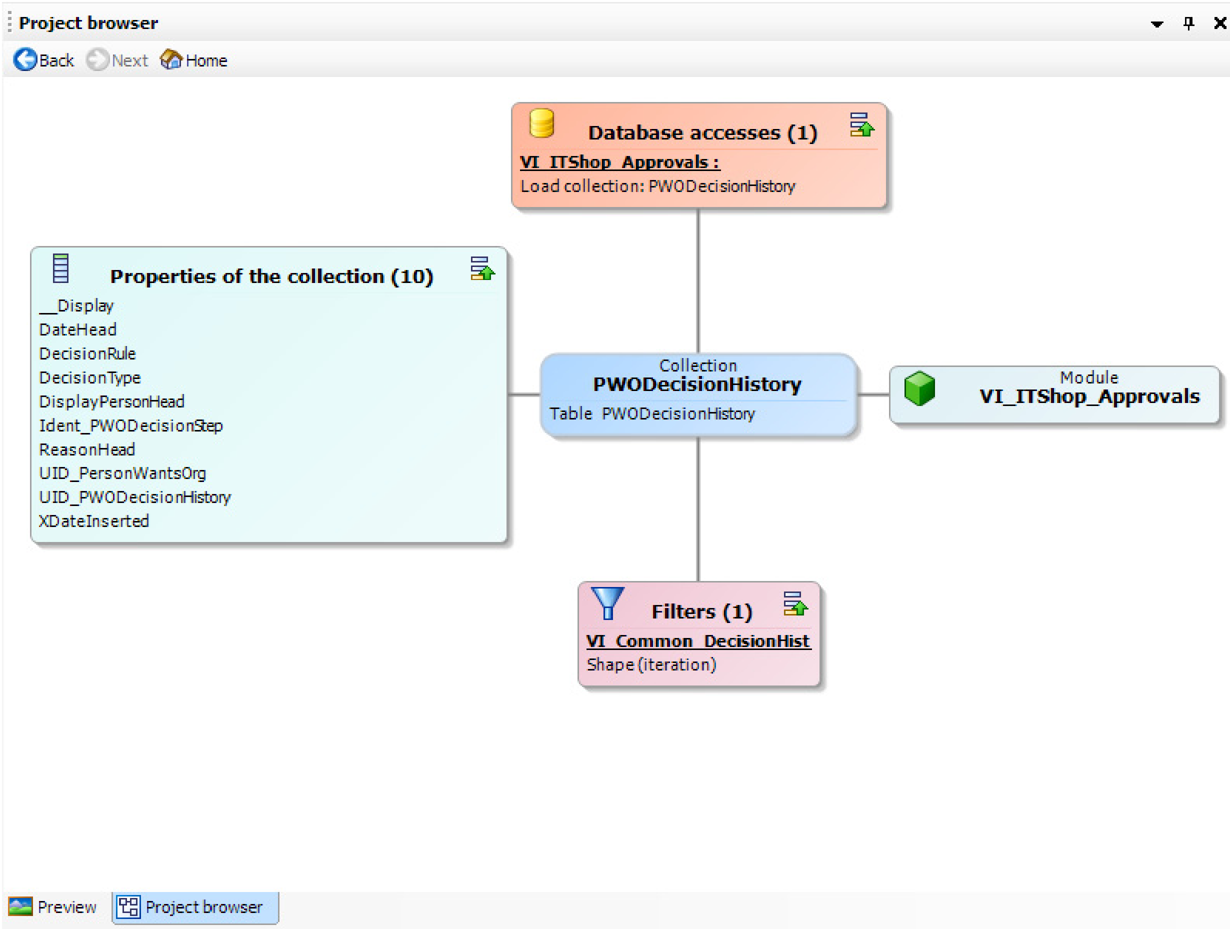


The following information is provided for a component, insofar as the information is available:

- Display nodes with actions
  List of all components that are defined in the module and the referenced components and that are associated with an executable action.
- Referenced components
  List of all components that are directly referenced by the component.
- Collections
  List of all collections that are defined in the component.
- Actions in collections
  List of all actions (insert, update, delete, fill, load, save) that are realized in these collections.
- Referenced by nodes
  List of all nodes that directly call up the component.
- Indirectly referenced by nodes
  List of nodes that call up the component indirectly, for example through an intermediate step

realized by at least one other node.

Specimen Project Browser View for a Collection



The following information is indicated for a collection:

- Module
  Module in which the collection is defined.
- Filters
  List of all nodes that query this collection.
- Properties of the collection
  List of all properties of a “Data base object” collection that are used in the module or a list of all properties that have been defined for this collection (applies to all other collections).
- Database accesses
  List of all nodes that can be loaded using this collection.
- Operations
  List of all operations (insert, update, delete, save) that relate to this collection.

Object State

The object state window contains a list of all modules, components, project files, form types, layout definitions and web projects that are currently in the computer’s memory. The following columns are shown:

- Object identifier

  This column contains the name of each relevant object.
- Don‛t compile

  This column contains a check box for each “Module” object. Checked modules in this column will be disregarded by the Compiler.

This option should be used in cases where only a handful of modules is to be processed, since this will reduce computing time. The status shown is local only and is only available during the session.

- DB status

  This column shows the results of the comparison of the status of the objects stored in the database and the locally loaded objects. “Up to date” status indicates that there is no difference between the two versions. The “Changed by ...” status indicates that a modified version is available in the database.
- Local status

  This column indicates whether the loaded objects have been modified. If modified objects are saved and the object state window view is then updated, these objects will revert to being marked as unchanged. Objects whose status is changed can likewise be reloaded from the database. In such a case, the changes that were realized locally will be discarded.

Objects whose status is “Changed by ...” and that were modified locally can no longer be saved. This database function prevents saved changes from being overwritten by re-saving of an old version of the object.

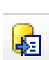
- Reload

  The check box in the “Reload” column is used to define whether the new development state should be exported to the local version. To begin the loading process per se, click the reload button. The local version will then be overwritten.

The object state window integrates an icon bar of its own containing three buttons.

Object State Window Toolbar

Functions in the Object State Window Toolbar

ICON	FUNCTION
	Refresh list Clicking this button updates the list in the object status window. This list is not updated automatically.
	Reload selected objects Clicking this button reloads the object whose check box is activated in the “Reload” column.
	Select all changed objects for reload Clicking this button activates all check boxes in the “Reload” column.

Node Editing, Tasks, Properties and Queries

At the bottom of normal view is a window for node processing and related functions, which you can toggle between using the tabs at the bottom of the window.

Node Editing Tabs

Node Editor

The node editor window allows for editing of the properties of a node that is selected in the definition tree window. Each type of node integrates a specific set of properties, which are listed in the node editor window. Hence the items on this list change in accordance with the type of node selected.

Normally, the properties listed first are those that are specific to the selected type of node (for example for labels such as text). A description of these properties can be found in Web Designer's context-sensitive help files. Following this, you have the option to enter a condition (insofar as one is available for the type of node selected), which normally is a visibility condition. However, there are other types of conditions which in some cases are available for one type of node only (for example "Active state condition" at the button node). These conditions must be formulated using Web SQL syntax, and the value True or False must be entered.

The conditions have differing effects depending on the type of node selected. Inasmuch as nodes can be subordinated to each other, the conditions are also hierarchical. Thus a display node will only be shown if all conditions for the superordinate container are True.

Types of Conditions

TYPE OF CONDITION	DESCRIPTION
Visibility condition	Visibility conditions occur only in certain display nodes such as containers, tabular columns, and tabular cells. These elements are only displayed if the result of the SQL expression is True.
Execution condition	Execution conditions occur solely at action nodes such as initialize and action. The action is only carried out if the result of the SQL expression is True.
Activate state	An active-state condition only occurs at button nodes. A button is only activated if the result of the SQL expression is True. Inactive buttons are grayed out and actions that are associated with the button are not performed, regardless of their execution conditions.
Condition for skipping this level	This condition is implemented at level nodes, which means that hierarchical levels can be foregone if they are not absolutely necessary. Example: Levels A, B and C are defined for a grid. If only one level B data set exists under a level A, the level C data sets are displayed directly below level A. In order for this to occur, the result of the entered condition must be True.

The lower portion of the list contains the properties that may potentially affect the browser display of the relevant node. Hence properties are only available in this area for display nodes. In addition, no display node contains the complete list of properties that is shown in the following table.

Node Processing and Display Properties

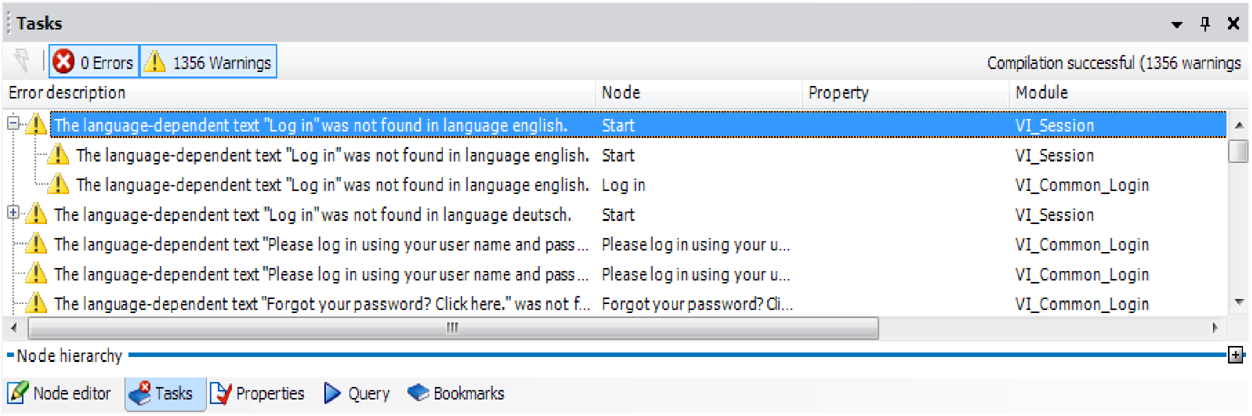
TYPE OF CONDITION	DESCRIPTION
Identifier	The entry in the identifier (optional) field is displayed at the relevant node if the hidden optional identifier in definition tree option is activated in the Global Settings dialog (see Connect on page 20). If an optional identifier is selected, it is helpful for orientation and for finding specific definition tree sections. At the same time, identifiers allow for unambiguous identification of a node and must therefore be unique within a document. This unique identification is necessary in cases where the web application is to be controlled solely through keyboard shortcuts. For more information in this regard, see the help files.
Layout	A defined layout object for the nodes can be selected here. The layout definitions that can be selected must be created first in the navigation window in the "Layout definitions" view. If no entry is made here, the default layout is used for the corresponding node. The default layout for a node type contains the name of the node type and the ending "default".
Style rules	Here, style rules can be defined for individual nodes. These rules overwrite the entries for the configured layouts – but with one exception. If "Inline element" is defined as container layout, the browser cannot implement the defined style elements owing to the properties of the CSS programming language. Any CSS rules can be applied in <css rule> or <css rule> syntax.
Tooltip text	The browser will display the texts entered here insofar as the mouse pointer is in the environs of the relevant control.
Alternative text	This property is available for image only. The texts entered here are read out by suitable software if the relevant image cannot be loaded.

Tasks

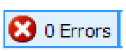
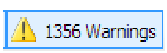
The task window contains a list of compiler errors as well as warnings. Compiler errors preclude successful web project compiling and must be eliminated. Non-compilable development states are not to be approved under any circumstances (see Edit on page 23) since no functional web applications are available in them. Compiler warnings relate to missing captions, or to messages concerning accessibility. If compiler warnings are the only type of message that is generated, the development state will be successfully compiled nonetheless.

If individual messages are displayed hierarchically, it means that the errors indicated occur at various places in the web project. This can happen, for example, in connection with a missing caption that is referenced by a number of nodes. In such cases, the error need only be eliminated once. To call up the relevant node in the definition tree window, double-click a table entry. Web Designer will then automatically go to the node editor window.

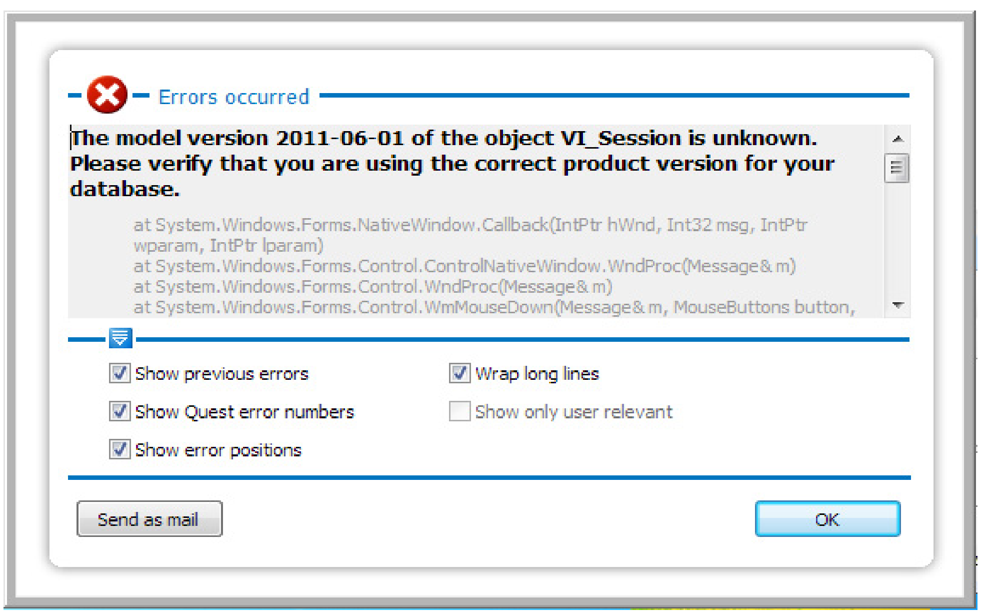
Tasks



Functions in the Object State Window Toolbar

ICON	FUNCTION
	Show error message This button is only active when an error message has been highlighted in the list. By clicking this button a detailed error description is displayed.
	Errors Clicking this button, errors can be switched on or off.
	Warnings Clicking this button, warnings can be switched on or off.

Detailed Error Description

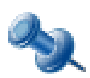


Properties

The functions in the properties window enable you to call up key information regarding specific nodes. In order for the properties window to be used, the web project in question must first be successfully compiled, whereupon the properties window can be used in standard mode or debug mode.

The properties of the nodes currently marked in the definition tree window are shown when the debugger is activated, as well as when it is in "Paused" state.

When the debugger is deactivated, the properties of the node that is currently selected in the preview window are shown.



This is only possible with display nodes, since only they are displayed in the preview window. For display nodes that are associated with an action, Suppress execution must also be clicked. Data and action node properties are only displayed in debug mode.

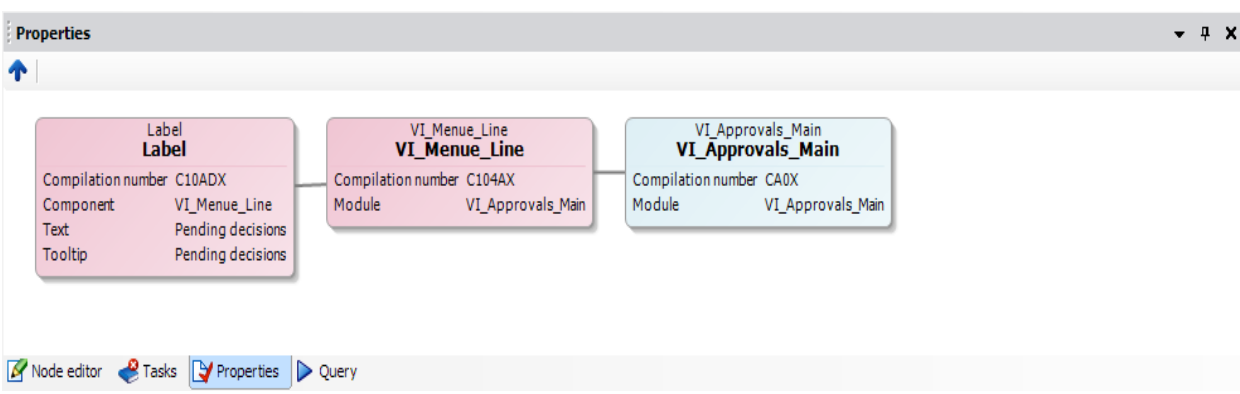
The bubble of the currently selected node shows the node compiling number (a sequential number defined by the Compiler) and the results of all expressions that are defined in the node. The bubble of the currently selected node also contains the following functions:

- Show definition object
  To mark the relevant node in the definition tree window, click in the upper portion of the bubble.
- Copy to query window
  To copy an expression to the query window, click the relevant expression, which will then be copied to the window automatically.

Functions in the Properties Window Toolbar

ICON	FUNCTION
	Go to parent node Clicking this button will display the properties of the node that is directly superordinate to the current node. To also mark this node in the definition tree window, click the node bubble.

Properties



Query

Web SQL statements can be formulated and implemented in the query window. However, in order for this to occur the compilation must be successfully completed. This window integrates a toolbar of its own, which is divided into the following three sections:

- On the left, the currently available session collections are shown, in such a way that the collections are bundled under the modules or components in which they were originally defined. To view the contents of a collection, right-click on collection and then select Show collection data. Collections are available for the following: the module currently shown in the preview window, the components referenced by this module and for the session module, which remains in existence throughout the session lifecycle. Collections that have not been loaded or filled are listed but have no content.
  Right-click on objects opens a context menu with the following additional options: Show definition object, Show last used WHERE clause, and Show in project browser.
- The area at the upper right is used to enter a statement that is to be implemented.
- The area at the lower right is used to display query results.

Functions in the Query Window Toolbar

ICON	FUNCTION
	Send request Clicking this button implements a Web SQL statement that has been entered. If more than one such statement has been formulated, mark the statement that is to be implemented (it will turn blue). Inasmuch as the exact marked area will be implemented, be sure that each valid statement is marked in its entirety. Only one statement at a time can be implemented.

Web SQL syntax is to be used when entering statements that are slated for implementation. The following application domains come into play here:

- Direct querying of the database
  Syntax example: exists(“person”, variable(isnull(getconfig("VI_Dashboard_Shop_Filter“), “1=1”))) – Output: true or false
- Computing of the effective Where clause
  For example, the Where clause that is to be implemented by the SQL server. Syntax example: format("select salutation from person uid_person = '{0}'", select uid_person from user) – Output: select salutation from person where uid_person = '923b9a92-beb5-4cc9-8f1f-7e6e4159debe'
- Querying of the effective Where clause of a collection (valid solely for database object collections)
  Syntax: whereclause(<collection name>) – Output: select salutation from person where uid_person = '923b9a92-beb5-4cc9-8f1f-7e6e4159debe'
- Querying data set collections
  Syntax example: select uid_person from user – Output: 923b9a92-beb5-4cc9-8f1f-7e6e4159debe

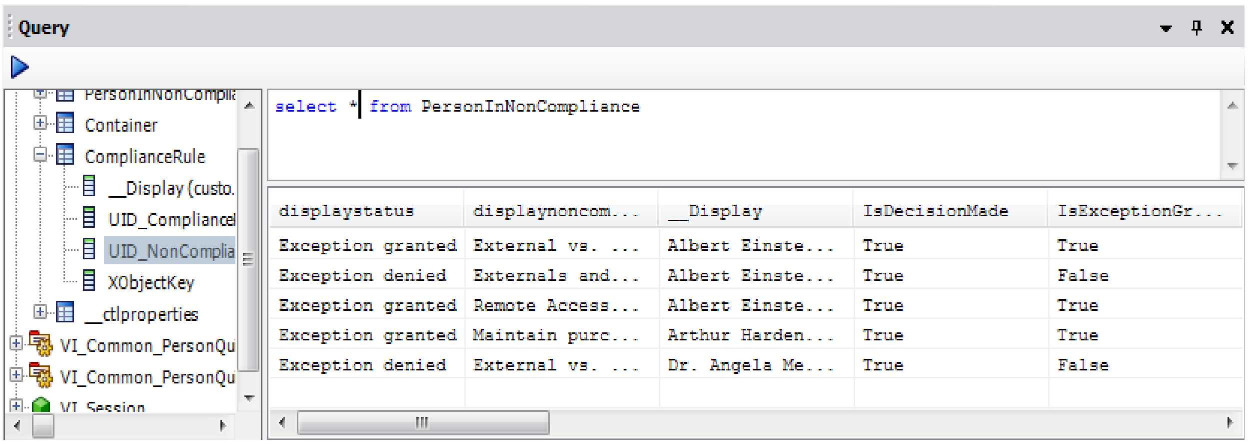
If the queried collection is a part of a component, the suffix displayed on the left side of the query window must be added to the collection name.

Web SQL syntax supports queries that are based on a “current” data set (select current...), which is indicated by a cursor. In order for such queries to be realized correctly in this window, the cursor must be defined through the Usingcursor () function. Following this, export a cursor name from the properties window by clicking a relevant node in the preview window; the cursor name may also be entered (or edited) manually. In this context, the hexadecimal characters to the left of the tilde represent the collection, and the characters to the right of the tilde represent the line. Two or more cursors can be attributed to one line. This means that a line in the third level of a grid has three cursors, which are separated by commas in the text field. A query can be carried out without using all cursors.

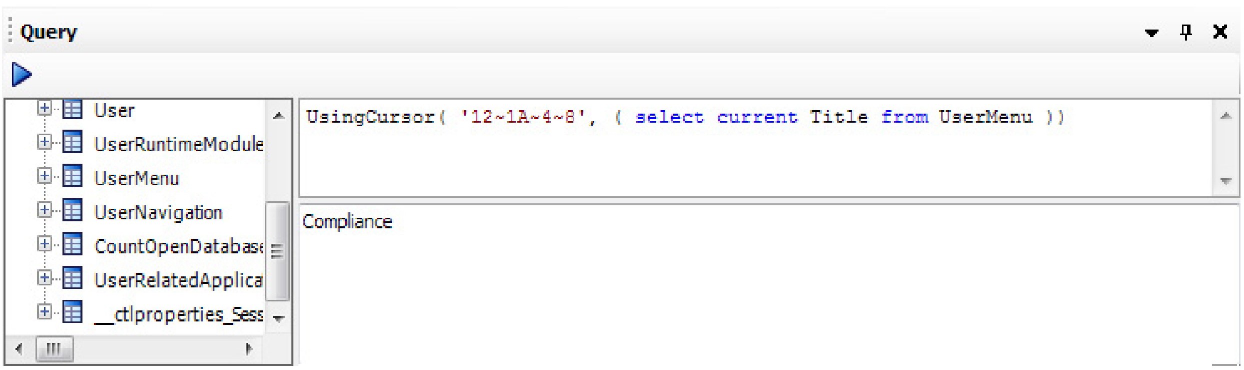
If an asterisk is inserted in a statement as a placeholder for the columns that are to be displayed, only those columns will be displayed in the module that are needed for (for example referenced by) the module.

If the dispatched SQL statement yields a result, it will appear in the lower part of the window.

Simple Query in the Query Window



Two-Cursor Query in the Query Window



Additional Functions

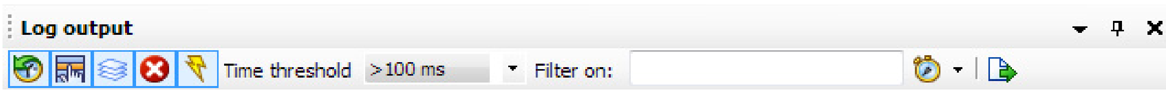
The functions described in this section support debugging. The debugging window is not opened in normal view for reasons of space, but it can be opened by clicking View in the menu bar.

Log Output

The log output window lists all current session log entries. Session logs are available solely in the client memory on the web server it was started on. They are discarded once the Web Designer is closed. The entries that are to appear in this window can be defined though the log output window menu bar.

Note: The log file on the web server, which is saved as a text file, is referred to as an application log. For further information in this regard, see Log Files on page 105.

Log Output Window Toolbar

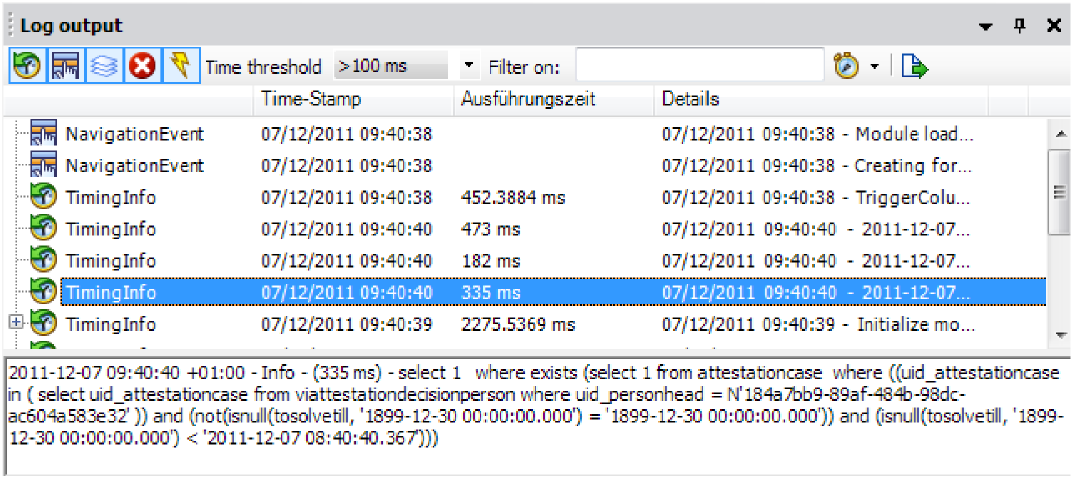


Functions in the Log Output Window Toolbar

ICON	FUNCTION
	Timing info Allows runtime-related log entries to be shown or hidden.
	Navigation event Allows log entries that are related to navigation results to be shown or hidden.
	Application info Allows log entries that are generated by a runtime environment to be shown or hidden.
	Unhandled exception Allows log entries that are related to an unhandled error to be shown or hidden.
	Log entry Allows log entries that are written by a log node to be shown or hidden.
	Filter threshold value Selecting a filter threshold value allows for the filtering of runtime-related log entries. For example, if the defined threshold value is > 1000 ms, only log entries will be shown for actions whose runtime exceeds 1 second.
	Filter The log entries are filtered relative to the entries in this text box.
	Filter session Only the log entries for one session are displayed. This button is used to swap between sessions. **Note:** Only one session can be started. The log entries for closed sessions are kept in client cache until the Web Designer is closed.
	Export log file This function allows for the export of all log entries or of only those entries that are visible in the log output window. The entries are exported to an xml or text file.

The following figure shows the Log Output window, whose upper section lists the log entries and whose lower section provides details for each marked entry.

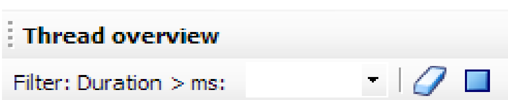
Log Output Entry



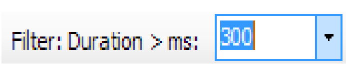
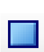
Thread Overview

The thread overview window shows the threads (http queries sent from clients to the in-house Web Designer web servers) of the current session. These threads are consistent with the filter criteria that are defined in the menu bar.

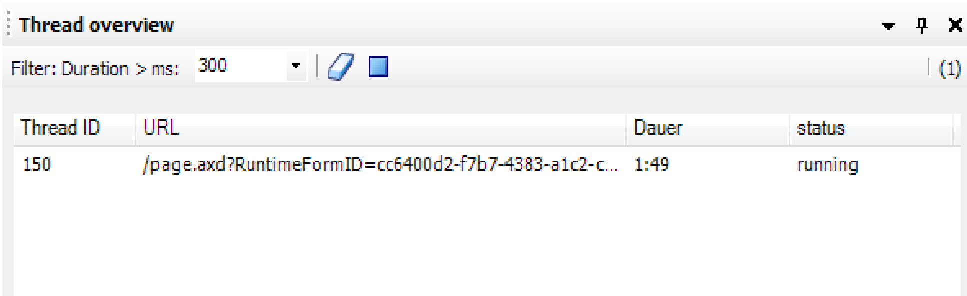
Thread Overview Window Toolbar



Functions in the Thread Overview Window Toolbar

ICON	FUNCTION
	Filter threshold value Selecting a filter threshold allows for the filtering of closed threads. For example, if a threshold value of > 300 ms is selected, only those threads will be shown whose implementation took more than 300 milliseconds.
	Clear all completed threads from the list. Deletes all completed threads from the list.
	stop selected thread This function allows for the retention of an ongoing thread. This function is useful in cases where an endless loop has been generated.

Thread List Window

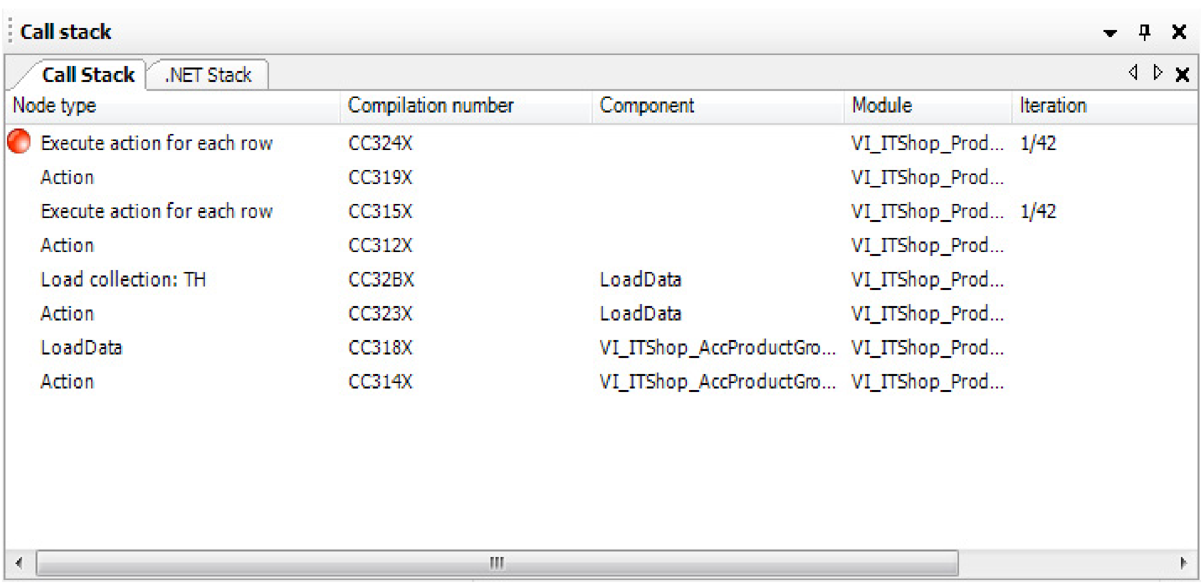


Call Stack

The call stack window contains a list of internal action calls when the debugger (see Preview Window on page 31) is in Step-by-Step mode. The list can be viewed on the Call stack tab.

To mark the relevant node in the definition tree, double-click a line. The underlying .Net calls are shown on the .Net Stack tab. This information is needed in cases where the cause of a faulty function is more serious than a faulty node definition in the definition tree.

Web Designer Call Stack Window

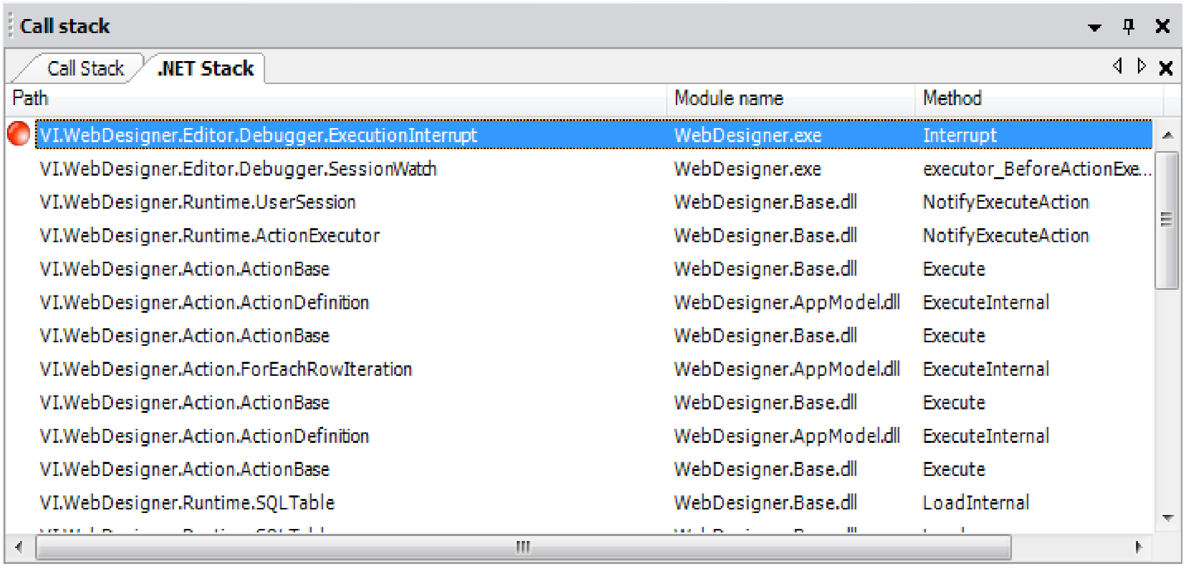


The call stack window contains the following information:

- Node type
  Type of the current node.
- Compilation number
  Compilation number of the current node.
- Component
  If the current node is defined in a component, the component name will be shown here.
- Module
  Name of the module in which the node was defined or that the component references.
- Iteration
  If the current node type is implemented as an iteration, the number of loops realized thus far

and the aggregate total of all loops will be shown in this column.

Web Designer .Net Stack

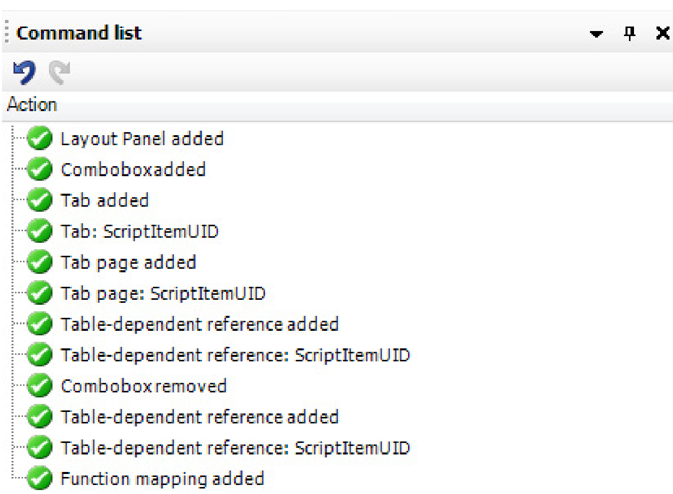


The .Net Stack window contains the internal Call Stack of the .Net runtime environment. This information can be useful for error troubleshooting and will be requested by the Quest support department if necessary.

Command List

The command list indicates all changes that have been implemented for the active object that is currently in the definition tree (see Definition Tree Window on page 50). A separate command list is generated for each object in the definition tree window. The menu bar buttons (or the CTRL + Z / CTRL + Y shortcuts) allow commands to be undone or repeated (Undo / Redo). Implemented commands are flagged with the icon in the list. This icon is missing for commands that were undone.

Command List



The use of wizards allows for automatic implementation of numerous commands, which in the command list are displayed as composite commands and the individual commands are shown at a second level. However, these commands can only be undone through the composite command.

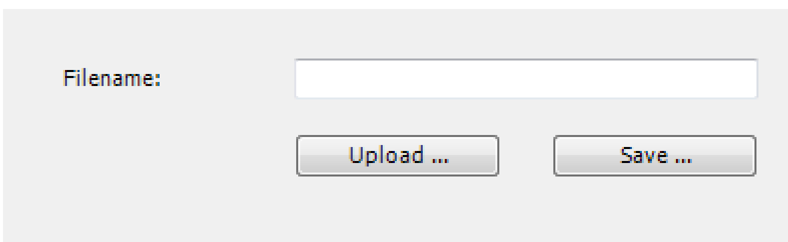
Functions in the Command Window Toolbar

ICON	FUNCTION
	Undo Undoes the last implemented command on the list.
	Redo Redoes the last implemented command on the list.

Definition Tree Window

The definition tree window shows in detail all of the modules, components, web projects, project files, form types, layout definitions, and test scripts that are listed in the navigation window. To this end, a tab is opened in the definition tree window for each object selected in the navigation window (through a double click). For web project, module, and component objects, the definition tree is edited in the definition tree window, for example nodes are inserted or deleted. Individual nodes are then edited in the node editor window (see Node Editor on page 40). For project file data objects, in lieu of the definition tree a dialog opens where the files can be uploaded and saved. The definition tree has a toolbar of its own.

Definition Tree Window / Project Files: File loading and saving Dialog

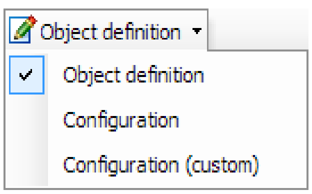


Definition Tree Window Toolbar

Functions in the Definition Tree Window Toolbar

ICON	FUNCTION
	Settings Opens a dialog window where the configuration keys that are defined in the current object can be edited. (see Project Configuration on page 55).
	Show in project browser Calls up the project browser (see Project Browser on page 34) for the currently active object.
	Display Hides optional node identifiers (see Node Editor on page 40).

Functions in the Definition Tree Window Toolbar

ICON	FUNCTION
	Show item help Click this button to call up the context-sensitive help function for node types. The information for the node type marked in the definition tree will be shown automatically.
	Compile script Starts play back of the currently script (see Scripting in the Web Designer on page 51).
	Search and replace Opens the Search and Replace window (see Search and Replace Dialog on page 64).
	Copy objects Opens the wizard to create individual object copies (see Copy Objects Wizard on page 75).
	Switch editing mode The options Object definition, Configuration, and Configuration (custom) are available for Quest defined objects in the "Object definition" list. The option configuration does not apply for custom objects. Refer to Project Configuration on page 55 for more information.

Scripting in the Web Designer

The Web Designer provides you with the possibility to record a procedure in a web application and to replay it in a web browser control. There are two different ways for you to do this:

- Execute scripts directly in the Web Designer. In this mode, the script is executed against the internal web server and the selected database in the Web Designer. This allows you to test and document the functionality of individual workflows.
- Use the WebDesigner.ScriptServer.exe and WebDesigner.ScriptClient.exe tools to run the script against a web application installed, for example, on an Internet Information Server (IIS). This mode allows several instances of the script to be started with the help of network computers. A load test is thus created on the web application in use.

System Prerequisites

The following prerequisites apply when using the script server application:

- Internet Explorer 8 or later has to be installed on the computer in use.
- An Identity Manager installation must exist on every client workstation involved.
- A Windows Management Instrumentation (WMI) access to the computers involved may not be denied due to firewalls or similar.
- A web application must be installed on an Internet Information Server (IIS).
- A compilable Web Designer project to record and execute a script is required.

Recording and Executing a Script in the Web Designer

Start and end recording a script using the menu items in the Edit menu in the edit bar once the successfully compiled web project is displayed in the web browser. Record the user activity directly in the project browser. Then enter a name for the script and save it in the database.

You can find existing database scripts in the navigation window under the area Test scripts. In the definition tree window you can play back an open script on the icon Compile script. The script is displayed as a list next to the web browser control. Using the option Start execution the command be currently executed is highlighted. Furthermore the options Slow motion and One step at a time are available. The time taken to run the script is displayed after successful execution.

Executability of a Script

Successful execution of a recorded script depends on the following:

- The same user that was logged in for testing the script must be logged in for running it.
- The first action in the script must be available on the page loaded in the web browser control.
- In general, you can also use scripts after the web application has been changed. If the changes affect in the structure that is used in the script, the correctness of the script cannot be guaranteed. In the event of an error the Web Designer displays a compile error when the script player is called. Then you need to record the script.
- The state of database cannot vary from the state at time recording in certain points. An example of this would be requesting an one-time only requestable product in the IT Shop. When such a script is executed the web application saves the request in the database. If the script is run immediately afterwards, the web application aborts the disallowed second request with an error.

Using Scripts with the Script Server

You need to run the script you want to use one in the Web Designer in preparation and then save it as a text file on the hard disk using the Save icon. This type of script can quickly become invalid just from small changes to the web project. In this case, you need to save the text file again or rerecord the script if necessary.

Use the WebDesigner.ScriptServer.exe to control the client workstation.

In this program, the workflow before a test runs starts looks like the following:

1. Open script file.
2. Enter web application URL.
3. Enter a user name and password for logging into the web application.
4. Enter the number of test runs that each client workstation should run simultaneously.
5. Enter the name or IP-address of the client workstation with user names and passwords if necessary.
6. Enter the file path for the Identity Manager on the client workstation.

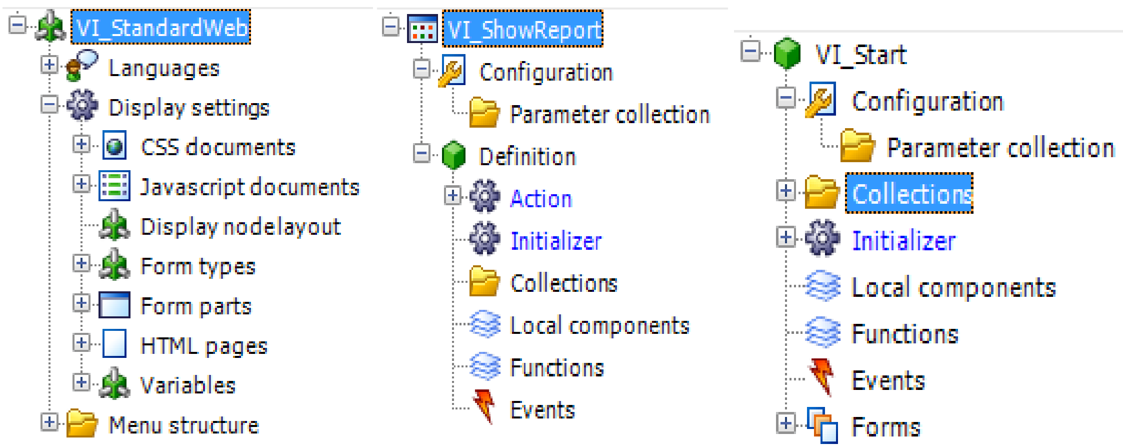
Ensure that the WebDesigner.ScriptClient.exe is started on every client workstation. Click on the up arrow displayed in the respective client table in the script server to trigger the start.

The script is executed in an endless loop on each client workstation during a test run. You can monitor the run times of completed scripts in a diagram on the locally started instance of the script clients even if your own workstation is in the client list.

Definition Tree

Upon being inserted, each web project, module and component is endowed with a minimum number of nodes in the object definition. All other nodes are inserted through a context menu (see Context Menu Functions on page 53).

Initial Web Project Definition Tree (left), a Component (center), a Module (right)



Context Menu Functions

To call up the context menu, right-click any node. The uppermost section of the context menu contains the node types and all available wizards that can be inserted into the currently selected node type.

The following wizards are available:

- Insert bundling node
  Eases the task of structuring nodes hierarchically later on. A container node is generated above the currently selected node and the currently marked node is pasted into the new container.
- Create data display
  Allows for the creation of forms, form elements, or tabular data views. This wizard is available at Container nodes and elsewhere (see Data Display Wizard on page 78).
- Embed Report
  Generates views from previously generated reports that are stored in the Identity Manager database. This wizard is available at action nodes (see Report Embedding Wizard on page 91).

If a defined number of node types is exceeded, the nodes will be re-bundled beforehand. Some node types do not allow for the insertion of additional node types. In such cases, the context menu begins with the cut function. The functions in the other sections (except for the last one) are either available or grayed out, depending on the type of node and its position in the definition tree.

Functions in the Context Menu

FUNCTION	DESCRIPTION
Cut	This function is available insofar as the node in question was not added automatically to the definition tree. All nodes that are subordinate to the selected node are likewise cut.

Functions in the Context Menu

FUNCTION	DESCRIPTION
Copy	This function is always available regardless of the type and status of the clicked node, and allows the selected node to be copied to the clipboard. All subordinated nodes are likewise copied.
Paste	This function is available if the clipboard contains a node that is also in the list of nodes that can be inserted in the currently selected node. The function inserts the clipboard content below the marked node.
Delete	This function is available insofar as the node in question was not added automatically to the definition tree. All nodes that are subordinate to the selected node are likewise deleted. Hold down the CTRL button to multi-select nodes to which you want to apply this function.
Set breakpoint	This function is available for controls and action nodes. Use this function to halt rendering at set points in debugger mode. Hold down the CTRL button to multi-select nodes to which you want to apply this function.
Set bookmark	This function opens a dialog box in which you can enter a description of a bookmark to set. The object is then labeled with the bookmark in the definition tree view.
Export	This function opens the “Save As” dialog. In this process, the xml definition of the selected node and all of its subordinate nodes are saved along with the file.
Import	This function opens a “file selection” dialog, where a previously exported definition tree structure can be selected. If the structure that is to be imported is compatible with the selected node, the structure will be inserted below the selected node.
Move up and Move down	This functions shift the position of the selected node within its branch. These functions are available if the node in question was not inserted automatically and if the current position of the selected node allows for the realization of the relevant action.
Undo and Redo	This functions are available if the relevant actions were carried out. The Undo function undoes the previous action, whereas the Redo function restores the state that existed prior to the realization of the Undo action.
Search	This function opens a dialog in which various search parameters can be entered using the CTRL + F shortcut (see Search and Replace Dialog on page 64).
Search next	This function continues the current search using the current search parameters and in so doing marks the next relevant node. Such a search can also be realized without opening the Search dialog.

4

Project Configuration

- Configuration Parameter
- Configuration Settings

Configuration Parameter

The important functions of default web can be customized without having to change the definition tree. Filter conditions (for example, searching for employees, products) can be changed, functions can be switched on or off, and properties displayed in the results list can be changed.

These changes are made by entering custom values for the corresponding configuration keys. It is not necessary to program new modules or components to do this.

Web Project Tab

These configuration keys are edited on the Web project tab in the "Project configuration" window (configurator for short).

To configure a web project:

1. On the menu bar, click **Edit**.
2. Select **Configure project | Web project...**.

   The web project is compiled in its actual state as soon as the configurator is opened. All declared parameters and references are assembled and then displayed. Therefore, what you see in the current configurator view is variable.

   - All nodes of type "parameter" are listed. Parameters are always added below a configuration section.The identifier given for a configuration step is shown both in the definition tree, and as an outline level in the project configuration window.
   - Configuration parameters that have been declared in different modules under a configuration step of the same name are grouped under one outline level in the project configuration window.
3. Mark your desired configuration parameter.

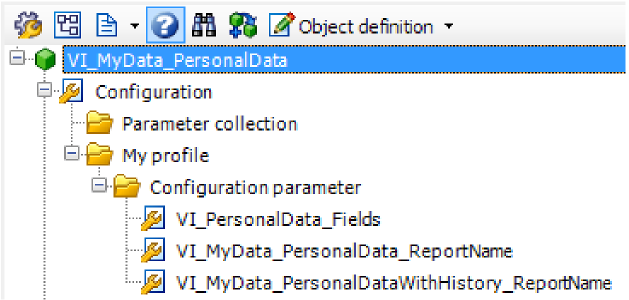
   More information are shown under Details.

The list contains the configuration section nodes that are implemented in the definition tree. The configuration parameters defined below this section are shown when you open the second level. Various input options are available in the configurator depending on the type of configuration parameters in the definition tree.

4. Select the check box in the **Customized** column.
5. Click the **Magic wand icon**.
6. Apart from selecting columns, you can also define column width and whether column content is to be <read-only>.
7. Click **Apply**.
8. Click **OK**.

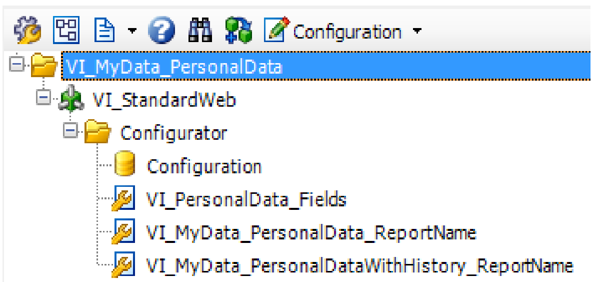
The parameters are declared directly in the module or components under the "configuration" node. The first sub node ("Parameter collection") is not used for declaring configuration parameters but for declaring passing parameters (as an example: forwarding from one module to the next). The parameters declared here are not listed in the configurator.

Declaring Module and Component Parameters in the Definition Tree



The assignment of a configuration key with a value is always applied to the module or component. They are shown if the selection in the drop-down list matches Configuration or Configuration (custom) (see Table Definition Tree Window Toolbar on page 50). All configuration parameters that are not listed here, are defined with no value.

Declaring Module and Component Parameters in the Definition Tree



The web project, for which parameters following under it are valid, is the first thing to be defined under the root node “Configuration”. This makes it possible for a module or a component, which is used by n different web projects (referenced), to provide its own parameter definitions for each web project. Assuming that no parameter definition under Configuration (custom) is entered or running under that module and no parameter definition for the web project either, the default parameter definition is used.

The type of parameter is defined in the node editor window. A selection of different types of parameters are available here. The remaining parameter types are:

1. Boolean values

   Choose a Boolean value and a check box is shown under “Details” in the lower section of the window. If the default value can be changed into a customer-specific value, set a check mark in the column Value (custom).

2. SQL quantity limit

   To set an SQL quantity limit, it is necessary to enter an SQL condition, which can be generated using the SQL wizard or can be edited in the displayed text box. The following should be borne in mind in this regard:

   a) Only a Where clause fragment is necessary. It must begin with the name of the primary key field of the relevant object, followed by the SQL keyword “in”. The clause can end with any syntactically correct condition. Here is an example for the “Selectable Reference Users” parameter: The relevant objects are thus in the person table, which means that the Where clause fragment begins as follows:
   uid_person in (<any SQL expression>)

   b) Only “pure” SQL can be used to formulate an SQL quantity limit. Web SQL cannot be used.

Configuration Settings

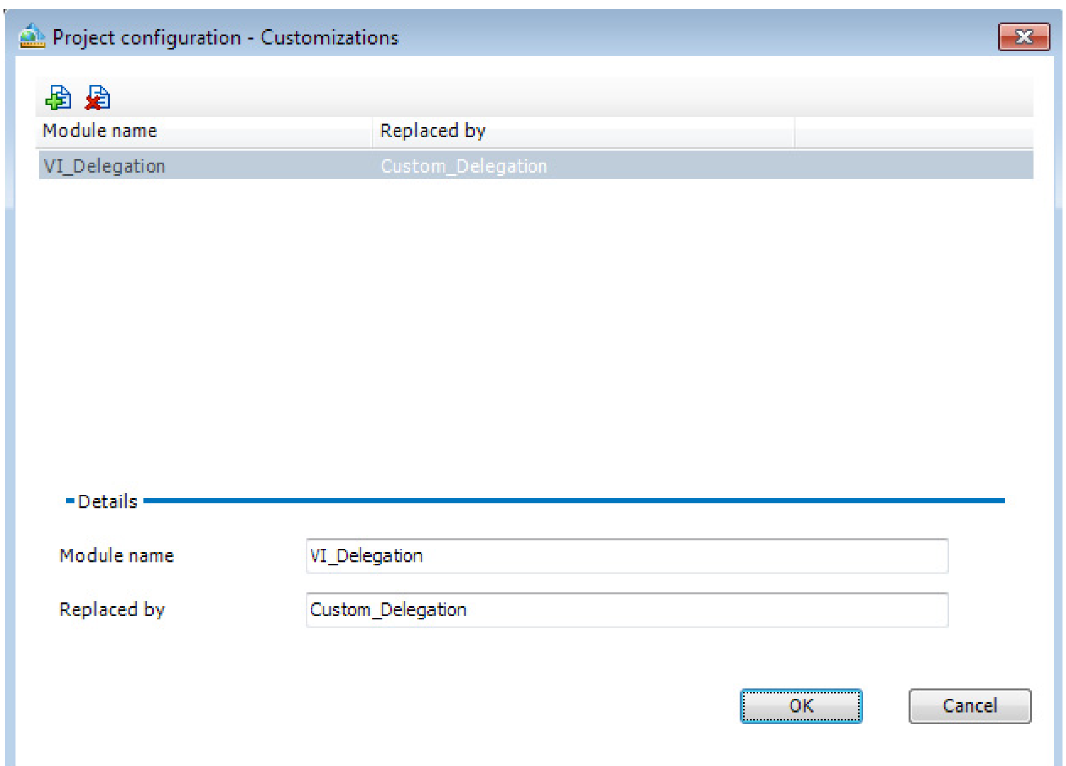
Project Configuration - Customization

This configurator domain allows for the creation and editing of substitution rules for module copies. The list also contains the substitution rules that are generated automatically by the Copy object wizard (see New Project Wizard on page 72).

The generated substitution rules have a higher priority than the references in the definition tree, which remain unchanged.

For the example shown in the screenshot, this means that redirecting is realized not to the VI_Delegation module, but rather to the Custom_Delegation module, even if the redirecting destination is still defined in the definition tree as VI_Delegation. If the customer module is no longer needed, the substitution rules can simply be deleted.

Project Configuration - Customization



Project Configuration - Search Fields

The menu item Search fields allows the user to define which database columns a search function should search.

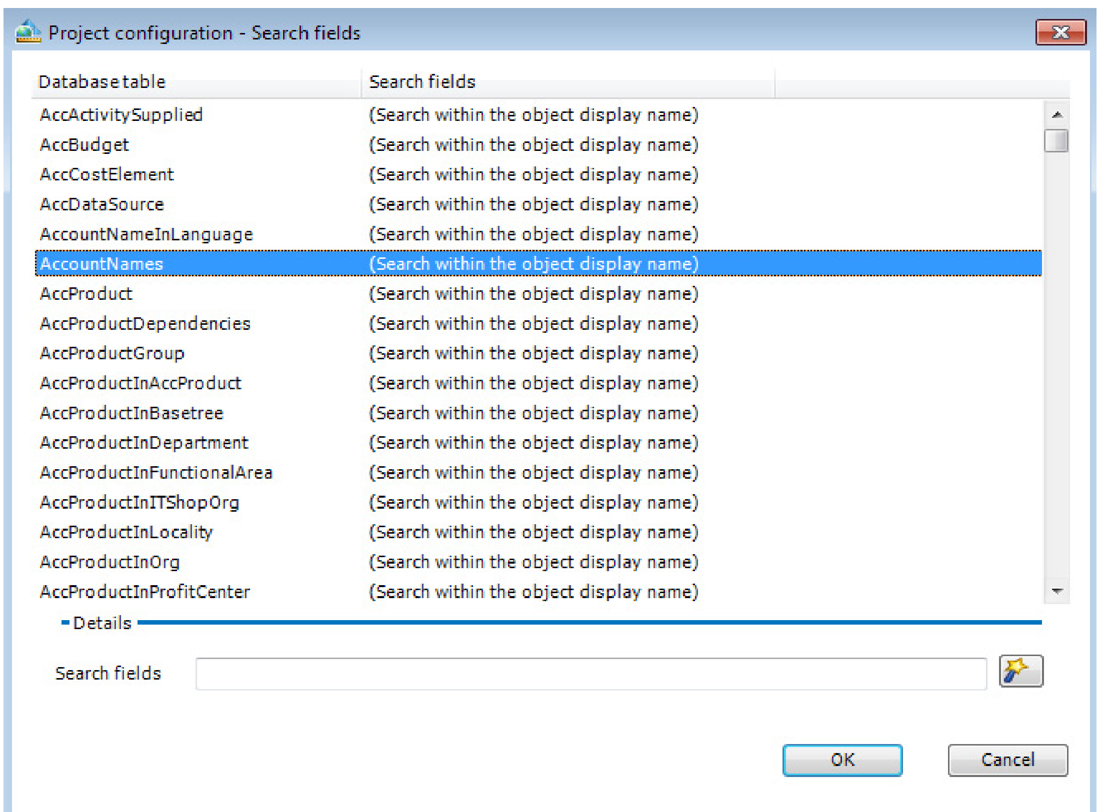
The configurator always lists all of the database tables here. In the standard delivery, each database table determine its own display pattern.

For example, the display pattern of the database table “Person” - “%InternalName%” - is deposited in the database (see Designer Database Schema | Table properties. If a different table or additional columns are to be searched, the search box list can be modified using a wizard.

In order for the search settings realized on this tab to be implemented, the Sqlsearch() function must be used in the relevant expression. This function needs the <Table Name> and <Find String> parameters. Example: sqlsearch(Person from various selected search strings)

This function is used by default in Quest-defined modules and components.

Project Configuration - Search Fields



Project Configuration - Column-Dependent References

A column-dependent reference is a conditional reference to a component. Depending on the type of column in a database table (for example, Boolean, DateTime or String) it is decided, which component will be used to edit the relevant content. To this end, the standard scope of delivery includes VI components which cover all possible column types and find them automatically.

Insofar as a module uses customer components rather than VI components, a relevant reference can be entered in the column-dependent reference menu item.

The references have a higher priority than the references that are implemented by default and are thus used as a matter of course.

To configure Column-Dependent References:

1. On the menu bar, click **Edit**.
2. Select **Configure project | Column dependent references...**.
3. Fill out or select from the drop-down list the following data to configure a reference:
   a) **Table**
   Database table of which the columns in question are part.
   b) **Column**
   The columns in which standard components are no longer to be used.
   c) **Component**
   Components that are to be used in lieu of the standard components.
   d) **Foreign key filter**
   the Where clause entered here places a limit on the number of objects that can be shown in the reference components. Note: the usual Web Designer syntax cannot be used here. If the UID_Person of the logged on user is to be used in the expression, this is nonetheless allowable if %useruid% is used.
4. Click **OK**.

Project Configuration - Object-Dependent References

A object-dependent reference is a proprietary node type that references components and as such is updated in the definition tree.

However, the component reference is expressed not through the component name, but rather through a combination of the properties Service categories or Attestation procedures and reference type.

Thus the menu item Object-dependent references is used to define which reference type and service categories or attestation procedures combination will reference which components.

Inasmuch as the available service category or attestation procedure is keyed to runtime, various components can be referenced.

The second part of the reference (the reference type) allows for the referencing of different components for the same service categories or attestation procedures at different positions on the definition tree.

To add a Object-Dependent Reference:

1. On the menu bar, click **Edit**.
2. Select **Configure project | Object dependent references...**.
3. Fill out or select from the drop-down list the following data to create a new reference:

a) **Name**
Reference name, which is used only at this place.

b) **Description**
Textual description of the reference; only shown in the configurator.

c) **Reference type**
Reference type selected from the reference type list, to which items can be added by updating the action type table.

d) **Component**
Selected components that are to be referenced.

e) **Service categories / Attestation procedures**
Selected object types to which the reference is to apply. The groups are selected automatically as well for all subordinate object types.

4. Click **OK**.

Project Configuration - Table References

A table reference is a proprietary node type that references components and as such is updated in the definition tree.

However, the component reference is expressed not through the component name, but rather through the database table that the data set currently being processed is part of.

Thus the configurator's table reference is used to define which component will be referenced by which database table. This is necessary in various situations, including in cases where collection objects from various database tables are to be displayed.

To edit Table References:

1. On the menu bar, click **Edit**.
2. Select **Configure project** | **Table references...**.
3. Mark your desired component.
4. Configure the table references.
   The configuration options are explained below.
5. Click **OK** to save the changes.

Table references are to be configured as follows:

- Type of reference
  Any of the reference types indicated in the table references node type can be used as names for reference types.
- Default
  Define a default component for every reference type. This component is then used whenever the reference type uses a table for which there is no specific entry. If properties are used in a default component that does not appear in all tables, the tables for which the default components can be used are to be indicated. (Creation for tables, entering a comma-separated list. ObjectCollection reference type uses the display property only. This property is available in all tables whose objects are displayed coherently in the web application.)
- For table
  Selection of the database table that is to be referenced.
- Component

Selected components that are to be referenced.

5

Dialogs

- Project Approval Dialog
- Search and Replace Dialog
- Captions Dialog
- Clean Language-Dependent Text Dialog
- Object Properties Dialog
- Change Label for Web Project Dialog

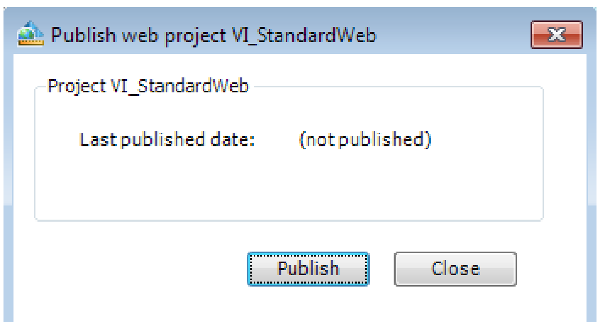
Project Approval Dialog

In order for a web project to be available as a web application in its current stage of development, the project must be approved, assuming that a web application has been published previously (see *Web Portal Installation Guide*). Approval is realized through a dialog that can be opened through the menu under Edit | Publish project. The title of the approved project will be indicated in the dialog's window title.

A project can only be approved if the logged on user is part of at least one permissions group that is authorized to carry out an approval procedure. This configuration can be viewed in the tool Designer under Permissions | Program functions | WebDesigner_PublishApplication, and can be modified if the user has the requisite permissions.

Start the compiler by clicking Publish. If a compiler error occurs, a message to this effect will be displayed and no approval will be realized. If no compiler error occurs, a new object will be saved in the DialogAEDS table, which mainly comprises a Zip archive with a "la" termination. This object contains all modules, components, project files and configuration files needed for a web project (except for captions) and is treated as a web application by the web server.

Project Approval Dialog



Search and Replace Dialog

To use the Search and Replace Dialog:

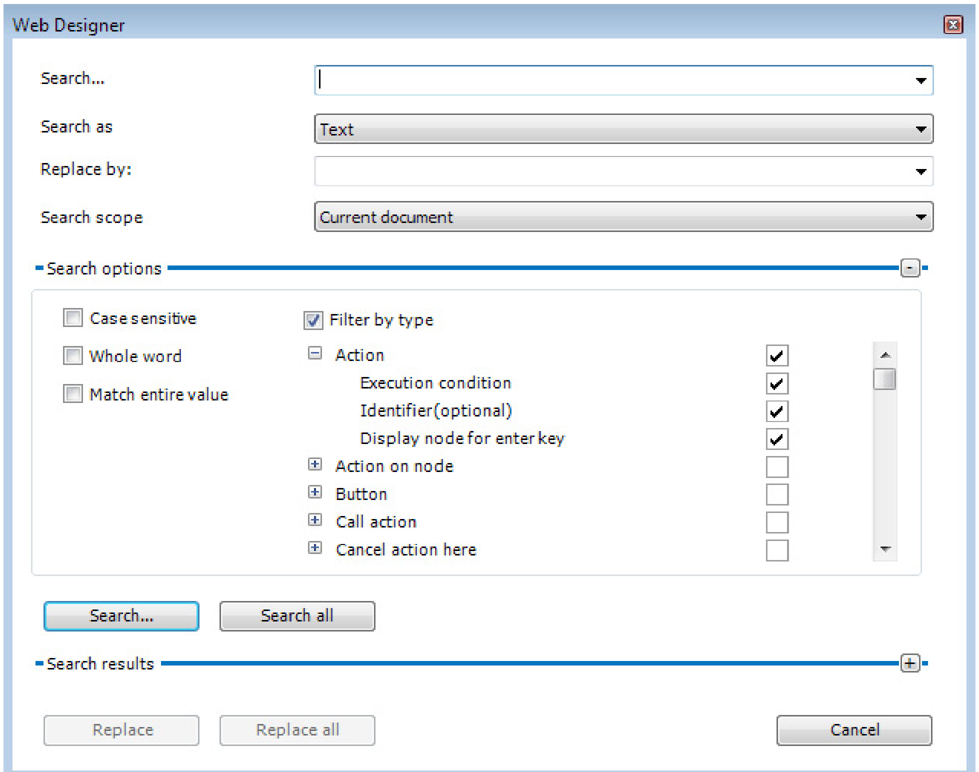
1. On the menu bar, click **Edit** | **Search**.
2. Use the search field to enter text strings (with or without wildcards) or standard expressions.
3. Select the search algorithm from the drop-down box Search as.

   The search algorithm used (text, placeholder or expression) must be adapted to the entry realized.
4. To replace texts or expressions, make the relevant entry in the **Replace by** field.
5. Select your preferred search scope from the drop-down list (Current document, Below the selected object, All Web Designer objects or Current web project).
6. If necessary, activate your preferred options under Search options:
   - **Case sensitive**
   - **Whole word**
   - **Match entire value**.
7. To select node types, click the **Filter by type** selection box.

   If at least one node type is selected, the search will be limited to the selected node types.
8. Click **Search** or **Search all**.

9. To mark the relevant node in the definition tree, double click a line in the search list.

Find Dialog



Captions Dialog

To open the captions dialog, go to Edit | Captions in the menu bar. This same dialog is also used for text editing for many different types of nodes. For this, the selected or entered caption is assigned to the relevant node in the same work step. The caption dialog also has a Quick edit tab.

MultiLanguage texts (short "Captions") from the standpoint of the database, captions are standalone objects in the DialogMultiLanguage table and are thus not saved in web project files. Instead, a reference to a DialogMultiLanguage object is created in the web project nodes. The caption dialog now performs two actions:

- it creates and edits the DialogMultiLanguage objects
- it creates the references in the Web Designer definition tree nodes.

DialogMultiLanguage objects are edited on the Captions tab. Dialog objects have four main properties:

- Language
  Language indicates the language in which the text is written.
- Key
  Key is a unique value that allows for referencing of a DialogMultiLanguage object.
- Text
  Text is the text that is to be displayed in the designated language.
- Customized text

Customized text is likewise a text in the designated language, but in the display this text overwrites a Quest-defined text, thus allowing texts in Quest-defined modules to be edited to the customer's specifications without generating a customer-specific module.

If a web application is to be available in more than one language, a DialogMultiLanguage object is to be created for each such language. All such objects must then integrate the same key (not to be confused with "primary key"). Thus the web application will only show the caption object whose language is the same as that of the defined language.

In the interest of enabling caption editing, a search function containing various options has been implemented in the upper portion of the mask:

- Search key and value
- Search for key only
- Search for value only
- Search in all available languages

During text searches, both the Quest-defined text and the customer-specific text are searched in all cases. The search results are shown in the results list, where they can be selected for editing by clicking them. Use the toolbar buttons to save and delete captions.

Caption Dialog Toolbar

Functions in the Caption Dialog Toolbar

ICON	FUNCTION
	Add Adds a new caption.
	Delete Deletes the active caption in the editing field after a database confirmation prompt is clicked.
	Save Saves a new or modified caption in the database.
	Assign caption This button is only activated if the dialog concerning a definition tree node is open. Clicking this button assigns the active caption in the Edit field to the node.

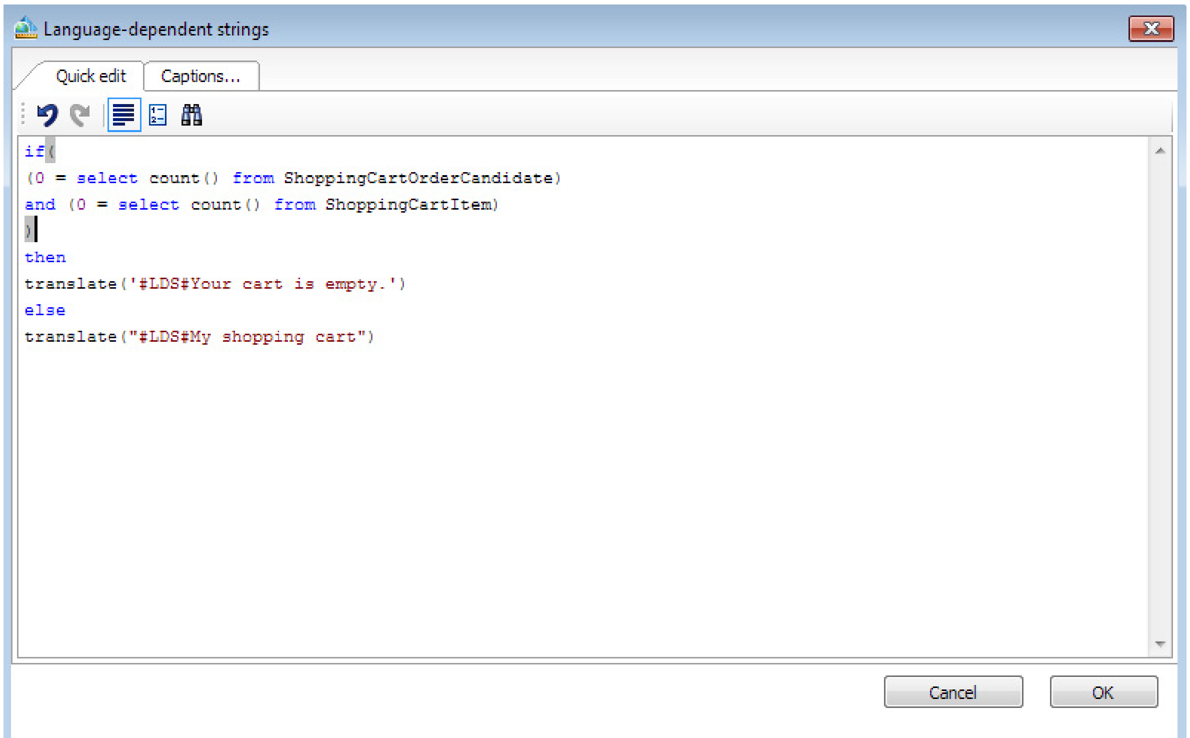
If the caption dialog is opened through a definition tree node, as previously noted this dialog contains the Quick edit tab, and below that an SQL editing window, which can be used in many other places as well. If a caption has already been assigned to the node, an SQL expression will be shown that takes the following form: translate("#LDS#<key>"). "Translate" here stands for calling up the corresponding Web Designer SQL function, whereas LDS (Language-dependent string) means that there is a caption after the key.

The translate function also allows for the use of parameters. The following figure shows an example of a caption with two parameters. In such a case, the parameter reference (for example {0}) need only be entered in the text. If the text thus parameterized is shown, it is always necessary to edit the function, for example enter the relevant parameters.

The Quick edit tab is available as well if no caption has been selected on the Caption tab. In this case, the SQL input field is not pre-filled either, but a relevant expression can be entered manually. If a key is used in such a case for which there is no caption in the database, the key name that was previously entered will be shown in the preview window and web application. Example: translate ("#LDS#Hello World!") will be shown in the web application as >Hello World!< if the key does not exist in the database.

This function is particularly useful during the initial phases of a project, when displayed texts are frequently modified. Until the customer initially accepts the text, only its key is kept updated. Once the text has been accepted, the captions and all necessary translations are implemented. A specific dialog has been implemented that makes this task easier (see Clean Language-Dependent Text Dialog on page 67).

Captions Dialog and Direct Editing Tab



Clean Language-Dependent Text Dialog

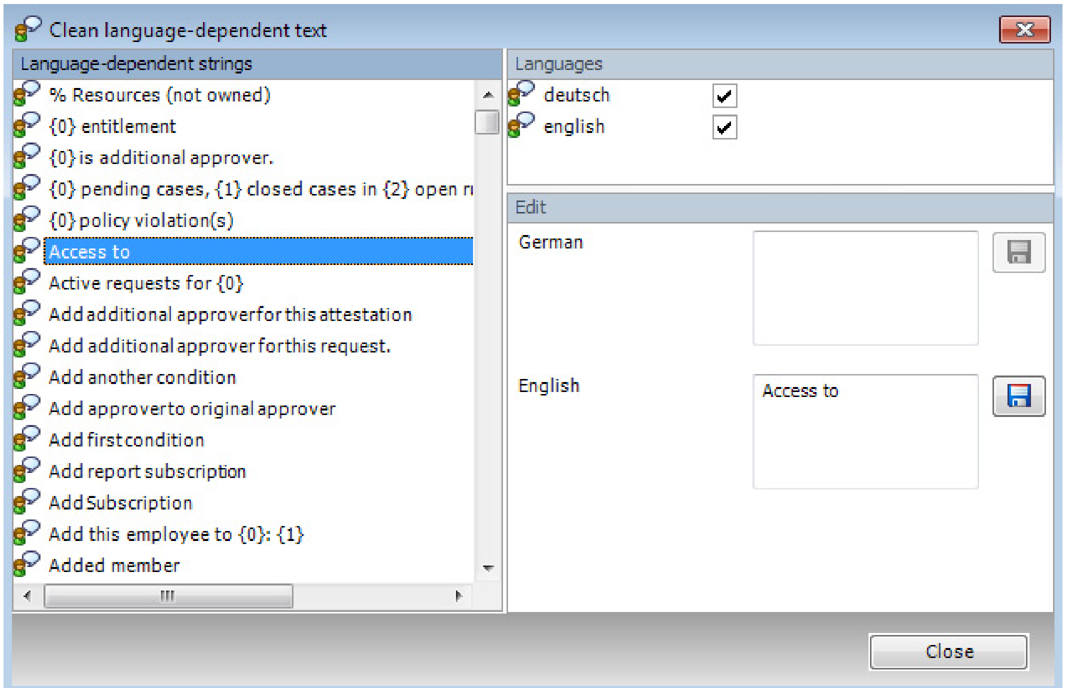
To open the dialog go to Edit | Clean language-dependent text in the menu bar. The left side of the dialog contains a list with the keys that are used for which the DialogMultiLanguage table contains no object in at least one language (see Captions Dialog on page 65 for further details).

When this list is compiled, the languages are taken into account that have been defined for the web project in question. To reduce the number of languages that are taken into account for the list, use the checkboxes in the Languages area.

If a key is marked in the list, the counterpart text will be shown on the right side, where only the languages that were defined using the checkboxes will be taken into account. A blank field indicates that there is no text for this language. The program will automatically suggest missing texts in languages that are defined in the database as default languages. Each such suggestion is based on the key that is used. In such a case, the text field will be filled and the Save button next to the text field will be activated.

A Save button is available for each text field since each translation comprises a separate database object, which the button will save to the exclusion of any other object.

Edit Language-Dependent Text Dialog

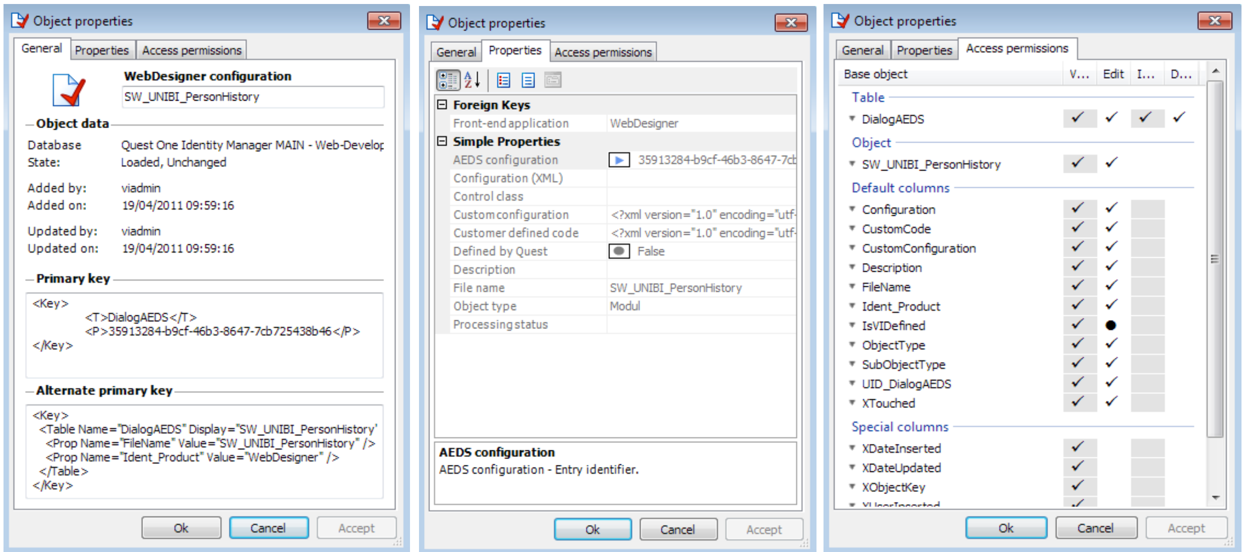


Object Properties Dialog

The object properties dialog shows the properties that are generated by Web Designer and the files that are saved as objects in the DialogAEDS table (see Files generated by Web Designer on page 15), where each module, component, project file and web project is represented as a separate file. Hence, once a relevant object has been marked, the object properties dialog can be opened through the toolbar icon  in the navigation window or through the context menu, insofar as such an object has been marked.

The dialog that now opens contains the <General>, <Properties> and <Access permissions> tab, the latter two of which can also be used to edit the relevant data, if the user has the requisite permissions.

Information shown in the Object Properties Window



Change Label for Web Project Dialog

Open this wizard using the  button in the toolbar. Use the function to book all database objects that make up a web project to one change label. This allows a web project to be transported to another database using the tool Transporter without problems. The function ensures that all necessary database objects are taken into account during transport.

To use the Change Label for Web project Dialog:

1. Click **Apply a change label to whole web project** in the toolbar.
2. Specify which groups of database objects should be linked to the change label:
   a) Modules, Components and Configuration

      The option links the database object created by the Web Designer (modules, components and configuration files) with the change label.

   b) Captions

      The option links all captions required by the web project (compare with Captions Dialog on page 65) with the change label.

      If any of the selected objects are already linked to one or more change labels, these assignments are not changed or deleted. In fact, another link is added.
3. Click **Next**.
4. Use the magic wand icon to open default dialog box for adding and selecting a change label.

   In this step of the wizard you define the change label, which will link the database objects that are affected.

Alternative, enter your own change label in the text box and click on the Next button to add and save the links in the database. Confirmation is displayed on the last page of the wizard if the action is executed successfully. To edit the change label, click the magic wand icon and select your change label from the list.

5. Click **Create a new change label** in the change label dialog toolbar.
6. Fill out the following text boxes:
   - **Change label**
   - **Description**
   - **Comment**
   - **Status comments**.
7. Select the Parent change label from the drop-down list (if necessary).
8. Select the Status from the drop-down list.
9. Select the Label type from the drop-down list.
10. Select the Locked status from the drop-down list.
11. Click **OK**.

Functions in the Change Label Dialog Toolbar

ICON	FUNCTION
	Show / hide edit view.
	Create a new change label.
	Delete the selected change label.
	Save change label.

See the chapter "Working with Change Labels" in the *Configuration Guide* about working with change labels.

6

Wizards

- New Project Wizard
- Migration Wizard
- Copy Objects Wizard
- New Module Wizard
- Data Display Wizard
- Report Embedding Wizard
- Mapping Generator Wizard

New Project Wizard

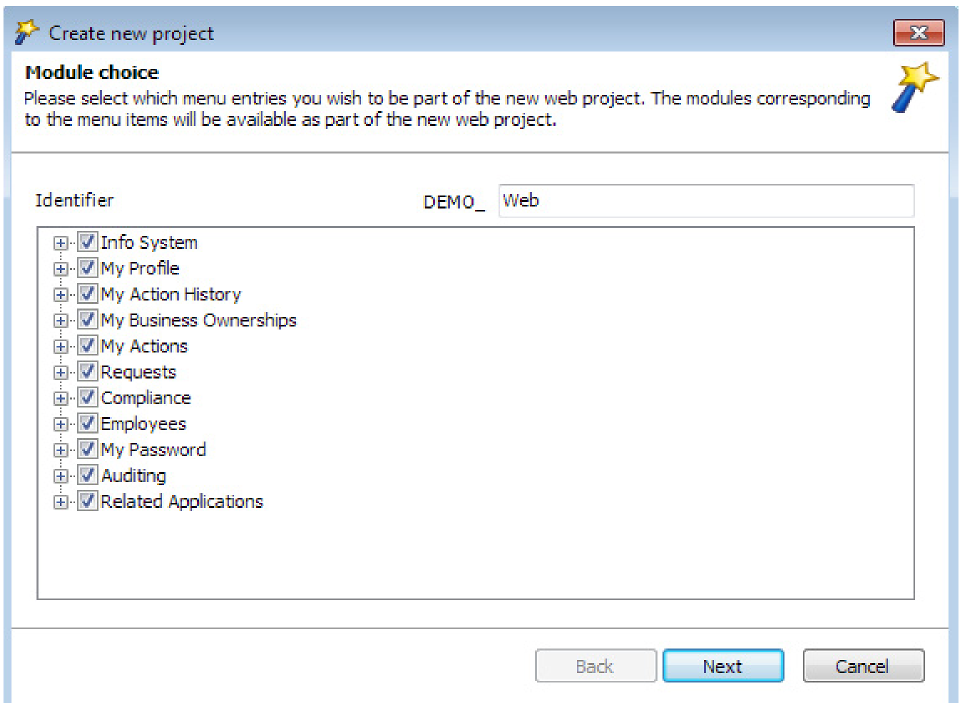
If properties that are defined in the web project (languages, display settings, menu structure) are going to be changed, it is necessary to create a copy of the default web.

This task is different from creating an object copy (see Copy Objects Wizard on page 75) because the menu structure is also part of a web project.

The Create new project wizard allow the editor of selecting which menu items and therefore which workflows from the default web project should be available in the new web project.

You start the wizard from the menu bar, Edit | Create new project. You need to enter a name (identifier) for the project on the first page of the wizard.

Create New Project – Entering the Identifier

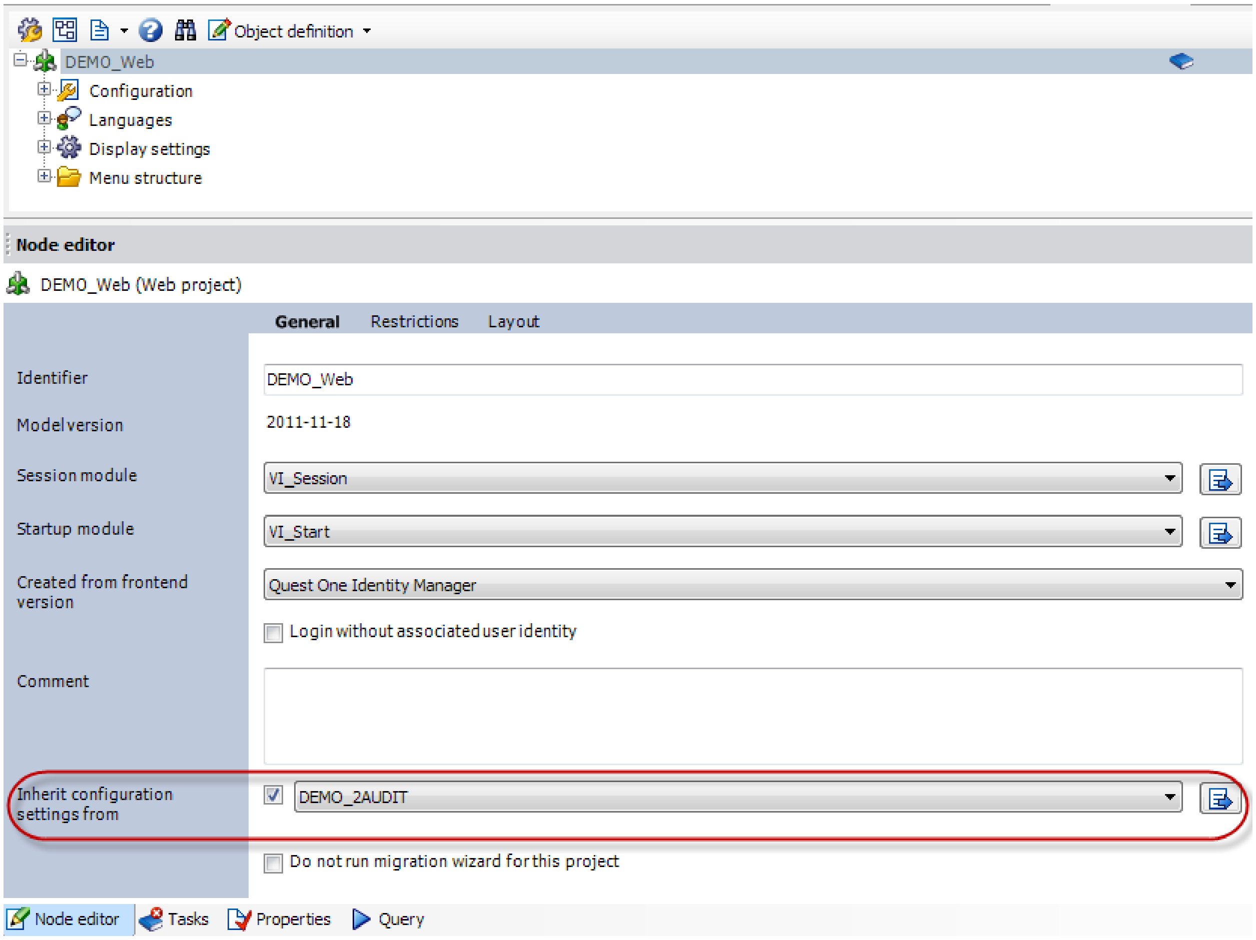


Then the menu items that are to be available in the copy have to be selected from the default web project. The default modules are still referenced from the customized web project. The new web project is added once you click the Next button.

Already custom configurations to the default web are automatically inherited by the newly created web project. This setting can be changed later if the configurations should be transferred from another web project. This function is available in the node editor window, Inherit configuration settings from.

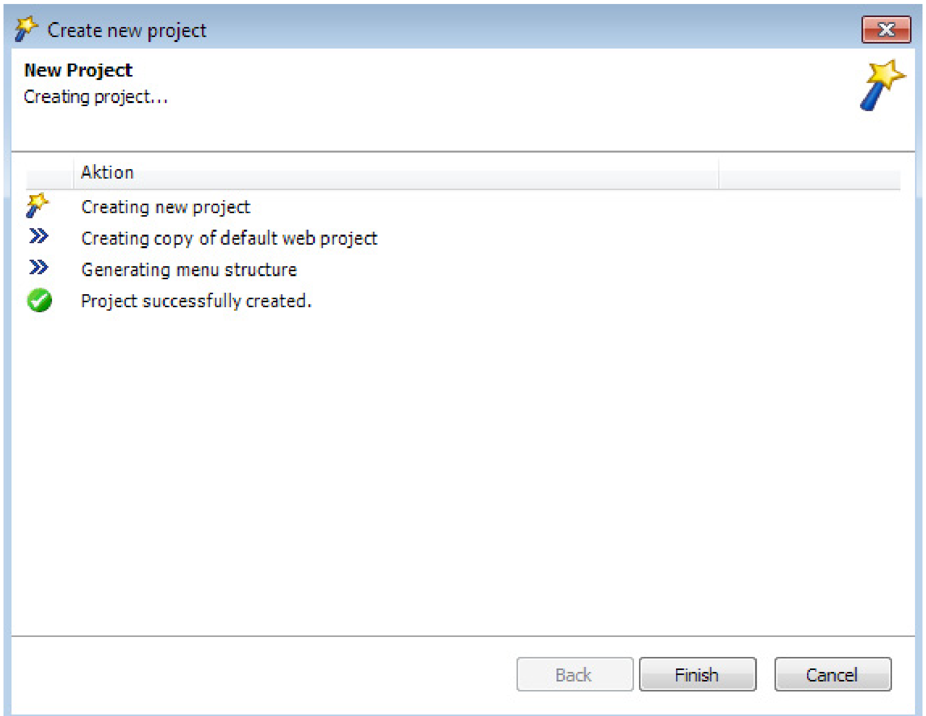
If you wish to display only menus with menu items: Open the web project in the definition tree window. Open the folder Menu structure and select a menu item by clicking. Activate the option Hide if no child menu items exist in the node editor window.

Change Configuration Settings



The last page of the wizard summarizes the changes that have been made. After you have closed the wizard with the Finish button the new web project definition is displayed in the edit window. The files have not yet been saved by the wizard. You have to do this yourself after everything has been checked. The new web project also represents a new web application. In order to start it using a browser, the web application has to be appropriately configured. See *Web Portal Installation Guide* for more information about this.

Create New Project - Confirming



Migration Wizard

Use the wizard to adopt available changes to a web project. If you open a web project in an older product version, a message is shown. Start the wizard by clicking the Yes button. You may launch the wizard at any time (Edit | Migration wizard).

Use the wizard by default, if compile errors occurs. You can disable the tip to the use of the wizard: Open the web project in the definition tree window and activate the option Do not run migration wizard for this project in the node editor window.

To work with the Migration Wizard:

1. Click **Edit** | **Migration wizard** to open the wizard.
2. If you want to integrate model changes, activate the option **Handle model changes**. Select the menu items you need from the shown list.

   All menu items are enabled by default. Remove the tick from the box for the menu items which you do not want to use.
3. Click **Next**.

   The migration starts.

4. Click **Finish**.

The changed objects have not been saved. In order to save changes permanently, click the Save button in the toolbar.

If you wish to display only menus with menu items: Open the web project in the definition tree window. Open the folder Menu structure and select a menu item by clicking. Activate the option Hide if no child menu items exist in the node editor window.

Copy Objects Wizard

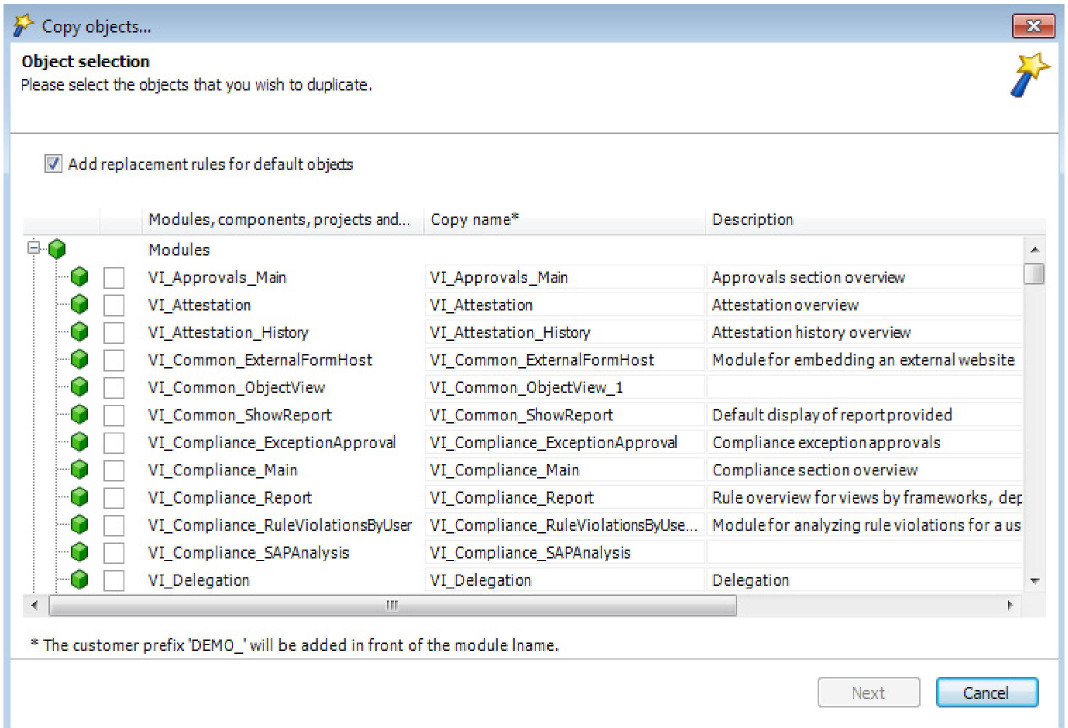
We mentioned already that default objects cannot be modified for customized installations. If modifications are required that cannot be achieved using the configurator, a copy can be made of the default objects, which contains the function to be modified. Use the Copy objects wizard to do this or for a individual object the Copy objects wizard in the definition tree window toolbar. Open the wizard with the Edit | Copy objects menu item.

The wizard displays a list of all objects that belong to the current web project. Select the objects to be copied and confirm the action. You can change the suggested name in the middle column of the selection table. The names suggested or provided are automatically inserted in front of the custom prefix in the database.

If the option Add replacement rules for default objects is set for Quest-defined objects then replacement rules that are used as copy templates for default objects are added automatically. The copies (the new custom objects) are automatically referenced in sequence instead of the default objects. If the custom objects should be used as copy templates, then no replacement rules are created by the wizard. If these should be required at a later date, they can be entered by hand into the configurator (refer to Project Configuration - Customization on page 58).

Use the Next button to create the copies and open the confirmation prompt. After you have closed the wizard with the Finish button the copies are displayed in the edit window. The files have not yet been saved by the wizard. You have to do this yourself after everything has been checked.

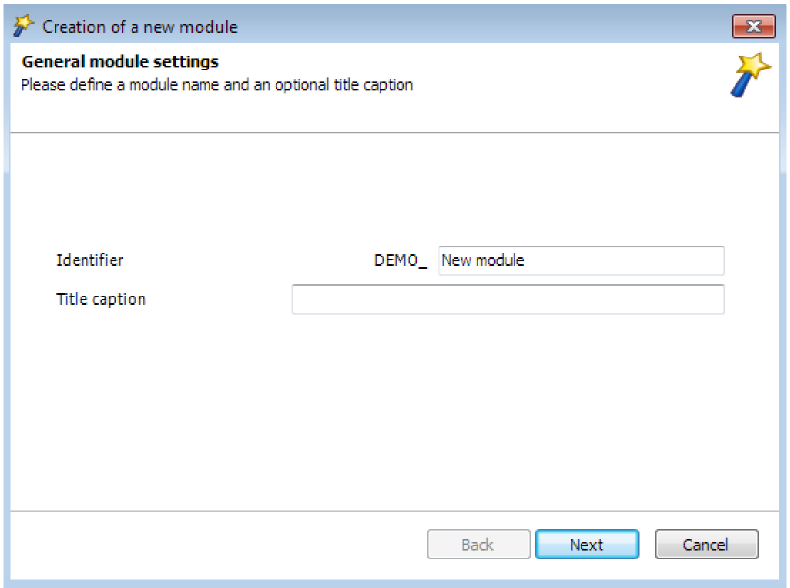
Copy Objects Wizard



New Module Wizard

Use this wizard to execute all the necessary steps to add a new module and reference in a web project. Start the wizard from the menu item Edit | Create new module…. Enter the identifier for the new module in the first step. A title for the new page can also be entered. Once this is done, a label node is automatically inserted into the title bar on the form. A corresponding caption object is not added. A warning is outputted the next time the project is compiled and the caption can then be added including all possible translations required (refer to Tasks on page 41).

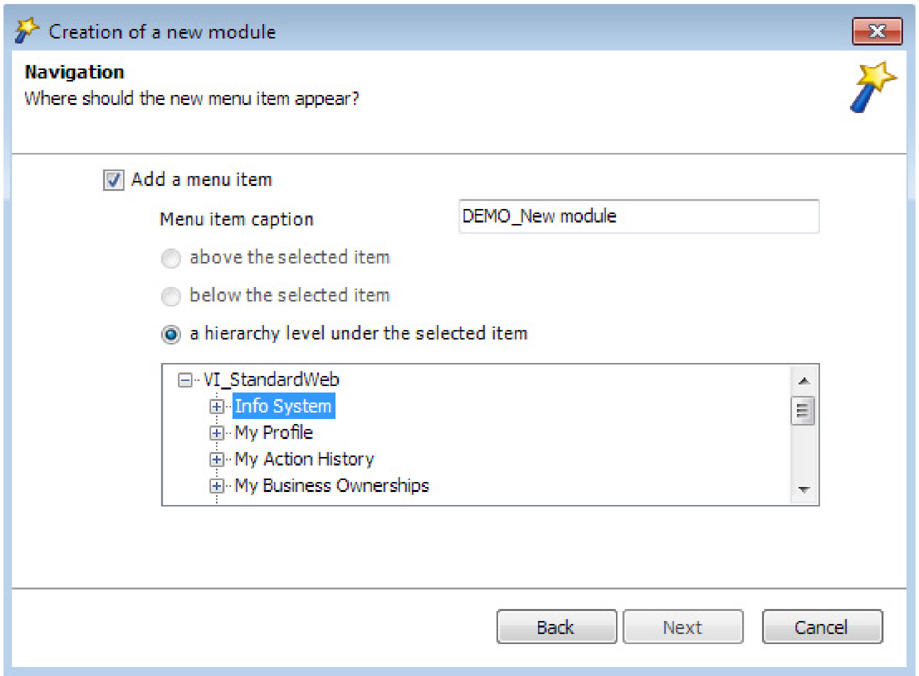
Create New Module Wizard – Basic Data



Use this page (see figure below) to specify whether a menu bar entry should be set up for the new module or not. However, this is not essential. If the appropriate option is set, more options are shown for defining its position in the menu bar and how is should be labeled.

Enter a name for the menu item in the Menu item caption field. This field is predefined with the module name or, if available, the text given for the title. This may be overwritten. A caption object can also be maintained for this text.

Create New Module Wizard – Adding and Configuring a Menu



The following step is only executed if the option Add a menu item has been previously selected. This defines the group of employees the menu item applies to:

- Visible everyone

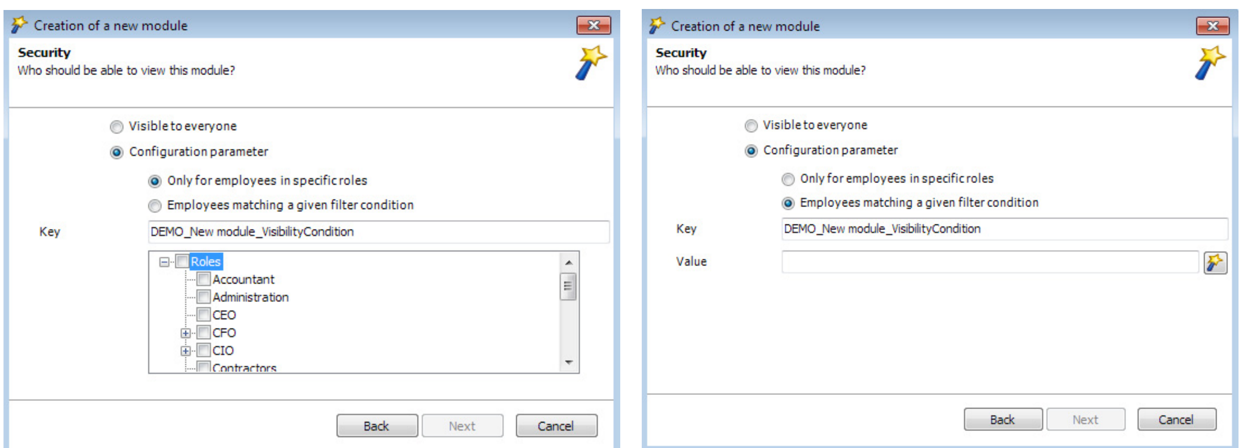
  No viewing restrictions are applied to the new menu item if the option Visible to everyone is selected.
- Configuration parameter

  Select this option to add a new configuration parameter. A predefined value is suggested for the key. There are more options available
  a) Only for employees in specific roles
  b) Employees matching a given filter condition.

If you select Only for employees in specific roles, multiple selection is possible in the Identity Manager treeview. The new menu item is available in the web application to all employees that are assigned to at least one of these selected structures.

If the option Employees matching a given filter condition is selected, the usual SQL wizard for formulating an SQL query is started. In this case the query always refers to the table "Person". Therefore, this must not be entered again.

Create New Module Wizard – Viewing Rules Configuration for Menu Items



Use the Next button to execute the changes in the web project, create a new module with all initial nodes and open the confirmation prompt. After you have closed the wizard with the Finish button the new module is displayed in the edit window. The files have not yet been saved by the wizard. If a menu item was added the web project is also opened in an edit window. You also have to save the changes to the web project yourself.

Data Display Wizard

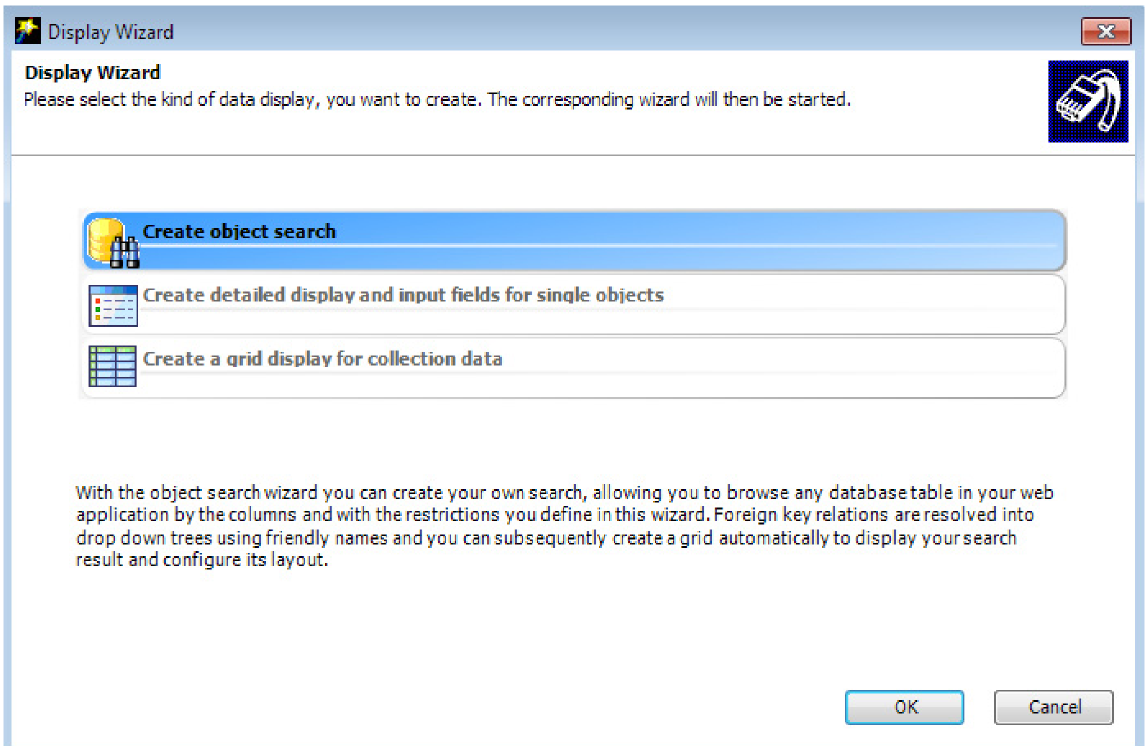
The core task of a web application created with the Web Designer is targeted visualization and manipulation of database objects. The Create data display wizard allows you to implement the three main tasks of a web application (setting up an object search, displaying a single object, displaying a group of objects) in a web project.

The wizard can be started from the context menu, but only from nodes that are permitted to have the relevant display (for example, containers).

The following figure shows the wizard's startup screen with the main tasks for selection:

- Create object search
  The resulting form allows you to search for database objects based on specified search criteria and to display the results in a list.
- Create detailed display and input fields for single objects
  This results in a list of single database objects and allows you to change (overwrite) selected properties.
- Create a grid display for collection data
  The resulting table shows all properties of n database objects. Relations between single database objects can be displayed in hierarchical form.

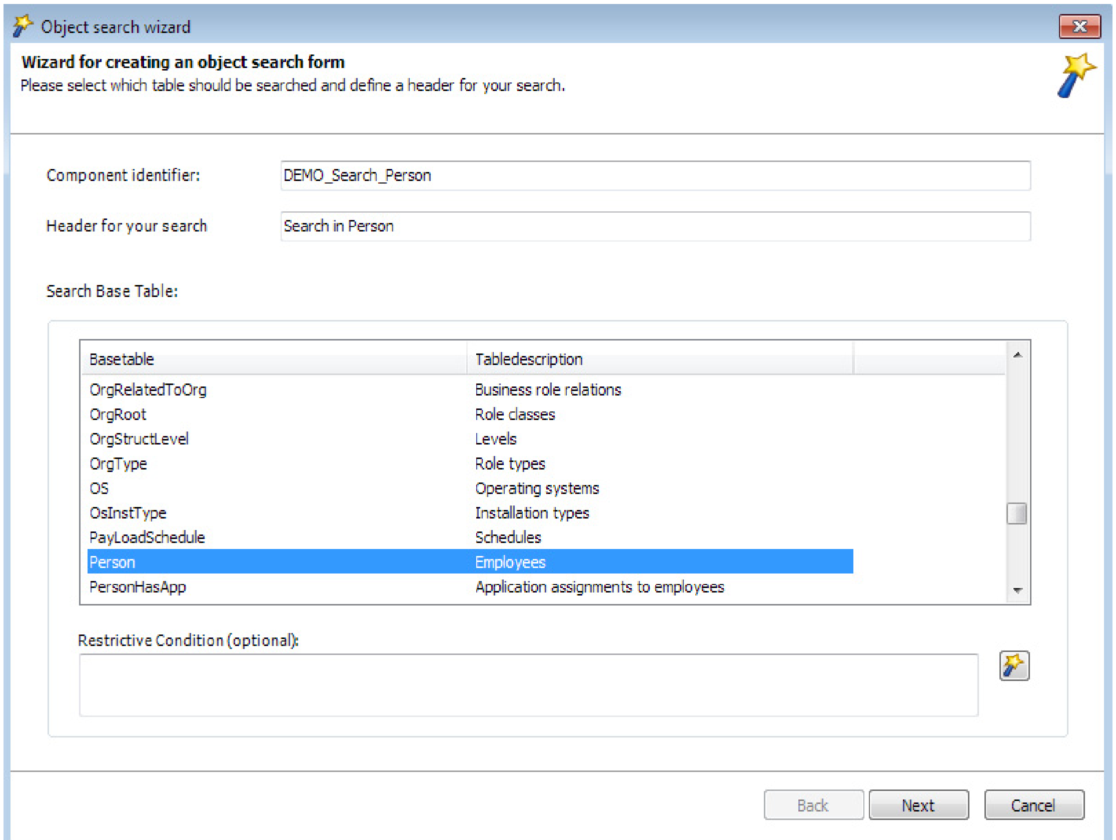
Create Data Display Wizard – Startup Screen



Creating an Object Search Form

This part of the wizard allows you to set up a search screen that can search any database table. Ultimately, the wizard creates a new component (with input fields for the search strings and a grid displaying the search results) and below the marked node a component link that references the new component.

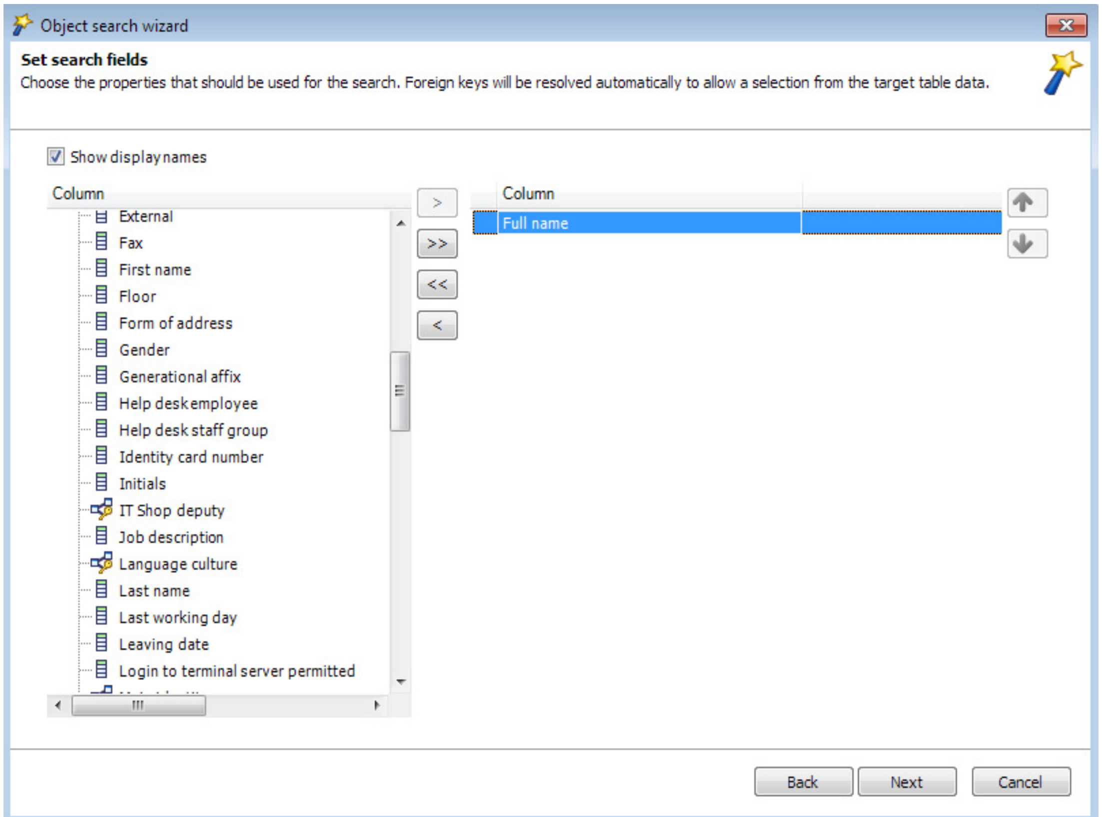
Creating an Object Search Form - Basics



On the first page of the wizard you need to select the table for the search. Suggestions are created automatically for relations to the components to be created and for a title for the search screen. These can be overwritten. You can also enter a Where clause statement that limits the search results more independently of the search strings. This means, for example, that the limiting condition “isnull(IsInactive,0) = 0” ensures that only those employee objects without a status of “permanently disabled” are found.

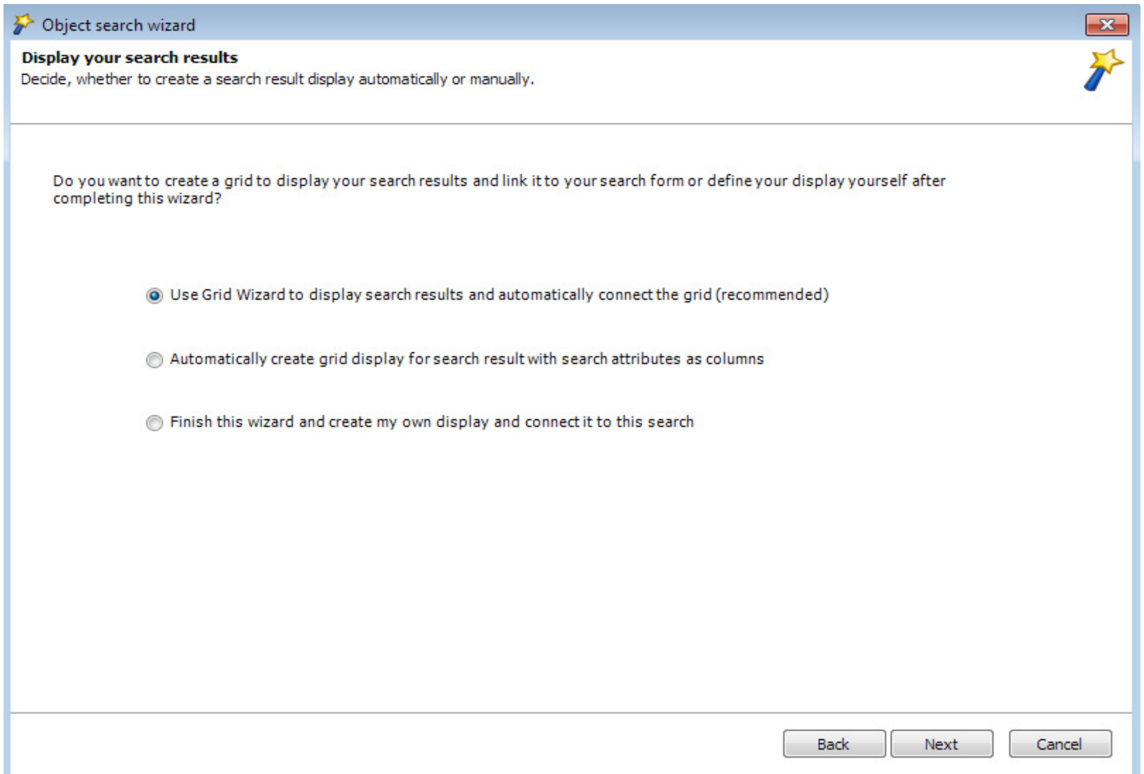
On the following page of the wizard, define which columns of the selected database table should be included in the search. Foreign key columns are automatically resolved for this. Therefore, a list is created on the search screen for columns of this type that displays the name ("DisplayValue") of the target table objects with a foreign key relation like this.

Creating an Object Search Form – Specifying the Search Criteria



Once the parameters have been specified for the search screen, you can configure how the search results are displayed. For this, you have the option to configure the columns to be displayed in the result list yourself (option Use Grid Wizard to display search results and automatically connect the grid) or to simply use the search columns in the result list as well (option: Automatically create grid display for search result with search attributes as columns). You also have the option of not configuring how to display the results in this step but to connect it later manually (option: Finish this wizard and create my own display and connect it to this search).

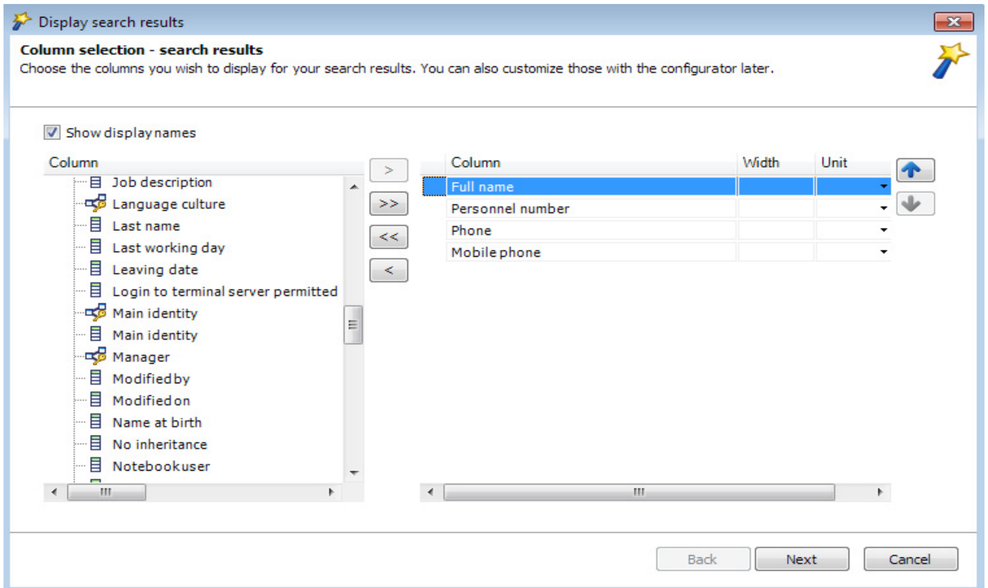
Creating an Object Search Form – Ways of Specifying the Search Criteria



If one of the last options was selected, the wizard configuration is complete. If the Use Grid Wizard... option should be used for displaying the search results then this also has to be configured.

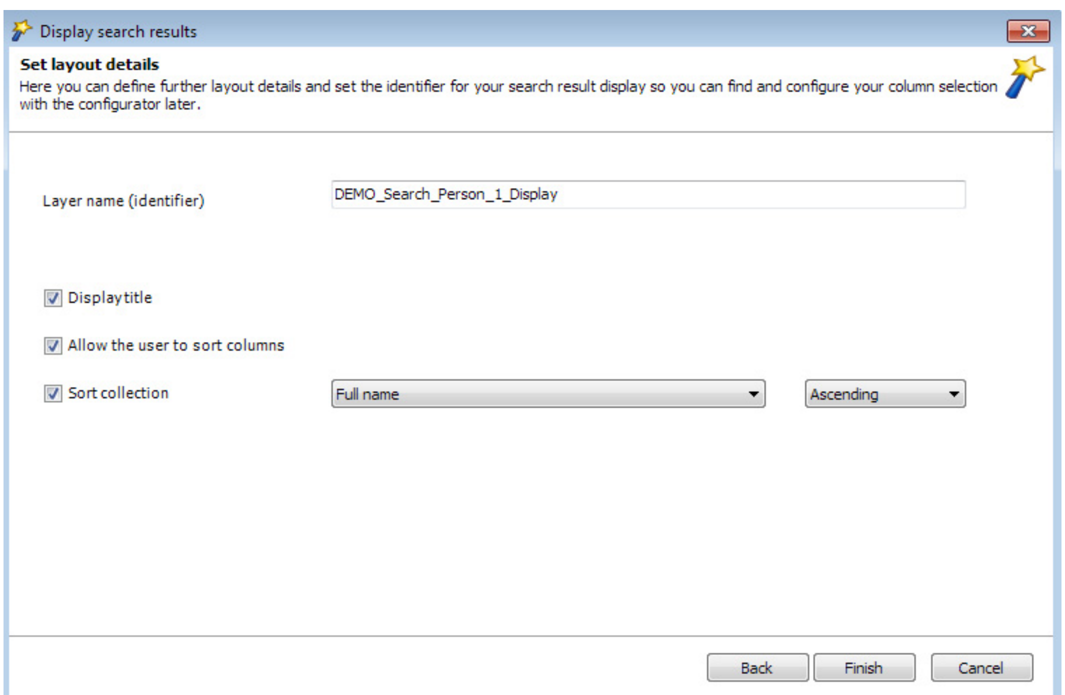
For this you first have to specify which object properties should be displayed in the search result list. The page for this is similar to the one for specifying the search attributes. You can change or extend the selected columns in the configurator at a later date.

Creating an Object Search Form – Specifying the Search Result Columns



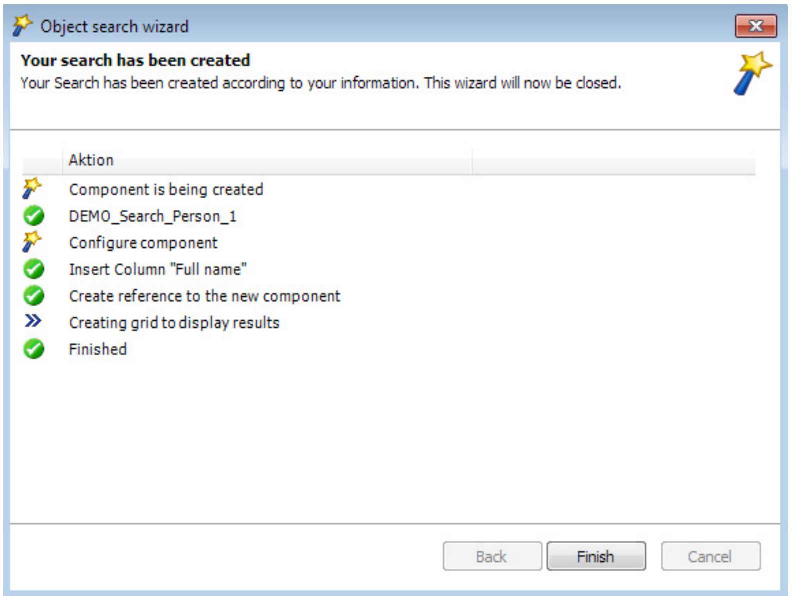
On the next page you can still configure how to display the search results. The configuration parameters are listed under the layer identifier. You can use these to alter the search results display at any time. The option Display title shows the columns titles and the option, Allow the user to sort columns, shows a button is each column that allow the table contents to be sorted by the respective column. On the other hand, the entries behind the option Sort collection refer to the initial table sort order.

Creating an Object Search Form – Configuring Search Result Display



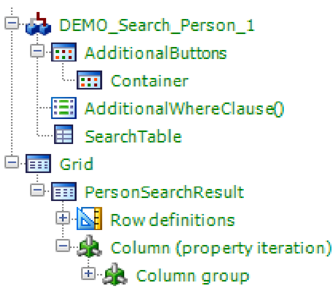
The relevant objects are created and a final status is displayed once the wizard has finally been configured and the Finish button clicked. Close the wizard with the final Finish button.

Create New Project – Creating an Object Search Form – Wizard Confirmation Screen



The nodes displayed in the following figure are those that have been added because they are the positions you can start the wizard from. Apart from this, the components created by the wizards are referenced here.

Creating an Object Search Form – Nodes Added to Definition Tree



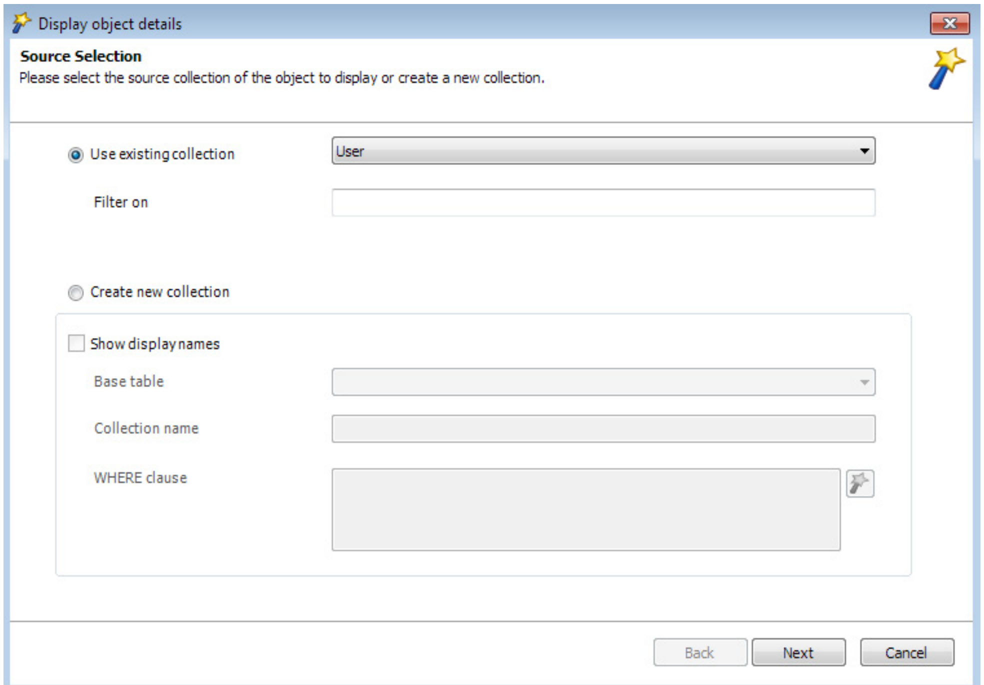
Displaying Single Objects

By selecting the option Create detailed display and input fields for single objects on the wizard startup screen you open a view for single data objects (that is why that it is not a grid display but a “section”). A section of this type is created for every object in the underlying collection (iteration). After the wizard is complete you can still add a filter condition to the added container (iteration). This is used to restrict the number of objects included in the iteration. Which of the object properties are going to be displayed is specified during the wizard configuration.

In the first configuration step of the wizard, enter the collection where the data objects are stored. You can select an existing collection or you can create a new collection.

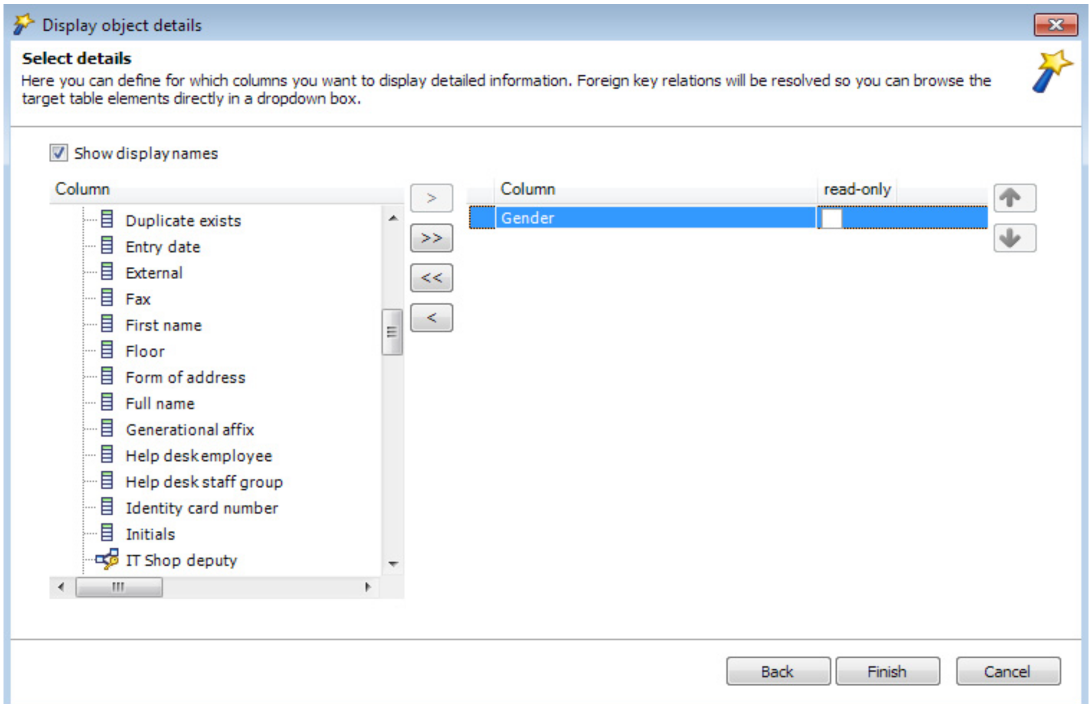
An SQL syntax editor is also available here to simplify Where clause definitions. The Where clause can also contain an internal variable as in the rest of the Web Designer (keyword "Web SQL"). In this case the SQL syntax editor cannot be used. The Where clause can still be added or edited after the wizard is closed.

Displaying Single Objects – Selecting the Collection



In the second step of the configuration, you need to specify which database properties should be displayed. Screens from other wizards are used to achieve this. If the option read-only is not set, you can edit the corresponding property in the web application. You need to add another button once the wizard is complete, so that the changes can be saved in the database. The columns that are shown can be changed later by editing the configuration key that has been added.

Displaying Single Objects – Selecting Properties to Display



When you select the Finish button the wizard creates the relevant display. The actions that have been executed are displayed on a confirmation page.

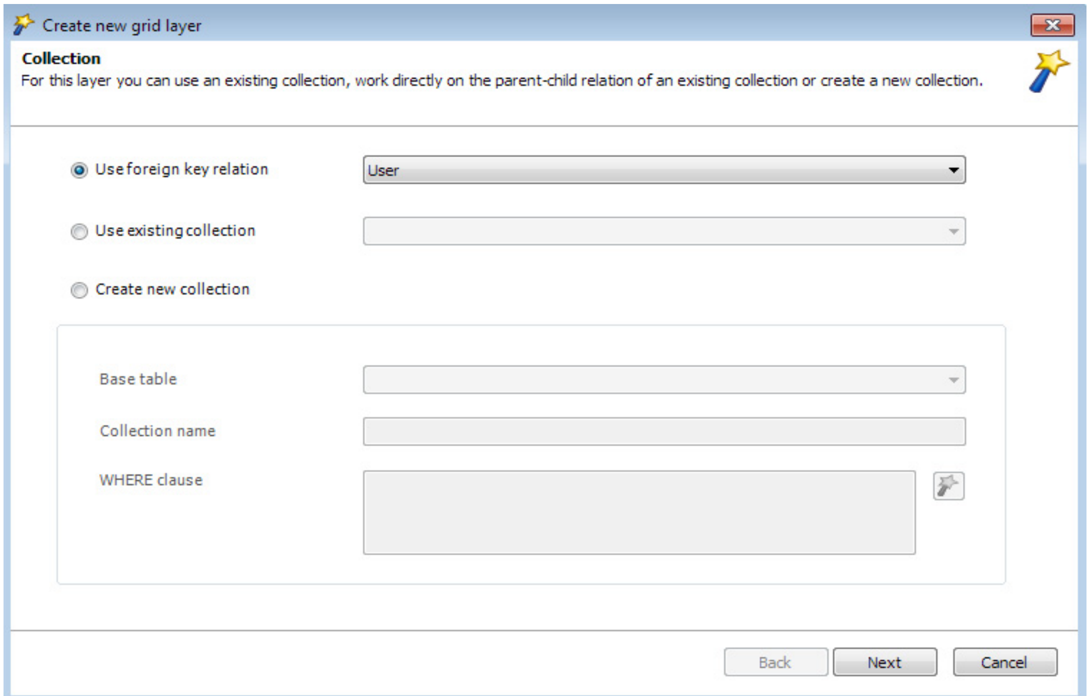
Grid Display for Collection Data

This part of the wizard creates a grid view for data from one or more collections. You can assign data object hierarchically in the process. That is why data objects that are defined as dependent (foreign key relation) on another database object that is stored in the same or another collection can be displayed in another layer. If you want to create a grid of this type you need to select the option Use foreign key relation. Select the option Use existing collection or Create new connection for grids that only contain one layer.

If the dependent data objects are in the same database table it may be referred to as "parent-child relation". This is really, however, a special case of a foreign key relation.

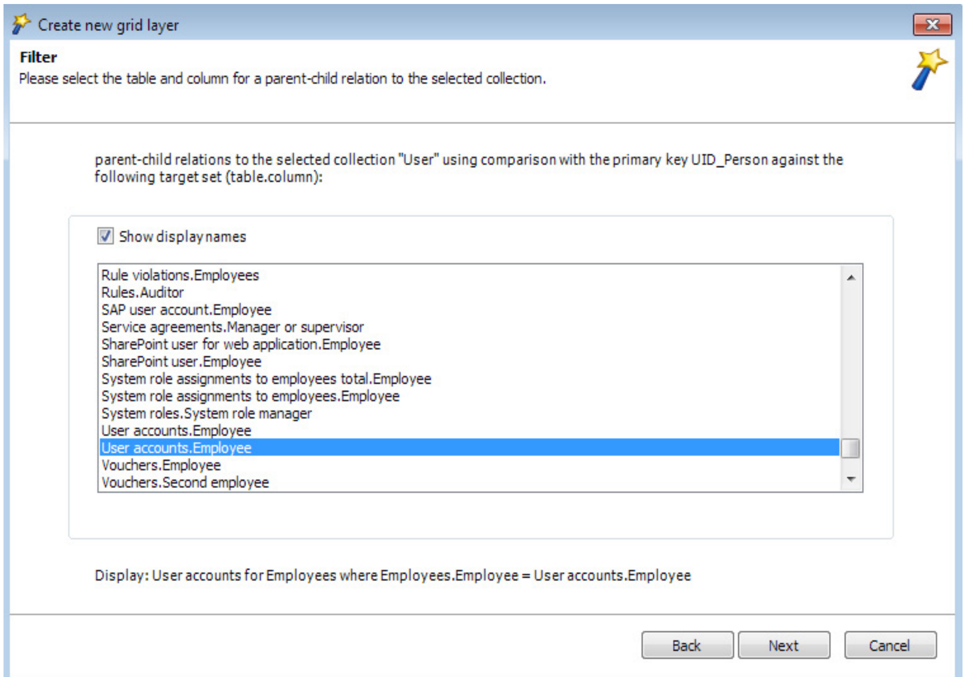
In the first screen in this part of the wizard you configure the upper layer of the display first. Then the dialog box allows you to add further layers.

Grid Display – Selecting the Collection



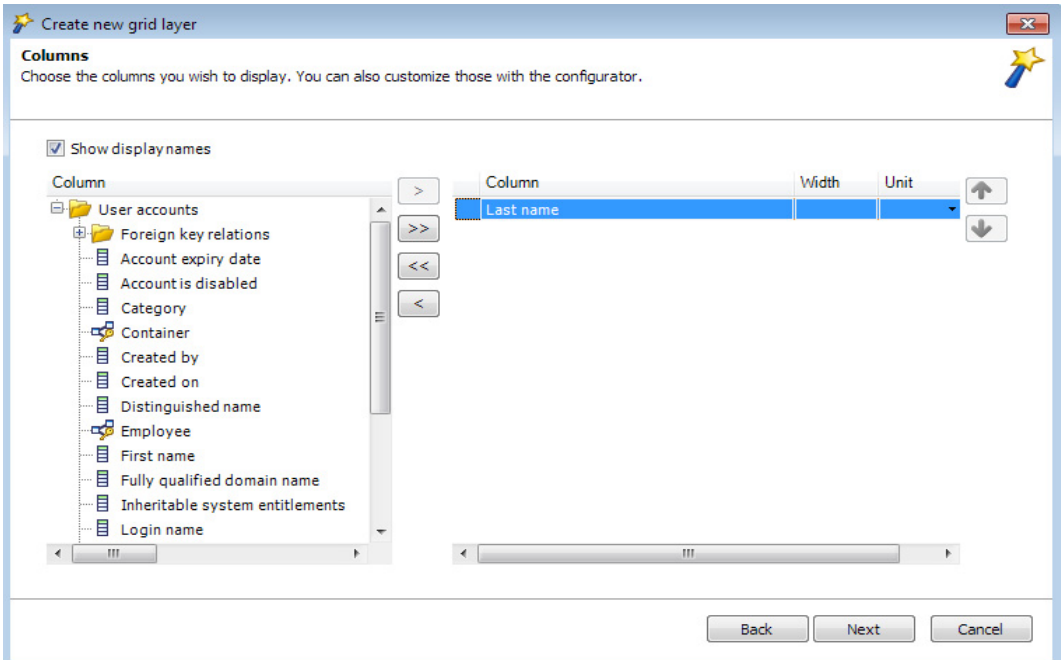
For each layer, you need to specify whether a foreign key relation is going to be used or not. This only makes sense to select this option for the top layer if a parent-child relation is used for the relation definition. Select the collection property that the parent-child relation is mapped to, on the following screen. This screen is missed out if no foreign key relation should be used. You must enter a foreign key relation for any further layers that you define.

Grid Display – Selecting the Foreign Key Relation



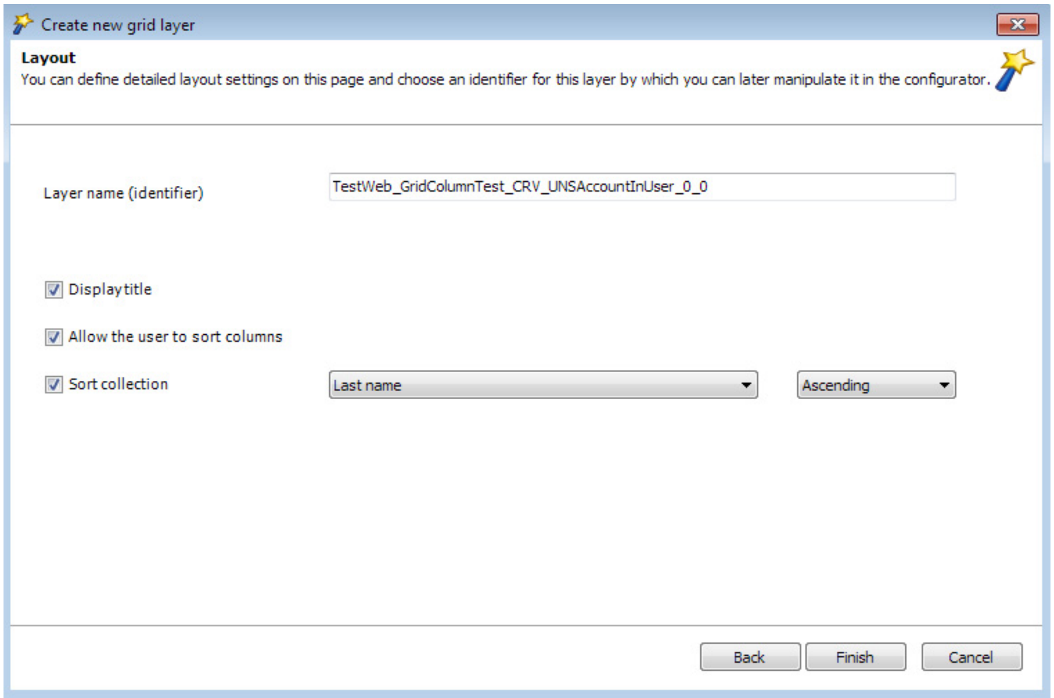
Select the collection properties that are going to be displayed in the following step. Double click or mark and click the selection button > to add the properties to the left hand column in the view (multi-select is possible using the Control button). This lists all the properties that will be finally contained in the grid. The order for displaying the properties in the grid (from left to right) corresponds to structure of the order from top to bottom. This order can be changed by marking a property in the structure and using the button on the right hand side of the view. The wizard can also resolve foreign key relations for the current collection up a depth of 2 steps. For foreign key relations that are resolved in such a way, the property name to be displayed can be altered in the structure. You can now add the desired width for each column selected. Note here that mixing percent and pixel values can lead to unexpected results. The checkbox display names can be swapped between column names (that means column display values entered in the database) and database internal column names.

Grid Display – Selecting the Properties to Display



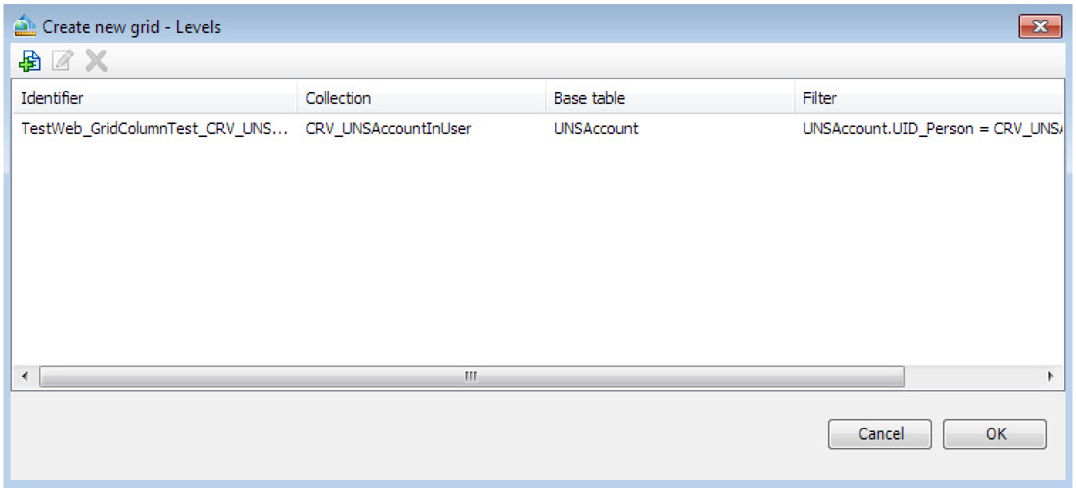
Subsequently, you can specify your own layout details and a caption for the current layer. The layer caption is only used help the search for properties that you may want to change later. Set the option Allow the user to sort columns if the users are allowed to sort database objects themselves. Then the grids can be resorted by clicking on the column title bar. The option Sort collection and specifying a property results in the database objects are initially sorted by this criteria.

Grid Display – Layer Configuration



This data completes the definition of the first grid layer. By clicking the Finish button you open another page. This displays the grid layers that are already defined. You can edit or delete a layer by selecting it. Use the plus button to define more layers.

Grid Display – Displaying Existing Layers



After you have selected the plus button, the screen for selecting or adding a collection is shown. Configure the second and further layers in the same way as the first layer but always enter a foreign key relation. By selecting the OK button on this screen, the grid with all layer defined so far is created and the wizard is closed.

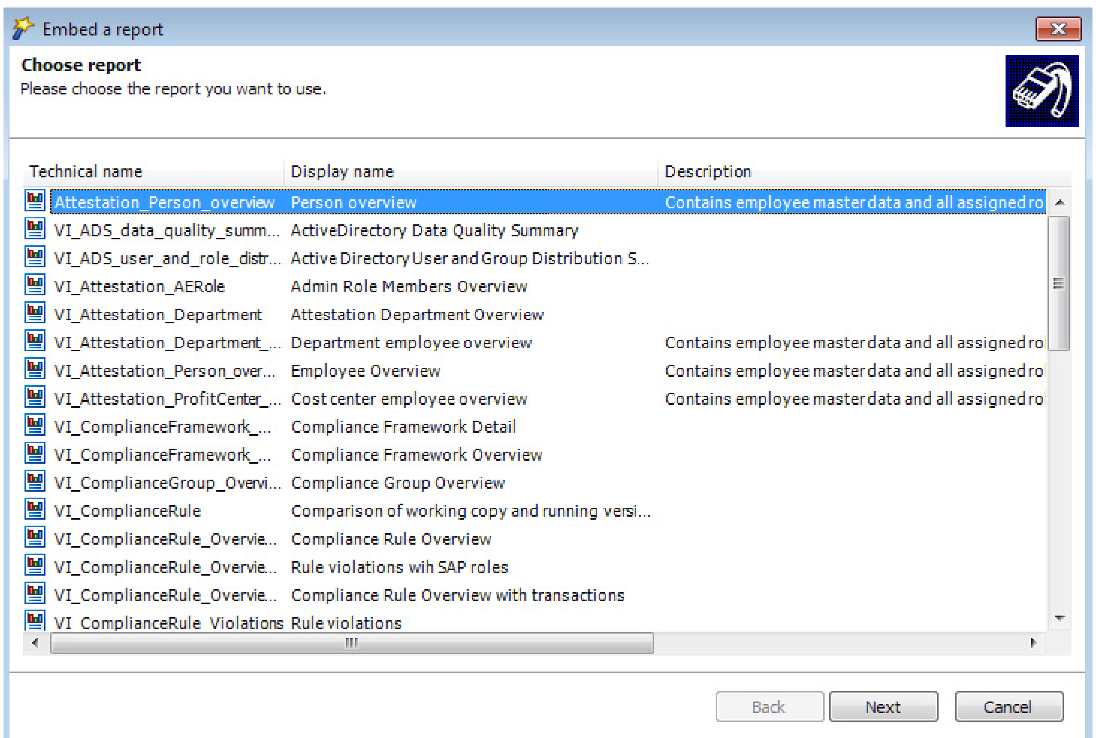
Report Embedding Wizard

Use the Report Embedding wizard to help you implement a function which calls a defined, meaning otherwise programmed report stored in the database from a web application. If the report requires a parameter, these are determined from the web application and passed to the report. Then the web application can display the report. The wizard can only be run from under an execution node because displaying report is preceded by a calculation. Therefore, the wizard is available in the action node context menu.

The wizard start screen lists all the reports stored in the database. Select the report you want to display and confirm with the Next button.

Reports are created and edited with the Report Editor tool.

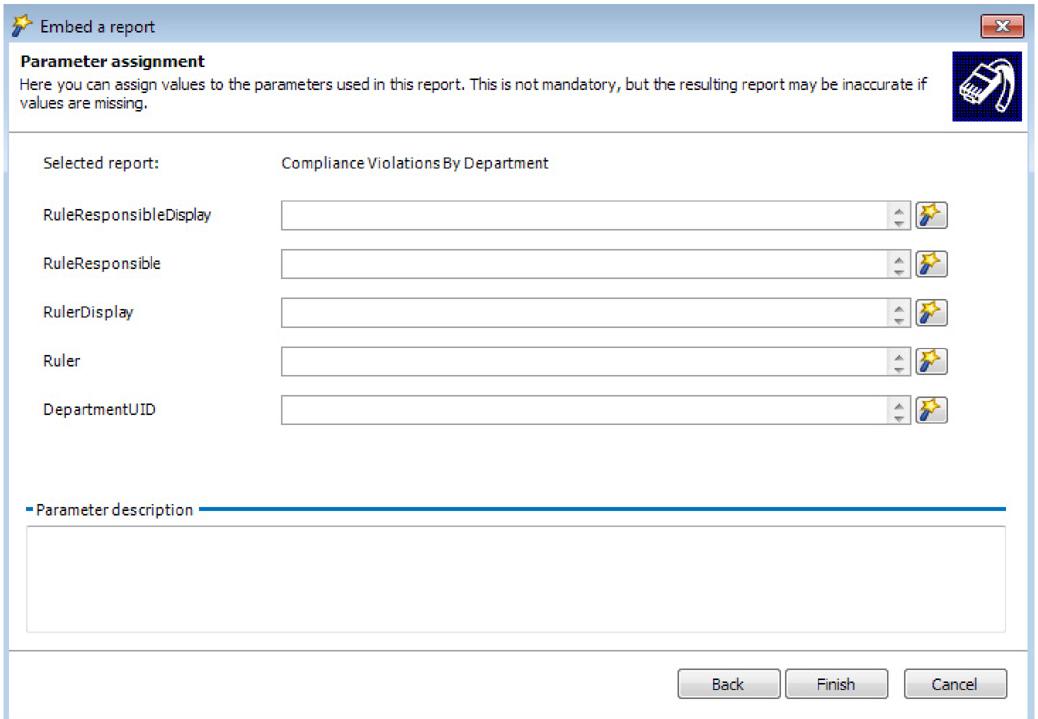
Report Embedding Wizard – Select Predefined Report



Define the parameter to use in the second step of the wizard. One input field is displayed on the screen for every parameter that is required. The magic wand icon opens a larger input window. You can add a

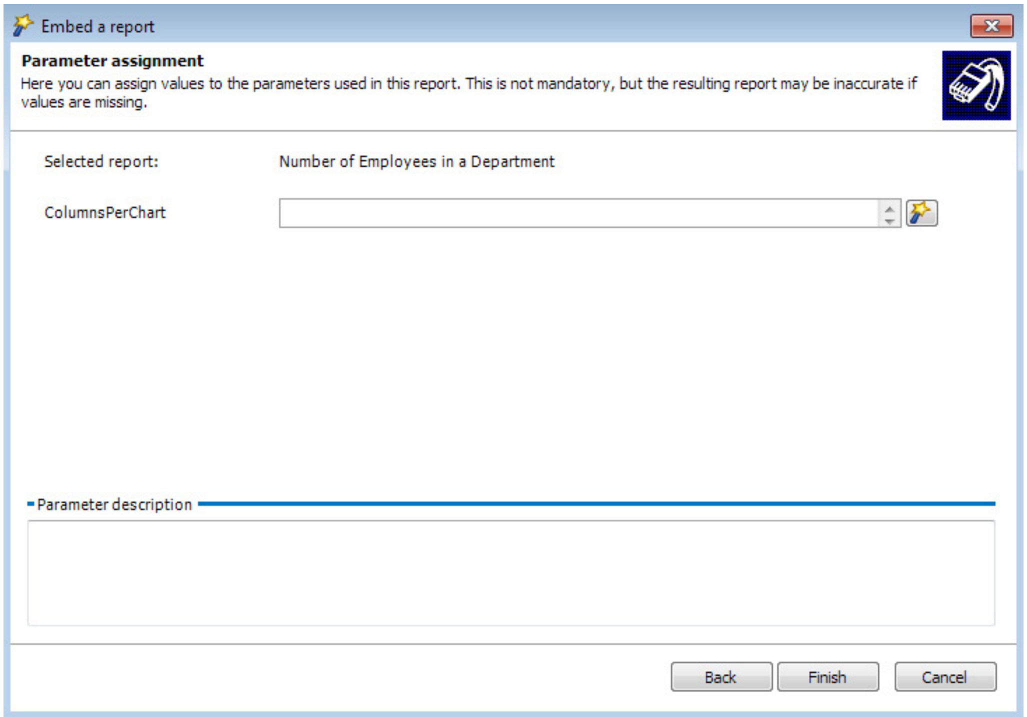
Web SQL statement that returns the required parameter value. This is normally involves one single primary key per parameter.

Report Embedding Wizard - Allocate Parameters, Ex. Primary



You can also apply quite different parameters. If a report does not need a parameter an appropriate comment is shown on the screen.

Report Embedding Wizard – Allocate Parameter, Example Fixed Value



After you have assigned all the parameters, close the wizard with the Finish button. As a result, the wizard creates a “Compile report” node and one called “Forwarding”. Forwarding is based on a default component. This takes over the task of displaying the report. The Web SQL statements are saved during configuration by the wizard under the node “Compile report” in the definition tree and can be edited at any time.

Mapping Generator Wizard

Use the node type from the category Referenced components to reference new components. Normally, you have to define additional information, which is expected by the referenced components, on the child nodes of type node mapping, function mapping and/or collection mapping. If not all the expected mapping nodes are defined it results in compiler errors.

The wizard assumes the task of loading the expected mapping nodes for the component to be referenced and automatically adds the component reference. Therefore, you do not need to manually check which mapping nodes are required. However, the mapping nodes added by the wizard can only be partially automatically configured. So you do need to manually post process them.

You can start the Mapping Generator wizard from the context menu but it is only available from nodes of the category Referenced components and Start execution. It is not necessary to be able to start the wizard from the Container (Link) node because the necessary mapping nodes are added immediately once the referenced component is selected. In all other cases, you first have to select the component to reference and then run the wizard. The mapping nodes are then added without opening another window.

7

Data Operations

- SQL Server versus Web Server
- Parameter and Function Usage
- Autocomplete

SQL Server versus Web Server

Data is selected and manipulated at many Web Designer nodes. The programming language used for this purpose – referred to in this document as Web SQL – can be regarded as an MS SQL offshoot to which special functions have been added in various places.

In this context, web operations can be realized by either an SQL server or a web server. Web server data operations can, of course, only be realized in data from previously loaded collections of the current module or the session module. All collections that can be loaded at any given time are referred to hereinafter as a data set.

Inasmuch as all collections are loaded via the SQL server, the "sql" function can be omitted from the SQL expression of the relevant node. In all other places where data is queried, it will be assumed that the query refers to a collection. The SQL server can always be queried directly using the "sql" function, a mechanism that opens up a broad spectrum of options for web project implementation. The decision as to which variant to use should be chiefly based on performance considerations, since SQL and web servers process the tasks that come into play here at differing speeds.

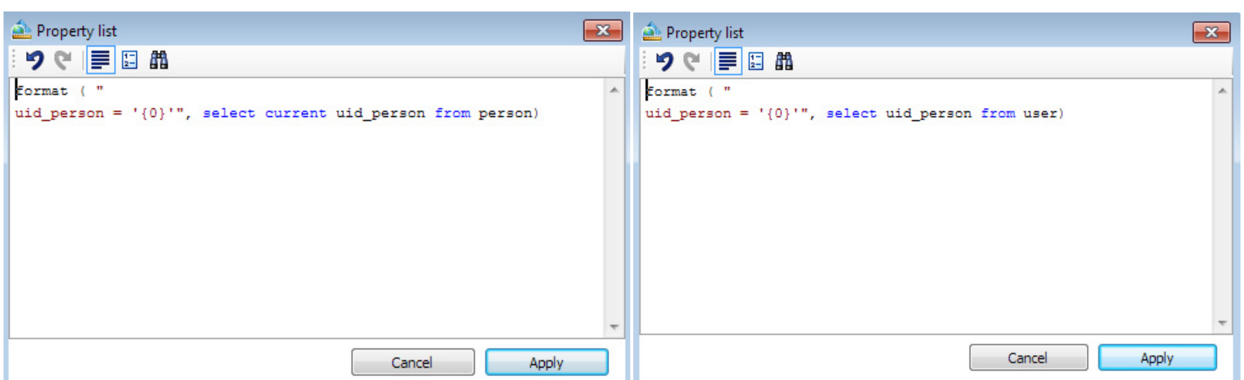
Parameter and Function Usage

Parameter-based data filtering is a standard web development task that is supported by many different types of nodes. The following two scenarios come into play here:

- The current cursor identifies the parameter in a grid and filtering is realized using the current function.
- The parameter (or in this case the parameter list) is generated via a process step while the web application is being used, and is available in a collection as a data set (for example, a person selected from a list of search results).

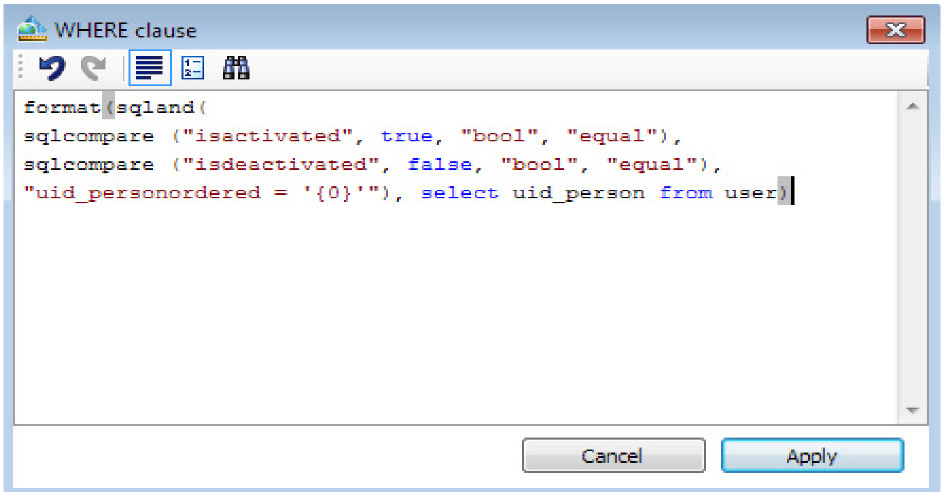
The use of parameters is enabled by the format function; and a {<counter>} placeholder is to be used initially in the query. The counter begins at zero and is increased by one digit for each additional parameter. To this query, which is to be evaluated by the SQL or web server, is added a statement for each parameter that identifies the parameter value that is currently needed from the data set.

Example of a Parameterized Expression



However, the format function cannot be used for parameters, but is instead used in cases where queries are to be realized dynamically, in which case functions can also be nested in an appropriate fashion. Functions are always implemented from inside to outside. In the example shown in the following figure, this means that the two sqlcompare functions are realized first, followed by the sqland function and then the format function, which computes the valid SQL query for the SQL server query.

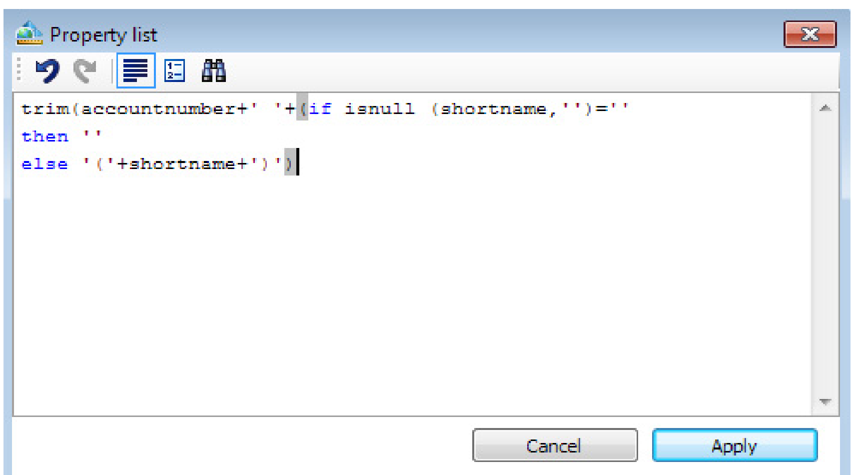
Example of a Function Nesting Procedure



Two character chains (concatenations) can be connected using <+>. In the example shown in the following figure, a line space has been inserted after the account number value using +' '+.

The specimen formulation '('+shortname+')' is the equivalent of using the {0} space holder and the select current short name from cost center formulation using the Format function in the following figure: Example of the connection between two concatenations.

Example of the Joining of Two Strings



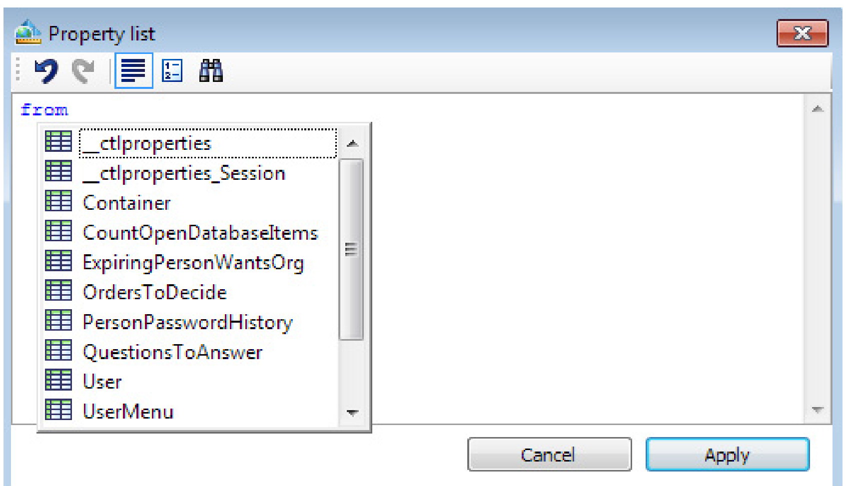
Autocomplete

The tool provides the developer with information and options that lighten the developer's workload and that, above all, reduce the amount of source code that needs to be entered manually. The tool's edit window assists the developer by showing him usable names for collections, properties and functions.

This feature can be called up at suitable places in a query via the ctrl + space shortcut. The actual content options that are offered by the program are determined by keywords in the query.

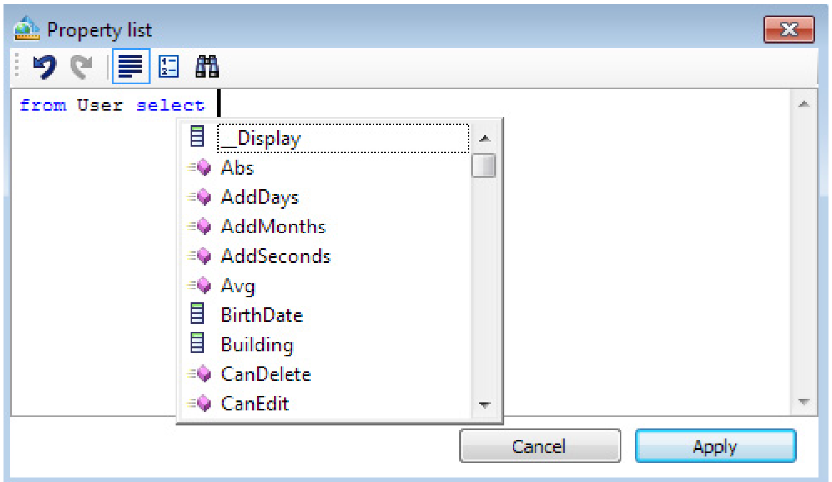
In view of the fact that the list of displayed available column names is unusually long when standard SQL syntax is used in the form select <Property> from <Collection>, the feature is not available in this case. However, the syntax that takes the form of from <Collection> select <Property> can be used instead. If this syntax is used, the available collections are shown in accordance with the keyword "from".

Autocompletion - Selection List for Collections Names



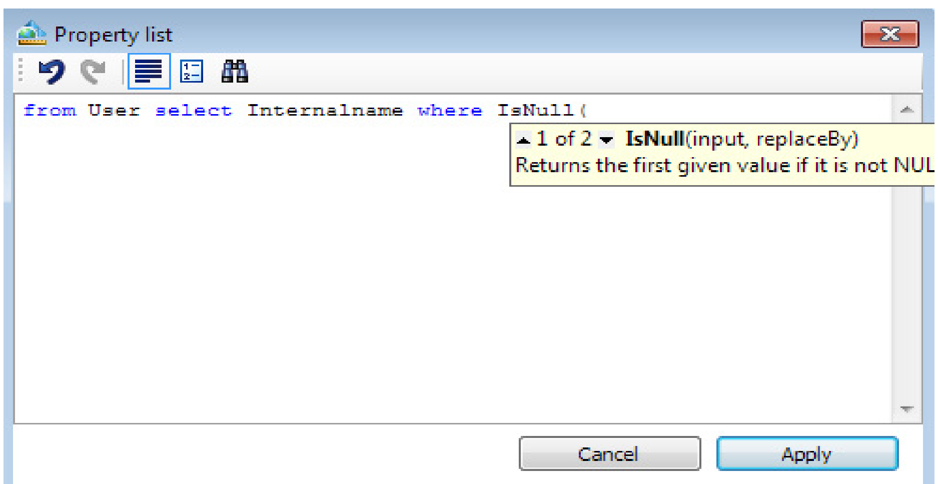
After the keywords “select” and “where,” as well as within a function (for example, after the open parenthesis), a list containing the available properties of the selected collections and the SQL functions is shown.

Autocompletion - Selection List for Column Names (list Icon) and Functions (red Rhombus)



If an SQL function is selected, after the open parenthesis is entered a tool tip appears containing the requisite function parameters. If there are additional allowable parameter combinations, they can be called up individually via the arrow icons.

Autocompletion Function Parameters



8

Web Application Monitoring

- Monitoring
- Status
- Sessions
- Assemblies
- Log Files
- Exceptions

Monitoring

A web application can be monitored from your own web site. This is called by appending "/monitor.axd" to the web application address. The file (monitor.axd) must exist in the web application directory.

The user groups that are authorized to call up the monitoring page are specified in the web application configuration file. By default, this is all the users that are members of the role "Builtin\Administrators".

The monitoring page currently has 5 tabs. The information shown there or rather the available functions are described in the following section. The pages are currently available in German and English. Which language used, depends on the language settings for the valid web application configuration file.

Status

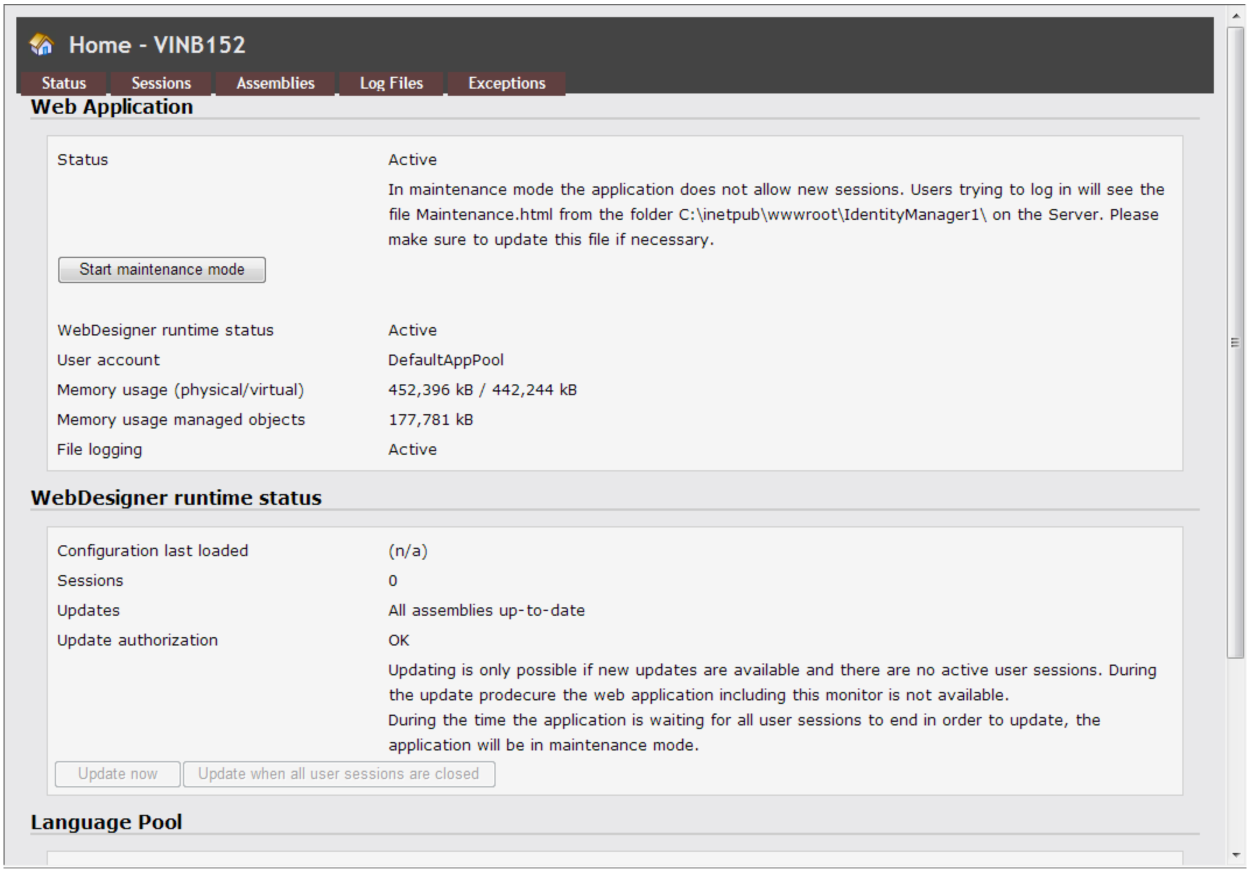
When the monitoring page is opened the contents of the Status tab are displayed. Various, mainly self-explanatory information about the web application status is shown.

It is also possible to put the web application into maintenance mode (button Start maintenance mode). If this is enabled the web server does not accept any new connections. An advice message is displayed to users that try to login.

Assuming new assembly files are stored in the database, it is possible to start a web application update. This can start immediately as long as there are no outstanding connections (button Update now). Otherwise the update can be delayed until everyone is disconnected (button Update when all user sessions are closed). This ensures that no one looses their session or any data they have entered.

The update functions on the monitoring page are independent of those in the configuration file settings.

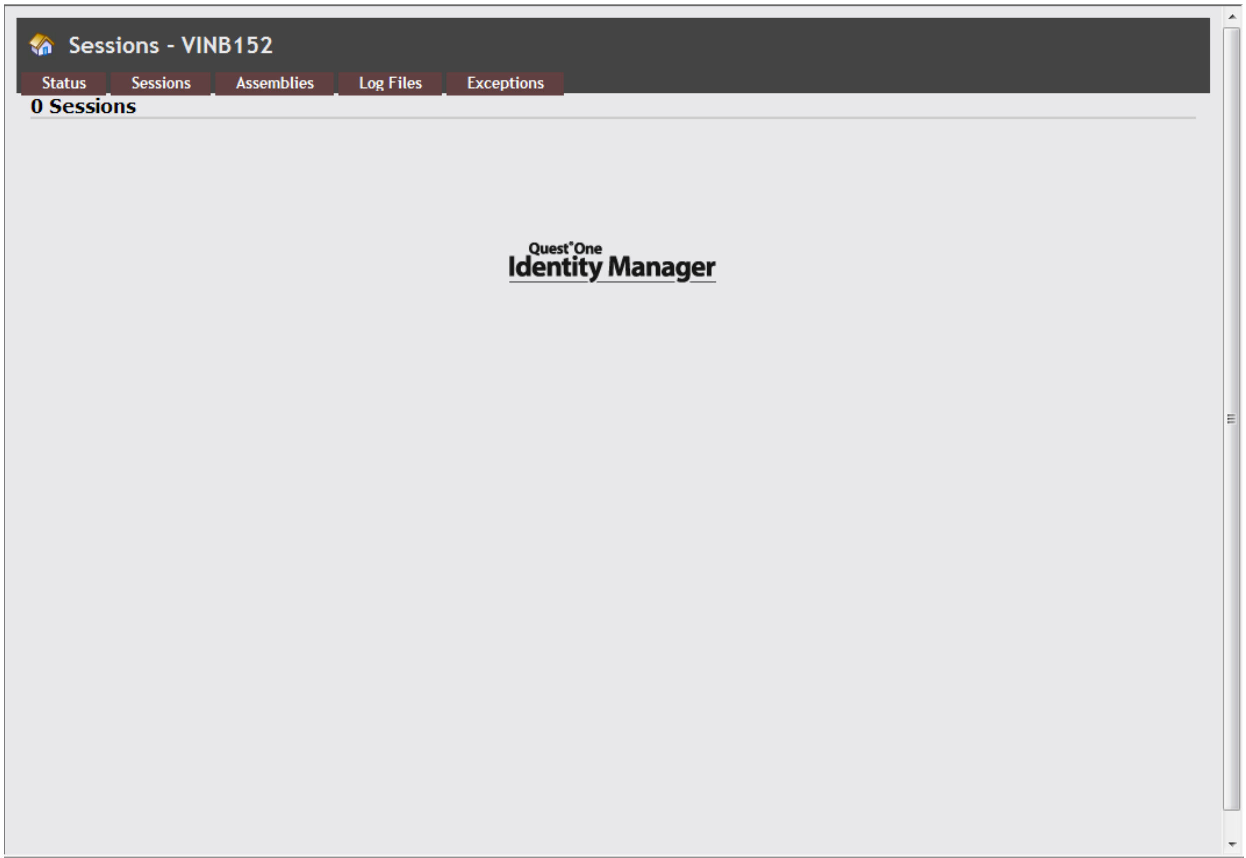
Monitoring – Status Tab



Sessions

The Sessions tab shows information about every single active session and provides the options to log them for debugging purposes (button Enable session log) or to abort them (button Terminate this session).

Monitoring – Sessions Tab



Assemblies

The web application assembly files are listed on the Assemblies tab along with their respective assembly and file version. Authorized employees can quickly get an overview of the development status currently in use. Usage of different file versions in different environment or on different server can sometime be a difficult source of an error to find.

Monitoring – Assemblies Tab

Assemblies - VINB152

Status | Sessions | Assemblies | Log Files | Exceptions

Name	Assembly Version	File Version
AE.WebDesigner.PlugIn	5.0.2011.331	5.0.104.0
AedsDatatypeLib	5.0.2011.331	5.0.61.0
App_global.asax.wbmammrw	0.0.0.0	0.0.0.0
CppCodeProvider	8.0.0.0	8.0.50727.762
Microsoft.JScript	8.0.0.0	8.0.50727.4927
Microsoft.VisualBasic	8.0.0.0	8.0.50727.5420
mscorlib	2.0.0.0	2.0.50727.5444
SgmlReaderDll	1.8.7.0	1.8.7.0
Stimulsoft.Report	2011.1.1000.0	2011.1.1000.0
Stimulsoft.Report.Web	2011.1.1000.0	2011.1.1000.0
System	2.0.0.0	2.0.50727.5420
System.Configuration	2.0.0.0	2.0.50727.5420
System.Core	3.5.0.0	3.5.30729.5420
System.Data	2.0.0.0	2.0.50727.5420
System.Drawing	2.0.0.0	2.0.50727.5420
System.EnterpriseServices	2.0.0.0	2.0.50727.5420
System.Transactions	2.0.0.0	2.0.50727.4927
System.Web	2.0.0.0	2.0.50727.5420
System.Web.Mobile	2.0.0.0	2.0.50727.5420
System.Web.RegularExpressions	2.0.0.0	2.0.50727.4927
System.Web.resources	2.0.0.0	2.0.50727.4927 (NetFXspW7.050727-4900)
System.Windows.Forms	2.0.0.0	2.0.50727.5420
System.Xml	2.0.0.0	2.0.50727.5420
VI.Base	5.0.2011.331	5.0.66.0
VI.DB	5.0.2011.331	5.0.70.0

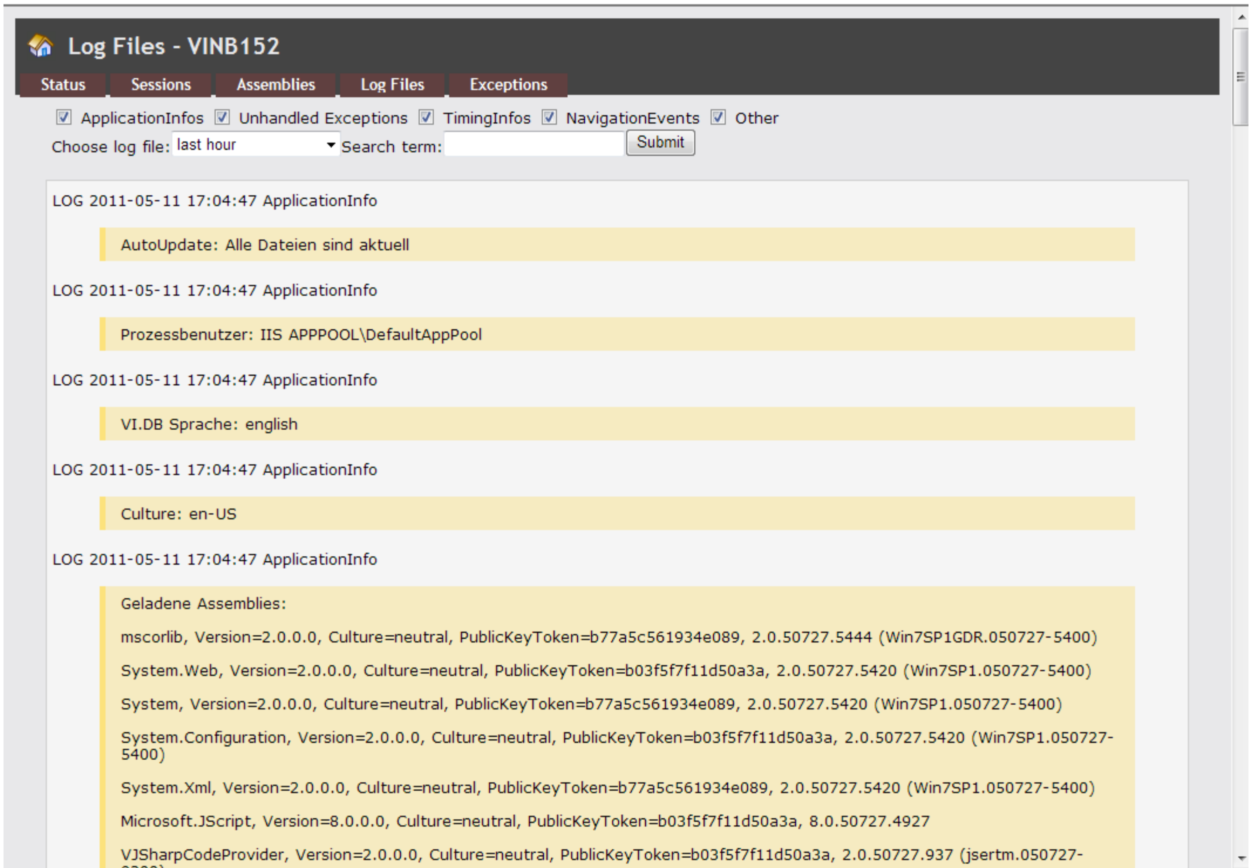
Log Files

The contents of the log files of a web application will be shown on the tab Log Files. The site use the path to the log files as it is deposited in the configuration file (web.config) of the web application.

To achieve a higher performance, only the latest 500 log entries are displayed. On reaching this limit the user receives a corresponding note. This restriction concerns only the transferred amount of data to the client. If the search function is used on a form, all existing log entries are searched.

The site offers a number of filtering options, with which the content of the log files can be analyzed. The filtering options are the same as in the configuration tool. For example, is an expected item not listed on this page, this only means that a corresponding log entry was not written. Details to configure the web application can be found in the *Web Portal Installation Guide*.

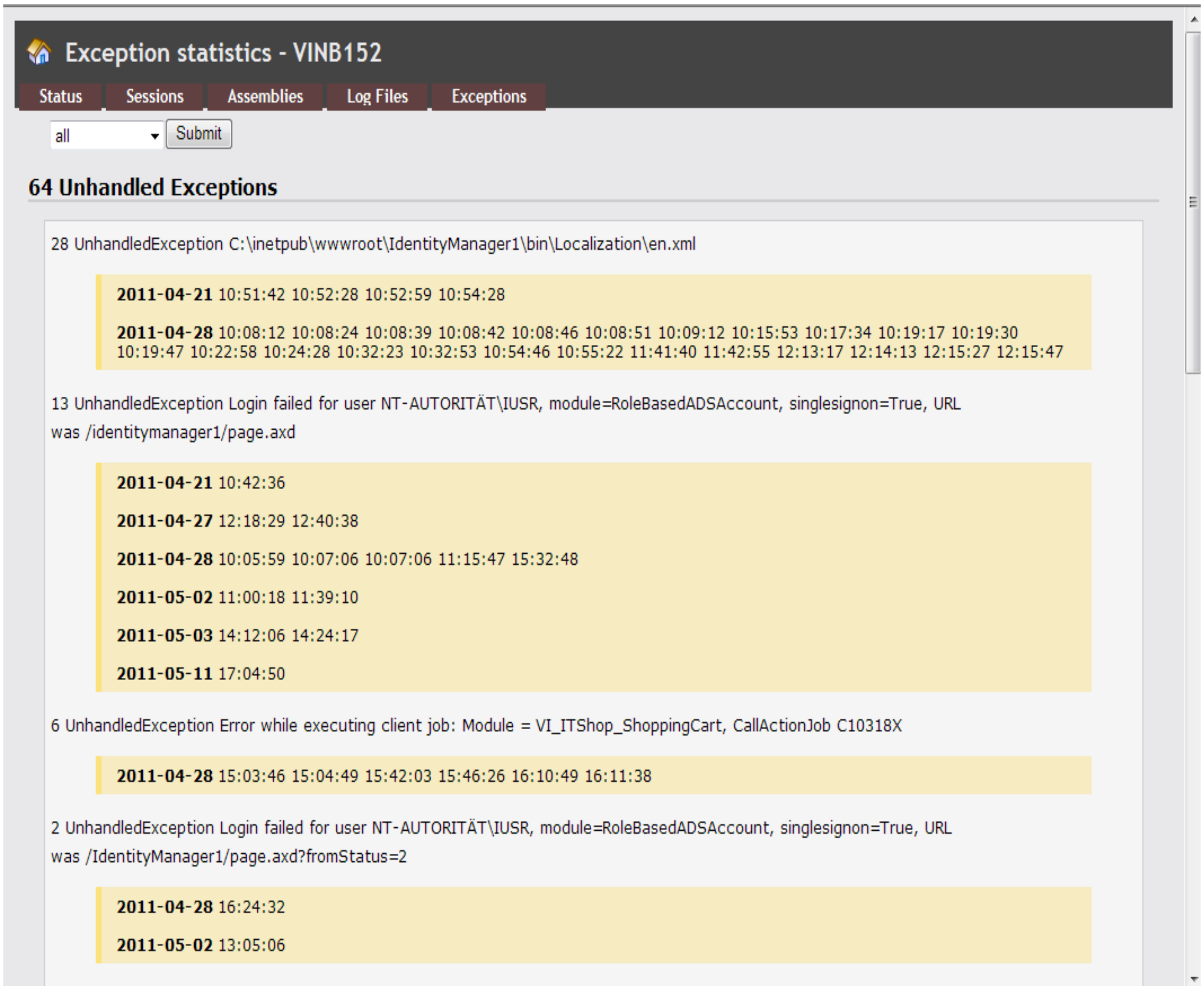
Monitoring – Log File Tab



Exceptions

If during the execution of web application unhandled error (exceptions) have occurred, are displayed on the tab Exceptions. It is a view of the log files of the web application (see also Log Files on page 105).

Monitoring – Exception Tab



9

Helpful Hints

- Editing Captions
- Adding Functions
- Pasting in Texts or Captions
- Pasting in Grids
- Grid Grouping Function
- General Grid Export Function
- Grouping Options for Bar Charts
- Visualizing Error Messages in the Web Portal
- Display Node Layouts
- Saving Login Data for Web Projects without User Related Login
- Replacing Resource Files Images
- Enhanced accessiblity settings

Editing Captions

In order to customize texts that are displayed in web applications, it is necessary to edit so-called caption objects, which, like all database objects, are labeled either as Quest-defined or customer-specific objects. Caption objects are the only Quest-defined objects that can be modified in a customer installation. This option can be used without creating module copies, as it suffices to merely modify web page captions. Moreover, it makes no difference whether a Quest-defined caption object is referenced in a Quest-defined or customer-specific module, since the caption can be edited in both cases. Customer-specific caption objects only occur in customer-specific modules.

The following caption editing rules and procedures apply:

- Caption objects can be referenced by more than one node. In such cases, the changes that are made also apply to these nodes, which may be on completely different pages of the web application. To find the relevant nodes, it is currently necessary to use the search function, which entails entering the relevant key and searching the entire web project concerned.
- For multilingual web applications, the translations (among other elements) also need to be edited, which entails editing a number of caption objects. These objects can be found in the caption dialog along with the Search in key only and Search in all available languages search settings.

First, load the relevant caption object in the caption dialog. Then open the caption dialog through Edit | Captions in the menu bar and find the text that is to be edited. Alternatively, the caption object that is to be edited can be found through the web interface of the Web Designer user interface.

Proceed as follows:

1. Compile the web project.
2. Navigate to the preview window and call up the web page that the text to be edited is located on.
3. Click the **Suppress execution** button.

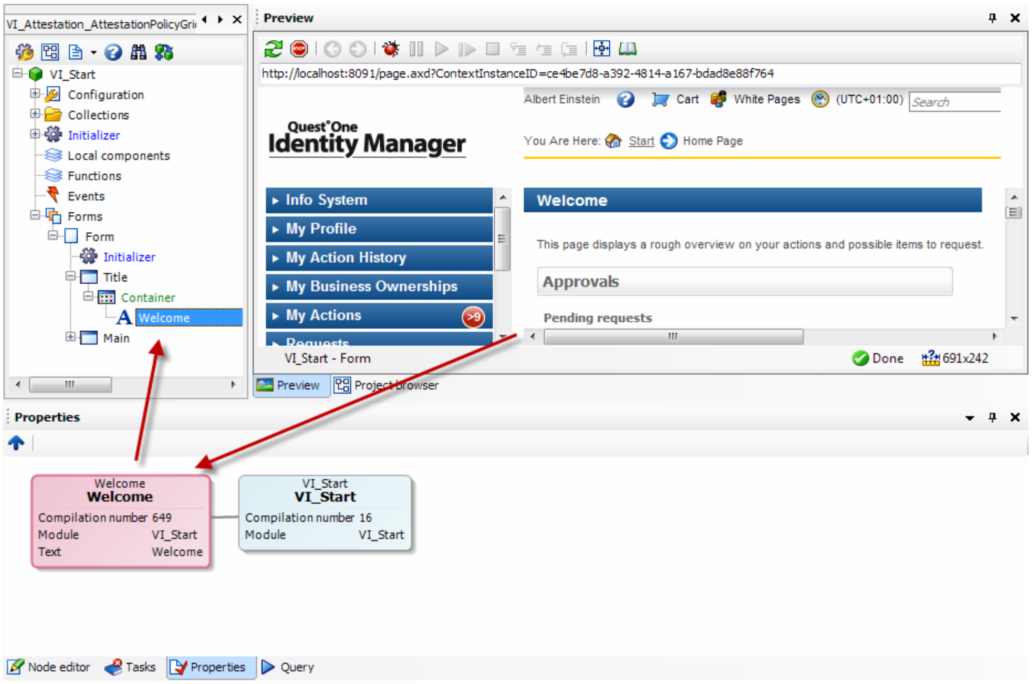
   Only necessary if a link label or button is involved.
4. Click the text in the preview window.

   The text need not be marked.
5. Go to the properties window and click the text bubble there.

   The node that points to the text will be marked automatically in the definition tree.
6. Click on the **Node editor** tab.
7. Click the **Edit** button next to the text box Text and the caption dialog opens.

8. Edit the text and click **OK**.

Helpful Hints: Editing Captions



For a Quest-defined caption object only the customer-specific text field can be edited, which means that the Quest-defined text remains intact. However, the web application automatically shows any customer-defined text.

For customer-specific caption objects, both the customer-specific text and text fields can be edited. Caption object changes must be saved immediately. The relevant save dialog opens automatically when the caption dialog closes.

For customer-specific caption objects, the content of the customer-specific text field is likewise shown, insofar as any such content is available.

The edited text is shown in the preview window following re-compilation.

If any customer-specific text was edited, prior to re-compilation the database scheduler must be run in order for the changes to be shown in the preview window.

If nothing but caption objects were edited, the project need not be approved in order for the changes to also show up in the web application.

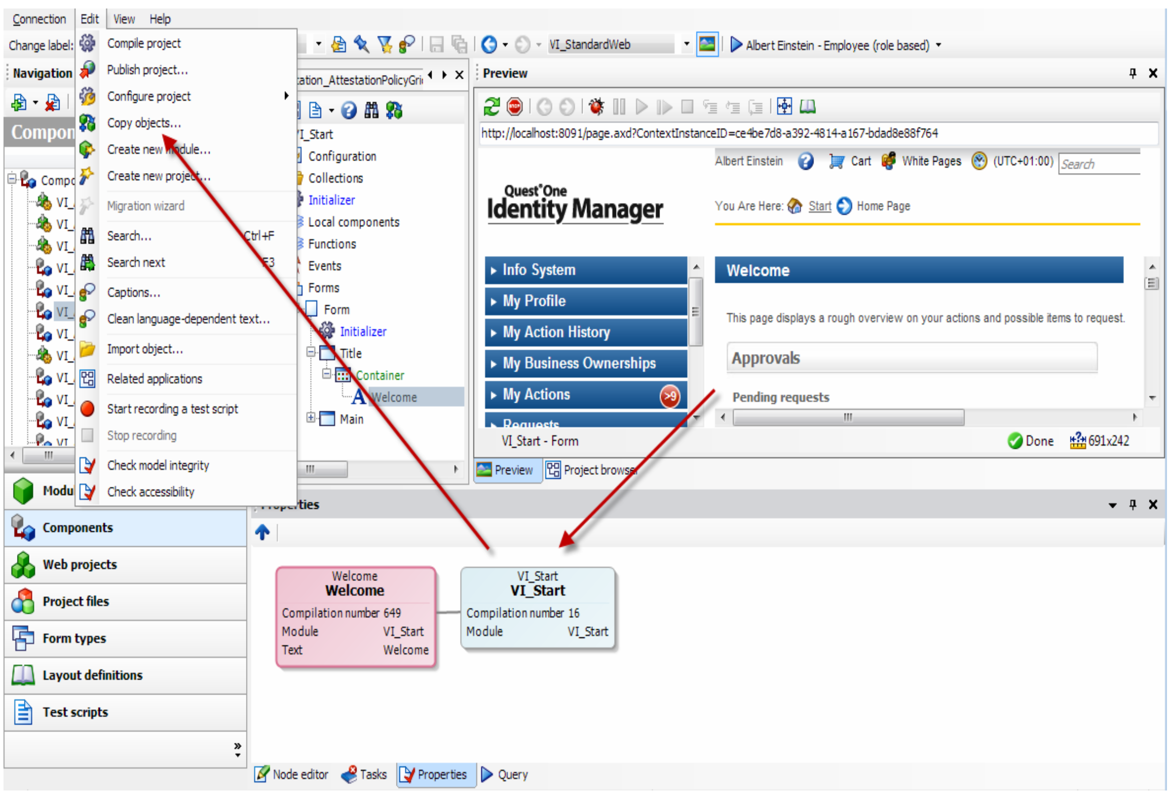
Adding Functions

It is possible to share functions that are already implemented but hidden by using the default web project configuration. If new functions should be added they need to be programmed. Operating processes that are not implemented can be programmed using new modules (see New Module Wizard on page 76). If an existing operating process is being dealt with the default module being used can be extended. To do this the module or components have to found first that the new function will be added to.

The procedure in this regard is as follows:

1. Create a new preview (this includes web project compiling).
2. Go to the preview window. Call up the web page where the new function is to be available and click the desired location on the page.
3. Go to the properties window and click the module bubble there.
4. Click **Edit** | **Copy objects...**.

Helpful Hints: Adding Functions



It is necessary to create an object copy if the module found is a Quest-defined module. Customer-specific modules can, of course, be edited without making a copy.

New functions can now be added to the customer-specific object thus created. The most common scenarios in this regard are described in Pasting in Texts or Captions on page 113 and Pasting in Grids on page 114.

Pasting in Texts or Captions

In order for texts or labels to be displayed on a web page, a label node must be pasted into the definition tree.

The procedure in this regard is as follows:

1. Find the place in the definition tree that represents the desired position on the web page.

   Then proceed as described in Helpful Hints on page 109.
2. Open the context menu by right clicking.

   The context menu will contain a list of additional node types, whose content will vary according the node type that you marked.
3. Select **Display nodes** | **Label**.

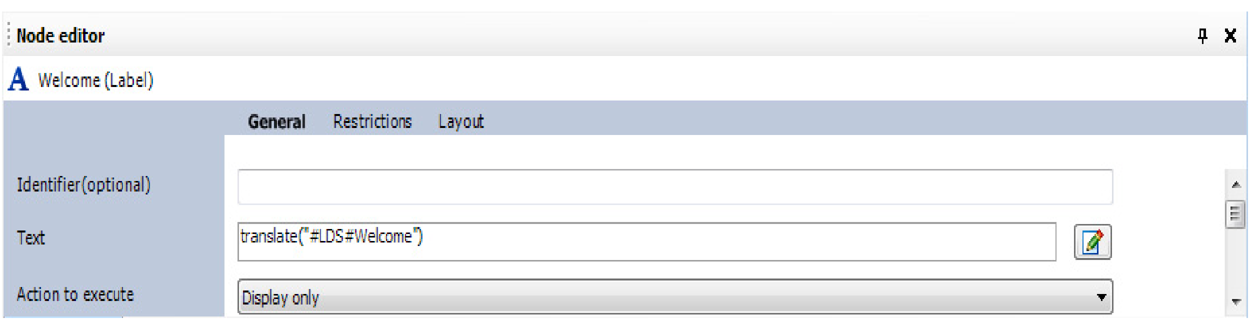
   Once the aforementioned selection has been made, a new label node will be pasted into the definition tree and automatically marked.
4. To configure a new node, go to the node editor window.

   For general information concerning the node editor window, and particularly concerning the Limitations and Display tabs, see Node Editor on page 40.
5. To open the caption dialog, which allows for text selection and creation, click the **Edit** button next to the text box Text.
6. In the **Action to execute** drop-down list, select the action that is to be realized when the label text is clicked in the browser. The following labels can be defined:
   - **Display Only**
   - **Implement Server Action**
   - **Implement Javascript.**

In such cases, a label only has a subordinate Implement Server Action node in an Implement Server Action state. To implement the Action or Javascript defined in the browser, click the label. Normally, labels whose implementation necessitates an action are displayed differently in the browser (depending on the template file entered in the web project).

7. To obtain a valid node definition, the label must contain at least one text.
8. Following compilation, the new text will then appear in the preview window.

Helpful Hints: Pasting in Texts and Captions



Pasting in Grids

A grid shows any database content desired in tabular form. To paste in a grid, first find the place in the definition tree that represents the desired grid position on the web page. To do this, you can proceed as described in chapter Editing Captions on page 110.

Because grids can only be inserted under container nodes, select a node of this type at the desired position or simply insert the node.

To insert grids:

1. Mark the relevant container using the right mouse button, which will open the context menu.
2. Click **Wizards** | **Create data display**.

   The wizard is described in detail in Data Display Wizard on page 78.

As with all definition tree nodes, the inserted grid can be edited in the node editor window (see Node Editor on page 40). The most important options are explained here (for more information see the corresponding entry in the help file, see table Functions in the Navigation Window Toolbar on page 28):

- Allow the user to sort columns

  If this option is selected, the user can double click on the column header to sort it in ascending or descending order. If this option is not set, the user cannot change the chosen sort order for the grid. This option is set in this window generally for all columns but you can also set it separately for each column in the grid.
- Sort collection

  Use this option to specify an initial sort order in the layer. If this option is not set, the data sets are show in the order that they are in the data set. For example, these have a sensible sort order if you have entered appropriate data when the collection was loaded.

The last options relate to the how the column width is fixed and can only be used alternatively.

- Dynamic column width

  Use the option to automatically fix the column widths based on the length of the entries in the rows by applying an internal optimization algorithm.
- Fixed column width

  The option required input in pixels. All columns in the grid are given the same fixed width.
- Custom column width

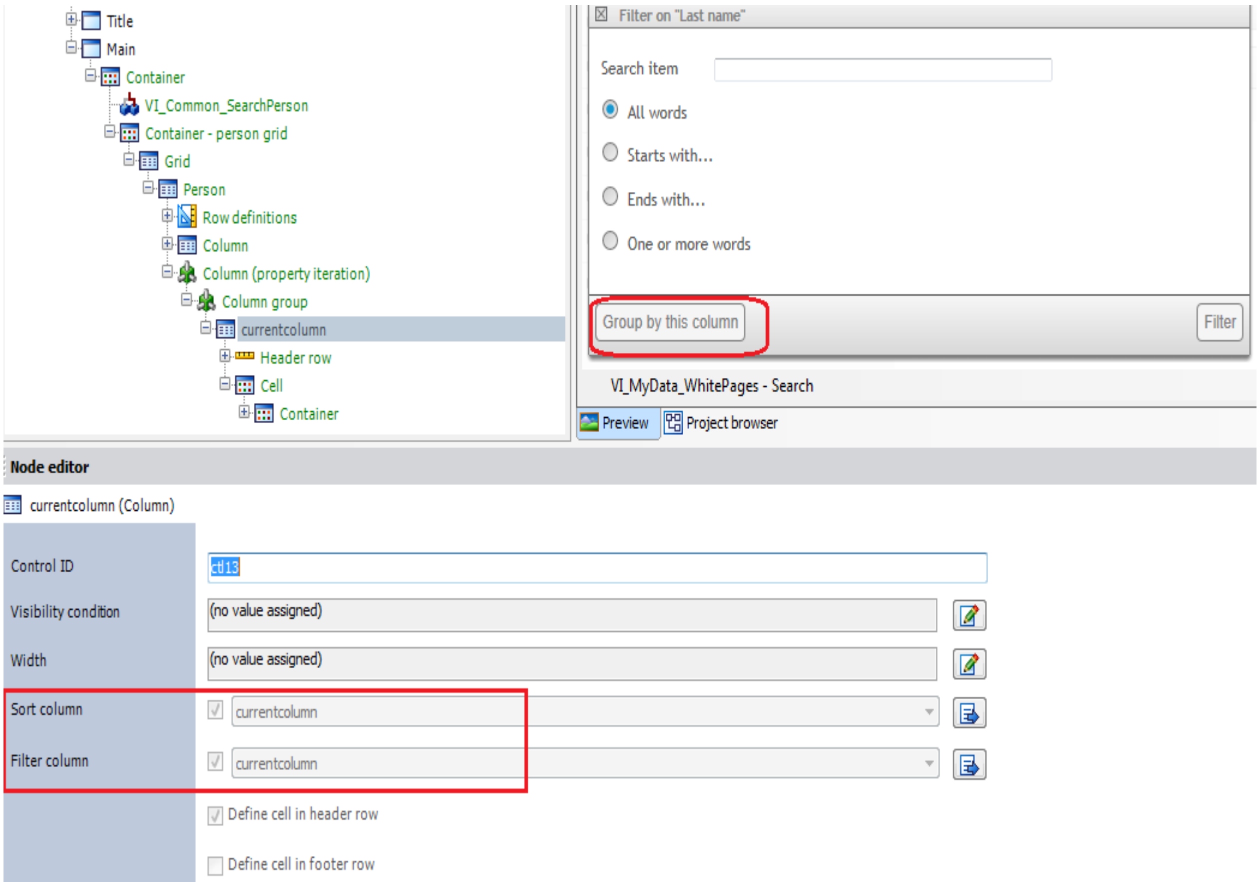
  The option offers the possibility to edit the width and the unit (px or %) for each column individually.

Grid Grouping Function

You can decide whether you want to be able to sort individual columns in the case of single or multi-step grids. If this function is set, a button is displayed in the column above which the columns are sorted alphabetically or numerically, in increasing or decreasing order.

In the case of both single and multi-step grids, you also have the option to group lines in a column. To enable this function, mark the option <Filter column>. The <Filter> button is also shown in the corresponding column. You will find the function <Group by this Column> in the dialog box <Filter by...>. Grouping is done in alphabetical or numerical order.

Grouping Columns



General Grid Export Function

The function <Export this view> is available for many lists (for example, white pages search, request overview). You can select either PDF or CSV file formats.

Export Function

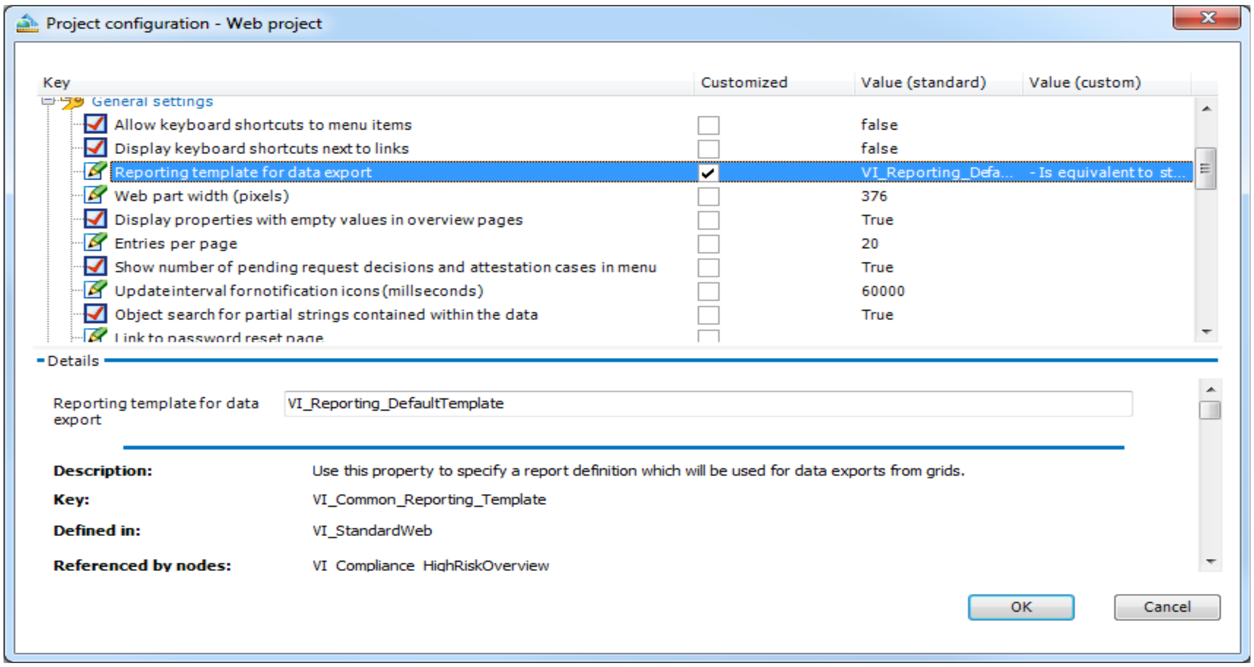
Modifying the Report Template

You can modify the template used for exporting in PDF format. There is a new key in “Reporting template for data export” in the configurator for this (Edit | Web project | General). The default layout in the Report Editor is called “VI_Reporting_DefaultTemplate”. You can copy this template in the Report Editor and modify it (modifying style).

This way, you can embed custom logos and use other color schemes. Specify the name of the new report definition yourself and enter it in the Web Designer Configurator as a custom configuration key.

Ensure the styles in the new report have the same name as in the default template.

Report Template



Hiding Controls

You can use a visibility condition to hide specific control when exporting by checking the environment variable GRIDEXPORT with the following expression:

variable("%GRIDEXPORT%")<>"EXPORT" - this hides the control on export

Replacing Images in the CSV Report

There is a property <Text export mode> on the node type image for CVS export. Use this to select whether the image should be completely left out of the CSV export or whether at least a tool tip should be displayed instead of the image.

Grouping Options for Bar Charts

In the node type diagram, you can change how the diagram is displayed and alter the layout settings with the node editor. You can, for example, customize the entire color scheme, change color of the axes (AxisLineColor) or define another color for the background (FillAreaUnderChart). Choose between the following display options for the diagram:

- Line diagram
- If you

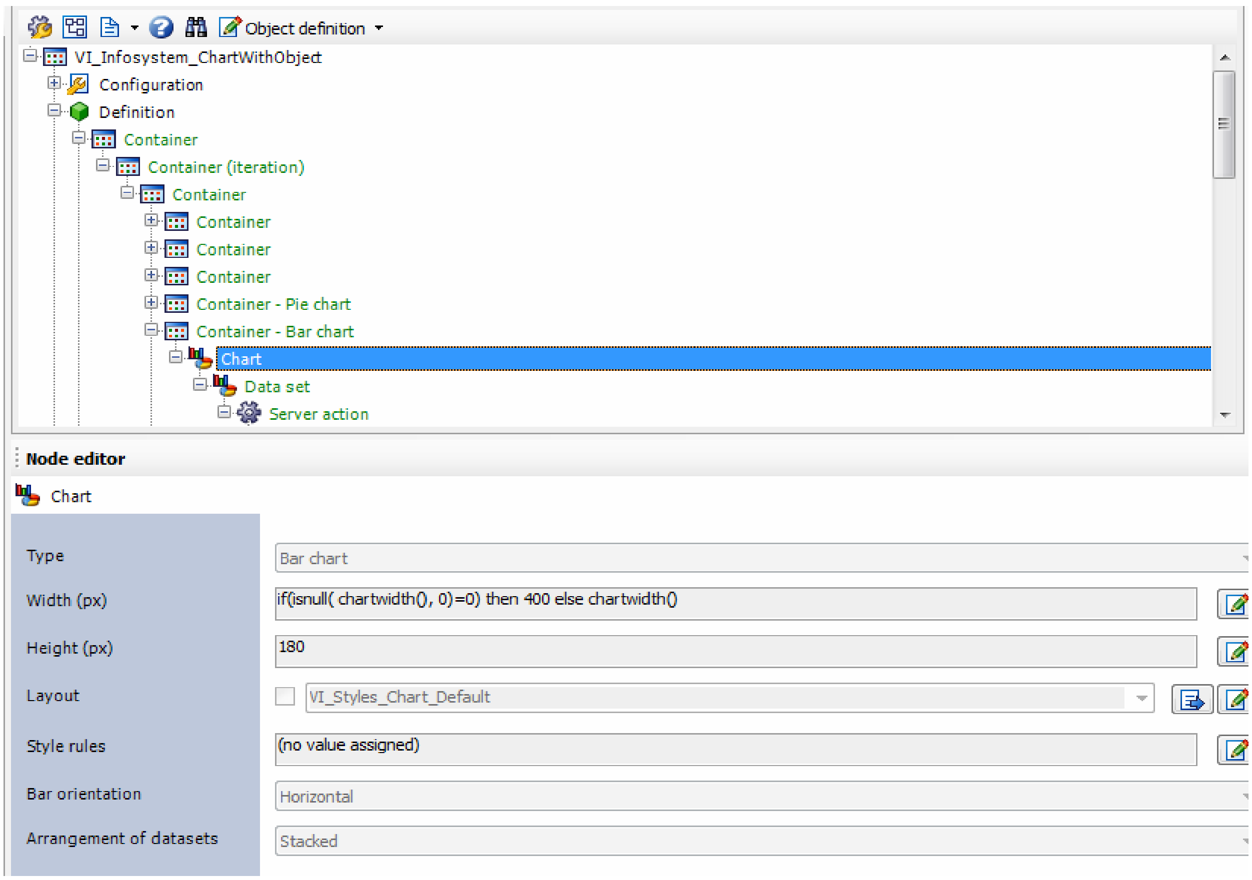
select line diagram, you are also given the option <Smooth curves>. This option is set by default.

- Pie diagram
- Bar chart

  You can decide whether the bars are vertical or horizontal and/or the order of the data sets (stacked or adjacent). You can also create groups. That means you can, for example, enter data on the x-axis instead of numbers in order to compare these graphically. For example, number of staff in departments x, y and z in years 2008 to 2011.

The selected diagram can be displayed in a Quest defined layout or customized.

Diagram Settings



Use the context menu to add more data sets. The values for the x and y-axes are in each data set. Furthermore, you have the option for enter text and a tooltip for the x-axis.

Visualizing Error Messages in the Web Portal

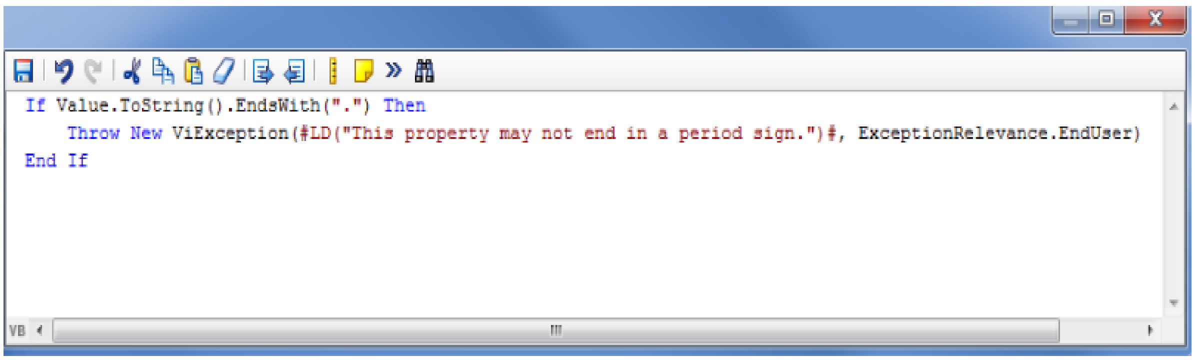
Error messages and exceptions (for example, format scripts on database columns) are displayed with a common error message in the Web Portal because exceptions, in most cases, contain too many technical details for the end user.

If you want a custom error caption to be shown in the Web Portal, an exception with matching relevance for the end user must be triggered. You have the following options:

- End user
- Default
- Intermediate
- Technical

Exceptions are always recorded in full in the web application log, independent of the relevance for the end user, and can be viewed in the Web Designer under View | Log output (if web applications are running, see the log files, Log Files on page 105).

No Period at End of Format Script from a Database Column

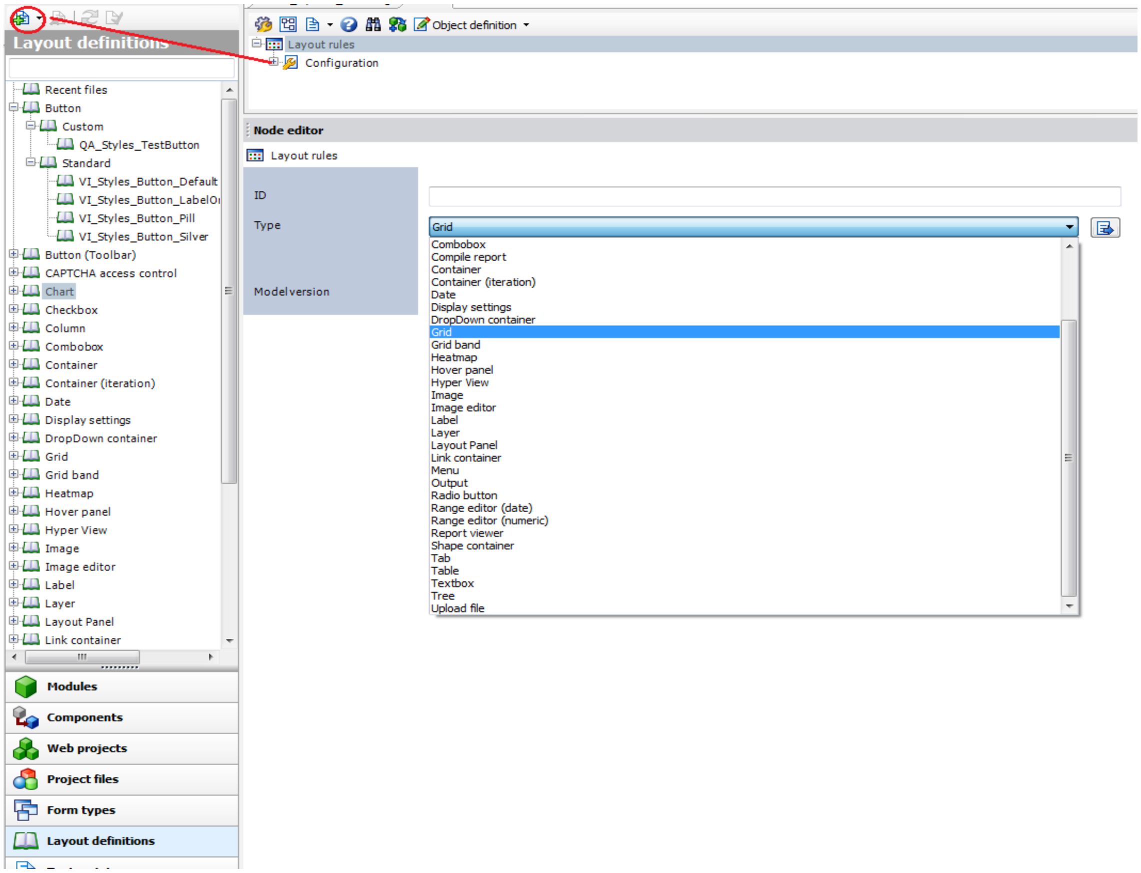


Display Node Layouts

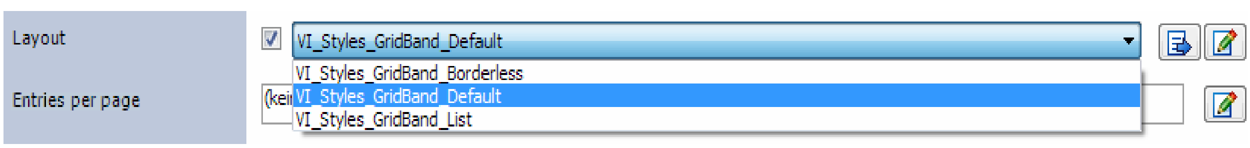
Add new layout definitions in the layout definition area using the navigation window toolbar. Use the key names from the following tables to do this. If you have added a new layout definition for a node type and then added a module to the node type from the context menu (text box or button), you can specify which layout definition to use, the default definition or your own in the node edit window. This gives you the option to assign layout individually.

You can specify different layout definitions for each level in multi-step grids.

Layout Definitions



Assigning the Layouts



The following table provides you with an overview of a number of node types and their parameters.

Display Node Layouts

DISPLAY NODE TYPE	KEY NAME	PERMITTED VALUE	DESCRIPTION
Image	CssClassName	CSS class name	Class for HTML node <IMG>
Button	AppendImage		
Button	CssClassName	CSS class name	Class for html node <BUTTON>
Button	CssClassNameContainer	CSS class name	
Chart	AxisLabelColor		
Chart	AxisLabelFontSize		
Chart	AxisLabelWidth		

Display Node Layouts

DISPLAY NODE TYPE	KEY NAME	PERMITTED VALUE	DESCRIPTION
Chart	AxisLineColor		
Chart	CssClassName	CSS class name	Class for html node
Chart	DrawXAxis	true, false	Should the X-axis be drawn or not?
Chart	DrawYAxis	true, false	Should the Y-axis be drawn or not?
Chart	FillAreaUnderChart	true, false	Should the area under the line be colored out (for line diagrams)?
Chart	StrokeColor		
Chart	StrokeWidth		
Check box	CssClassName	CSS class name	Class for html node
Combo box	CssClassName	Css class name	Class for html node
Container	CssClassName	Css class name	Class for html node
Container	TagName	Css class name	
Load file	CssClassNameButton	Css class name	Class for html node
Date Editor	CssClassName	Css class name	Class for html node
Date Editor	CssClassNameButton	Css class name	
Date Editor	CssClassNameCalendar	Css class name	
Drop-down container	CssClassName	Css class name	Class for html node
Drop-down container	CssClassNameDrop-DownImage	Css class name	Class for image cell (DIV) to expand
Drop-down container	FullRowDropDown	true, false	
Drop-down container	ShowDropDownImage	true, false	
Drop-down container	CssClassNameHeader-Container	Css class name	
Grid	AutoNormalizeLayout	true, false	Automatic default of 100% for the sum of column widths?
Grid	CssClassName	Css class name	Class for html node
Grid	PagingControlType	Type of .NET class	
Tree	PagingControlType	Type of .NET class	
Grid level	CssClassName	Css class name	Class for html node
Grid level	CssClassNameBlockDiv	Css class name	Class for a single grid block
Grid level	CssClassNameCell	Css class name	Class for a single cell

Display Node Layouts

DISPLAY NODE TYPE	KEY NAME	PERMITTED VALUE	DESCRIPTION
Grid level	CssClassNameCell1	Css class name	Class for a single cell in the first row (also for corresponding second, third, and so on).
Grid level	CssClassNameContainerResize	Css class name	Class for displaying containers for changing column width
Grid level	CssClassNameContainerResizeInactive	Css class name	Class for displaying disabled containers for changing the column width
Grid level	CssClassNameHeaderRowAboveColumns	Css class name	Class for title above column titles
Grid level	CssClassNameHeaderRowBelowColumns	Css class name	Class for title under column titles
Grid level	CssClassNameNaviCell	Css class name	Class for image to expand or collapse the next level.
Grid level	CssClassNameRow	Css class name	Class for a single row
Grid level	CssClassNameRowAlternate	Css class name	Class for rows with even numbers
Grid level	CssClassNameRowFooter	Css class name	
Grid level	CssClassNameRowHeader	Css class name	
Grid level	CssClassNameSortImage	Css class name	Class for displaying the sort order
Grid level	CssClassNameTable	Css class name	
Grid level	IconMinus	Path for image file	Image for collapsing the next level (“Minus character”)
Grid level	IconPlus	Path for image File	Image for expanding the next level (“Plus-character”)
Grid level	Image_SortAsc	Path for image File	Image for increasing sort order
Grid level	Image_SortDesc	Path for image File	Image for decreasing sort order
Grid level	Image_SortNone	Path for image File	
Grid level	CssClassNameHeaderCell	Css class name	
Label	AppendImage		
Label	CssClassName	Css class name	Class for html node

Display Node Layouts

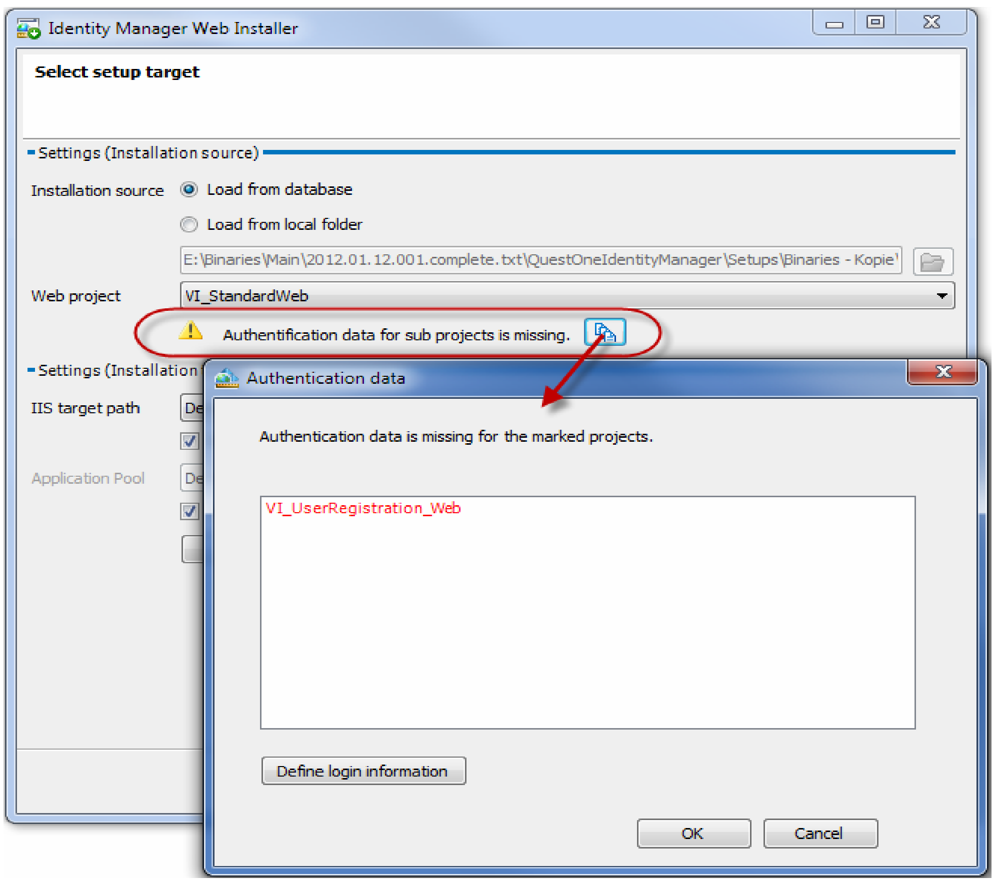
DISPLAY NODE TYPE	KEY NAME	PERMITTED VALUE	DESCRIPTION
Layout-Panel	CssClassName	Css class name	Class for html node
Layout-Panel	CssClassNameColumn-Prefix	Css class name	
Radio button	CssClassName	Css class name	Class for html node
Tab	CollapseImage	Path for image File	Image for collapsing a part of an accordion control
Tab	CssClassNameBodyCon-tainer	Css class name	
Tab	CssClassNameHeader-Container	Css class name	
Tab	CssClassNameHeader-Row	Css class name	
Tab	CssClassNamePageAc-tive	Css class name	
Tab	CssClassNamePageInac-tive	Css class name	
Tab	ExpandImage		
Tab	HeaderBelow	true, false	
Tab	NaviClass	Css class name	Only for user in “Sec-tion” mode
Tab	OpenFirstPage	true, false	
Table	CssClassName	Css class name	Class for html node
Text box	CssClassName	Css class name	Class for html node
Text box	CssClassNameManda-tory	Css class name	Class for html node, if text box is mandatory
Text box	CssClassNameTextArea	Css class name	Class for HTML node, if text box is multiline
Tree	CssClassName	Css class name	Class for html node
Tree	TreeLevelCssClassPrefix	Css class name	
Tree level	CssClassNameBlockDiv	Css class name	

Saving Login Data for Web Projects without User Related Login

An anonymous login with its own user account has been added so that new users can apply for a user account on the login page. This login data is saved directly in the configuration file (web.config) of the web application.

To enter the anonymous login data, apply the WebDesigner.ConfigFileEditor to the web application or the Web Designer Installer during installation of a web application. Whether in the ConfigFile Editor or in the Installer, the information about required anonymous login or whether this information was supplied is displayed below the default login data. You can use the button next to the advice shown to enter missing information.

Installer - Login Information



To store login data:

1. Click **Web projects** in the navigation and select the web project you want with a double-click.
   This opens the web project in a definition tree window.
2. Set the option **Login without associated user identity** in the Node Edit pane if you want to allow anonymous login for this project.

Replacing Resource Files Images

You can replace images, which reside as compiled resource in one of the files <VI.ImageLibrary.dll> or <WebDesigner.ImageLibrary.dll> with project files.

Before you can replace an image you must find the replacement image and related information, like source, file name and, if necessary, size and status.

Finding the Source

There are various options for getting more information about properties of an image. You can find the information through the image's URL, for example. Use the properties display to find out the correct source. You will recognise the source <VI.ImageLibrary> from the expression <stockImg>. The expression <Img> stands for the source <WebDesigner.ImageLibrary>.

Finding the File

You can find the name of the image file through the image's URL. The file name is placed right behind the source and could, for example, look like:

```
stockImg=AssignedDirect
```

Finding the Size

This value is only required for files from the source <VI.ImageLibrary.Images>. You find the value through the image's URL. The value could, for example, look like:

```
size=Small
```

Finding the Status

The status is only relevant for image files from the source <VI.ImageLibrary.Images>. You find this value also through the image's URL. The status could, for example, look like:

```
state=Normal
```

For both different resource file
There are different rules for the resource file names. These are dealt with separately in the following sections:

Rules for Naming an Image File from a VI.ImageLibrary.dll

Resources in the file <VI.ImageLibrary.Images.dll> are stored under the following naming convention:

```
VI.ImageLibrary.Images.<size>.<image name> <Pixel> <Status> p.png
```

For example:

```
VI.ImageLibrary.Images.Small.DeleteDocument 16 n p.png
```

The resource file <size> and <pixel> values must match.

Possible Values for Resource Size and Pixels

<SIZE>	<PIXEL>	DESCRIPTION
Small	16	Image in 16x16 Pixel
Medium	24	Image in 24x24 Pixel
Large	32	Image in 32x32 Pixel

The resource status can have the following values:

<STATUS>	DESCRIPTION
d	Disabled
n	Normal
h	Highlighted

Rules for Naming an Image File from a WebDesigner.ImageLibrary.dll

Resources in the file <VI.WebDesigner.ImageLibrary.dll> are stored under the following naming convention:

```
VI.WebDesigner.ImageLibrary.<image name>.png
```

For example:

```
VI.WebDesigner.ImageLibrary.filter-small.png
```

To replace an image from a resource file

- Create a project file in the Web Designer, which is saved under the same name as the image file.

  For more information, see Navigation Window on page 27.

Ensure the resource file images fit into the browser's cache.

Enhanced accessiblity settings

New settings for improved accessibility have been added in this version. To ensure backwards compatibility, these settings have to be explicitly activated using the script **VI_GetWebSettings**, which you can define using the Designer.

On every successful login to the web application, the system executes the **VI_GetWebSettings** script if it exists in the script library. The script must return a list of key-value pairs that enable different accessibility settings. These settings apply only to the current session, meaning that accessibility configurations can be configured for each user.

The settings address the following areas:

- New style sheets improve usability when high zoom settings are configured in the browser.
- High-contrast styling can be activated.
- Keyboard accessibility has been improved for a number of controls.
- Navigation in grid tables and tree controls is now easier.
- Controls make better use of ARIA attributes that make the meaning of the page layout clearer.

The following keys are supported:

List of available settings

NAME OF THE KEY	POSSIBLE VALUE	DESCRIPTION
AutoCreateGridCaptions	True	Add <caption> tags to grid tables, describing the table content.
BaseCssClass	q1im-highzoom	Applies the CSS class for high-zoom layout to every page's <body> tag.
BaseCssClass	q1im-highcontrast	Applies the CSS class for high-zoom layout to every page's <body> tag.
BaseCssClass	q1im-highzoom q1im-highcon-trast	Combination of the previous two set-tings.
DisableTopLevelMenuPages	True	Disables links on top-level menu items.
GridResizeDisabled	True	Disables the option to change grid col-umn widths
GridRenderParentLinks	True	Shows additional links in recursively structured grid tables. Clicking such a link sets the focus to the parent grid row.
UseButtonExpanders	True	Renders the buttons to expand or col-lapse a grid or tree node using <but-ton> tags instead of <a> tags.
HideMainContentOnPopup	True	Modifies all popup dialogs so that the main page area is hidden while the popup is open.
HideCalendarMonthNaviga-tors	True	Disables the option to navigate by month in the date control.
HideImageCrop	True	Disables the function to crop an uploaded image.
ToolbarButtonRender-AsCombobox	true	Enables a different renderer for the tool bar button control, using an accessible drop down menu.
UseAlternateTree	true	Enables a different renderer for the tree control.

The following example script shows how to activate a number of the described settings:

```
Public Function VI_GetWebSettings() As System.Collections.Generic.Dic-
tionary(Of String, Object)
```

```
Dim settings As New System.Collections.Generic.Dictionary(Of String, Ob-
ject)

settings.Add("GridGroupByDisabled", True)
settings.Add("ToolbarButtonRenderAsCombobox", True)
settings.Add("GridResizeDisabled", True)
settings.Add("UseButtonExpanders", True)
settings.Add("GridHiddenHeaderRowDisabled", True)
settings.Add("BaseCssClass", "q1im-highzoom q1im-highcontrast")
settings.Add("GridRenderParentLinks", True)
settings.Add("DisableTopLevelMenuPages", True)

' return settings
VI_GetWebSettings = settings

End Function
```

About Dell Software

- About Dell Software
- Contacting Dell Software
- Technical support resources

About Dell Software

Dell listens to customers and delivers worldwide innovative technology, business solutions and services they trust and value. For more information, visit www.software.dell.com.

Contacting Dell Software

Technical Support: Online Support

Product Questions and Sales: (800) 306 - 9329

Email: info@software.dell.com

Technical support resources

Technical support is available to customers who have purchased Dell software with a valid maintenance contract and to customers who have trial versions. To access the Support Portal, go to http://software.dell.com/support/.

The Support Portal provides self-help tools you can use to solve problems quickly and independently, 24 hours a day, 365 days a year. In addition, the portal provides direct access to product support engineers through an online Service Request system.

The site enables you to:

- Create, update, and manage Service Requests (cases)
- View Knowledge Base articles
- Obtain product notifications
- Download software. For trial software, go to Trial Downloads.
- View how-to videos
- Engage in community discussions
- Chat with a support engineer