One Identity Manager 8.1.4

Web Designer Reference Guide

Copyright 2020 One Identity LLC.

ALL RIGHTS RESERVED.

This guide contains proprietary information protected by copyright. The software described in this guide is furnished under a software license or nondisclosure agreement. This software may be used or copied only in accordance with the terms of the applicable agreement. No part of this guide may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopying and recording for any purpose other than the purchaser's personal use without the written permission of One Identity LLC .

The information in this document is provided in connection with One Identity products. No license, express or implied, by estoppel or otherwise, to any intellectual property right is granted by this document or in connection with the sale of One Identity LLC products. EXCEPT AS SET FORTH IN THE TERMS AND CONDITIONS AS SPECIFIED IN THE LICENSE AGREEMENT FOR THIS PRODUCT, ONE IDENTITY ASSUMES NO LIABILITY WHATSOEVER AND DISCLAIMS ANY EXPRESS, IMPLIED OR STATUTORY WARRANTY RELATING TO ITS PRODUCTS INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTY OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NON-INFRINGEMENT. IN NO EVENT SHALL ONE IDENTITY BE LIABLE FOR ANY DIRECT, INDIRECT, CONSEQUENTIAL, PUNITIVE, SPECIAL OR INCIDENTAL DAMAGES (INCLUDING, WITHOUT LIMITATION, DAMAGES FOR LOSS OF PROFITS, BUSINESS INTERRUPTION OR LOSS OF INFORMATION) ARISING OUT OF THE USE OR INABILITY TO USE THIS DOCUMENT, EVEN IF ONE IDENTITY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. One Identity makes no representations or warranties with respect to the accuracy or completeness of the contents of this document and reserves the right to make changes to specifications and product descriptions at any time without notice. One Identity does not make any commitment to update the information contained in this document.

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

One Identity LLC.
Attn: LEGAL Dept
4 Polaris Way
Aliso Viejo, CA 92656

Refer to our Web site (http://www.OneIdentity.com) for regional and international office information.

Patents

One Identity is proud of our advanced technology. Patents and pending patents may apply to this product. For the most current information about applicable patents for this product, please visit our website at http://www.OneIdentity.com/legal/patents.aspx.

Trademarks

One Identity and the One Identity logo are trademarks and registered trademarks of One Identity LLC. in the U.S.A. and other countries. For a complete list of One Identity trademarks, please visit our website at www.OneIdentity.com/legal. All other trademarks are the property of their respective owners.

Legend

WARNING: A WARNING icon highlights a potential risk of bodily injury or property damage, for which industry-standard safety precautions are advised. This icon is often associated with electrical hazards related to hardware.

CAUTION: A CAUTION icon indicates potential damage to hardware or loss of data if instructions are not followed.

One Identity Manager Web Designer Reference Guide
Updated - 19 October 2020, 07:44
Version - 8.1.4

Contents

The Web Designer editor **8**

Web Designer structures and functions **9**

Starting the Web Designer 9

Setting up or installing a web application 9

Selecting and editing a web application 10

The user interface layout 12

Start page 13

Title bar 14

Status bar 14

Menu bar 15

Toolbar 18

Navigation view 20

Navigation menu functions 21

Definition tree view 23

Node editor 24

Global settings 26

The Web Designer object model **29**

Web applications 29

Web projects 29

Subprojects 30

Modules 30

Start module 30

Session module 31

Menu structure 31

Structure of breadcrumb navigation 31

Current user 32

Session module instances 32

Components 33

Form types 33

Local components 33

Controls 34

Functions 34
Layout objects 34
Using the Web Designer object model 34

Working with the Web Designer 36
Find and replace 36
Sharing the web project 37
Working with tabs 38
Context menu functions 38
Working with the preview 40
Additional preview functions 41
Debugging mode support functions 42
Footer toolbar functions 43
Solution 44
Viewing generated code 45
Bookmarks 46
Command list 47
Properties 48
Multilingual captions 49
Import object 52
Related applications 53
Check model integrity 54
Check accessibility 54
Object properties 55
Change labels for web projects 55
File-based workflow in the Web Designer 57
Exporting a module from the database to the hard drive 58
Importing from a solution to the database 60

Web project configuration options 61
Project configuration - web project 61
Selecting a database object 61
Configuring a web project 62
Finding a specific configuration key 62
Finding references to a configuration key 64
Calling up the "Settings" view 64

Parameter type "Boolean Value" 64
Parameter type "SQL filter condition" 65
Parameter type "Selection from value list" 66
Parameter type "Free text" 66
Parameter type "Property list" 67
Parameter type "Image" 69
Parameter type "Configuration Objects" 70
Parameter type "Color value" 74
Parameter type "Color table" 75
Project configuration - customization 76
Project configuration - customization overview 77
Statistical references 78
Dynamic references 79
Column-dependent references 79
"Column-dependent references" tab 80
Editing components for modifying 82
Editing display components 83
Editing foreign key filters 83
Editing foreign key values 83
Using a column-dependent reference 84
Object-dependent references 84
Defining reference types and references 85
Defining a new object-dependent reference 88

Customizing the Web Portal 90

Creating new projects 90
Creating new modules 92
Adding new components 95
Customizing object definitions 96
Creating object copies with the wizard 97
Extensions 98
How extensions work 98
Extension rules 99
Creating new Hyper Views 101
Adding new nodes 103
Create data display 103

Displaying single objects ... 104
Creating a grid display for collection data ... 105
Mobile view - grid display and list view ... 106
Customizing the list view ... 107
Disable list views ... 108
Generating mapping definitions ... 108
Embedding reports ... 109
Linking to a page ... 110
Basics of Web Designer programming ... 111
Node types ... 111
Defining with Web SQL ... 111
Web SQL functions ... 112
Loading collections ... 113
Querying data from a collection ... 114
Filtering data from a collection ... 115
Collections ... 116
"Database objects" as a collection ... 117
Managing a database-based collection ... 117
Loading database objects through relations ... 119
Loading database objects from multiple tables ... 120
Using the database query wizard ... 121
Edit database queries ... 124
Loading a historical object state ... 124
Loading a change history ... 125
View definitions ... 126
Collections as data sources for controls ... 126
Collection events ... 126
Assigning collections to components ... 128
Declaring configuration keys in modules and components ... 129
Declaring context parameters ... 130
Executing Microsoft .NET Framework code ... 131
Runtime API ... 131
Integrating code into object definitions ... 132
Integrating C# code into a Web SQL expression ... 133
Access to environmental data ... 133

Referencing controls ... 134
Referencing collections ... 134
Customizing documentation ... 135

Compiling and debugging ... 137
Compiling a web application ... 137
Compiling with the Database Compiler ... 138
Viewing error messages ... 138
Querying a web application ... 139
Metadata from a web application query ... 141
Evaluating the Web SQL expression of a property ... 141
Viewing information specific to just one database object in the query window ... 142
Debugging ... 142
Setting breakpoints ... 143
Call stack ... 144

Monitoring ... 145
Status ... 145
Sessions ... 146
Assemblies ... 146
Log files ... 147
Exceptions ... 147
Web Portal performance indicators in Windows performance monitoring ... 147

Frequent tasks in the Web Portal ... 149
Editing captions ... 149
Adding functions ... 150
Pasting in texts / captions ... 151
Inserting grids ... 152
Presets for grouping grids ... 153
Visualizing exceptions in the Web Portal ... 154
Replacing images in resource files ... 154
Settings for improved accessibility ... 156

About us ... 158
Contacting us ... 158
Technical support resources ... 158

1

The Web Designer editor

The Web Designer is a development environment for One Identity Manager web applications. The web application resulting from a web project is based on ASP.NET. The Web Designer uses a simplified development model that helps you to quickly develop a robust web application without any prior knowledge in Microsoft .NET Framework.

The development module used by the Web Designer forms the basis of the web project. A web project represents the later web application and is made up of reusable modules and components amongst other things.

The installation already contains numerous default modules and components as well as a default web project that uses them. You can configure and extend the default web project to create your own web application. You can also add a new web project where you can reuse default modules and components.

The Web Designer development environment contains an editor with which you can program your web application.

In addition, the Web Designer has a compiler to compile your web project. Compiling is a prerequisite so that the web project can run as a web application. You can debug a web project running as a web application with the help of the integrated debugger in the Web Designer.

2

Web Designer structures and functions

The Web Designer is used to configure and extend the Web Portal. It has an extensive GUI in which the web application functions can be brought together.

Each Web Designer interface element and its functions are described in detail in the following chapters.

Starting the Web Designer

To start the Web Designer

1. Select **Start | One Identity | One Identity Manager | Configuration | Web Designer**.
2. Select the database connection data and the authentication method.
3. Log in to the program.

Setting up or installing a web application

NOTE: Certain key Web Designer functions, such as web project preview or debugging, can only work if a web application is installed. An important prerequisite for ensuring that the web application will work, is a functioning WCF connection.

NOTE: To set up a new web application, a web application must already be installed.

Perform one of the following tasks:

- Install a new web application
- Select a web application

For more information, see Selecting and editing a web application on page 10.

To install a new web application

1. On the start page, click the **Web applications | Install new web applications** entry.

   If applicable, the **User account control** window opens if the corresponding option is enabled.
2. Click **Yes** to run the file.

   This opens the **One Identity ManagerWeb Installer** wizard.
3. Select a database connection and click **Next**.
4. Select an installation target by entering an application name in the field.
5. Click **Next** until you see a menu.
6. In the **Change in** menu, select the application you entered.
7. Click **Next** and then **Finish**.

NOTE: After you have created a new web application, important settings must be made in the database. You can configure the settings in the Web Designer. The exact procedure is explained in **To edit web application settings**. You can also configure the setting in the Web Designer Configuration Editor. For more information about settings, see the One Identity Manager Installation Guide.

Selecting and editing a web application

To select a web application

1. Select **Web application | Select web application**.

   This displays a list of existing web applications.
2. Select your web application in the list.

   The selected web application is loaded and display under **Manage your web portal environment**.

To edit web application settings

NOTE: A web application must be selected.

1. Click **Edit web application settings**.

   This opens **Edit web application settings**.

   NOTE: The **URL** field shows you the address where the web application can be found. This value cannot be edited and serves as a key for the matching set in the database.
2. In the **Web project** menu, select the web project to be shown in your web application.

NOTE: The project you selected in "Web project" must be stored in the database. Otherwise, you cannot select it. Ensure that you have saved the web project.

The selected web shown in the menu.

3. In the **Authentication module** menu, select the module the web application user will use for authentication in future.
4. In the **Try single sign-on, use the following module if single sign-on fails:** menu, select a module.

   If the module selected under **Authentication module** supports single-sign-on, you have the option to specify an alternative authentication method here. This is used as a fall-back if single sign-on fails for any reason. For detailed information about the One Identity Manager authentication modules, see the *One Identity Manager Authorization and Authentication Guide*.
5. Click **Debug**.

   The debugging environment is enabled and you can use it. For more information, see Debugging on page 142.

   NOTE: Test your web application with this setting before you generate a web project compilation through **Release**.

To edit web application settings with an OAuth authentication module

NOTE: If you use OAuth authentication, you can enter an ID in the field provided. This automatically identifies the web application. This authentication ID is assigned in the OAuth system and must be added to the configuration in the Web Designer. If you do not use OAuth authentication, ignore this field.

1. Follow the steps 1 to 3 described in the **To edit web application settings** step-by-step.
2. Select one of the following modules in the **Authentication module** menu.
   a. OAuth 2.0 / OpenID Connect
   b. OAuth 2.0 / OpenID Connect (role-based)

Enter values for your web application in the **OAuth** section of the **Edit web application settings** dialog. These fields are normally preset with values from global configuration parameters.

The following values can be entered.

Table 1: Values for using an authentication module

Values	Description
Client ID for OAuth authentication	If this is empty, the value from "QER | Person | OAuthenticator | ClientID | Web" is used. For example, "urn:IdentityManager/Web".
Thumbprint for the OAuth certi-	This is used to find the certificate in the certificate store. Either the thumb nail or the issuer of the certificate is required.

Values	Description
ficate	For example, "7612FF79D99FCC56BEDDOC758412025B284DC327".
Issuer information for the OAuth certificate	This is used to find the certificate in the certificate store. Either the thumb nail or the issuer of the certificate is required. For example, "O=[company name], OU=[organizational unit], CN=[server IP]".
Resource	Is only required if the OAuth service needs this value. The value entered here is passed as a URL parameter "Resource" to the OAuth server.
Certificate endpoint	Supplies the URL from which to download the certificate. The certificate's thumbprint or issuer information is required for validating the downloaded certificate. For example, "https://certificateServer/certificate.crt".

The user interface layout

You can control the Web Designer graphical user interface with a mouse and key combinations. A minimum screen resolution of 1280x1024 pixels is recommended with at least 16-bit color in order to optimize the user interface graphics.

NOTE: You can customize the Web Designer's default layout, as in the other One Identity Manager tools, by moving, closing, or hiding to suit your requirements. You can set the hide mode with **Auto Hide**.

After you have logged in, you are presented with an empty start page. This page contains a title bar, a status row, menu bar, a toolbar, a navigation view, and other edit areas, as described below.

A. Title bar

   In the title bar you can see the program icon, program name, and connected database.

B. Menu bar

   The menu bar gives you different menus and menu items.

C. Toolbar

   Each program component comes with its own toolbar.

D. Start page

   On the start page you can see various functions and information on the status of the currently selected web project and for other web projects.

E. Navigation view

The navigation view is on the left-hand side of the screen and organizes database objects into different categories.

F. Preview

The preview window shows how a specific web application will look and respond in the browser.

G. Window in lower area

More edit options are available in the lower area of the screen. Some of these options, such as the **Node editor,** are open by default.

H. Status bar

The status row includes the connected database display.

Start page

In the middle of the start page, you can see various functions and information about the status of the currently selected web project and for other web projects. For more information, see Web projects on page 29.

On the left-hand side of the start page different tasks are available concerning web project installation, configuration, and editing.

Table 2: Tasks on the start page

Task area	Task	Description
Web applications	Install new web applications	Installs a new web application for the Web Designer. You can install several web applications.
	Select a web application	Selects a web application. You work on this selected web application.
Configuration	Web project	Configures a web project. Defined configuration keys are available for web project configuration.
	Customizations	For creating or editing substitution rules for module copies.
	Search fields	For establishing database columns in which to execute searches.
	Column-dependent references	For configuring column-dependent references. These are used as a reference for a

Task area	Task	Description
		component's column definition.
	Object-dependent references	For configuring object-dependent references. These are part of dynamic references.
Edit	Add new	Edits the web application. You can create new modules and projects, column-dependent, or object-dependent references and object copies.
	Edit texts	Edits control element texts.

To reopen a closed start page

- In the toolbar, in the **View** menu, select the **Start page** menu item.

  The **Start page** tab is displayed as an empty page with the usual selection options and settings.

Detailed information about this topic

- Selecting and editing a web application on page 10
- Web project configuration options on page 61
- Customizing the Web Portal on page 90
- Multilingual captions on page 49

Title bar

In the title bar you can see the program icon, program name and connected database in the following notation:

Web Designer - <User>@<Database server>\<Database (Description)> - ...project

Status bar

The status bar displays different status data. Some status data is shown by way of icons. Which icons are displayed is partially dependent on the program settings selected. The status bar comes in different colors.

Table 3: Meaning of the colors

Color	Meaning
none	Development environment database is connected.
Red	Simulation mode is enabled.
Green	Test environment database is connected.
Yellow	Productive environment database is connected.

Table 4: Status bar icons

Icon	Meaning
	Current user.
	Shows information about the project.
	The database is connected.
	Shows database access.
	Quick edit mode is enabled.
	The database must be compiled.
	The program is in simulation mode.
	A warning has been written to the error log.
	An error message has been written to the error log.

In the status bar, you can additionally see the connected database under <Server>\<Database (Description)>.

TIP: Double-click the connected database in the status bar, to copy the database path to the clipboard. If you double-click the current user symbol, you can access further information about the current user.

Menu bar

You can use the menus on the menu bar to access submenus and to access and execute many functions in the Web Designer quickly. The menus are described in more detail in the following table.

Table 5: Menu items in the menu bar

Menu	Menu item	Description
Connection	Settings	Opens the **Global settings** dialog in the Web Designer.

Menu	Menu item	Description
		For more information, see Global settings on page 26.
	Exit	Closes the program.
Edit	Refresh preview	Refreshes the Web Designer preview.
	Publish	Opens a submenu with other options. For more information, see Sharing the web project on page 37.
	Configure project	Opens a submenu with other options. s For more information, see Web project configuration options on page 61.
	Copy objects	Opens the **Copy objects** dialog. For more information, see Creating object copies with the wizard on page 97.
	Creating new modules	Opens the **Create new module** dialog. For more information, see Creating new modules on page 92.
	Create new project	Opens the **Create new project** dialog. For more information, see Creating new projects on page 90.
	Find and replace	Opens the **Find and replace** dialog. You can also access the search function using the **Ctrl + F** shortcut. For more information, see Find and replace on page 36.
	Find next	Performs the search with the search parameters specified in the **Find and replace** dialog. The search is performed without opening a dialog. Use **F3** to continue searching.
	Captions	Opens the **Multilingual captions** dialog. For more information, see Multilingual captions on page 49.
	Import object	Open the Windows, **Please select a source file**, default dialog. For more information, see Import object on page 52.
	Related applications	Opens the **Related applications** dialog. For more information, see Related applications on page 53.

Menu	Menu item	Description
	Check model integrity	Checks the current web project. For more information, see Check model integrity on page 54.
	Check accessibility	Checks the web project's accessibility. For more information, see Check accessibility on page 54.
View	Tabs	Opens the **Tabs** dialog. For more information, see Working with tabs on page 38.
	Restore standard layout	Restores the default layout.
	Restore standard layout (including size)	Adjusts the default layout and the window size.
	Restore saved layout	Restores the saved layout.
	Save layout	Saves the Web Designer layout after you have modified it.
	Start page	The tab is open by default. For more information, see Start page. Opens the start page. For more information, see Start page.
	Solution	Opens the **Solution** window. For more information, see Solution on page 44.
	Node editor	Opens the **Node editor** window. For more information, see Node editor.
	Tasks	Opens or enables **Tasks**. For more information, see Viewing error messages on page 138.
	Command list	Opens the **Command list** window. For more information, see Command list on page 47.
	Bookmarks	Opens the **Bookmarks** window. For more information, see Bookmarks on page 46.
	Navigation	Opens the navigation view.

Menu	Menu item	Description
		For more information, see Navigation view on page 20.
Debug views	Monitor page	Opens the monitor page. The monitor page is displayed on a page that is separate from the Web Designer page. For more information, see Monitoring on page 145.
	Preview	Opens the preview. For more information, see Working with the preview on page 40.
	Properties	Opens **Properties**. The view is open by default and displayed in the lower pane of the Web Designer. For more information, see Properties on page 48.
	Query	Opens **Query**. The view is open by default and displayed in the lower pane of the Web Designer. For more information, see Querying a web application on page 139.
	Call stack	Opens **Call stack**. The view is opened in the lower pane of the Web Designer. For more information, see Call stack on page 144.
	Compilation	Opens the **Compilation** window. Here you can view the compilation log. The view is opened in the lower pane of the Web Designer.
	Compiled objects	Opens **Compiled objects**. All compiled objects are listed here with more details. The view is opened in the lower pane of the Web Designer.
Help	Info	Opens a dialog. The dialog displays several tabs providing information about the Web Designer, such as version number, third party contributions, installed modules, and so on.

Toolbar

The Web Designer has a toolbar. This toolbar cannot be configured.

Table 6: Toolbar functions

Icon	Description
Change Label:	Selects change labels. You can select a change label in this option box. Changes in the One Identity Manager database are stored under the change labels selected here.
	Managing change labels You can insert, modify, or delete change labels in **Change labels**. For detailed information about working with change labels, see the *One Identity Manager Operational Guide*.
	Uses the current change label as default and selects it automatically later This function sets the currently selected change label as a default label that will be selected automatically when the Web Designer is restarted. The selection is client-specific and has no impact on other One Identity Manager database users.
	Applies a change label to the entire web project. You can assign a special change label to all referenced objects in a web project in **Change Labels for Web Projects**. Some web projects can be moved completely and independently of previously selected assignments.
	Configures a web project. For more information, see Web project configuration options on page 61.
	Opens the **Multilingual captions** dialog. For more information, see Multilingual captions on page 49.
	Opens **Hyper View wizard**. For more information, see Creating new Hyper Views on page 101.
	Suppress background actions Use this button to suspend and start actions running in the background. These actions are defined through nodes of type **Timer**. This results in the web application executing these actions are regular intervals. The behavior may be disturbing when debugging.
	Save This button is enabled if you have modified a database object. Use this button to save changes to the current database object. If you want to save changes to other database objects, you must select the corresponding tab in the definition tree view and click the button again.
	Save all Use this button to save all database objects that have been modified.
	One node back/forward

Icon	Description
	Use this button to navigate back or forwards in the history of the selected object. This displays the selected object in the definition tree view. If an object or node was deleted, the next existing object is shown. If an object is selected in the history that' no longer exists, the previous object is shown. The number of objects displayed in the history can be specified in the settings in the menu bar under **Connection | Global settings**. You can delete the history using **Delete history**.

Navigation view

NOTE: The navigation view is hidden by default if you are not currently using it. This feature is designed to provide ease of use.

In the default view, you can find the navigation view on the left of the screen.

To show and hide the navigation view

1. Proceed as follows.
   a. Click or mouse over the **Navigation** control, which is positioned vertically on the left-hand side of the start page.
   b. In the **View** menu, select **Navigation**.

   NOTE: Use **Auto Hide** in the navigation view toolbar to set the navigation view.

   NOTE: If the **Navigation** control is not visible where it should be, you can select **Restore default layout** or **Restore saved layout** instead of **Navigation**.

   The navigation view is displayed.
2. Click or mouse over outside the navigation view to close the navigation view again.

   Now, only the **Navigation** control is visible in the program's sidebar.

In the navigation view, select the database object that you wish to edit in the Web Designer. Database objects are grouped into different categories. The different categories have different functions which you see depending on what you select. The following categories are available:

- Modules
- Components
- Web projects
- Project files
- Form Types
- Layout definitions

The structure of displayed database objects varies within each category. The following table describes the different category display structures individually. Categories with similar display structures are described together.

Table 7: Display structure of database objects in the different categories

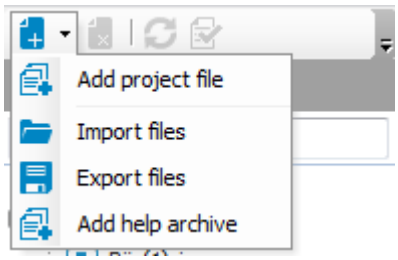
Category	Description
Display structure in the "Modules" category	Database objects are displayed in a tree structure. Database objects are found in **Custom** and **Default**. Objects grouped under the **Default** node cannot be edited in a custom installation of One Identity Manager and are part of a default installation of the Web Designer. You can copy these objects or extensions. All custom objects in **Modules** are grouped in **Custom**. Modules, that are added to a custom installation are automatically groups under these root nodes. **Custom** root nodes are displayed above the **Default** root nodes in the tree structure.
Display structure in "Component" and "Layout definitions"	Database objects form individual groups of objects. Database objects from these categories are not grouped within the **Custom** and **Default** root nodes but form individual groups of objects. Each group of objects can contain custom as well as default objects that have been available since initial installation.

To make database objects easier to find, they are displayed in **Search results** and **Recent files** groups. For more information, see Navigation menu functions on page 21.

Extensions are displayed as subnodes for the module or component for which they were created. You also see these subnodes under groups, search results, and recent files. For more information, see Extensions on page 98.

Navigation menu functions

Table 8: Functions in the navigation view toolbar

Icon	Description
	You can select a function in this context menu. The functions available depend on which database object you selected beforehand.

Icon	Description
	You can delete a database object with this button.
	You can load a new database object with this button.
	You can display database object properties with this button.

To execute a function

1. Select either a module, component, project file, form type, layout definition, or test script in the category you want.

NOTE: You cannot add a web project through this context sensitive submenu. You can add new modules using the corresponding wizard. This procedure also applies to creating new web projects.

   Depending on your choice, other functions are available in the context sensitive submenu displayed using the .

2. In the navigation view toolbar, click .
3. In the submenu, select a function.

You can run the following functions in the **Project files** pane.

- Add project files
- Import files
- Export files
- Add help archive

To delete a database object

- In the navigation view, select a database object and click .

  This deletes the selected object after confirming the security prompt from the database.

NOTE: is only enabled if you have selected a database object.

NOTE: The delete process cannot be undone.

To load new database objects

- Click the database objects in the appropriate category to refresh the database.

To view database object properties

- In the navigation view, select the a database object and click .

  This display the database object properties.

To search in the navigation view

1. In the navigation view, select the category in which you want to search for database objects.

   This displays the selected category in the navigation view.

2. Enter the search term in the text field and click RETURN.

   The search results are listed under the text field.

Detailed information about this topic

- Creating new modules on page 92
- Creating new projects on page 90

Definition tree view

The Web Designer's graphical user interface, which you can use to edit data objects in your web application, is displayed as a definition tree view. The definition tree view is displayed as a tab next to the start page. You open the definition tree view by selecting a database object from the navigation view for example.

Almost all database objects consist of several nodes that are shown as a tree structure in the definition tree view and can be edited. If you add a new database object (web project, module, component) to your web application, this database object is created with a predefined number of nodes in the object definition. The object definition is a type of view found within the definition tree view. For more information, see Menu bar icons in the definition tree view on page 23.

The predefined nodes of a database object provide the foundation for an XML document in your web application and, as a result, cannot be deleted. You can add additional nodes in addition to the predefined nodes using the context menu. These additional nodes can be removed again. When a database object is being edited, the user is always located at a defined position in the definition tree view.

A database object generally consists of several sub-documents that can be edited in separate views in the definition tree view. You can switch between different views using the toolbar in the definition tree view. The toolbar is explained in the following table.

Table 9: Menu bar icons in the definition tree view

Icon	Description
	Object definition (read-only). This view is mainly used to view existing database object nodes. This view is preset when you switch to the

Icon	Description
	definition tree view. If the database object comes from the standard version of the Web Designer, the view is called **Object definition (read-only)**. For more information, see Customizing object definitions on page 96.
	Configuration (read-only).
	Configuration (custom).
	Generated code (read-only).
	Settings.
	Find and replace.
	Help for displaying selected node type.
	Create object copy.
	Compiles a single object without writing the changes to the database.

Detailed information about this topic

- Customizing object definitions on page 96
- Web project configuration options on page 61
- Viewing generated code on page 45
- Project configuration - web project on page 61
- Find and replace on page 36

Node editor

Node editor allows you to edit the properties of a node that you have selected in the definition tree view. Each node type has a special set of properties that are listed in the **Node editor**. Therefore, the list of properties varies depending on which node type you have selected.

The functions in the toolbar are listed in the following table.

Table 10: Functions in the "Node editor" toolbar

Button	Description
Node name / node	Show the node identifier and the node type icon that marks it in

Button	Description
type icon	the definition tree view. This cannot be edited.
Database	Displayed if a database object is selected. This cannot be edited.
and name / icon of the base object	Displayed if a file-based object is selected. When you hover over , the custom object's local path is displayed in a tool tip. When you hover over and click the right mouse button, your Windows Explorer context menu opens. For more information, see File-based workflow in the Web Designer on page 57.
	Filters the current settings. Only the settings are filtered, not the settings' values. Enter at least part of a setting's name to display the settings whose names contain this string.
	Sorts the settings alphabetically in ascending order within each category. If this is enable the icon's border is highlighted. If sorting is enabled, it follows by default, a logical order that is graded by importance. Compulsory fields have the highest priority and are displayed at the top.
	Sorts the settings alphabetically in descending order within each category. See the entry about sorting in ascending alphabetical order when sorting disabled.
	Enables/disabled settings groups in the node editor window. This means the settings are not divided into categories.
	Expands the category contents. These buttons are only active when categories are grouped. You can use to expand a group separately.
	Collapses grouped categories. These buttons are only active when categories are grouped. You can use to collapse them separately.
Show extension values	Highlights setting that have had an extension added to them. Use to switch to the extension. A node that has had an extension added to it, is labeled in the definition tree view with . Show extension values is highlighted if the function is enabled.

The node editor is divided into several categories. These categories could be the following. Which categories are shown, depends on the type of node selected.

Table 11: Categories in the "Note editor" window.

Category	Description
General settings	Settings in this category include node ID, node type, viewing condition or condition for executing. Not all settings are not listed here. Which settings are shown, depends on the type of node selected. This applies to all categories.
Workflow control	This category is available for module or container nodes, for example. You can edit the data **Identifier of the associated menu item** for a module node. For a container node, you can edit the data **Display node for enter key** and **Expression to trigger an AJAX update**.
Style	Menu for selecting a layout that you can modify with an editor. First check the box to enable this function and add a condition to the layout.
Advanced settings	In this category, there can be a control ID that you can overwrite.
Viewing options	This category is available for nodes of type "Chart", for example. You can change styles or set heights and width of charts.
Source data collection	This category is available for nodes of type "Combobox". You want to make source data from other collections available to the current collection. This means the user can select the collection with the desired source data over the menu.

Global settings

You configure the basic settings for the Web Designer in the **Global settings** dialog. The settings are sometimes preset. You can adapt the settings to your needs.

To adjust Web Designer settings

1. Select **Connection | Settings**.

   The **Global settings** dialog opens. Detailed information concerning global settings is described in the following table.
2. Open the relevant settings area and apply any necessary changes.
3. Save the changes.

Table 12: Global settings for the Web Designer

Settings area	Setting	Description	Default setting
Common	language	Sets the language used.	German
	Resolution	Sets the screen resolution.	1280 X 1024
	Maximum history length	Sets the maximum number of history items displayed.	25
	Maximum number of recently opened items	Sets the maximum number of recently opened items displayed.	10
Editor	Use colors to distinguish data, display, and action	Enables color differentiation.	Enabled
	Shorten long property values	Enables a shortened display.	Enabled
	Show XPath navigation	Enables XPath navigation.	Not set
	Show messages after syntax errors are found after editing	Enables message display.	Not set
	Check for problems before publishing	Enables checking.	Enabled
	Save and load Web Designer size within layout	Enables the Web Designer to be saved and loaded with layout.	Enabled
	Show message, if new assemblies are available	Enables information on new assemblies to be displayed.	Enabled
	Check if web is not published, when closing	Enables checking.	Enabled
Save and backup	Ask for change label if none is selected	Enables queries.	Enabled
	Keep backup of unsaved objects on the local machine	Enables caching.	Not set
Compiling and debugging	Show warnings of this database modules	Manual input of the module displaying the warnings.	Empty field
	Perform enhanced check	Enables enhanced checking.	Not set
	Suppress background actions	Disables background actions.	Not set

Settings area	Setting	Description	Default setting
Keyboard layout	Zoom in on preview	Zooms in on the preview.	Ctrl + D
	Recompile	Recompiles the web application.	F6
	Show next bookmark	Displays the next bookmark.	F2
	Enable/Disable the debugger	Enables or disables the debugger.	F9
	Resume debugging	Resumes debugging after stoppage.	F7
	Debugger - Step Into	Locates the error at debugging.	F11
	Suppress background actions	Enables or disables background actions.	F8

To switch the keyboard layout back to the default layout

- In the **Keyboard layout** settings pane, click **Default layout**.

  Resets the settings to the default layout.

3

The Web Designer object model

In Web Designer, database objects, such as modules, components, or styles, are saved as XML files in the database. They are shown as nodes in a tree structure in the Web Designer definition tree view. Different nodes are labeled with icons to provide a better overview and to distinguish between them more easily.

For more information, see the One Identity Manager Web Designer Object Model Documentation.

Web applications

The Web Portal is a web application. A web application is a published, live website on a web server, which interacts with the One Identity Manager database. For more detailed information about the Web Portal, see the One Identity Manager Web Portal User Guide.

A web application manages web projects and all other database objects. Functions, such as editing or saving colleague or order data are available. A web application is a published web project with its subprojects, if these have been defined.

Each web application has a dedicated intranet or Internet address.

Web projects

In the Web Designer you can define and manage as many web projects as you want. Generally, a web project includes a web application and a web portal. However, the link between the `VI_StandardWeb` and `VI_RegistrationWeb` web projects is a special case. These web projects are already in the database and are part of the business workflow. `VI_RegistrationWeb` becomes a subproject of `VI_StandardWeb` and must be published separately.

You can view a web project as a web application before publishing it in the Web Designer preview.

A web project is generally made up of modules, components, and other database objects. You can add or edit database objects in the Web Designer. Modules with and without parameters are linked in the web project.

To assign a web project to a web application that is on the web server, use the web application configuration file. For more detailed information about the web server, see the One Identity Manager Installation Guide.

In every web project there is a set of specific data to be maintained in the definition tree view. For more information, see Definition tree view on page 23.

Subprojects

You can also select another web project as an additional web project. You can update or compile this project on the start page in the same way as the web project `VI_StandardWeb`. `VI_RegistrationWeb` is an example of a subproject. Subprojects are required to operate several web projects from within an installed web application. Without subprojects, you would need to install as many web applications as there are web projects. In addition, a subproject enables a part of the web application, for example with different authentications, to control another menu structure or layout.

Modules

A module is a logically closed unit within a web project. A module can only contain one web page with a short welcome text (as in the start page) but can also map a comprehensive workflow, which stretches over several web pages (such as, ordering products).

Modules contain at least one form or main page. They can contain components that are required for defining nodes visible in the browser. This allow the contents of a form node to be rendered in a start-up module or on the browser's start page if the start page is displayed to the user. The size of the module or number of modules is not limited.

NOTE: An effective and structured mapping of a web application's target functionality in modules makes any debugging and later implementation of other functions easier.

Start module

The start module for a web project is a special module. It can configured and is called `VI_Start`. It specifies which main content page is displayed first. Each module can have a start module, assuming **Allow quick navigation to this module with a URL parameter** is enabled. You cannot pass any parameters.

Session module

The VI_Session session module is a special module. Exactly one session module is used in each web project. It can be used by several web projects.

The lifetime of a module is a deciding factor. As long as the session is valid on the web server, you can always access the session module data.

This module contains data about the current user and other session related data.

A session module is initially loaded at the start of a session and retains its data during the entire lifetime of the session. A session module must contain information about the current user. To do this, a session module must contain:

- Collections with the name User
- UserRuntimeModule
- UserMenu
- UserNavigation

NOTE: The collections UserRuntimeModule, UserMenu and UserNavigation are Microsoft .NET Framework object collections.

You can define your own user groups for the web application, for example, by assigning an employee to an organizational unit. These current users assignments are determined once in the session module after logging into the web application and are available for the period of the session thereafter, to hide or show menu items or parts of forms.

Menu structure

The UserMenu contains data about the user's enabled menu items. Data in this collection may not be changed because changes can be overwritten on recalculation.

Structure of breadcrumb navigation

UseNavigation contains information about the user's navigation breadcrumb trail. Each entry corresponds to a generated form object. The following collection properties are available:

Table 13: UserNavigation collection properties

Property	Description
UidForm	Provides the main key.
UidModule	References the entry in the <UserRuntimeModule> collection

Property	Description
	containing the associated module entry.
UidModulePrevious	References the entry in the <UserRuntimeModule> collection corresponding to the previous module in the navigation path. This information is also used for building up the breadcrumb trail.
UidFormPrevious	References the entry in the <UserNavigation> collection corresponding to the previous form. This information is used for building up the breadcrumb trail.
Title	Specifies the form title.
CanReturnToThis	Boolean value specifying whether this form can be opened or not.
SkipInNavigation	Boolean value specifying whether this form in the breadcrumb trail should be skipped.
Number	Specifies the sequential number of this form in the breadcrumb trail.
FormName	Specifies the form identifier.

Current user

The collection `User` contains the database of type `Person` that corresponds to the current user.

Session module instances

The `UserRuntimeModule` collection contains all module instances generated during a session. A module can be called up several times. Every time a module is called up, a new instance is generated. A module instance is a fixed instance of a module definition.

Table 14: `UserRuntimeModule` collection properties

Property	Description
UidModule	This property provides the main key for the module instance.
ContextID	This property contains the name of the corresponding Web Designer module.
LinkObjectID	This property contains the menu item identifier triggering the generation of this module instance.

Property	Description
TimeStamp	This property contains the time the module instance was generated.

Components

A component is a part of a module, which has been separated out. The module references these component using cross-references. It can be use by several modules and components and rendered multiple times in a browser simultaneously. Instances arise from components that are used more than once. To differentiate various instances from each one another, each instance is given an automatically generated suffix at runtime. All nodes of this instance have the same suffix.

Apart from that, components are part of the definition tree and keep it free from redundancies. This property reduces the effort of programming and ensure a uniform appearance as well as homogeneous functionality of the web application. Another feature is the different component type. The position of the component in the definition tree determines which component type can be referenced.

Form types

A form type specifies the structure of HTML pages, which are rendered in the browser. It defines the available form parts and their positions in the HTML tree structure. The **Form types** area is found in the navigation view and the form definitions are found as `DialogAEDS` objects in the database.

The default version supplies different VI objects as templates. To edit a form type, you must make a copy of the object. For more information, see Creating object copies with the wizard on page 97. Each form type is made up of the form parts `Title` and `Main` and other output nodes. Form types are often differentiated by mobile and desktop view.

Local components

A local component is a node in the tree structure of a module and a component. It provides a better overview for programming and can be referenced in other modules and components.

Controls

The same applies to a control as to a local component. They are defined within a module and are only found there.

As a result, a reference to a control can only be set in the module in which the control was defined. A benefit of controls is simple referencing.

Functions

A function provides a better overview and is suitable for checking readability. That ensures increased quality. Function are available only in the respective module or component.

NOTE: Functions defined in a session module are an exception. These functions can be implemented everywhere.

Layout objects

Layout objects are layout definitions and describe the appearance of display nodes (nodes of type `View`) that are rendered in the browser. There is at least one layout variation defined for each display node where possible. Layouts with the ending `default` are used by the associated node type without having to be entered on individual nodes.

Each object in the layout definition list corresponds to a CSS class and is linked to a node type. This layout definition is therefore available for each node of a particular node type. You can set up more new layout definitions in this list. A wizard is available to you to customize a default layout definition. For more information, see Project configuration - customization on page 76.

For a better overview, layout definitions are divided into type and group in the navigation view.

Using the Web Designer object model

To select a web project

- In the navigation view, in the **Web projects** category, double-click on a web project.

  The selected web project is displayed as a tab in the definition tree view.

To select an additional web project

NOTE: Your web project must already be selected and open in the definition tree view.

1. In the **Object definitions** view, expand the **Display settings** .

   This displays subnodes.
2. Mark the **Subprojects1** subnode and select **Web project** from the context menu.

   The view switches to the **Configuration (custom)** view in the definition tree view and the **Web project** node is marked.
3. In the **Node editor** view, select a subproject from the **Identifier*** menu.

   This selects the subproject of your choice.

To navigate to a specific module

Quick navigation can be used for opening a link in an email for example.

NOTE: The relevant module must already be selected and opened in the definition tree view.

1. In the **Object definition** view, mark the root node of the module.
2. In the **Node editor** view, set the **Allow quick navigation to this module with a URL parameter** option.

The option **Allow direct navigation to this module with an URL parameter** allows you to navigate directly to a specific module. For example, with the help of defined passing parameters, you can navigate directly to the overview page of a particular database object or approve an order.

To open the `VI_Session` session module

- In the navigation view search box, open the session module.

  The session module is shown as a tab in the definition tree view.

To see `UserNavigation` collection properties

1. In the lower pane of the Web Designer, click the **Query** tab.
2. Click the **VI_Session** node on the left-hand side of the query view and double-click the **UserNavigation** collection.

   The properties and content of the collection are displayed at the bottom left of the query view.

To add and edit form elements

NOTE: The relevant module or component must already be selected and opened in the definition tree view.

1. In the **Object definition** view, select the **Forms** node.
2. From the context menu, select the **Object in extension | Form** menu item.

   The view switches to **Configuration (custom)** and the new added node is selected.
3. Enter a name for the new form element in the **Identifier*** field in the **Node editor** and adjust the settings.

   You can also add talking selectors.

4

Working with the Web Designer

In the following, you will be shown in detail, how to work with the Web Designer.

Find and replace

To run a search

1. Select the **Edit | Find and replace** menu item.

   This opens a dialog with search options.
2. Enter a search term in the **Find** field and select the relevant entry from **Find as** menu to narrow down your search.
3. Enter the relevant information in the **Replace by** field and select the relevant entry from the **Find scope** menu.
4. In the **Find options** pane, click the button and, in the **Find options** pane, enable any necessary criteria.
5. In the **Find options** pane, enable the **Filter by type** option if required, and select the relevant node types.

   TIP: If you select a node type, the search is limited to the selected node type.
6. Click **Find** or **Find all**.

   The search results are displayed.
7. In the **Find results** pane, click the and double-click a result in the results list.

   This marks and displays the corresponding nodes in the definition tree view.
8. In the results list, mark the result that you want to replace.
9. Click **Replace** or **Replace all**.

To conduct a more detailed search, you can select the following entries to narrow down your search.

Table 15: "Find scope" entries

Entry	Description
Current document	Only the current document is included in the search.
Current document and its extensions	The current document and its extensions are included in the search.
Current web project	The current web project is included in the search.
Objects below current object	Only the objects that are below the current object are included in the search.
All Web Designer objects	All objects are included in the search.

You can use the following settings for searching. Multi-select is available.

Table 16: Options under "Find options"

Option	Description
Upper/lower case	The search is case sensitive.
Whole word	The whole word is included in the search.
Match entire value	The complete value is included in the search.

Sharing the web project

You can share web applications that depend on a web project for debugging or release. These functions are available on the start page in **Edit**. Both procedures are identical. In the steps below, the menu bar procedure is described. On the start page, you can access the functions using **Debug** and **Release**.

To share a web project for debugging

1. In the menu bar, click the **Edit** menu and select the **Share** item.

   This displays a context menu containing other menu items.
2. Select the **Debug** menu item.

   Selected web applications opens with a list of relevant web applications.
3. Confirm the prompt with **OK**.

This starts the compilation process. At the end of the compiling process, the log is shown in **Compiling**.

To share a web project for release

1. In the menu bar, click the **Edit** menu and select the **Share** item.

   This displays a context menu containing other menu items.
2. Select the **Release** menu item.

   Selected web applications opens with a list of relevant web applications.
3. Confirm the prompt with **OK**.

   This starts the compilation process. At the end of the compiling process, the log is shown in **Compiling**.

Working with tabs

You can open **Tabs...** using the **View** menu. This provides you with an overview of tabs open in the Web Designer.

To get an overview of open tabs

1. In the **View** menu, select the **Tabs** menu item.

   This opens the **Tabs** dialog with a list of open tabs.
2. In the list, mark the relevant tab you want to work with and select one of the following actions:
   a. Enable
   b. Save
   c. Close the tab

   TIP: You can multi-select by selecting items in the list whilst holding the Ctrl key.
3. Close the dialog.

Context menu functions

To call up the context menu, right-click any node. The content of the menu depends on the type of node clicked ("context-sensitive"). The top 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 node of the type **Container** is created above the currently selected node and the currently marked node is inserted in the new container.

- Create data display

  This wizard creates forms, form elements, or grid views from available data. You will find this wizard on a **Container**-type node, for example. For more information, see Create data display on page 103.

- Create/Embed Report

  Generates views from previously generated reports that are stored in the database. This wizard is available on action nodes. For more information, see Embedding reports on page 109.

If a defined number of node types is exceeded, the nodes will be re-bundled beforehand. Some node types do not allow additional node types to be pasted into them. In such cases, the context menu begins with the **Cut** function. All other functions are either available or grayed out, depending on the type of node and its position in the definition tree window.

Table 17: 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. Any child nodes for the selected node are also cut.
Copy	This function is always available. 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.
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. You can multi-select nodes with the Ctrl key to apply the function once to all selected nodes.
Set break-point	This function is available for controls and action nodes. Use this function in debugger mode to halt rendering at the selected points. You can multi-select nodes with the Ctrl key to apply the function once to all selected nodes.
Setting bookmarks	This function opens a dialog, in which to add a description of the bookmark. The object is then labeled with a bookmark in the definition tree view. For more information, see Bookmarks on page 46.
Export	This function opens **Save as**. In this process, the XML definition of the selected node and all of its subordinate nodes are saved in the file.
Import	This function opens **File selection**. 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 it.

Function	Description
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. **Undo** undoes the previous action, whereas the **Redo** restores the prior state.
Search object references	Search all references, which reference the current object. If the search is successful, the object references are displayed with position, module, and object type. If the search is for a reference within a collection, the results are grouped by related columns.
Search	This function opens a dialog in which various search parameters can be entered (Ctrl + F). For more information, see Find and replace on page 36.
Find next	This function continues the current search using the current search parameters andmarks the next relevant node. The search is executed even if the **Search** dialog is not open.

Working with the preview

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. The preview is displayed in the Web Designer preview area.

NOTE: Other browsers or other versions of the same browser can generate views other than that you see in the preview. To avoid display errors, conduct layout tests for the web application with specific browsers and browser versions.

NOTE: The preview function also writes data to the database or deletes database objects if this function is performed by the user. You need to be careful when you test the corresponding functions.

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.

Additional preview functions

Table 18: Additional toolbar functions in the preview

Icon	Function
	One node back/forward.
	Use **Back** and **Forward** to navigate backward or forward between selected objects in the history in the Web Portal preview mode. However, the history is not shown here.
	Zoom in on preview. The preview window switches to full screen mode. 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. The button is enabled after compiling has successfully completed. It opens a popup window containing the HTML source code generated by the Web Designer. You can debug directly in the source code.
Window size: 1024 x 768	Customize window size. This option is only available when the window is not docked. You can open the list with the arrow and select a value from the listed monitor resolutions. After selecting a resolution, the preview is automatically set to this size. This makes it possible to estimate whether the web application required scrollbars. The available space in the web browser to be used also plays a significant role as well as the number of items shown in the menu bar.
	Use this button to swap from the mobile view to the desktop view.
	Use this button to swap from the desktop to the mobile view.
language	**Language** allows you to set the language for the preview. You can choose from the languages in which you have logged in to the Web Designer or other One Identity Manager tools. For more detailed information about login languages, see the *One Identity Manager Configuration Guide*.
Address bar	In the address bar, you can see the web application URL as in a browser. You can enter additional URL parameters either here or in the browser in which you opened the web application. Parameter examples: • **imx_layout**: This parameter accepts the values **tablet** and **desktop**. • **imx_culture**: This parameter accepts the values **de-DE** and **en-US**.

To enter a parameter with a value

1. Click the address bar next to the URL in the preview window.
2. Enter the following data, for example:

```
&imx_layout=tablet
```

3. Press **Return**.

   You preview changes to mobile view.

   NOTE: Entering a URL parameter works in the same way as the and buttons as well as the **Language** options box.

Debugging mode support functions

NOTE: A WCF connection must be established for functions which support debugging mode.

Table 19: Function supporting debug mode in the preview

Icon	Function
	Updating the preview. The Web Portal start page is displayed on update. This button has varying displays. The following visuals are possible: • If the button is displayed in combination with the start icon, modifications made to the web project in the definition tree view have not been compiled. After you click the button, the modifications are compiled and displayed in the preview. • The button is combined with a question mark if it is unclear whether modifications have been made to the web project. Use the arrow next to this button to select other menu items. The following menu items are available: • Create new predefined connections: You can add any amount of login data for different users. Predefined login data is listed under this menu item. If you are one of the users in this list, you can login simple by mouse click. This saves you entering your login name and password in the login on the start page preview. Predefined connections are deleted the moment you close the Web Designer. • Copy to clipboard: If you select this menu item, you copy the link to your web project start page that you see in the preview. • The browsers you use on your computer are automatically inserted into the menu as extra menu items. The preview is displayed with the browser of your choice.
	Suppress action execution. Use this button to prevent action execution in the preview. This can be helpful when you want to view a certain node in the definition tree view in more detail, for example.

Icon	Function
	One node back/forward. Use **Back** and **Forward** to navigate backward or forward between selected objects in the history in the Web Portal preview mode. However, the history is not shown here.
	Enable debugger. Use this button to toggle the debugger on and off. If the button is blue highlighted, the debugger is enabled. All actions are executed in succession (for example one node at a time) when the debugger is enabled. If the debugger is stopped in "Pause" mode, To continue the debugger, click **Resume execution**.
	Single step mode. The single step mode button is only enabled when the debugger is enabled. If single step mode is enabled, the executed action is suspended at every execution step in the definition tree view. If single step mode is disabled, the action continues on to the next break point (break points are inserted on action nodes in the definition tree view using the context menu).
	Resume execution This button is enabled when the debugger is enabled insofar as any program steps remain to be carried out, for example no user entries are anticipated. Clicking this button launches execution of the next program command.

Footer toolbar functions

In addition to a toolbar, the preview window also has a footer toolbar. The following functions are available to you.

Table 20: Functions in the preview window status bar

Icon	Function
	Status display. Here, "Done" means that compilation and rendering of the current project state have been completed and that entries can be made in the relevant pages.
769x485	Pixel lines. The pixel line count indicates the dimensions of the window area that is 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 is influenced by the screen resolution and the number of menu bars displayed in the browser.

Icon	Function
	Use this button to switch from the preview to the monitor page.
	Use this button to switch from the monitor page to the preview.

Solution

You can open the solution view from the **View** menu. It lists all the Web Designer objects that can be opened from the navigation view. The solution view depends on your choice of project. If a solution project was selected when the Web Designer was started, the entries are listed by database module and you can disable compilation of individual database modules.

You can identify objects whose status has changed in the Web Designer on the **Solution** tab. Differences between changes in the Web Designer and changes to resources in the database or on the hard disk are marked. A new tab is opened in the definition tree view by double-clicking on an object. The following table described the features of the **Solution** tab.

Table 21: Database project solutions and solution projects

Project	Description
Database project	In database projects, all the elements are listed under the **Solution** node that can be opened from the navigation view. **Table 22: Solution columns of a database project** **Column** / **Description** Object identifier — Name of the object. DB status — If the database is up to date, this shows "Up to date". Otherwise the user is who made the change is displayed. Status Web Designer — This shows "Changed" if changes have been made or "Unchanged" if not. Reload — This show a option if the object is new. Selected object can be reloaded. Select **Reload selected objects** to reload the selected objects.
Solution Project	In a solution project, the database modules are listed under the **Solution** modes as subnodes. Database modules whose status has not changed, are shown as collapsed nodes.

Project	Description
	If you mark a subnode, you can run additional tasks. Use the right mouse button to open the same context menu that is available in Windows browsers. Available items are, for example, Apache Subversion, Git or similar.

Table 23: Solution columns of a solution project

Column	Description
Object identifier	Name of the object.
Status drive	If the drive is up to date, this shows "Up to date". Otherwise a time when the changes will be made is shown.
Status Web Designer	This shows "Changed" if changes have been made or "Unchanged" if not.
Reload	This show a option if the object is new. Selected object can be reloaded.
	Select **Reload selected objects** to reload the selected objects.
Path	Path to file on the drive.

Additional functions can be carried out on the solution project's sub nodes.

The following tables shows the actions you can execute on the **Solution** tab.

Table 24: Actions tab "Solution"

Button	Description
	Refreshes the list of database objects.
	Reloads selected objects.
	Marks all modified objects to be refreshed.
	Disables module compilation. Compilation runs without the disabled module. (This action is only available if a solution project was selected when the Web Designer was started.)
	Filters modified objects only.

Viewing generated code

You can view and copy generated code with help of the buttons from the database object toolbar in the definition tree view. This function is particularly useful if you are

looking at C# code in detail and want to reuse parts of the code or want to examine an error in more detail.

NOTE: If you want to search for a specific point in the code, you can use the shortcut **Ctrl** + **F**.

To copy generated code

1. In the definition tree view, mark the position in the code and select **Copy**.

   The code is copied into the clipboard.
2. Inset the copied code at another point.

Detailed information about this topic

- Viewing error messages on page 138
- Executing Microsoft .NET Framework code on page 131

Bookmarks

You can set bookmarks for custom and defined objects at any place in the definition tree view.

To set a bookmark

NOTE: The relevant data object must already be selected and opened in the definition tree view.

1. Mark the relevant node in the definition tree view.
2. Select the **Set bookmark** function from the context menu and enter a description for the bookmark.
3. Click **OK**.

To remove a bookmark

- Delete the bookmark from the definition tree view context menu.

  This removes the bookmark.

To manage bookmarks

1. In the **View** menu bar, select the **Bookmarks** item.

   The **Bookmarks** pane is displayed in the lower pane of the Web Designer. This lists bookmarked objects in chronological order.
2. Double-click an object in the **Bookmarks** pane.

   NOTE: The marked object is also shown in the definition tree view. Navigate between bookmarks by pressing **F2**.

Bookmarks has its own toolbar which is described in the table below.

Table 25: The "Bookmarks" toolbar

Icon	Function
	Delete all bookmarks. All bookmarks are permanently deleted from the list after the security prompt is confirmed.
	Edit description. This opens a dialog in which a description can be entered for the bookmark. The description can be extended, changed, or delete. This function can also be called by right clicking on the selected object.
	Delete current bookmark. This deletes the current bookmark permanently from the list after the security prompt is confirmed. All bookmarks of the marked modules or components are deleted when modules/components are grouped together. This function can also be called by right clicking on the selected object.
	Group by module/component. Bookmarked objects are grouped by their respective affiliation under modules or components and are listed in alphabetical order.
	Show bookmarks for current web project only. This filters bookmarks found in the compiled web project only. Hidden bookmarks are shown again by clicking on the button.

Command list

The command list shows all modifications carried out on the current object in the definition tree view. A command list is kept for each object in the definition tree view. You can use the buttons in the toolbar to undo or redo commands.

Implemented commands are flagged with the icon in the list. This icon is missing for commands that were undone.

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.

Table 26: Functions in the command list

Icon	Function
	Undo. Undoes the last implemented command in the list.

Icon	Function
	Redo. Redoes the last implemented command in the list.

Properties

Once you have successfully compiled a web project, you can call up important information on individual nodes in the **Properties** window. In this window, you can check the property values of controls during application runtime.

NOTE: A WCF connection must be created for this function.

The **Properties** window is divided into two parts. On the left, the ASP.NET control hierarchy is displayed as it was on the web server when the web page was created. The associated database objects and IDs are also listed with the controls. On the right, the properties and property values for the controls are displayed.

To view properties of a node you can, for example, click a point in the preview window to view it in more detail in the **Properties** view. The point is highlighted in **Properties**.

NOTE: This is only possible with display nodes, since only they are displayed in the preview window. Data and action node properties are only displayed in debug mode.

To display a definition object

1. Click the point in the preview that you want to see in more detail.

   The clicked object is marked in the **Properties** hierarchy.

   NOTE:**Show definition object** and **Query here** are shown in the context menu if the marked object is a control. No context menu is available if you selected another object type.

2. In the **Properties** view, right-click the marked object.

   Show definition object and **Show query** are shown in the context menu.

3. Click the **Go to definition object** menu item.

   The definition object is marked in the definition tree view.

Table 27: Toolbar functions in the "Properties" view

Icon	Function
	Marks superordinate nodes. After clicking this button, the parent nodes for the definition object are marked in **Properties**.
	Marks subordinate nodes. Use this button to navigate in the view to a subordinate node in the node tree

Icon	Function
	structure. On the right-hand side of **Property** column, the property associated with the node marked is highlighted in color.
	Closes all nodes and subnodes. You can close all nodes and sub-nodes in the view with **Collapse all entries**. Only the root node is visible.
	Opens all nodes and subnodes. Use **Expand all nodes** to expand all nodes and sub-nodes. This displays all levels in the hierarchy.
	Displays more details. Use this button to mark the definition object in the definition tree view and view more details.

Multilingual captions

In the **Multilingual captions** dialog, you can add and edit multilingual captions and generate references to nodes in the definition tree view. Open the dialog using **Edit | Captions** in the menu bar.

Multilingual captions are standalone **objects** and are not saved in web project files. Instead, a reference to the multilingual caption is created in the web project nodes.

Before adding or editing multilingual captions, in your web project you need to define objects that output captions in your web application. Keys are created for these objects. These keys contain a caption for each language. This means that the keys are translated into the different languages you wish to use.

Adding and editing multilingual captions

To add and edit keys and translations, use the **Captions** tab in the dialog. Here, the toolbar displays the different functions available. These are listed and described in the table below.

Table 28: Toolbar functions for "Multilingual captions"

Icon	Function
	Add Adds a new caption.
	Delete. Deletes from the database the caption that is active in the edit view after confirming the security prompt.

Icon	Function
	Save. Saves a new or modified caption in the database.
	Assign caption. This button is only enabled if the dialog was called by a node in the definition tree view. Pressing this button assigns the active caption in the **Edit** area to the node.

NOTE: If you open **Multilingual Captions** when editing a node in the node editor, it contains an extra tab, **Quick edit**, with additional settings. In this case, the selected caption is assigned to the relevant node in the same work step.

The following properties are edited to add and edit multilingual captions in **Captions**.

Table 29: Multilingual captions properties

Property	Description
language	Indicates the language in which the caption is written.
Key	A unique value assigned for object referencing.
Text	The caption to be displayed in the corresponding language.
Custom caption	Custom caption that replaces the standard caption displayed. With custom captions, it is possible to modify captions in the standard modules without having to create a custom module.

NOTE: You can set the regional language for captions in **Language**. For example, for the **English** language, you can select **English - United Kingdom [en-GB]** or **English - United States [en-US].** The settings for the web browser in which you open your web application also play an important role when adding captions in different languages. If a translation into the selected language or regional language is not available in your web browser, a translation from the next family of languages above is sought.

The key that outputs the caption in the web application can be entered and edited manually in a SQL editor. If the node does not have a key, the SQL editor is empty. If a caption exists for a key, the following SQL expression appears in the SQL editor:

```
translate ("#LDS#<Key>")
```

In this case, `translate` stands for calling the associated Web Designer SQL function. `#LDS#` means that a key containing the caption follows.

If no appropriate translation for a key in the language you want to offer is found, you can create a new caption. The SQL field in **Quick edit** is empty when there is no translation. You can manually enter the translation as a SQL expression in the field.

NOTE: If there is no translation for a language, the preview and web application display the caption that is saved in the **Key**.

Example:

```
translate("#LDS#Hello World!")
```

This SQL expression appears as `Hello World!` in your web application. There is no translation or caption for the key in the database.

This function is particularly useful during the initial phases of a project, when captions 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.

Translate also allows you to use parameters. When several parameters are used, you do not need to enter the parameter references (`{0}`) in the key. It is sufficient to enter the parameter references in the caption. If you want to display the parameter reference in the caption, you need to enter the relevant parameters in the function. The following example shows what this variation looks like.

Example with two parameters:

```
translate('#LDS#Please fill in the field "{0}" on the request "{1}".', datacaption
(shoppingcartitem, currentcolumn), from shoppingcartitem select current displayvalue
(uid_accproduct))
```

How to add a caption to a **Label**-type node is described below.

To add a node directly to a caption

1. In the definition tree view, select a node of **Label** type to which you want to add a caption.
2. Select the **Node editor** view.
3. Click next to the **Text** field.

   This opens **Multilingual Captions**.
4. Select the **Quick edit** tab.
5. Perform one of the following tasks:
   a. Enter a SQL expression.
   b. Edit the SQL expression.
6. Save the changes.

Searching for captions

To edit existing multilingual captions, a search function with several options in available at the top of the dialog.

Table 30: Search options in the "Multilingual captions" dialog

Option	Description
Search key and value	Searches keys and captions.
Search for key only	Searches only in keys.
Search for value only	Searches captions only.
Search in all available languages	Searches all existing languages.

During caption searches, both the standard and custom captions are searched. The search results are shown in the result list, where they can be selected for editing by clicking them.

To search for a caption

1. t**Multilingual captions** from the web project menu bar or from the node of choice in the definition tree view.
2. Narrow your search down using one of the options.
3. Select a language or regional language to be applied to the search in the **Language** field.

   NOTE: In the Web Portal, captions and other values, such as numbers or dates, are displayed depending on the regional language settings of the browser you use. When maintaining multilingual captions, you must specify a language which applies for all captions. For more detailed information on languages, see the *One Identity Manager Configuration Guide*.
4. Enter part or all of the caption in the search box and click .

   All captions matching the search string are displayed.

   NOTE: If one of the **Search key and caption** or **Search for key only** options is set, the keys shown in the result list are labeled with an asterisk (*).

Import object

You can import modules and components into the Web Designer. to using the standard Windows file import dialog. The function recognizes automatically whether complete modules or components are involved and imports them as such. In all other cases the selected file is loaded as a project file.

To import an object

1. In the **Edit** menu, select the **Import object** menu item.

   This open the Windows, **Please select a source file**, default dialog.

2. Select an object and click **Open**.

   This imports the object.

Related applications

This function can insert, delete, and edit external web applications which are shown in the Web Portalunder **Applications**.

To edit related applications

1. In the **Edit** menu, select the **Related applications** menu item.

   This opens **Related applications**.
2. Select an application from the list of displayed applications.

   The name of the selected application is shown in the **Related application*** field. This name is the unique key for the application.

   NOTE: Fields marked with * are compulsory.
3. Make any necessary changes to application settings.
4. Save the changes.

The following settings are available.

Table 31: Settings for a related application

Property	Sub Property	Meaning
Language-dependent text	Language	Language selection.
	Display name	Enter a name for the application. This application name is displayed. It does not have to be the actual name of the related application.
	Description	Enter a description.
URL		Enter a base URL.
Type of display		Display type selection.
Parent applic-ation		Parent application selection. The parent application entry serves as a reference and enables sub-forms of a main application to be accessed directly.
Image		Uploads an image file. Image uploading is optional.

To add a new related application

1. In the **Add in the Related applications** dialog, click the **Add** button.
2. Enter a unique key for the web application in the **Related applications*** field.
3. Fill in the additional fields as described in the table.
4. Save the changes.

To delete a related application

1. In the **Related applications** dialog, select an application from the list of displayed applications.
2. Click **Delete**.

   The application is deleted from the list.
3. Save the changes.

Check model integrity

This function check 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.

To check your project for model integrity

- In the **Edit** menu, select the **Check model integrity** item.

  Your web project is checked. This can take a few minutes. When the check is finished, a message is shown with the results of the check.

Check accessibility

This function checks the web project to ensure rules meet current standards for web application accessibility. A rule violation example is, when no alternative text is given for missing visuals. Violations of accessibility rules do not generate compilation errors.

To check your project for accessibility

- In the **Edit** menu, select the **Check accessibility** menu item.

  Your web project is compiled. After compiling, a log is shown in the **Compiling** window. If there is an inconsistency, a detailed warning message is shown in the task view.

Object properties

Object properties shows the properties of the files that are generated by the Web Designer and that are saved as objects in the `DialogAEDS` table. Each module, component, project file and web project is represented as a separate file.

To open object properties

1. In the navigation view, select an object.
2. Click in the toolbar.

   This opens the **Object properties** dialog. There are three tabs available. You can edit the data on **Properties** and **Access permissions** if you have the right access permissions.

Change labels for web projects

With change labels for web projects, you can move a web project to another database. To do this, all web project database objects are booked to a change label. The Database Transporter is used for moving the web project. You can create and edit change labels in different One Identity Manager tools.

In the Web Designer toolbar, you can use to open the dialog for managing change labels. The following functions are available here.

Table 32: Functions for managing change labels

Icon	Function
	Open/close edit view. Displays the edit fields for the change label. You can edit the change label.
	Create a change label. Is the only button enabled after clicking . Opens the edit fields for a new change label.
	Delete change label. This button is enabled once you have selected a change label. After clicking , respond to the message "Do you really want to delete the change label?" with **Yes** or **No**.
	Save change label. This button is only enabled when you have modified change label settings or created a new change label.

To create or edit change labels

1. Click next to the **Change labels** list.

   A dialog for creating and editing change labels opens.
2. Perform one of the following tasks:
   a. Create a new change label using .
   b. Select a change label from the list and open the edit view using .
3. Enter the change label data.

   Table 33: Change label properties

Property	Meaning
Change label	Change label name. This name is used to select the change label for allocating the changes or creating a customer transport package.
description	Detailed description of the change label.
Parent change label	Specifies a parent label (optional).
Status	Status of the object changes, for example, development, test, and production.
Status comments	Additional comments in relation to the status
Comment	Additional information to enable tracking of changes to a change label.
Label type	Label type for more detailed classification. Label type "Change" is used by default.
Locked	Indicates if the change label is locked. If a change label is locked, no further changes can be booked to this label.

4. Save the changes.

   This closes the dialog. The change label is preselected in the **Change label**.

To delete a change label

1. Click next to the **Change labels** list.

   A dialog for creating and editing change labels opens.
2. Select the change label you want to delete and click the button.
3. Confirm the security prompt with **Yes**.

4. Close the dialog using **Cancel**.
5. Click **OK** to close the **Edit change labels...** dialog.

To apply a change label to a web project

1. Click in the toolbar.
2. Select the group of database objects you want to link to a change label:
   a. **Modules, components, and configuration**

      This option links the databased objects generated by the Web Designer with a change label.

   b. **Captions**

      This option links all the multilingual captions required by the web project with a change label. For more information, see Multilingual captions on page 49.

      If any of the selected objects are already linked to one or more change labels, these links are not changed or deleted. An additional link is added.
3. Click **Next**.
4. Select the change label and click **OK**.

   The change label is copied.

Detailed information about this topic

- One Identity Manager User Guide for One Identity Manager Tools User Interface

File-based workflow in the Web Designer

With the Web Designer, you can edit custom objects locally on your computer. To do this, export all existing custom objects from the database. All object types are taken into account and copied to the hard drive.

After exporting these objects, they can be edited, removed, or supplemented by additional objects on your computer. Object editing on your computer is no different than editing in the database project.

NOTE: You can only export complete objects for file-based editing. A default object can be custom configured on the hard drive. A hybrid type of single object is not provided for.

To edit files in the Web Designer

1. Start the Web Designer and, in the **Solution project in Select project save type** dialog, click the **Solution project** option.
2. Select the **Load solution** option.

3. Perform one of the following tasks:
    a. In the Windows dialog, select the `MyWebDesignerSolution.wds` file from the folder in which the solution is stored.
    b. Select the name of the solution project that was opened last.
4. In the navigation view, select the file you want to work.

    NOTE: Files that come from the local system are found in the From file system **node.**
5. Edit the objects that you want to change and save the modified objects to the hard drive.

To open the folders of the edited file

NOTE: The custom module you want to change must be open in the Web Designer.

1. Mark the tab for your custom module and select the **Node editor** view.
2. Open the Windows Explorer by clicking .

    The associated folders for the edited file are show in your Windows Explorer.

To open the Windows Explorer's context menu in the Web Designer

NOTE: The custom module you want to change must be open in the Web Designer.

1. Mark the tab for your custom module and select the **Node editor** view.
2. Mouse over and open the context menu with right mouse button.

    This displays the Windows Explorer's context menu.

Detailed information about this topic

- Exporting a module from the database to the hard drive on page 58
- Importing from a solution to the database on page 60

Exporting a module from the database to the hard drive

To export data from the database to a local computer, you require read permissions for specific database tables. Read permissions are needed for the following tables.

- `DialogAeds`
- `DialogAedsAction`
- `DialogAedsActionType`
- `DialogAedsActionHasObject`

NOTE: If these database tables do not have read permissions, the wizard will not open.

During export, all objects in the selected module are copied to the hard drive. This includes the following objects.

- Modules
- Components
- Layout definitions
- Web projects
- Project files
- Form Types

In the steps below, the **Create solution** option is selected.

To export files to a local computer

1. Start the Web Designer and, in the dialog, select the **Solution-Project** option as the type of storage for the project.

   The following options are displayed under the **Solution-Project** option.

   Table 34: Selection options for solution-project

Option	Description
Create solution	Creates a new solution containing all the necessary objects.
Load solution	Loads an existing solution.
File path to MyWebDesign-erSolution.wds	Opens the file containing the solution settings.

2. Create a new solution using **Create solution**.
3. On the **Export current project** page, select the file path for the new solution from the **Path to output folder** menu and enter a name in the **Solution name** field.
4. Create the new solution with **Next**.

   The solution is created containing the following objects.

   Table 35: New solution objects

Object	Description
Export module objects	All custom files
MyWebDesignerSolution.wds	Solution settings
Project.xml	Project file, for organizing individual files
ActionTypes.xml	File for references

5. Click **Finish**.

When the wizard is closed, you can continue working with the Web Designer as before. In addition, you can perform other tasks for file-based editing.

NOTE: If you enable **Use selected project as default**, the same project is opened the next time you launch the Web Designer.

To disable the "Use selected project as default" setting

1. Use the **Menu | Settings** item to open the **Global settings** dialog.
2. Open **Editor** and enable **Ask for save type after every start up**.

   When you next start up, **Select project save type** is displayed.

Importing from a solution to the database

To use the wizard for importing, the solution must manage database module objects. If this is not the case, you must import using another method.

To import files into the database

1. Perform one of the following tasks:
   a. On the Start page or from **Edit menu**, open the **Import data object into database** dialog.
   b. Publish your project in the database and confirm the import object prompt.

   NOTE: If you enable **Do not ask again**, this box will not open again.

   If you want to display this dialog again, activate the setting **Ask for import after publishing** in the **Global settings** dialog, which you can access through the **Edit** menu.
2. Select the objects you wish to import into the database from dialog on the **Object selection** page.

   NOTE: If the **Import object-dependent references** option is enabled, files used by references can no longer be deselected, thus avoiding inconsistencies.

   NOTE: If the **Remove deleted objects from the database** option is enabled, objects that are not on the hard drive are removed from the database

   TIP: Only deselect objects you know have not been modified or are not needed to publish your project.

   Otherwise, files may be missing during compilation or assemblies created later may be incomplete.

Web project configuration options

You can customize the default Web Portal. You can modify filters for searching employees or products, enable, and disable functions, and change properties displayed in result lists.

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.

Detailed information about this topic

- Project configuration - web project on page 61
- Project configuration - customization on page 76
- Project configuration - customization overview on page 77
- Statistical references on page 78
- Dynamic references on page 79

Project configuration - web project

A web project is defined through the configuration key under **Configuration**. This definition only applies for this web project. They can consist of modules and components that are used by several web projects and can also contain their own configuration key definitions. The web project uses default configuration keys, if none are defined under **Configuration (custom)** or within the web project used by a module's configuration key.

NOTE: The configuration keys are declared directly in the modules or components under **Configuration**.

Selecting a database object

To select a database object

1. Select the relevant report in the navigation view.

   The selected category is listed with the corresponding database objects .

2. Proceed as follows.
    a. Double-click a database object.
    b. In the field for searching the navigation view, enter the name of the required database object.

    This displays the selected database object in the definition tree view.

NOTE: The assignment of a configuration key with a value is always applied to the module or component. These configuration keys are shown when you click **Configuration** or **Configuration (custom)**. All configuration keys that are not listed here, are not given a value.

Configuring a web project

To configure a web project, you edit configuration keys in **Settings...**.

To configure a web project

1. In the toolbar, click **Edit**.
2. In the **Configure project** submenu, select the **Web project** item.

    The **Configure project** tab is displayed with the existing configuration sections.

    NOTE: If the debugging environment is deactivated or the web application is not running correctly, it is possible that the current status of the web project is compiled before the data is displayed.

    NOTE: The view in the **Configure project** tab is the same view that you access by pressing the **Settings...** button.
3. Open the required configuration section.

    This displays the configuration key stored under the configuration section.

Finding a specific configuration key

To find a special configuration key

1. Perform one of the following tasks:
    a. Enter part or the entire search term on the **Configure project** tab and click .
    b. Click to open the **Options** pane.

    A select of options is displayed, which you can use for searching. The following options are available:

Table 36: Configuration key search options

Option	Description
Parameters	Searches in parameters only.
Key	Searches in keys only.
Description	Searches in descriptions only.
File Name	Searches in file names only
Only from type	Searches in a specific parameter-type only.

NOTE: The **Only of type** option has a menu that you use to select a parameter type.

2. Check one or more of the boxes and click .

   Configuration keys matching your search criteria are displayed.

3. Select a configuration key.

   More information about the selected configuration key is displayed in the **Details** pane.

   NOTE: In the **Details** view, detailed information about the configuration key of the configuration section is displayed, which is implemented in the **Object definition** view in the definition tree view. Depending on which parameter type is used for the configuration key, when editing the configuration key in the **Settings...** view, different input options are available.

4. Enter the value **- Is equivalent to standard value** in the **Value (custom)** column, by clicking .

5. Perform one of the following tasks:

   a. Click to open the wizard.

   b. Click to jump to the definition object.

      A new tab is opened in the definition tree view and the corresponding node is marked in the **Object definition**.

   c. Set the custom value back to default by clicking and confirming the prompt.

   NOTE: Different editing options for the configuration key values are described in more detail in the following sections on individual parameter types.

The following parameter type can be assigned to a configuration key:

- Boolean value
- SQL filter condition
- Selection from value list
- Free text
- Property list
- Image

- Color value
- Color dictionary

Finding references to a configuration key

The Web Designer can display all the places where a specific configuration key is used.

To view references for a configuration key

1. On the **Configure project** tab, mark a configuration key.

   For more information, see Finding a specific configuration key on page 62.
2. In the **Details** pane, click the **Show definition object** button.

   The definition object in which the configuration key is defined is marked in the definition tree view.
3. Click the marked definition object and select the **Search object references** context menu item.

   The object references corresponding to the search are listed in a dialog.
4. Double-click a reference to access more details in the **Node editor** view.

Calling up the "Settings" view

To call the "Settings..." view

NOTE: A database object must be selected to be edited in the definition tree view.

- In the definition tree view, click the **Settings** button.

  NOTE: If several configuration keys exist, they are grouped together in one configuration section. You can expand the configuration sections to go to the configuration key of your choice.

Parameter type "Boolean Value"

Configuration keys of the type "Boolean value" can be used to configure "Yes" or "No" questions. An example for such a question would be to allow navigation with keyboard shortcuts.

This chapter describes how to edit a configuration key value. Declaring a configuration key is covered in another section of this guide. For more information, see Declaring configuration keys in modules and components on page 129.

To edit the value of a configuration key value of the "Boolean value" parameter type in the "Settings..." view

NOTE: Before you can edit the values in a database column, you must add a matching entry in **Configuration (custom)**. You can recognize this entry because it is marked with an **X** in the **Value (custom)** column in the list of configuration keys in the **Settings...** view of the selected database object.

1. Open the required database object in the definition tree view and select the **Settings...** view.
2. In the **Settings...** view, select the configuration key of the parameter type "Boolean Value", which you created previously in the **Object definitions** view.
3. Enter the ✖ value **- Is equivalent to standard value** in the **Value (custom)** column, by clicking ✚.
4. Enable or disable the configuration key option under **Details**.

Parameter type "SQL filter condition"

You can use the configuration key of the "SQL filter condition" parameter type to, for example, query database objects with certain properties.

This chapter describes how to edit a configuration key value. Declaring a configuration key is covered in another section of this guide. For more information, see Declaring configuration keys in modules and components on page 129.

To edit a configuration key value of the "SQL filter condition" parameter type in the "Settings..." view

NOTE: Before you can edit the values in a database column, you must add a matching entry in **Configuration (custom)**. You can recognize this entry because it is marked with an **X** in the **Value (custom)** column in the list of configuration keys in the **Settings...** view of the selected database object.

1. Open the required database object in the definition tree view and select the **Settings...** view.
2. In the **Settings...** view, select the configuration key of the "SQL filter condition" parameter type that you created previously in the **Object definitions** view.
3. Enter the ✖ value **- Is equivalent to standard value** in the **Value (custom)** column, by clicking ✚.
4. Perform one of the following tasks:
   a. Enter the filter condition in the field.

      NOTE: You can only use SQL to formulate your expression. Web SQL cannot be used.

   b. Open the Web SQL Editor by clicking 🔧.
5. Enter a SQL condition and close the Web SQL editor using **Apply**.

Parameter type "Selection from value list"

A use case for the configuration key of parameter type "Selection from value list" would be gender data. For example, the values `male`, `female` and `other` can be chosen.

This chapter describes how to edit a configuration key value. Declaring a configuration key is covered in another section of this guide. For more information, see Declaring configuration keys in modules and components on page 129.

To edit a configuration key value of the "Selection from value list" parameter type in the "Settings.." view

NOTE: Before you can edit the values in a database column, you must add a matching entry in **Configuration (custom)**. You can recognize this entry because it is marked with an **X** in the **Value (custom)** column in the list of configuration keys in the **Settings...** view of the selected database object.

1. Open the required database object in the definition tree view and select the **Settings...** view.
2. In the **Settings...** view, select the configuration key of the "Selection from value list" parameter type that you created previously in the **Object definitions** view.
3. Enter the ✖ value **- Is equivalent to standard value** in the **Value (custom)** column, by clicking ✚.
4. In the **Details** menu, select the preset value of your choice.

   The selected value is displayed in the **Key** column in **Configure project**.

Parameter type "Free text"

The configuration key of the "Free text" parameter type is used when you want to provide several links. You can enter a link to the company's home page, for example.

This chapter describes how to edit a configuration key value. Declaring a configuration key is covered in another section of this guide. For more information, see Declaring configuration keys in modules and components on page 129.

To edit a configuration key value of the "Free text" parameter type in the "Settings..." view

NOTE: Before you can edit the values in a database column, you must add a matching entry in **Configuration (custom)**. You can recognize this entry because it is marked with an **X** in the **Value (custom)** column in the list of configuration keys in the **Settings...** view of the selected database object.

1. Open the required database object in the definition tree view and select the **Settings...** view.
2. In the **Settings...** view, select the configuration key of the "Free text" parameter type that you previously added in the **Object definitions** view.

3. Enter the ✖ value **- Is equivalent to standard value** in the **Value (custom)** column, by clicking ✚.
4. Enter your text in the field below **Details**.

   The entered text is shown next to the edited configuration key in **Value (custom)** in **Configure project**.

Parameter type "Property list"

The configuration key for the "Property list" parameter type can, for example, be used to modify columns in the **Address Book** view that is displayed in the Web Portal.

This chapter describes how to edit a configuration key value. Declaring a configuration key is covered in another section of this guide. For more information, see Declaring configuration keys in modules and components on page 129.

To edit the configuration key value for the "Property list" parameter type in the "Settings..." view

NOTE: Before you can edit the values in a database column, you must add a matching entry in **Configuration (custom)**. You can recognize this entry because it is marked with an **X** in the **Value (custom)** column in the list of configuration keys in the **Settings...** view of the selected database object.

1. Open the required database object in the definition tree view and select the **Settings...** view.
2. In the **Settings...** view, select the configuration key of the "Property list" parameter type that you created previously in the **Object definitions** view.

   NOTE: If you want to see more details for a configuration key without editing the value, open **Configure dashboard display** by clicking 🔨 without converting the value in the **Value (custom)** column beforehand. This read mode is available for configuration keys with "Configuration object" and "Property list" parameter types.
3. Enter the ✖ value **- Is equivalent to standard value** in the **Value (custom)** column, by clicking ✚.
4. Open the **Database column selection** dialog by clicking 🔨.

   On the left-hand side, all selectable columns of the preset table are displayed. On the right-hand side, all columns in use are displayed.
2. Mark the relevant entry in the list of available columns and click ❯.

   The marked entry is displayed in the list of used columns. The following fields and options for editing selected columns are displayed in the lower part of the dialog.

Table 37: Settings for database columns

Setting	Description
Condition	SQL field specifying the condition.
Property	This setting makes properties available. Enable the setting and select a property.
Expression	SQL field for entering an expression.
Database table for simulated foreign key references	SQL field for entering a database table for simulated foreign key references.
Name of column for simulated foreign keys	SQL field for entering a column name for simulated foreign key references.
Comment	Field for entering a comment on using a database column.
Component for editing a value	This setting makes components for editing a value available. Enable the setting and select a component.
Component for displaying a value	This setting makes components for displaying a value available. Enable the setting and select a component.
Foreign key candidate filter	SQL field for entering a foreign key candidate filter.
Expression for display values	SQL field for entering an expression for a display value.
Descriptive text	SQL field for entering a descriptive text.
Name	SQL field for entering a caption.
Property values are not editable	SQL field for entering non-editable property values.
Maximum number of characters	Field for imposing a maximum on the number of characters used.
Minimum number of characters	Field for imposing a minimum on the number of characters used.
Control-ID*	Field for entering a Control-ID*.

NOTE: Using the ABC button, you can specify that the database column display name be used. If you click ABC again, the original database column name is displayed.

3. Configure your settings.
4. Save the changes.

To add a validation node to a database column

1. In the **Database column selection** dialog, select the relevant database column.
2. In the toolbar, click and select the **Validation** item.

   A new validation node is added under the marked entry and a number of edit boxes are displayed. For more information, see Table 37 on page 68.
3. Configure your settings.
4. Save the changes.

To add a parameter to a database column

1. In the **Database column selection** dialog, select the relevant database column.
2. In the toolbar, click and select the **Parameter** item from the context menu.

   A new parameter is added under the marked entry and a number of edit boxes are displayed.
3. Enter an identifier for the new parameter in the **Identifier*** field using the button.
4. Enter a value for the new parameter in the **Value*** field using the button.
5. Save the changes.

NOTE: Added validation nodes or parameters can be deleted using the button. This button is only enabled if you have added new properties.

Parameter type "Image"

You can use the configuration key of the "Image" parameter type for logos, for example.

This chapter describes how to edit a configuration key value. Declaring a configuration key is covered in another section of this guide. For more information, see Declaring configuration keys in modules and components on page 129.

To edit a configuration key value of the "Image" parameter type in the "Settings..." view

NOTE: Before you can edit the values in a database column, you must add a matching entry in **Configuration (custom)**. You can recognize this entry because it is marked with an **X** in the **Value (custom)** column in the list of configuration keys in the **Settings...** view of the selected database object.

1. Open the required database object in the definition tree view and select the **Settings...** view.
2. In the **Settings...** view, select the configuration key of the "Image" parameter type that you created previously in the **Object definitions** view.
3. Enter the value **- Is equivalent to standard value** in the **Value (custom)** column, by clicking .

4. Perform one of the following tasks:
    a. In the **Use image from resource** field, select an image.

       Your selection is shown in the **Image** preview.

       NOTE: Only images that have already been loaded once are available in **Use image from resource**.
    b. In the **Use stock image** field, select an image.
5. In the **Large** field, select the size of the image.

   In **Configure project**, you can see the file name of the selected image and the method of loading the image, in **Value (custom)**.

   NOTE: Use **Delete** to remove your selection again.

Parameter type "Configuration Objects"

The configuration key of the "Configuration Object" parameter type is used when you want to add or remove statistics.

This chapter describes how to edit a configuration key value. Declaring a configuration key is covered in another section of this guide. For more information, see Declaring configuration keys in modules and components on page 129.

To edit the configuration key value for the "Configuration objects" parameter type in the "Settings..." view

NOTE: Before you can edit the values in a database column, you must add a matching entry in **Configuration (custom)**. You can recognize this entry because it is marked with an **X** in the **Value (custom)** column in the list of configuration keys in the **Settings...** view of the selected database object.

1. Open the required database object in the definition tree view and select the **Settings...** view.
2. In the **Settings...** view, select the configuration key of the "Configuration objects" parameter that you created previously in the **Object definitions** view.

   NOTE: If you want to see more details for a configuration key without editing the value, open **Configure dashboard display** by clicking without converting the value in the **Value (custom)** column beforehand. This read mode is available for configuration keys with "Configuration object" and "Property list" parameter types.
3. Enter the value **- Is equivalent to standard value** in the **Value (custom)** column, by clicking .
4. Open the **Configure dashboard display** dialog by clicking .

   In this window, you can view the statistics and heatmaps for the selected element. The following functions are available in this dialog.

Table 38: Functions in the "Configure dashboard display" dialog

Function	Description
	Opens **Select statistics**. You can select and add statistics in the list of available statistics. Multi-select is available.
	Deletes marked statistics from the list of statistics used.
	Moves the marked statistics higher in the list of statistics used.
	Moves the marked statistics lower in the list of statistics used.

Table 39: Functions in the "Select statistics" dialog

Function	Description
	Shows you all the global statistics from the list of available statistics.
	Shows you all object-related statistics from the list of available statistics.

TIP: You can use multi-select by selecting items in the list whilst holding the Ctrl key.

2. Select a function from the toolbar.
3. Close the dialog after editing the relevant settings.

To edit chart settings

NOTE: You have enabled the **Chart** option.

1. In the **Type** field, select a chart type.
2. Enter a title for the chart in the **Title caption** or **Text** fields. This is displayed later for the organizational unit in the web application.

   For more detailed information on organizational units, see the One Identity Manager Web Portal User Guide.

   NOTE: The available fields that you need to edit depend on the chart type you selected in the **Type** menu. The properties **Title caption**, **Text** and **Tooltip** in this view use a SQL editor. Enter the captions for the organizational units shown in the web applications. For more information, see Multilingual captions on page 49.
3. Configure the selected statistic with the available fields and menus.
4. Click **OK**.

   Your settings are saved. The number of selected columns is displayed under **Details** in the field and in the **Settings...** view in the **Value (custom)** column.

To edit heatmap settings

NOTE: You have enabled the **Heatmap**. The dialog mask changes and divides into three parts, **General**, **Current database**, and **Historical data**.

NOTE: Statistics that need to be represented in heatmaps must return values for the `ElementValueZ` column from a query.

- Configure the selected statistic with the available fields and menus.

Table 40: Heatmap configurations

Area	Setting	Description
General	Additional Where clause	Used when loading a heatmap.
	Title	Replaces the statistic's display value. Parameters available in this expression are documented in the `ChartTitleInfo` class.
	Measurement parameters represented by rectangles	Display name for measurement parameters represented by rectangles.
	Measurement parameters represented by colored rectangles	Display name for measurement parameters represented by rectangles.
Data	Text	Specifies the caption for rectangles. Parameters available in this expression are documented in the `HeatmapDataPoint` class.
	Tooltip	Specifies the tooltips for rectangles. Parameters available in this expression are documented in the `HeatmapDataPoint` class.
	Limit	Specifies the color of rectangles according to data values. The color spectrum goes from red to yellow to green. Red means a critical value, green an uncritical value, and yellow a neutral or average value. Limits can be assigned to data values directly using double values. Alternatively, maximum, and minimum values can be set for the outer colors. The corresponding data value is then dynamically assigned. For neutral values a value can be optionally configured.
historical data	Statistics for monthly historical	Used to calculate historical comparison values.

Area	Setting	Description
	data	You can select one of the statistics available in the option box to be used to calculate historical comparison values.
	Statistics for annual historical data	Used to calculate historical comparison values. You can select one of the statistics available in the option box to be used to calculate historical comparison values.
	Text	Specifies the caption for tiles. Parameters available in this expression are documented in the `HeatmapDataPoint` class.
	Tooltip	Specifies the tooltips for tiles. Parameters available in this expression are documented in the `HeatmapDataPoint` class.
	Limit	Specifies the color of tiles according to data values. The color spectrum goes from red to yellow to green. Red means a critical value, green an uncritical value, and yellow a neutral or average value. Limits can be assigned to data values directly using double values. Alternatively, maximum, and minimum values can be set for the outer colors. The corresponding data value is then dynamically assigned. For neutral values a value can be optionally configured.

Parameter type "Color value"

You can use the configuration key of the "Color value" parameter type to change all the colors used.

This chapter describes how to edit a configuration key value. Declaring a configuration key is covered in another section of this guide. For more information, see Declaring configuration keys in modules and components on page 129.

To edit a configuration key value of the "Color value" parameter type in the "Settings..." view

1. Open the required database object in the definition tree view and select the **Settings...** view.
2. In the **Settings...** view, select the configuration key of the parameter type "Color value" that you created previously in the **Object definitions** view.
3. Enter the ✖ value **- Is equivalent to standard value** in the **Value (custom)** column, by clicking ✚.
4. Open the **Color** edit window by clicking 🔨.
5. Select the settings from the available options and click **Add colors**.

   The selected colors are displayed under **Custom colors**.
6. Click **OK**.

   In the **Settings...** view, the selected color value is displayed in the **Value (custom)** column and under **Details** in the field.

Parameter type "Color table"

The configuration key of the "Color dictionary" parameter type is used to determine the color scheme of statistical values. If the same values occur in different statistics, the values should be viewable in consistent colors. Thus, the "Approved" value is always "green" no matter which statistic this value appears in.

This chapter describes how to edit a configuration key value. Declaring a configuration key is covered in another section of this guide. For more information, see Declaring configuration keys in modules and components on page 129.

To edit a configuration key value of the "Color table" parameter type in the "Settings..." view

NOTE: Before you can edit the values in a database column, you must add a matching entry in **Configuration (custom)**. You can recognize this entry because it is marked with an **X** in the **Value (custom)** column in the list of configuration keys in the **Settings...** view of the selected database object.

1. Open the required database object in the definition tree view and select the **Settings...** view.
2. In the **Settings...** view, select the configuration key of the parameter type "Color table", which you created previously in the **Object definitions** view.
3. Enter the ✖ value **- Is equivalent to standard value** in the **Value (custom)** column, by clicking ✚.
4. Open the **Change color dictionary** dialog by clicking 🔨.

   You can add and remove color values in **Numbered values** and in **Predefined values**.

5. Click ✚ in a pane.

   A new color bar is shown below the corresponding pane.
6. Click the color of the new color bar.
7. In the **Color** dialog, select a color.
8. Click **Add color**.

   This displays the selected color under **Custom colors**.
9. Click **OK** to close the window.
10. Perform one of the following actions with the new color bar in **Change color dictionary**.
    a. Assign an item in the statistics to the color bar by hovering the mouse over the dashed area of the required color bar in the **Numbered values** area. Drag this bar to the position of choice whilst holding down the mouse button.

       The color bar is now in the right position. If the color bar in in the first position, it will appear in the first position of the statistics. This bar represents the value with the highest priority. The importance of the other bars is set according to their position in the list of color bars.

    b. Enter a name for the color bar in the corresponding field.

       The color bar now has a name. For example, the green color bar is called "Approved".
11. Close the **Change color dictionary** dialog using **Apply**.

    Your settings are saved. The number of selected columns is displayed under **Details** in the field and in the **Settings...** view in the **Value (custom)** column.

Project configuration - customization

This dialog allows you to create and edit substitution rules for module copies. Substitution rules are listed that were added with **Copy objects**.

NOTE: Substitution rules have higher priority that references in the definition tree. References in the definition tree are no changed in this case.

For example, a substitution rule from both the `VI_Delegation` module and the `Custom_Delegation` module copy can exist. The modules are mentioned by way of example in the steps below.

To edit a substitution rule

1. In the **Edit** menu, select **Configure project | Customization**.

   This displays **Configure project - Customization**. If substitution rules already exist, they are listed here. The following editing options are available in this dialog.

Table 41: Functions in the "Configure project - Customization" dialog

Function	Description
	Creates a new substitution rule. In the **Module name** column, (no name) is shown and the **Module name** and **Replace by** fields are empty.
	Deletes the marked substitution rules from the list.
Input field **Module name**	Specifies the module name. When you create a substitution rule the field is empty. Enter the module manually. Edit a substitution rule, overwrite the module name (for example, `VI_Delegation`) with another module name.
Input field **Replace by**	Specifies the module copy. When you create a substitution rule the field is empty. Enter the module copy manually. Edit a substitution rule, overwrite the module copy name (for example, `Custom_Delegation` with another module copy name.

2. Create an edit rule or modify an existing edit rule.
3. Save the changes.

NOTE: If a module is no longer needed, the substitution rules can simply be deleted.

Detailed information about this topic

- Creating object copies with the wizard on page 97

Project configuration - customization overview

On the **Customization overview** tab, you can view the modified sections of the configuration. Objects that exist in the object model, can be edited in the definition tree view. The Web Designer jumps directly to the object when you double-click on the respective node. These objects are labeled in the following table.

Table 42: Tab "Customization overview"

Objects	Description	Handling
Configuration parameter	Modified configuration parameters and subnodes are grouped by document in the list.	x

Objects	Description	Handling
Extensions	Added extensions are grouped by document in the list. • Extension nodes, for example, "Add node", "Remove node" and "Modify property" are listed under the added extensions. • Other child elements are listed under the extension node "Add mode".	x
Customizations	Overwritten modules are listed as objects you can also find listed under **Edit | Configure project | Customizations**.	x
Column-dependent references	Lists modified, custom, column dependent references.	x
Object-dependent references	Lists modified, custom, object dependent references by reference type, reference, and object.	
Root node	Root nodes are always displayed, whether with or without subnodes. This means that no changes have been made.	

The following table describes which editing options you have on the **Customization overview** tab.

Table 43: Functions on the "Customization overview" tab

Editing Option	Description
	Deletes all child elements • Mark the nodes you want to delete in the tree view and click . All nodes under a root node (**Configuration**, **Extensions**) can be edited.
	Deletes single nodes, like extension nodes, configuration nodes, customizations, or column or object dependent references. ***Deleting single nodes.*** • Mark the nodes you want to delete in the tree view and click .

Statistical references

A component can be referenced in two ways. Normally, a static reference is used to reference components directly using their names.

An example of a static reference is calling the component `VI_Popup`.

NOTE: It is not possible to use object-dependent references on the login page if the web

application is executing against an application server.

Detailed information about this topic

- Dynamic references on page 79

Dynamic references

The second way of referencing components is dynamically. Dynamic references can reference more than one component at a time. A specified criteria is used to determine which component will effectively be referenced at runtime. This method of referencing makes it easier to build extendable generic components.

Detailed information about this topic

- Statistical references on page 78
- Column-dependent references on page 79
- Object-dependent references on page 84

Column-dependent references

A column-dependent reference references a component's column definition. This reference can be used everywhere where contents of a specific database table is shown or edited. An instance of the component is generated depending on which data type and other metadata is linked to the column.

By default, the following components are used as column-dependent references for editing.

Table 44: Overview of components as column-dependent references for editing

Type	Component name
String	VI_Edit_Default VI_Edit_TextLong (if the column is multi-row defined)
Int32	VI_Edit_Int
DateTime	VI_Edit_Date
Foreign key	VI_Edit_FK
Dynamic foreign key	QBM_Edit_FKDynamic
Limited value columns	VI_Edit_LimitedValues

Type	Component name
Multi-value columns	VI_Edit_MultiValueProperty
Multi-value columns with limited values	VI_Edit_MulitLimitedValues (for parameter objects)
Multi-value foreign key columns	VI_Edit_MultiFK (for parameter objects)
Numeric value range	VI_Edit_NumericRange (for parameter objects)
Date range	VI_Edit_DateRange (for parameter objects)
Boolean	VI_Edit_Checkbox
Double	VI_Edit_Double
Decimal	VI_Edit_Decimal

By default, the following components are used as column-dependent references for visualizing.

Table 45: Overview of components as column-dependent references for visualizing

Type	Component name
Default	VI_Edit_View_Default
Binary column with contents "Picture"	VI_Edit_View_Picture
Column with contents "URL"	VI_Edit_View_URL

"Column-dependent references" tab

A column-dependent reference is a conditional reference to a component. The type of column in a database table (for example, `Bool`, `DateTime` or `String`) determines which component is used to edit the relevant content. To this end, the standard scope of delivery includes VI components which cover all possible column types and finds them automatically.

NOTE: There are also custom entries in the web project file. You can tell which of the various tabs an entry belongs to by the prefix.

To open the "Column-Dependent References" tab

1. In the **Edit** menu, select the **Configure project | Column-dependent references** menu item.

This opens **Column-dependent references**. The tab lists all existing columns with the corresponding default components, which are each given their own tab.

2. Select the required tab.

   The following tabs are available.

Table 46: "Column-dependent references" Tab

Tabs	Description
Edit values by column	Select columns whose data can be edited in components. For example, you want to use these components when you add a new employee in the Web Portal. The prefix "ColumnEditor_" indicates that the entry belongs on **Edit values by column**.
Display values by column	Select columns whose data can only be viewed in components and not edited. For example, you want to use a specific component for displaying an image. "ColumnViewer_" indicates that the entry belongs on **Display values by column**.
Foreign key filters	Create a filter to limit the number of foreign key values. An example of a filter would be, for the user to limit the number of departments an employee can use in the Web Portal. In this case, the filter should be set in "UID_Department" in the "Person" table. The list of filters includes the filters given on the tab **Editor** tab as well as the filters that can be used without components. "Filter_ColumnEditor_" indicates that the entry belongs on **Foreign key filters**.
Edit foreign key values	Specifies a component for all foreign key references to a table. This means the user enters a combination of tables and components. The selected components are used in the Web Portal when the table is referenced by foreign key. In the case of "Person", this means that the same component is always used if an employee and a group owner or request recipient is selected, whether a manager is selected or not. "FkColumnEditor_" indicates that the entry belongs on **Edit foreign key values**.

Tabs	Description
Edit multi-valued foreign key values	Specifies a component for all multi-value foreign key references to a table. The user enters a combination of tables and components. The selected components are used in the Web Portal when the table is referenced by multi-value foreign key. An example for using a multi-value foreign key would be a request for multiple employees. In the case of "Person", this means that "AutoComplete" is not only used to select the first recipient but also for subsequent recipients that are selected in the same request procedure.

NOTE: You can only delete entries that you added yourself. Default entries are write-protected. The table and columns of a reference can only be edited if the entry does not have any default values.

Detailed information about this topic

- Adding new components on page 95

Editing components for modifying

Components of a column-dependent reference can be edited for modifying.

To edit a component for modifying

1. Open the **Column-dependent references** tab and, on the **Edit values by column** tab, select the desired entry.
2. Check the box for the selected entry.

   If there is a custom value for this entry, the value is shown in the **Component (Customized)** column.

   When you enable the entry a custom value is added, just like when configuring the web project, which corresponds to the default value.

   The control box cannot be disabled if the foreign key is customized.
3. Edit the selected component by selecting another one from **Component**.

   The selected component is the one to be used in the default web application.

   NOTE: Instead of selecting a component, you can delete the configured entry using ✖.
4. Specify a filter by clicking in the **Filter** field on 🔧 and then creating a filter.

   You can limit foreign key values with the help of the filter.

Editing display components

Components of a column-dependent reference can be edited for displaying.

To edit a component for viewing

1. **Column-dependent references** and select the desired entry on **Display values by column**.
2. Check the box for the selected entry.

   If there is a custom value for this entry, the value is shown in the **Component (Customized)** column.

   When you enable the entry a custom value is added, just like when configuring the web project, which corresponds to the default value.

   The control box cannot be disabled if the foreign key is customized.
3. Edit the selected component by selecting another one from **Component**.

   The selected component is the one to be used in the default web application.

   NOTE: Instead of selecting a component, you can delete the configured entry using ✖.

Editing foreign key filters

Components of a column-dependent reference can be edited for a foreign key filter.

To edit a filter for foreign keys

1. Open the **Column-dependent references** tab and select the required entry on the **Foreign key filters** tab.
2. Check the box for the selected entry.

   If there is a custom value for this entry, the value is shown in the **Component (Customized)** column.

   When you enable the entry a custom value is added, just like when configuring the web project, which corresponds to the default value.

   The control box cannot be disabled if the foreign key is customized.
3. Specify a filter by clicking in the **Filter** field on and then creating a filter.

   You can limit foreign key values with the help of the filter.

Editing foreign key values

Foreign keys can be edited in components of a column-dependent reference.

To edit foreign key values

1. **Column-dependent references** and select the desired entry **Edit foreign key values**.
2. Check the box for the selected entry.

   If there is a custom value for this entry, the value is shown in the **Component (Customized)** column.

   When you enable the entry a custom value is added, just like when configuring the web project, which corresponds to the default value.

   The control box cannot be disabled if the foreign key is customized.
3. Edit the selected component by selecting another one from **Component**.

   The selected component is the one to be used in the default web application.

   NOTE: Instead of selecting a component, you can delete the configured entry using ✖.

To edit multi-value foreign key values

- Proceed as in the step-by-step for 'To edit foreign key values'.

  NOTE: Use **Edit multi-valued foreign key values** to offer components in the same context in the Web Portal any number of times.

Using a column-dependent reference

NOTE: A cursor must be on the collection to be able to add a column-dependent reference.

To add a column-dependent reference

1. Select a control node (a container, for example) with the right mouse button.
2. Select **Column-dependent references**.
3. Assign the added node to the collection and the edited property.

Object-dependent references

Components are referenced on the basis of two factors in the case of object-dependent references:

- Reference type specified on the reference node
- Reference object identified by its object key

NOTE: Use **Only apply object-dependent references that are explicitly assigned to this project** to ensure that only object-dependent references explicitly assigned to this web project apply. If this option is not set, both the assigned object-dependent

references as well as object-dependent references without assignments are taken into account. This option is set by default.

You can find **Only apply object-dependent references that are explicitly assigned to this project** in the **Node editor**.

Defining reference types and references

Object-dependent references are divided into reference type, which define each area of application.

These reference types are defined in the table `DialogAEDSActiontype`. The following reference types are defined by default.

Table 47: Reference type overview

Reference Type	Use Case	Reference Table	Interface	Default component
Attestation_ EditCases	Displays a list of attest-ations pending approval.	AttestationObject	VI_Interfaces_ ObjectSwitch_ Attetation_ EditCases	VI_Attestation_ EditCases_Default
AttestationCase_ Detail	Detailed inform-ation about an attest-ation case.	AttestationPolicy	VI_Interfaces_ ObjectSwitch_ AttestationCase_ Detail	
AttestationCase_ DetailInit	Preset detailed view for an attest-ation case.	AttestationPolicy	VI_Interfaces_ ObjectSwitch_ AttestationCase_ DetailInit	
Clone_ ShoppingCartIte-m	Duplicate an item in the shopping cart	AccProduct	VI_Interfaces_ ObjectSwitch_ Clone_ ShoppingCartIte-m	VI_Clone_ ShoppingCartIte-m_Default
Detail_Dialo-gDashboardDef		Dialo-gDashboardDef	VI_Interfaces_ Objectswitch_ Detail_Dialo-	

Reference Type	Use Case	Reference Table	Interface	Default component
			gDashboardDef	
Details_ AccProduct	Detailed inform-ation about a requestab-le product.	AccProduct	VI_Interfaces_ ObjectSwitch_ Details_ AccProduct	VI_Details_ AccProduct_ Default
Details_ PersonWantsOrg	Detailed inform-ation about a request.	AccProduct	VI_Interfaces_ ObjectSwitch_ Details_ PersonWantsOrg	VI_Details_ PersonWantsOrg_ Default
Details_ ShoppingCartIte-m	Detailed inform-ation about an item in the shopping cart.	AccProduct	VI_Interfaces_ ObjectSwitch_ Details_ ShoppingCartIte-m	VI_Details_ ShoppingCartIte-m_Default
Edit_PersonWant-sOrg		AccProduct	VI_Interfaces_ ObjectSwitch_ Edit_PersonWant-sOrg	VI_ITShop_PWO_ Detail
Edit_ ShoppingCartIte-m		AccProduct	VI_Interfaces_ ObjectSwitch_ Edit_ ShoppingCartIte-m	VI_ITShop_ ShoppingCart_ DetailPane
GroupOwnerEdit		DialogTable	VI_Interfaces_ ObjectSwitch_ GroupOwnerEdit	VI_UNS_ UNSGroup_ EditOwners
HyperView		DialogTable	VI_Interfaces_ ObjectSwitch_ HyperView	VI_HyperView_ Default
Insert_ ShoppingCartIte-m	Adding an item to the shopping	AccProduct	VI_Interfaces_ ObjectSwitch_ Insert_ ShoppingCartIte-	VI_Object_ Overview

Reference Type	Use Case	Reference Table	Interface	Default component
	cart.		m	
Object_Overview	Overview of an object.	DialogTable	VI_Interfaces_ ObjectSwitch_ Object_OverView	VI_Object_ Overview_Default
ObjectCollcetion		DialogTable	VI_Interfaces_ ObjectSwitch_ ObjectCollection	VI_ObjectCol-lection_Default
ObjectEdit		DialogTable	VI_Interfaces_ ObjectSwitch_ ObjectEdit	VI_ObjectEdit_ Default
ObjectSelection		DialogTable	VI_Interfaces_ ObjectSwitch_ ObjectSelection	VI_ObjectSelec-tion_Default
ObjectSheet		DialogTable	VI_Interfaces_ ObjectSwitch_ ObjectSheet	VI_ObjectSheet_ Default
ObjectView		DialogTable	VI_Interfaces_ ObjectSwitch_ ObjectView	VI_ObjectView_ Default
Parameters_ ShoppingCartIte-m	Display compon-ents of an additional request parame-te-r.	AccProduct	VI_Interfaces_ ObjectSwitch_ Parameters_ ShoppingCartIte-m	
Select_Dialo-gDashboardDef	Selecting a data point in a statistics overview.	Dialo-gDashboardDef	VI_Interfaces_ ObjectSwitch_ Select_Dialo-gDashboardDef	VI_Select_Dialo-gDashboardDef_ Default
Verify_ ShoppingCartIte-m	Checking an item in the shopping cart.	AccProduct	VI_Interfaces_ ObjectSwitch_ Verify_ ShoppingCartIte-m	VI_Verify_ ShoppingCartIte-m_Default
VerifyDecision_ PersonWantsOrg	Verifying an approval.	AccProduct	VI_Interfaces_ ObjectSwitch_ VerifyDecision_ PersonWantsOrg	

The references are defined in the table `DialogAEDSAction`. Such a reference is defined by:

- A reference type
- An assigned component
- A list of assigned objects from the reference type's reference table

The list of assigned objects is stored in the table `DialogAEDSActionHasObject`.

Defining a new object-dependent reference

In this section, you will find out using an example, how a component, displaying detailed information about a request, can be defined for a specific service item or entire service category request. This component is required if detailed information about requesting this service item should be displayed without taking into account, which page of the web application is currently open.

To assign an object-dependent reference of reference type `Details_PersonWantsOrg` to a service item

1. the **Object-dependent references** tab by selecting **Edit | Configure project | Object-dependent references** in the menu.

   The tab is divided into two parts. References are shown under reference types and grouped in a hierarchical structure on the left-hand side.
2. **Details_PersonWantsOrg** and select the reference you want from the grouped entries, for example, "Group request".
3. Edit the input data on the right-hand side of the **Object-dependent references** tab.
   a. Select the custom components for `Details_PersonWantsOrg_Default` that apply to the service categories in your web application.
   b. select the service items under the objects by double-clicking .

      If the object has been successfully selected, is displayed on the object. This reference applies to the selected object.
4. Save the changes.

   Recompile the Web Designer so that the setting take effect in the web application.

To add a new object-dependent reference

1. Select the **Edit | Configure project | Object-dependent references** menu item to open the **Object-dependent references** tab.
2. Expand the `Details_PersonWantsOrg` reference type and add a new reference.
   a. Mark the reference type and right-click to open the context menu.
   b. Click the **Add object-dependent reference** item.

      A new reference with empty property fields is created.

3. Enter the data and configure the following settings.
    a. Give the new reference a name.
    b. Enter a description about how to use the reference (optional).
    c. Select a component that the reference applies to.
    d. Select the object by double-clicking . An object can be a service item from a service category, for example.

       Object can be grouped in a similar way to references and displayed in a hierarchy. This means, the service item is under a service category.

       If the object has been successfully selected, is displayed on the object. that this reference applies to the selected object.

   NOTE: If you want to take object-dependent references into account that are explicitly assign to your web project, set **Only apply object-dependent references that are explicitly assigned to this project** in **Note editor**.
4. Save the changes.

   Recompile the Web Designer so that the setting take effect in the web application.

Detailed information about this topic

- Adding new components on page 95

6

Customizing the Web Portal

The Web Portal has many options for customizing the application according to what it is used for and your requirements. One option for configuration is to change predefined settings using the Web Designer configurator.

The second possible configuration provides customization of the object definition for requirements without predefined configuration settings.

Detailed information about this topic

- Creating new projects on page 90
- Creating new modules on page 92
- Adding new components on page 95
- Customizing object definitions on page 96
- Creating new Hyper Views on page 101
- Adding new nodes on page 103
- Create data display on page 103
- Displaying single objects on page 104
- Creating a grid display for collection data on page 105
- Mobile view - grid display and list view on page 106
- Generating mapping definitions on page 108
- Embedding reports on page 109
- Linking to a page on page 110

Creating new projects

If properties that are defined in the web project (languages, display settings, menu structure) are going to be changed, you must make a copy of the default web project.

NOTE: This task is different to creating an object copy. For more information, see Creating object copies with the wizard on page 97.

To create a new project

1. In the **Edit** menu, select the **Create new project** item.
2. Enter the name of the new web project in the **Identifier** field.
3. In the **Template** field, select a template.

   NOTE: Other projects are available to you as templates, not just the default project. You will see different menu items and workflow for use in your new project, depending on which web project you took as a template.
4. In the option box, select the menu items or modules from the web project you selected as the template and want to use in your copy.

   The modules are still referenced from the customized web project.

   NOTE: Check the modules options in the tree selection. To add the modules to a new project, you must set the option next to the module. Simply disable or do not enable the modules you do not want to have in the web project. Use **Select all/deselect all** to select or deselect all modules at once.
5. Click **Next**.

   Continue to add the new module. Next, the **Create new project - New project** view shows all changes made.
6. Click **Finished**.

   Details of the new web project are displayed in the definition tree view.
7. Click **Save**.

   The web project is added as a new file.

To view the new web project

1. In the main view, select the **Start page** tab.
2. Click .

   This opens **Edit web application settings**.
3. In the **Web project** field, select the new web project.
4. Click **OK**.
5. On the start page, click **Refresh preview**.

   The new web project is loaded in the preview.
6. On the start page, click **Debug** or **Release**.

   This publishes the web project.

To copy configuration settings from another project

1. In the definition tree view, select the node containing the web project.
2. In the **Node editor** view, enable the **Inherit configuration settings from** option.
3. From the drop-down menu, select the web project whose configuration settings you want to copy.

To add a new menu item to web project

1. In the definition tree view of the new web project, select the **MenuStructure** node.
2. In the context menu, select **Menu item**.

   This adds a new subnode under **Menu structure**.

To hide a menu without submenu.

1. Select the new subnode.
2. In the **Node editor** view, select the **Hide if no subnodes exist** option.

   The menu is not displayed in the web project if it has no submenu.

Creating new modules

Use this wizard to execute all the necessary steps to add a new module and reference it in a web project.

Enter the identifier for the new module in the first step. You can also enter a title for the new page. If you have added the identifier as well as the title, a new `Label`-type node is automatically added under the `Form` node type in the definition tree view.

A corresponding caption object is not added. A warning is output the next time the project is compiled and the caption can then be added including any translations required.

To create a new module

1. In the **Edit** menu, select the **Creating new modules** menu item.

   This display **Wizard for creating a new module**.
2. On the **General module settings** page, enter the following information and click **Next**.
   a. A name for the new module.
   b. A title for the new page.

   The general data for a module page closes and the **Navigation** page is displayed.

   Here, you can set whether a menu item is created for the new module or not. If this option is set, you will see more settings for defining the position of the menu item in the menu bar and its caption.
3. On the **Navigation** page, configure the following settings and click **Next**.

Table 48: Navigation settings for the new module

Setting	Description
Add a menu item	Adds a new menu item.

Setting	Description
Name	Displays the title on a new page. A title is only displayed if you entered one on the first page of the wizard. You can override this title and add a caption object.
Above the selected item	Options for selecting those which specify the position of the menu item.
Below the selected item A hierarchy level under the selected item	With the option "A hierarchy level under the selected item" you can first select the content view in the Web Portal and then open the sub menu items by clicking on and continue selecting.

4. On the **Security** page, define the group of employees who can use this menu item.

   NOTE: You can multi-select employees in the editor tree structure with the option **Only for employees in specific roles**. The new menu item is available in the web application to all employees that are assigned to at least one of these selected structures.

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

5. On the **Security** page, select one of the following options and click **Next**.
   - Visible to everyone

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

     Select this option to add a new configuration parameter. A predefined value is suggested for the key. These options are also available.

     a) Only for employees in specific roles

     b) Employees matching a given SQL query

   The visibility settings are closed. The new module is added.

6. Close the dialog using **Finish** and save all the changes made to your web project by clicking in the toolbar.

   The new module is shown as a tab in the definition tree view.

Use case

You want to add a new menu item, which can only be seen by users who belong to a particular department. In this example, only the menu item of the department "Sales" should be visible. This example is described in detail in the following step-by-step instructions.

To add a new menu item, which can only be seen by certain users

1. Work through the steps in **To add a new module** until you get to step 5.
2. On the **Security** page, select the **Configuration parameters** and **Employees matching a given filter condition** options.
3. Perform one of the following tasks:
   a. In the **Value** field, enter the following text.

```
(UID_Person='%useruid%')
AND (EXISTS
(
     SELECT 1 FROM
     (SELECT UID_Department FROM Department WHERE DepartmentName =
     N'Sales') as X
     Where X.UID_Department = Person.UID_Department
))
```

   b. Open the WHERE clause wizard by clicking on and use the link **Add expression** to set the following properties.

Table 49: WHERE clause wizard filter conditions

Setting	Description
References to other objects	Object selection in a hierarchical structure. Expand the hierarchy by clicking on and select the **Primary department** item.
At least one record set exists / Add expression	Link under the new condition for select an additional condition. Click the link and select **Department** item from **Value comparison** hierarchy.
The following applies to the value in the Department column: The value contains equals ""	Use the link **""** to open a field and enter "Sales".

4. Create the new module and click **Next**.

   This generates the new module and displays it as a new tab in the definition tree view.

Adding new components

You can add new components in the following ways. If you want to create a new component for an object dependent reference, you use a wizard to add the object dependent reference as well as the new component conveniently. The wizard can also help you set up column-dependent references. Both variations are described in the steps below.

To add new components using the navigation view

1. Select **Components** in the navigation view.
2. In the navigation view toolbar, click .
3. Select .

   The new component is shown in a new tab in the definition tree view.

   NOTE: New components are automatically prefixed. You can change the name later. Input fields marked with an asterisk '*', are mandatory fields.
4. Select the component root node in the definition tree view.
5. Select the **Node editor** view for the root node and edit the predefined name.
6. In the toolbar, click or .

   This renames the component.
7. In the component's definition tree view, select the **Definition** node.

   The selection **Type** is shown in the **Node Editor**. Use the selection to define which type of nodes the new component will have.
8. In the **Type** field, select a type.

   This displays the selected node type under **Definition** and you can continue editing.
9. In the definition tree view, select the new node to modify other settings.

   You can specify conditions, rules, and extensions.

To create a new component through object-dependent references.

1. On the start page, click the **Edit | Add new | Create component for object-dependent reference** menu item.
2. In the **Create component for object-dependent reference** dialog, click **Next**.
3. On the **Set display for identifier** page, enter a name in the **Reference name** field and click **Next**.
4. On the **Select a reference type** page, select a reference type, and click **Next**.

   NOTE: Depending on which reference type is marked, before you confirm with the **Create component for object-dependent reference** button, this reference type is preset.
5. On the **Select objects** page, select one or more objects to show and click **Next**.

   NOTE: The nodes are grouped on the basis of the reference type's grouping column. You can select products and product groups from the `AccProduct` table. If

you have selected the product group, its products do not have to be selected explicitly.

6. On the **Enter the component name** page, enter a name for the component and click **Next**.

   The object-dependent reference and the component are generated.

7. Click **Finish**.

   The reference and component are added and shown on a new tab.

To create a new component through column-dependent references

1. On the start page, click the **Edit | Add new | Create component for column-dependent reference** menu item.
2. In the **Create component for column-dependent reference** dialog, click **Next**.
3. On the **Enter component name** page, enter a name in the **Component name** field and click **Next**.
4. On the **Select an interface** page, select an interface and click **Next**.
5. On the **Select tables and columns** page, select a table and column then click **Next**.

   Your settings are processed and the fitting elements are generated.

6. Click **Finish**.

   The reference and component are added and shown on a new tab.

Customizing object definitions

Copy existing objects with Web Designer and customize them. Not only is there a wizard to help you here, but you can also copy the desired object directly within the definition tree view.

Extensions can be used as another way of customizing objects. For more information, see Extensions on page 98.

To create a copy of an object

1. In the definition tree view, select an object.
2. Click .

   This opens the **Copy object** dialog.

3. the field in the **Copy name*** column on the **Object copy properties** page and overwrite the preset name.
4. Click the field in the **Description** column and enter an optional description for the object.
5. Enable the **Add substitution rule for current object** option and click **Next**.

Your entries and settings are processed and displayed on the **Processing** page.

6. Click **Finish**.

   The object copy is displayed in the definition tree view.

   NOTE: If you view an object in the definition tree view for which a substitution rule was defined, a header with a corresponding comment is shown above the object.

   NOTE: After an object has been assigned a substitution rule, a button is added in the **Node editor** view below to provide further information. Click to display the object being substituted in the definition tree view.

   NOTE: If you copied a default object, future changes to the default object are not transferred to the object copy. This will be the case after a migration, for example.

Creating object copies with the wizard

You must make a copy of a default object if you want to add new functionality to it that cannot be added in the configurator. The **Copy objects** wizard helps you with this.

To add a copy of an object with the wizard

1. In the toolbar, select the **Edit | Copy objects** menu item.
   Copy objects... opens after compiling. All existing database objects are ordered by different object types into a hierarchical structure.
2. Enable the option next to the database object you want to copy.

   NOTE: The database objects cannot be multi-selected.

   NOTE: The wizard automatically adds substitution rules for the objects used for copying if **Add substitution rule for current object** is set. The copies are automatically referenced instead of the default objects. The Web Designer does not generate substitution rules if custom database objects are used as copy templates. You can use the configurator to enter these substitution rules manually. For more information, see Project configuration - customization on page 76.
3. Enter a name for the copy in the **Copy name*** column, if required.

   A custom prefix stored in the database is automatically prefixed to the entered or suggested name.
4. Click **Next**.

   A copy of the selection is created and a conformation prompt is displayed. After you have closed the wizard, new tabs are created in the definition tree view displaying the copies.
5. Save the copy manually from the toolbar.

Extensions

In some cases, it is sufficient to add or delete a node to change single properties within an object. In such cases, extensions are best suited to configuring the web application. This method of configuration allow you to describe changes to an object.

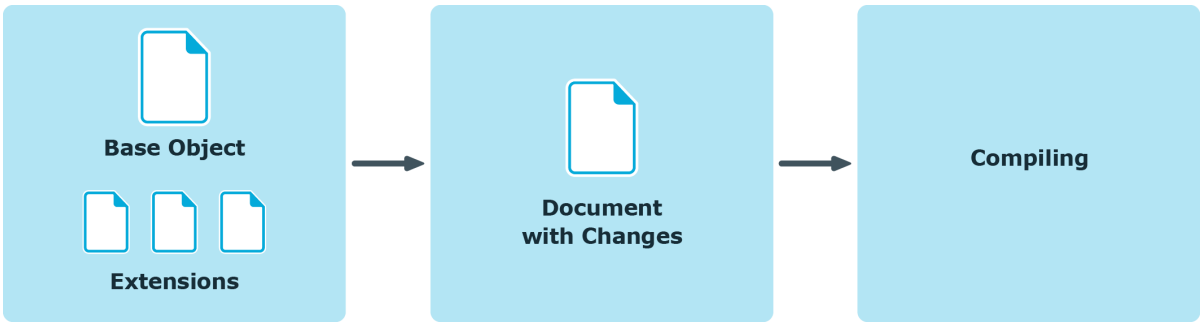
How extensions work

You can add any number of extensions to the Web Designer's base object module, component, and web project. Some default extensions are already supplied when the database modules are installed. You can add custom extensions yourself.

Extensions are edited when you configure the base objects. That means you add an extension to the object and change the property value. You then compile the base object and the modification resulting from the extension is highlighted in the Web Designer.

You can identity base objects, which you want to extend, by their colored nodes in **Objects definitions**. You edit extensions in **Configuration (custom)**.

Figure 1: Workflow in the Web Designer



To overwrite a single property in an extension

NOTE: An edited extension node is labeled with in the Object definition view in the definition tree view.

1. In the **Object definition** view, mark an extension node.
2. In the Node editor view's toolbar, click Show extension values to mark the edited setting of the extension node in color.
3. Click at the marked place.

   This switches you into the **Configuration (custom)** view and marks the node with the modified property value.
4. To return to the **Object definitions** view, click on the changed setting in the node editor window.

   NOTE: You can switch between the **Configuration (custom)** and **Object**

definition views with the button, if the extension is found under the **Add node** or **Remove node** node.

Extension rules

Each extension is its own object in the Web Designer. The advantage of extensions is that database objects that have been extended are updated by migration without the database objects being overwritten. This does not apply in the case of object copies. In that case, the copy is not updated on migration.

The following extension rules are listed in the table.

Table 50: Overview of extension rules

Extension rule	Description
Inserting a node	Adds a new node to the read-only base object.
Moving a node	Removes a node from the base document and places it in an extension.
Editing a property	Modifies a node property in the read-only base object.
Deleting a node	Deletes a node from the read-only base object. A deletion is carried out, for example, when you want to delete a bookmark on the Web Portal start page.

NOTE: You recognize an extension node by its colored bar.

You add a new node in the definition tree view using the context menu on your selected node. The entries in the context menu vary and depend on which type of node you are adding the extension node to. You can, for example, add new form or component nodes.

Because there are numerous types of nodes that can be added, the example is based on a node of type title.

To add a new node

1. In the navigation view, double-click the required base object.

   The selected base object is displayed in **Object definition (read-only)**.
2. Mark the node 'Title', for example, and select **Object in extension | New node**.

   NOTE: You can also select another item as **New node** if you know the node type to base the extension on.
3. In the **New nodes** dialog, select the **button** node type from the list and click **OK**.

The new node is added with a custom prefix and a bar under the selected node. is displayed at the bottom of the definition tree view. Use to go to the definition object.

4. Edit the new node in **Node edit**. You may configure the following settings.

NOTE: Fields marked with * are compulsory.

- Enter a name for the button in the **Text** field.

  NOTE: The text must be enclosed in quotes.

  The button is displayed in the definition tree view with the name you entered.

To modify the position of an extension node.

1. Mark the nodes you added in the step-by-step "To add a new node" in **Object definition**.
2. Switch to the definition object by click in the definition tree view.

   The edited new node is displayed in **Configuration (custom)**.

   In the custom configuration view, you can, for example, edit the position of the button. This is how you do this.
3. In the **Configuration (custom)** view, mark the **Add node** node above the node you added.
4. Enter the value "0" in **Sort order**.

   The value is displayed after the **Add node** node (sort order: 0). You will see the change after compiling the preview. When you swap back to the **Object definition (read-only)** view, you will see that your added node is placed above the extension you marked.

You can move nodes, which are on a parent node in an extension. This assumes that the node to move is on a custom object which is not write protected. An example of this would be objects found in a container node. Other examples of moveable extension nodes are columns in a collection or parameters of a configuration parameter section.

NOTE: The menu item **Move to extension** is only available to custom objects.

To move a node

1. In the navigation view, double-click the required base object.

   The selected base object is displayed in the **Object definition** view in the definition tree view.
2. Enable the view in the definition tree view using .
3. Mark a node and select **Move to extension** from the context menu.

   The node is marked in color as in the extension mode.

A change can take the form of extending viewing conditions for a container, for example, as explained below. Other changes are not gone into here. For more information, see How extensions work on page 98.

To edit a property

1. In the **Node editor** view, click next to the **Viewing condition** field.

   The view switches to **Configuration (custom)** and the **Modify property value** extension is selected. You can edit the extension.

2. Enter the value "1=1" in **Condition** in the Node editor view.

   The edited extension is displayed with the modification in the definition tree view and in the preview. The edited node is marked with in the definition tree view.

Before you can delete the node, you must create a deletion rule for the mode in question. You can only add a deletion rule to a node that does not contain an extension.

A use case would be if you want to remove a container node from a write protected base object.

To create a deletion rule

- In the **Object definition** view, mark the node to delete and, in the context menu, select the **Delete node as an extension** menu item.

  The node changes its colored mark and you can see the message, **Remove node** in the **Node editor**. You can also see this message on the extension when you switch to **Configuration (custom)**.

To remove a deletion rule

1. In the **Configuration (custom)** view, mark the **Remove node** node if you want to remove the extension again, and, in the context menu, select **Delete**.

   NOTE: The menu item **Delete** is only available to custom objects.

2. Delete the extension from the definition tree view by selecting **Yes**.

Creating new Hyper Views

You can use this wizard to create new Hyper Views or Hyper View shapes or to import existing ones.

Table 51: Options available in the "Hyper View wizard" dialog

Option	Description
Import a complete Hyper View	Imports a complete Hyper View definition from the database dialog structure to be used in the Web Portal.
Import one or more Hyper View shapes	Adds one or more shapes, imported from the database dialog structure, to a Hyper View.
Define a new Hyper View shape	Adds a new shape, which can represent data from any table, to a Hyper View.

To import an complete Hyper View

1. Click in the toolbar.
2. In the **Hyper View wizard** dialog, select the **Import a complete Hyper View** option.
3. On the **Select central element** page, select an element from the list and click **Next**.
4. On the **Select dependent elements** page, disable the dependent elements you do not wish to display and click **Next**.
5. On the **Details and layout** page, enter the component identifier in the field and click **Next**.

   The **Wizard closed** page is shown and the component is created.
6. Click **Finish**.

To import one or more Hyper View shapes

1. Click in the toolbar.
2. In the **Hyper View wizard** dialog, select the **Import one or more Hyper View shapes** option.
3. On the **Extend an existing Hyper View** page, select a **Hyper View** from the list and click Next.
4. Disable the dependent elements you do not wish to display on the **Select dependent elements** page and click **Next**.

   The **Wizard closed** page is shown and the component is created.
5. Click **Finish**.

To define a new Hyper View shape

1. Click in the toolbar.
2. In the **Hyper View wizard** dialog, select the **Define a new Hyper View shape** option.
3. On the **Extend an existing Hyper View** page, select a **Hyper View** from the list and click Next.
4. On the **New HyperView shape** page, select a database table in the **Base table selection** field.
5. In the **WHERE-clause** SQL field, enter a definition and in the field, enter a name for the new shape.

   The **Wizard closed** page is shown and the component is created.
6. Click **Finish**.

Adding new nodes

To add a new node

1. Select a database object in the navigation view.

   The selected database object is displayed in its own tab in the definition tree view.

   NOTE: The procedure for adding new nodes differs somewhat if the database objects are standard objects or custom objects. In the case of standard objects, you must first move the node to the extension using the context menu before you add a new node. You have additional write permissions for custom objects and you can add a node directly under the node in the definition tree view.

2. Mark a node of `Container` type, for example, in the definition tree view and select the New node entry from the context menu.

   NOTE: You will see different submenu structures in the context menu depending on which database object or which node types you have selected.

3. In the **New Node** dialog, select the node you want from the list or search for it in the search field.
4. Close the dialog with **OK**.

   The new node is displayed in the definition tree view.

Create data display

The core task of a web application created with the Web Designer is targeted visualization and manipulation of database objects.

NOTE: The wizard is only available for node types to which corresponding displays can be added.

You can create the following data display types with the wizard.

- This results in a list of single database objects and allows you to change (overwrite) selected properties.

  This results in a list of single database objects and allows you to change (overwrite) selected properties.

- Grid display for collection data

  The resulting table shows all properties of database objects. Relations between single database objects can be displayed in hierarchical form.

Detailed information about this topic

- Displaying single objects on page 104
- Creating a grid display for collection data on page 105

Displaying single objects

Use this wizard to include an edit mask into your web project. You can edit several properties of individual data set as a bundle. Use filters to limit how many objects are displayed. Data objects are not mapped in tables but in sections. Each individual data object from the selected or new collection is displayed in such a section (iteration). Therefore, each individual data object produces a data set and is displayed in a list.

To view the object details

NOTE: You must have already selected a database object. If you have selected an object that belongs to the default version, you will find the wizard in the context menu under **Element in extension**.

1. For example, mark a node of type `Container`, and, in the context menu, select the **Wizards | Create data display** item.
2. in the **Display wizard** dialog, mark the **Create detailed display and fields for single objects** and click **OK**.

   This opens **Source selection**. You can apply the following settings.

Table 52: Settings for selecting source data

Option	Setting	Description
Use existing collection		Selection box with a list of available collections.
	Filter on	Input field for filter conditions.
Create a new collection		Adds a new collection.
	Use display names	Enables/Disables displaying of database names.
	Base table	Database table selection.
	Collection name	Input field for the collection name. The data object to be displayed are saved in this collection.
	WHERE clause	SQL input box for setting filter conditions. Use to open the filter condition setting wizard and narrow down a search.

NOTE: 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.

3. On **Source selection** page, adjust the settings as required and click **Next**.

   NOTE: If **Read-only** is disabled, the property in the web application can only be edited with special permissions. You need to add another button once the wizard is complete, so that the changes can be saved in the database. The displayed columns are written as value list directly under **PropertyList()**.

4. On **Select details** page, select the database columns for which you want to display detailed information.
5. Click **Finish**.

   NOTE: The list of editors is displayed on the web page under each other. To view individual data sets, you must change the container style in the definition tree view, which encloses **VI_Common_PropertyEditor**.

Creating a 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. This means that data objects which are defined as dependent (foreign key relation) on another database object stored in the same or another collection can be displayed in another layer.

To create a grid display for collection data

1. For example, mark a node of type `Container`, and, in the context menu, select the **Wizards | Create data display** item.
2. In the **Display wizard** dialog, select the **Create a grid display for collection data** option and click **OK**.

   This displays **Collection**. You can apply the following settings.

Table 53: Collection settings

Option	Setting	Description
Use foreign key relation		Selection box with a list of available foreign key relations.
Use existing collection		Selection box with a list of available collections.
Create a new collection		Adds a new collection.
	Base table	Database table selection.

Option	Setting	Description
	Collection name	Input field for a collection name.
	WHERE clause	SQL input box for setting filter conditions. Use to open the wizard for setting filter conditions.

3. On the **Collection** page, configure the settings and click **Next**.
4. On the **Columns** page, select the database columns for which you want to display detailed information and click **Next**.
5. Configure the following details on the **Layout** page and enter a name for the layer.

Table 54: Layout settings

Setting	Description
Layer name (Identifier)	Gives the layer a name. The layer caption is only used help the search for properties that you may want to change later.
Display title	Enables/Disables the title display.
Enable the user to sort columns	Allows the user to sort data objects in their own way. Grid contents can be resorted by clicking the column title bar.
Sort collection	Sorts the collection according to a set criterion. The criterion is set using both selection boxes.

6. Click **Finish**.

   This data completes the definition of the first grid layer. Another window opens. This window displays the grid layers that are already defined.
7. Perform one of the following tasks:
   a. Add another grid using .
   b. Mark the existing grid and edit it using .
   c. Mark the existing grid and delete it using .
8. Confirm the **Create a new grid - layers** pane with **Ok**.

Mobile view - grid display and list view

The Web Portal is has been designed not only for use with desktops but also mobile devices, so you can modify the mobile view settings. You have different setting options in the Web Designer.

Detailed information about this topic

- Customizing the list view on page 107
- Disable list views on page 108

Customizing the list view

The Web Designer offers different setting options for adjusting the usual desktop grid view in the Web Portal to suit a mobile mode. The **List view** node is available at every grid level. This means, you can make changes in the Web Designer anywhere where these nodes are available.

The following table describes the setting options for the list view.

Table 55: Display options for the list view

Display Option	Description
Disable list view	The grid view is used as apposed to the list view in mobile mode in the Web Portal. This setting is not recommended because it makes operating and displaying difficult due to lack of space on mobile displays.
Automatically generate the list view	This setting automatically generates the list view for mobile mode. This is sufficient for simple tables, meaning, a list view without other options and buttons.
Define a list view	This setting allows you to customize the list view. The list view works independently of the table view for this setting, meaning, if changes have been made to the table view, you must also make them in the list view, otherwise, the changes are not visible in the list view.

NOTE: The groups and filter function is not available in mobile mode. You can 'leaf' backward and forward within the list view.

Customizing the list view is described using an example based on the component `VI_ITShop_ApprovalItem`, which is used in the **Pending requests** menu in the Web Portal.

To customize the list view

1. Open the component `VI_ITShop_ApprovalItem` in the definition tree view.
2. Create an object copy and ensure that **Add substitution rule for current object** is not set.
3. Open **Grid band - PersonWantsOrgGrid** under **Definition**in your object copy.
4. Mark **List view** and select the entry you want from the **List view mode** menu in the **Node Editor** view. The following items are available:
    a. Disable list view
    b. Automatically generate the list view

c. Define a list view

Disable list views

You can disable list views in mobile mode in the Web Portal with the help of the configuration parameter `VI_Common_GridListViewDisabled`.

To disable list views

1. Open **Edit | Configure project | Web project** and search for **VI_Common_GridListViewDisabled**.

   The configuration is displayed with the default setting 'false'.

2. In the **Value (custom)**, click ✚.

   This sets the configuration parameter to "true". Enable "Disable all grid list views".

Generating mapping definitions

Use the node type from the category **Component references** 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`.

NOTE: If all the mapping nodes expected by the component are not defined it results in compiling errors.

The **Mapping generator** wizard reads the expected mapping nodes for the component to be referenced and creates a component reference. As a result, the user does not need to check the mapping nodes. The mapping nodes added by the wizard may only be partly configured. You therefore need to rework these automatically generated mapping nodes.

For example, if you switch components, the mapping is automatically modified. In a dialog, you are prompted as to whether you want to modify the mapping or not.

Use the wizard when the mapping nodes required for the called up component have been modified so much that manually adjusting the structure is too time-consuming. The wizard generates the new node.

NOTE: When you use the **Mapping generator** wizard, all existing nodes are overwritten. Values from defined nodes are deleted.

You can start the **Mapping generator** wizard from the context menu but it is only available for **Component references** nodes and `Call action` nodes.

To create a mapping

1. Mark a `Container(reference)` type node, for example, and select the **Wizards | Mapping generator** menu item.

The wizard runs automatically and generates a mapping node below the component without displaying a dialog.

2. Mark this related node in the definition tree view.

   In the lower area of the definition tree view, the **Mapping node** is displayed in the `Node editor` view next to the node label. You can add more information about the referenced component in the next step.

3. In the **Virtual node*** and **Type*** menus, select the relevant settings.

Detailed information about this topic

- Assigning collections to components on page 128

Embedding reports

You can create a report to be saved in the database with the **Embed report** wizard. This wizard can be found in the context menu of the action node. The web application determines values for the report parameter, if required. The calculated report is displayed in the web application.

To embed a report

1. **Wizards | Embed report**.

   The **Embed report** wizard is opened with a list of reports.

   NOTE: Reports are created and edited with the Report Editor tool.

2. Select a report from the list and click **Next**.
3. On the **Allocate Parameters** page, click the button next to the field you want to edit.

   NOTE: The data you enter depends on the report you selected. Normally, there is one primary key per parameter. Other, quite different types of parameter are also possible. If no parameters are required for the report, the wizard displays a corresponding message.

4. Define the parameter used in the SQL input box and click **Apply**.

   Your entry is displayed on the Allocate parameters page.

5. Repeat steps 3 and 4 until all the parameters have been assigned.
6. Click **Finish**.

   As a result, the wizard creates a **Compile report** node and a **Forwarding** node.

Forwarding is based on a default component. This takes over the task of displaying the compiled report.

NOTE: The Web SQL expressions used when the parameters were defined are saved under the **Compile report** node in the definition tree view and can be edited in the **Node editor** at any time.

Linking to a page

Links can be set for certain Web Portal modules. Use the following template for setting a link:

```
<BasisURL>/page.axd?ContextID=<Module name>
```

To set a hyperlink to a specified page in the Web Portal

1. Use the base URL of the Web Portal instead of <BaseURL>.

   NOTE: The base URL is saved in the system under the configuration parameter `QER\WebPortal\BaseURL`.

2. Use the name of the target module instead of `<module name>`.

   NOTE: If there are mandatory parameters, you must give values for these parameters.

Context parameters can be attached as parameters to the URL, for example:

```
<BasisURL>/page.axd?ContextID=<Module name>&<parameter name>=<parameter value>
```

NOTE: If a user is not yet authenticated when the link is called, the Web Portal opens the login page before the linked page is called.

Detailed information about this topic

- Declaring configuration keys in modules and components on page 129

Basics of Web Designer programming

This section is all about the basics of Web Designer programming. Collections are explained here in detail, among other things.

Detailed information about this topic

- Node types on page 111
- Defining with Web SQL on page 111
- Collections on page 116
- Collection events on page 126
- Assigning collections to components on page 128
- Declaring configuration keys in modules and components on page 129
- Declaring context parameters on page 130
- Executing Microsoft .NET Framework code on page 131
- Customizing documentation on page 135

Node types

You will find a detailed description of node types in the One Identity Manager Web Designer Object Model Documentation.

Defining with Web SQL

You can or must select or customize data on many of the Web Designer node types. This data often comes from the database. However, you will not have direct access to the database due to the Model-view-controller architecture of the Web Designer programming

model. Instead, you access it through the Web Designer's data layer. You can query this data layer with Web SQL, a SQL-related language.

Web SQL limits to access to `SELECT` statements. `INSERT`, `UPDATE` or `DELETE` are not supported. `JOINS` are not supported either in Web SQL.

Query example with Web SQL

You want to view the employees in your department in a grid.

To view data

1. Load the necessary data from the database table `Person` into a Web Designer collection.
2. Create this collection on the grid node.
3. Using the following Web SQL query, create the individual employee fields you want to display.

   ```
   FROM Colleages SELECT CURRENT LastName
   ```

Detailed information about this topic

- Web SQL functions on page 112
- Loading collections on page 113
- Querying data from a collection on page 114
- Filtering data from a collection on page 115

Web SQL functions

Web SQL provides a large number of functions. The most important functions are described briefly in the following. See the Web Designer help for a full description.

Table 56: Overview of the most important Web SQL functions

Function	Description
CanDelete, CanInsert	These functions allow collection delete and insert permissions to be queried.
CanEdit, CanSee	These functions allow edit and view permissions for individual columns of collection rows to be queried.
Count(*)	This function queries the number of rows in a collection.
DbCount, Exists	Queries the number and existence of rows in a database table. This can be limited with a WHERE clause.
Display, DisplayValue,	Queries the display values of a complete collection row or a

Function	Description
DisplayValueLong	specific column of a collection row.
Format	This function formats a string by entering a mask with wild cards and insert values.
GetConfigParm	Returns the configuration parameter value with the given path. NOTE: To use the configuration parameter function, the session must be authenticated. This function cannot be used until the after the login page if the Web Portal is being used against an application server.
IsNull, IsNullOrEmpty	Checking for undefined values or empty strings is allowed.
PrimaryKey	This function queries the primary key value for a row from a collection.
Translate	Translates a string.
SqlCompare, SqlAnd, SqlOr	This function allows conditions to be written in database SQL.
Try	This function allows alternative values, if an error occurs during evaluation of a Web SQL expression.

Loading collections

Web SQL is the SQL language in the Web Designer. This language is used when loading and populating collections with data from the database.

NOTE: A filter expression is used when loading collections. A filter expression is a Web SQL expression. This filter expression must be able to interpret the Web SQL expression for a string which is valid in the database SQL language.

When collections are loaded, the result must be a valid condition in the `WHERE clause` of a `SELECT` statement for the queried database table. The same Web SQL expression can be invalid for loading another table.

Example

An example of an invalid Web SQL expression could be using the `LastName` column in the `Department` table. The `LastName` column does not exist in the `Department` table.

To create a "Define database collection" collection type

1. Select the database table that the data is going to be loaded from.
2. Use a filter expression.

   ```
   sqlcompare ("LastName", "Schm*", "string", "LIKE")
   ```

   This expression specifies which data is to be loaded.

- OR -

```
format(("LastName like '%{0}%'", FROM Var SELECT searchString)
```

This expression bypasses the problem when loading an invalid Web SQL expression.

Generally, Web SQL functions are the prefixed by `Sql`. You can create an optional condition for the collection or `Load collection` node. For more information, see Web SQL functions on page 112.

To load data from a database table into a collection with the help of the "Load collection" action node

- Specify the collection into which to load the data.

  NOTE: To limit the data to load, you can enter an optional condition for a WHERE clause. You can specify this on the `Load collection` node. A WHERE clause on the action takes precedence over a WHERE clause on the collection. The WHERE clause on the collection is only used if there is no WHERE clause on the action node.

  TIP: In the query window, you can query the WHERE clause used for the last loading with the Web SQL function `WhereClause("CollectionName")`.

Querying data from a collection

You can query the collection at any point where you can enter a Web SQL expression. The collection must be visible to you.

The following collections are visible to you in modules and components.

- Self-defined collections
- Virtual collections
- Collections in session modules

To define a Web SQL query

```
FROM <Collection> SELECT [CURRENT] <Column>
```

- OR -

```
SELECT [CURRENT] <Column> FROM <Collection>
```

The query listed first is advantageous in terms of auto completion because both the collection name and the columns of the collection can be auto-completed.

IMPORTANT: If you query several columns, you must enclose the column names in curved brackets.

Example for a multi-column query

Query with Web SQL:

```
FROM Persons SELECT FirstName
```

```
FROM Persons SELECT (FirstName, LastName)
```

The Web SQL query relates to the collection. `Persons` is the name of the collection. `LastName` and `FirstName` are properties of a collection in Web SQL.

To show several rows (the names of all loaded employees for example) from a collection there are several Web Designer node types that are iterated through a collection. If you want to reference the current data set within one of these iterations, use the Web SQL keyword `CURRENT`.

Example for a multi-row query

```
FROM Persons SELECT CURRENT FirstName
```

There are also aggregate functions, which group rows of a collection together into a single value.

Example of querying a single value of a collection

This query returns the number of rows of the collection `Persons`

```
FROM Persons SELECT Count(*)
```

This query returns the minimum, maximum, and average number of days

```
FROM AttestationPolicies
SELECT (Min(SolutionDays), Max(SolutionDays), Avg(SolutionDays))
```

Filtering data from a collection

You can limit a query to the relevant rows by using a filter condition when querying a collection with Web SQL. Use Web SQL to enter the keyword `WHERE` and a condition in the `SELECT` statement. The condition is a comparison of one or more columns with a preset value.

Example of a simple filter

```
FROM AttestationPolicies SELECT Count(*) WHERE SolutionDays > 4
```

Example of a filter with nested Web SQL expressions

```
FROM AttestationPolicies SELECT Count(*)
WHERE SolutionDays >
(FROM AttestationPolicies SELECT Avg(SolutionDays))
```

Example of a filter to use within an iteration

You can also combine a filter with the keyword `CURRENT` if you are in an iteration.

```
FROM AttestationCase SELECT Display()
WHERE UID_AttestationPolicy >
(FROM AttestationPolicy SELECT CURRENT UID_AttestationPolicy)
```

Example of a filter with a condition

There are several places where you can only enter a filter condition with Web SQL.

If all employees are loaded in the collection "Persons" and you want to display men and women separately, proceed as follows:

- Define two grids that iterate on "Persons" and only differ in their filter conditions:
  a. Gender = 1
  b. Gender = 2

Collections

Collections play an important role when you are using the Web Designer. The data contained in a collection is required for working the components. Data in a collection is stored in rows and columns as in database tables.

A collection is referenced by name within a component or module. You can query collections with Web SQL. For more information, see Defining with Web SQL on page 111.

A component's data model defined with collections server as the basis for displays and further processing of data. The contents of collection can be loaded from the database but other data sources are also possible.

Detailed information about this topic

- "Database objects" as a collection on page 117
- Managing a database-based collection on page 117
- Loading database objects through relations on page 119
- Loading database objects from multiple tables on page 120
- Using the database query wizard on page 121
- Loading a historical object state on page 124
- Loading a change history on page 125
- View definitions on page 126
- Collections as data sources for controls on page 126

"Database objects" as a collection

In most use cases, the database is the best data source for collections. All table defined in One Identity Manager can be used as data source for collections. Not all database table entries must be loaded in this case. Normally, you need to limit which data sets are loaded from the database with a WHERE clause on performance grounds.

TIP: An object from a collection based on a database behaves no differently than other objects. All the database object's templates, formatting rules and access permissions are apply.

To load a simple database based collection

1. Open the module or component to which to add the collection.
2. Select the **Collections** node with a right click.
3. Select the **Database object** option.
4. Click the new node.
5. In the **Node editor** view, under **Object type**, select the database table.
6. Enter the name of the new collection under **Identifier**.

To load the contents of a database table

1. Select the component's **Initializer** node by right-clicking on it.
2. In the **Data actions** submenu, select the **Load collection** option.
3. Click on the node.
4. In the **Node editor** view, select the relevant collection under **Collection**.
5. Using the **Filter condition** property, enter an filter condition expression if you wish.

Managing a database-based collection

Working with a database based collection includes adding properties or object as well as removing and delete objects from a collection or database.

All important procedures for managing database-based collections are described below:

To add a proxy property to a database based collection

Proxy properties allow transparent use of database columns (base property) with modified metadata. Metadata includes:

- Name
- Description
- Specifying a property as mandatory or optional

- Structure for displaying the property value
- Component selection for displaying and editing property values

You can modify a property within a component to match the requirements of the component exactly using proxy properties. Ensure the database of a proxy property is always the base property value. Updating the value through proxy properties always have additional effects on the base property.

1. Right-click to select a collection.
2. Select the **Proxy property** option.

   A proxy property is added under the selected collection node.
3. In **Node editor**, select a property in the **Base property** menu.
4. In the **Identifier** field, enter the name for the proxy property.

To add more proxy properties to a database based collection

1. Right-click to select a collection.
2. Select the **Property** option.
3. Enter at least a name and data type for this property.

NOTE: Each object, which is in the collection at runtime, now has an additional property. You can handle this like a database based property, for example, to store temporary data.

To add a new object to a collection

1. Right-click to select an action node.

   This could be the **Initializer** node of the component, for example.
2. Select **Paste** and click the new node.
3. In the **Node editor** view, select the required collection under **Collection**.
4. Using the **Filter condition** property, enter an filter condition expression if you wish.

NOTE: Ensure the object is only added in the collection.

To modify a property in a database-based collection

In the steps below, the display value in **Entry date** in the **Address Book** is modified from `<dd.MM.yy>` to `<yyyy-MM-dd>`, as an example.

NOTE: The steps for this function can be used for the **Child relation column** and **Foreign key relation column**. For more information about these functions, see the One Identity Manager Web Designer Object Model Documentation.

1. Log in to the Web Portal with your user information and select the **Address Book** view.
2. Select the additional **Entry date** column, if it is not displayed, and look at the current display value.

   In our example, we presume that the `<MM/DD/YY>` display value is set.

3. In the **Address Book** view, click the title and switch to the Web Designer.
4. Open the **Properties** view and click to switch to the definition object in the definition tree view.
5. Open the **Collections** node using and mark the **Person** data object.
6. In the context menu, select the **Object in extension | Modify property** item.

   The display in the definition tree view switches to **Configuration (custom)**.
7. In the node editor view, in the **Base property** menu, select the `EntryDate` field and enter the new expression for the display value in the **Expression for display values** field.

   In our example, the following expression could be used.

   ```
   FormatDate(EntryDate, "yyyy-MM-dd")
   ```
8. Switch to preview mode and re-compile the web application.
9. Select the **Address Book** view and the additional **Entry date** column.

   In **Entry date**, the display value is `yyyy-MM-dd`.

To remove objects from the collection or delete them from the database

1. Right-click to select an action node.

   This could be the **Initializer** node of the component, for example.
2. Click **Delete** and select the new node.
3. In the **Node editor** view, select the required collection under **Collection**.
4. Perform one of the following tasks:
   a. **Delete row only from collection** under **Delete type** if you want to delete the object from the collection.
   b. Under **Delete type**, select the **Delete database object and save** option.
5. Using the **Filter condition** property, enter an filter condition expression if you wish.

To save objects in the database

1. Right-click to select an action node.

   This could be the **Initializer** node of the component, for example.
2. Select the **Save and click** action and click the new node.
3. In the **Node editor** view, select the required collection under **Collection**.
4. Using the **Filter condition** property, enter an filter condition expression if you wish.

Loading database objects through relations

Data is linked in a database by relations to each other. You can use these relations to load data from the database in the simplest way.

Prerequisite for using such a collection is a collection containing source data.

There are different type of relations for defining a collection:

- Defining with a foreign key relation:

  A foreign key relation helps automatically load those objects in a database view that were referenced by objects from another collection through a foreign key relation.

- Defining with child relation:

  A child relation helps automatically load those objects in a database view that reference objects from another collection through a child relation.

To define a collection with a foreign key relation

1. Open the module or component to which to add the collection.
2. Select the **Collections** node by right-clicking on it.
3. Select the **Database view by foreign key relation** menu item.
4. In **Source data collection**, select the collection with the source data.
5. In **Foreign key column**, select the foreign key column to create the relation to the target table.
6. Enter the name for the new collection in **Identifier**.

You do not have to trigger loading for this type of collection. The moment the data is available in the source data collection, the foreign key referenced objects are automatically loaded.

To define a collection with a child relation

1. Open the module or component to which to add the collection.
2. Select the **Collections** node by right-clicking on it.
3. Select the **Database view by child relation option** menu item.
4. In **Object type**, select the database table from which to load the objects.
5. In **Source data collection**, select the collection containing the objects to be referenced.
6. In **Child relation column**, select the foreign key column to create the relation to the target table.
7. Enter the name for the new collection in **Identifier**.

You do not have to trigger loading for this type of collection. The moment the data is available in the source data collection, the referenced objects are automatically loaded.

Loading database objects from multiple tables

For different use cases, it is necessary to examine database objects from multiple tables together. The special collection type `Database objects from multiple tables` in the object

model is available for this.

To define a collection with the option "Database objects from multiple tables"

1. Open the module or component to which to add the collection.
2. Select the **Collections** node by right-clicking on it.
3. Select the **Database objects from multiple tables** menu item.
4. Enter the name for the new collection in **Identifier**.

   A collection like this is database based, but can accept objects from multiple database tables.

For more information, see "Database objects" as a collection on page 117. You must explicitly provide the database table from which the objects will be loaded.

Using the database query wizard

NOTE: You can use the database query wizard in a module or component of a collection or when you want to add a new module or component.

You cannot execute SQL expressions directly in Web Designer for security reasons. Instead, this wizard helps you create SQL expressions that are stored in `QBMLimitedSQL` and linked to at least one permissions group in `QBMGroupHasLimitedSQL`. Your own collection node is added for the query.

To create a database query

1. Perform one of the following tasks:
   a. In the navigation view, select a module or a component.
   b. Create a new module or component.
2. In the definition tree view, mark the **Collections** node in the **Object definition** view and, in the context menu, select the **Wizards | Database query wizard** or the **Object in extension | Wizards | Database query wizard** menu item.

   NOTE: The navigation depends on whether you have selected a custom module/component or a default module/component. Default database objects first require an object extension. For more information, see Extensions on page 98.

   Wizard for creating or selecting a database query is displayed for a SQL expression to be entered or selected.
3. Select the **New** option.

   This displays the following fields:

Table 57: SQL expression settings

Setting	Description
Collection name	If necessary, overwrite the standard name for the collection that you are adding with this query.
Identifier	In this field, enter a unique identifier that is used later to identify the entry.
SQL expression	In this field, enter the entire expression. You can also declare parameters.
Description	In this field, enter an explanation of this expression. This makes it easier for other users to understand and classify this expression.

4. Modify the settings, then click **Next**.

   Specify the dialog groups that can execute the SQL snippet in **Wizard for creating or selecting a database query**. All dialog groups are selected.
5. Specify the relevant dialog groups.
   - Disable the relevant dialog groups by double-clicking .
   - Disable **Select all / deselect all** to deselect all listed dialog groups and to enable the relevant dialog groups one by one.
   - Enable the relevant dialog groups by double-clicking .
6. Specify the parameters you wish to select and set the values in the following **Wizard for creating or selecting a database query** page.
   - If necessary, disable parameter columns you do not want to include in the query or that were recognized by the parser as parameters but are not correct.
   - Disable **Select all / deselect all** to select all parameter columns individually.
   - To enter a value for the parameter, click next to the required parameter column.

   NOTE: You can check your database query with **Test statement**. As soon as the database processes running in the background are complete, you can test the new expressions.

   This display the **Processing** page and generates the new collection and SQL expression.
7. Click **Finish**.

To select a database query

1. Perform one of the following tasks:
   a. In the navigation view, select a module or a component.
   b. Create a new module or component.

2. In the definition tree view, mark the **Collections** node in the **Object definition** view and, in the context menu, select the **Wizards | Database query wizard** or the **Object in extension | Wizards | Database query wizard** menu item.

   NOTE: The navigation depends on whether you have selected a custom module/component or a default module/component. Default database objects first require an object extension. For more information, see Extensions on page 98.

   The **Wizard for creating or selecting a database query** page is displayed and you can enter or select a SQL expression.

3. Select the **Existing** option.

   This displays the following input and text fields.

   Table 58: Settings for an existing SQL expression

Setting	Description
Collection name	If necessary, overwrite the standard name for the collection that you are adding with this query.
SQL snippet	Select an existing fragment that you can use and modify in the **SQL expression** field.
SQL expression	In this text field, modify the selected snippet and the preset parameters if required.

4. Modify the settings, then click **Next**.

   Specify the dialog groups that can execute the SQL snippet in the following **Wizard for creating or selecting a database query** page.

5. Specify the relevant dialog groups.
   - Disable the relevant dialog groups by double-clicking .
   - Disable the relevant dialog groups by double-clicking .

     NOTE: Dialog groups that have already been specified in a collection of a default module cannot be deselected.

   - Disable **Select all / deselect all** to select or deselect all listed dialog groups and to enable or disable the relevant dialog groups one by one.

6. Specify the parameters you wish to select and set the values in the following **Wizard for creating or selecting a database query** page.
   - If necessary, disable parameter columns you do not want to include in the query or that were recognized by the parser as parameters but are not correct.
   - Disable **Select all / deselect all** to select all parameter columns individually.
   - To enter a value for the parameter, click next to the required parameter column.

   NOTE: You can check your database query with **Test statement**. As soon as the

database processes running in the background are complete, you can test the new expressions.

This display the **Processing** page and generates the new collection and SQL expression.

7. Click **Finish**.

Edit database queries

You do not have to select a node to edit database queries using this dialog. You can create new expressions or delete and edit existing expressions.

To select a database query

1. In the **Edit** menu, select the **Edit database query** item.
2. On the left-hand side of the **Edit database queries** dialog, mark a database query list.
3. Perform one of the following tasks:
    a. Overwrite the database expression name in the **Identifier**.
    b. Enter explanatory text in **Description**.
    c. Edit the SQL expression.
    d. Select or de-select other dialog groups.
    e. Select or de-select all dialog groups.
    f. Test the expression by clicking on **Test statement...** and enter a parameter value in the next dialog.
4. Save the changes.

Loading a historical object state

You can reset the state of a database object in a collection if you have enable recording of historical data in the database.

To reset a database object to a previous state

1. Add a database collection.

    For more information, see "Database objects" as a collection on page 117.
2. Add the **Load objects including history** action to an action node.
3. At this node, select the collect.
4. Choose between two options:
    a. Select **Load objects including history** to load historical object statuses for an individual object.

Here you can enter an object key (`XObjectKey`) that uniquely identifies the object.

b. Select **Load table with history** to load historical table data.

You can enter a column name and its value as restriction criteria.

NOTE: Once historical object states are loaded as described, you can simply set the collection to any point in time within the loaded time period.

5. Add the action **Set object to historical state** on the action node.
6. Select the collection.
7. Enter the date under **Date**.

The collection is set to the state that was current at that time.

Loading a change history

A change history contain a list of all operations recorded on a defined amount of data. You can load a change history into a special collection type designed for the purpose.

To define a collection for a change history

1. Open the module or component to which to add the collection.
2. Select the **Collections** node by right-clicking on it.
3. Select the **Change history** option.
4. Enter the name for the new collection in **Identifier**.
5. In the **Type** field, select the type of change history.

To load a change history

1. Right-click to select an action node.
2. Select the **Load change history** action.
3. Under **Collection**, select a collection.
4. In the **Type** field, select the type of change history defined for the collection.

NOTE: The setting in **Type** must be identical for the collection and the change history to be loaded.

5. Enter the date up to which you want to load the change history in **Load historical data back to (date)**.
6. Select one of the following scenarios to continue:

a. To load the change history of an individual object, enter the Web SQL expression of the (`XObjectKey`) object key of the required object.

b. Enter the relevant table name to load the change history for a table or an assignment object.

NOTE: Optionally, you can specify the column name and the value for a filter

condition.

TIP: Column-dependent references are advised for displaying historical data. For more information, see Column-dependent references on page 79.

View definitions

A collection layer view behaves in the same way as a database layer view. The content of the view is defined with a Web SQL expression. The view is updated during runtime. The view is automatically updated if the source data changes.

To add a view

1. Select the **Collections** node by right-clicking on them.
2. Select the **Add collection data view** action.
3. In the **Identifier** field, enter your name for the new collection.
4. In the **View expression** field, enter the Web SQL expression for representing the view content.

NOTE: Add a property with the matching data type for every view property in Web Designer.

Collections as data sources for controls

Some control types can load required data from the database themselves, assuming `Database` was selected as data source.

- The control type `Tree` load data from a database table depending on hierarchy level. First of all, the root level is loaded; the second level is loaded by expanding a node and so on.
- The control type `Grid` loads data depending on the selected page. If, for example, 20 entries are displayed per page, then initially only the first 20 database objects are loaded from the database.

This reduces memory usage and the control can be displayed faster.

Collection events

You can define actions to be executed if a specified operation is applied to the collection data. Events can be triggered by the following.

Table 59: Overview of the operations used

Operation	Description
`Insert`	A single row was added to the collection.
`Delete`	A single row was deleted from the collection.
`Update`	A single collection data value was updated.
`Bulk`	A set of rows was added to the collection or deleted from it.

To define an event

1. In the navigation view, select a module or component.
2. In the definition tree view, mark the `Events` node and, in the context menu, select the **Object in extension | Event handling** menu item.

   The view switches to **Configuration (custom)** and the new `Insert` subnode is selected. In **Node editor**, you can manage the following settings.

   Table 60: Settings for event handling

Setting	Description
Identifier (optional)	Enter the name for the event.
Operation*	Select one of the operations listed above.
Collection*	Select a collection in the option box. You can navigate to the selected collection using and apply further settings if necessary.
Control-ID*	A Control-ID is preset. You can overwrite this.

3. Adjust any settings you need to and then click in the toolbar.

To define an expression-based event

1. In the navigation view, select a module or component.
2. In the definition tree view, mark the `Events` node and, in the context menu, select the **Object in extension | Expression-based event** menu item.

   The view switches to **Configuration (custom)** and the new subnode is marked. In **Node editor**, you can manage the following settings.
3. Open the SQL field of the **Value*** box using .
4. Enter a value for the expression-based event and click **Apply**.
5. Save with .

Assigning collections to components

Before you can assign a collection to a component, you must prepare the custom component to make it suitable for the collection.

To do this, the component requires a "virtual collection", which you will find under the component's **Property** node in the definition tree view. Collections are assigned in the "virtual collection".

"Virtual collections" defined on components are marked as compulsory or optional.

Compulsory "virtual collections" must be assigned to a location collection. This assignment is made where the component's usage is defined.

Before you assign a collection to a component, a node must be configured under **Collection mapping**

Collection mapping nodes are automatically assigned when a component reference is added. These nodes are added automatically but not configured. You configure them in **Node editor**. You can also add a **Collection mapping** node with the help of a wizard that you can call from the context menu. Another option for adding these nodes, is to add a single node that you can also access through the context menu of your component in the definition tree view.

NOTE: The following instruction steps are based on an example of a direct container reference.

To assign a collection to a component

1. In the definition tree view, select the tab of the component or module, in which you want to use a **Container** type component.
2. Mark a **Container** type node and, in the context menu, select the **Component reference | Container (reference)** menu item.

   NOTE: If you add a component reference, the subnodes for collection mapping are automatically added. If more nodes are required for collection mapping, you can either add a **Collection mapping** type node yourself or by using the wizard. Other nodes may be added when you add a node with the wizard. You can delete these afterward.
3. In the **Note editor** view, in the **Identifier** field, select the component you want to reference.
4. In the definition tree view, select the **Collection mapping** node.
5. In the **Node editor** view, select the relevant collection under **Virtual collection**.
6. In the **Map to local collection** field, select a collection.

To define a virtual collection for a user-defined component

1. Select a custom component to which you want to assign a collection.

   This displays the object in the definition tree view.

2. Mark the **Properties** node and, in the context menu, select the **Virtual collection** menu item.

   This adds a new virtual collection in the definition tree view.
3. In the **Node editor**, in the **Identifier** field, enter a name for the virtual collection.

   NOTE: Use **Object type** and **Cursor required** to limit collection assignment further. Use **Object type** to ensure that the assigned collection must be a database collection of the given type. Use **Cursor required** to specify the point of the assigned collection which must be within an iteration above this.

Declaring configuration keys in modules and components

To declare a configuration key for a module or a component

1. In the definition tree view, click and open the **Configuration** node.
2. In the context menu, select the **Configuration section** item.
3. In the definition tree view, mark the new configuration section.

   You can see that the new configuration section is selected with empty fields in **Node editor**.
4. Enter the name for the new configuration section in the **Identifier** and **Description** fields in **Node editor**.

   The configuration section is displayed with its name in the definition tree view.
5. Mark the configuration section.
6. In the context menu, select the **Parameter** item.

   The new parameter is added under the configuration section.
7. Mark the new parameter.
8. In the **Key** field, enter a name for the configuration key.
   - Example of a configuration key name: `CCC_ConfigParm_Hyperview_HR`
9. In the **Identifier** field, enter the name from the **Key** field in the following way:
   - Example for input: `translate('#LDS#HR_Hyperview_Colums')`
10. In the **Description** field, enter the field to be display in the **Settings...** view under **Details**.
    - Example for input: `translate('#LDS#Choose Columns for HR-Hyperview')`
11. In the **Type** field, select the parameter type and edit the configuration key as in the sections about parameter types.

    NOTE: You may have to edit further settings depending on which parameter type you selected.

Detailed information about this topic

- Parameter type "Boolean Value" on page 64
- Parameter type "SQL filter condition" on page 65
- Parameter type "Selection from value list" on page 66
- Parameter type "Free text" on page 66
- Parameter type "Property list" on page 67
- Parameter type "Image" on page 69
- Parameter type "Configuration Objects" on page 70
- Parameter type "Color value" on page 74
- Parameter type "Color table" on page 75

Declaring context parameters

NOTE: You can only declare context parameter in modules.

Context parameters are utilized in things like automatically generated emails. These emails contain a link referencing a fixed point in the web application. This option removes step-by-step navigation in the application from the user.

Example: approving a request

A manager receives an email that an employee has been added and can request permissions. The email contains a link to a custom request page with preset data of the new employee.

To disable a context parameter for a module

NOTE: You have already selected a module. In the above example, the `CCC_ITShop_Approvals` module is used.

1. In the definition tree view, click and open the **Configuration** node.
2. In the `Context parameters` subnode, select the `UID_PersonWantsOrg` data object.

   Now you can edit the following settings for this context parameter in **Node editor**.

Table 61: Context parameter settings

Setting	Description
Identifier	The name of the selected data object is preset.
Option "Mandatory parmeter".	If this option is enabled, the parameter is mandatory.

Setting	Description
Type	Parameter type selection.
Comment	Specifies how the parameter is used.

3. Modify settings, then click .

Detailed information about this topic

- Linking to a page on page 110

Executing Microsoft .NET Framework code

The web project defined in the Web Designer is translated by the compiler in C# language. This compiles the complete logic of the definition document into a directly executable Microsoft .NET Framework assembly. It is possible to incorporate your own C# source code into this procedure. This way, you have access to the complete functionality of various execution layers. In this section, it will be explained how you incorporate your own C# source code.

Runtime API

Web Designer objects written in C# code can be integrated directly into the object definition. This makes customizing the web application very flexible.

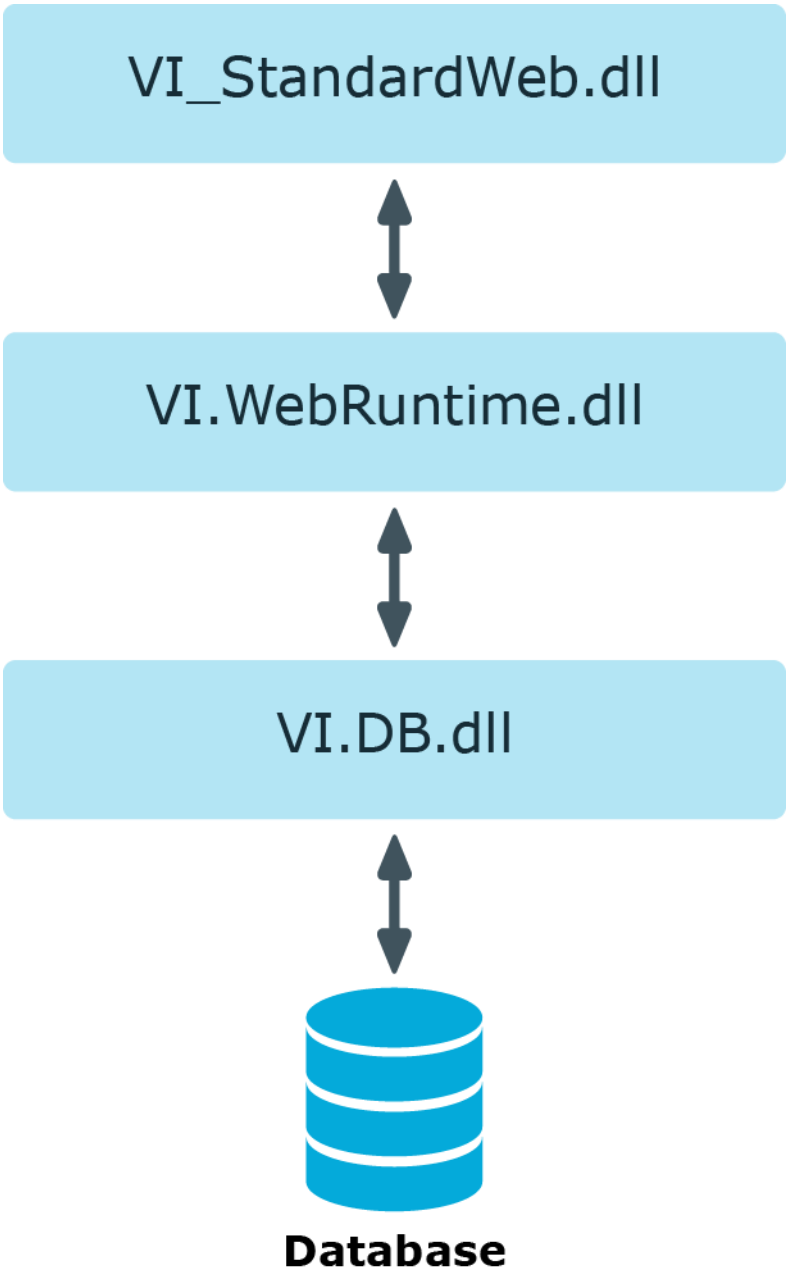
The web project code uses the runtime API of the Web Designer. The runtime API provides the following functionality, which is required for execution of the web project.

- Managing database connections with VI.DB.dll
- Authenticating and authorizing HTTP traffic
- Web controls
- Debugging Web Designer interfaces

The following diagram exemplifies the layer structure of components from the web project to the database.

For detailed information about runtime API classes, see the `OneIM_QBM_WebRuntime.chm` file.

Figure 2: Structure from the web project to the database



Integrating code into object definitions

In the Web Designer, you can insert a C# code fragment into an object's code using the `Code fragment` node type.

```
//------------------------------------------------------------------------------
// <auto-generated>
```

```
// This code is tool generated
// Runtime version:4.0.30319.34209
//
// Changes to this file may cause incorrect behavior and be overwritten when
// the code is regenerated.
// </auto-generated>
//------------------------------------------------------------------------------
```

NOTE: Changes to the code remain intact even on regeneration when you make them in the Web Designer.

Integrating C# code into a Web SQL expression

You can integrate C# code into every Web SQL expression. This can be either a C# expression or a command string.

To integrate a C# expression into a Web SQL expression, enclose it in double curly brackets, as in the example:

To integrate a C# expression into a Web SQL expression

- Enter the expression in double curly brackets.

```
format ("The database user is: {0}",
          {{ _Connection.User.Display }} )
```

NOTE: More complex commands are also enclosed in double curly brackets. The return value of the embedded expression is labeled with the keyword `@return`.

To integrate multiple C# expressions into a Web SQL expression

- Enter the expression in double curly brackets and use the keyword `@return`.

```
format ("User is: {0}",
          {{ string display = _Connection.User.Display;
               @return display; }} )
```

NOTE: The keyword `@return` must always be the last command in the embedded expression in this case.

Access to environmental data

The following statistical member variables are defined in all classes that can be modified by code. You can use these to gain access to the execution environment.

Table 62: Member variables overview

Name	Type	Description
_UserSession	VI.WebDesigner.Runtime.IUserSession	User session
_Form	VI.WebDesigner.Runtime.IForm	Form on which the code is executed
_Module	VI.WebDesigner.Runtime.IModule	Module in which the code is executed.
_Connection	VI.DB.IConnection	Actual database connection

Referencing controls

The generated Microsoft .NET Framework code creates a structure of ASP.NET controls. A local variable is defined for each control. The name of this variable corresponds to the identifier given to the Web Designer node.

The local variable can be accessed directly from within the same method.

Furthermore, all controls coming from the base class `System.Web.UI.Control` have an identifier (ID), which also corresponds to the Web Designer node identifier. This makes it easier to locate controls outside the visible range of the local variable.

Referencing collections

Data values can be loaded from or changed in collections using C# code.

Local collections of a module or component are defined in the component's TableStore.

Example:

```
TableStore.GetTable("SomeCollection")
```

NOTE: Virtual collections must use a slightly different call. Virtual collections are member variable of the class and can be accessed directly from the C# code using their name.

Customizing documentation

The menu bar of the Web Portal contains the **Help** button. The One Identity Manager Web Portal User Guide can be opened from this help function.

The Web Designer gives you the option to extend the guide and to modify it with your Web Portal custom modifications. You can add text content as well as visuals and logos. To add a help archive, copy the current zip files (Help_WebPortal_DE.zip and/or Help_WebPortal_EN.zip) into the Web Designer directory. Automatic updating keeps these up-to-date. Custom help is maintained as a zip file under the project files.

If you are in **Project files**, the **Add help archive** function is available in the toolbar. This archive is required to make changes or extensions to existing default One Identity Manager Web Portal User Guide. A new help archive is added in the definition tree view when you select this function.

Select the project to which to assigned the help archive in the top part of the view. All existing projects are listed under **Projects**. Then enter the language of your choice. A file name is generated from the selected project and language.

This file name cannot be configured. The lower part lists HTML documents, CSS documents and default help mappings. By double-clicking (or right-clicking) on an HTML document, for example, you can assigned custom help.

The selected file can be edited using **Export...**. You cannot edit directly in the Web Designer.

After you have finished editing the file, you must import it again into the Web Designer. The imported file is immediately available. You can even add your own HTML documents to the help archive. Custom help files replace the files from the default help archive after processing. You cannot delete default help files.

The help folders are regenerated if the help function is called in the Web Portal or if a help file is changed is one of the help archives default or custom).

Existing help files are updated if a new version is loaded. Custom help files are not overwritten.

You have the same function available to you when you right-click a selected file as in the menu bar. You also have the option to change the file name.

Table 63: Functions in the help archive menu bar

Icon	Function
	Delete. Deletes the selected file after confirming the security prompt.
	Export. Opens the default Windows dialog for saving the selected file to a data storage medium.
	Import.

Icon	Function
	Opens the default Windows dialog to select a file for importing.
	Adds a new folder in the selected directory. You can add files to the folder with **Import...**.

8

Compiling and debugging

Before you can execute a web project created with the Web Designer in a browser, you must compile with the Web Designer compiler.

Detailed information about this topic

- Compiling a web application on page 137
- Querying a web application on page 139
- Debugging on page 142

Compiling a web application

The Web Designer compiler can be called from the Web Designer as well as from the Database Compiler. For more detailed information Database Compiler see the *One Identity Manager Operational Guide*.

You must compile a web project in the following cases.

- After changing a definition (module, component, or similar) in the Web Designer.
- After changing the Web Designer configurator For more information, see Web project configuration options on page 61.
- After changing system settings, which required the Database Compiler to be run.

The Web Designer compiler creates a set of DLL files from the web project's XML definition and saves them in the database. The web application loads these DLL file and uses them to display the web interface.

There are basically two ways to compile a web project.

- Release compilation is run to release a specific version of the web project for use.
- Debug compilation is used during the development phase for testing and debugging purposes. Debug compilation creates additional code to support the Web Designer debugger. This means the DLL files are somewhat larger.

Changes within the Web Designer do not effect web application as long as there is no release compilation.

A web application loads the most recently compiled DLL file for the respective web project. If these DLL files are updated, the web project reloads the new DLL files; however, only new sessions run with the code from the newly loaded DLLs.

Compiling with the Database Compiler

Use the Web Designer to compile a web application outside the Database Compiler. For more detailed information about the Database Compiler, see the *One Identity Manager Operational Guide*.

To compile a database

1. Open the Launchpad and select the **Compile the database** entry.
2. On the Database Compiler's start page, click **Next**.
3. On the **Connect to database** page, enter your credentials for the One Identity Manager database and click **Next**.
4. In the menu, select the **Do not compile scripts** menu item.

   More options are enabled in the dialog and can be set.
5. Disabled the options you do not want to compile.

   NOTE: Expand the options fields to view additional settings and edit them.
6. Configure the additional settings, then click **Next**.

   Compiling takes place and the progress is displayed.
7. Close compiling using **Next**.
8. Close the program with **Finish**.

Viewing error messages

Tasks contains a list of compiler errors and warnings. Compiler errors preclude successful web project compiling and must be eliminated. Development states that cannot be compiled cannot be released because the corresponding web application do not run.

Compiler warnings relate to missing extensions, 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 extension 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 view, double-click a table entry; the Web Designer will then automatically go to **Node editor**. This behavior applies if the error is in

the node. If the error occurs when the resulting code is compiled, you can double-click it to display **Generated code (read-only)** in the definition tree view.

Table 64: Functions in the object state window toolbar

Icon	Function
	Display 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. You can send an email message in the default email program using **Send as Mail**. The error message is automatically transferred to the email.
	Errors. By clicking this button, errors can be shown or hidden.
	Warnings. By clicking this button, warnings can be shown or hidden.

Querying a web application

Web SQL expressions can be created and executed 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. The query window is divided into three sections.

NOTE: A WCF connection must be created for this function.

On the left, the currently available session collections are shown. The internal currently selected session is preset in the toolbar selection. Individual collections are bundled under the modules or components in which they were originally defined. The number of rows within the collection is shown next to the collection.

If you mark a database object in the hierarchy, the following context menu items are available. Which menu items are shown, depends on the database object you have marked.

Table 65: Context menu entries for marked collections

Entry	Description
Show definition object	Takes you to the definition object in the definition tree view and marks the nodes with a color.
Show collection data (this entry is only available for a selected collection)	This displays an expression for the query in the SQL input box. More information about the collection is displayed below the SQL input box.
Show last used WHERE clause (this entry is only available for a selected collection)	This displays the last WHERE clause in the SQL input box. More information about the collection is displayed below the SQL input box.

On the right-hand side of the window is a SQL query window. Here you can enter a query statement to be executed. The results of the query are also displayed in the part of the window.

To select another session

- In the query view's toolbar, click .

  This displays a list of active sessions.

To view the current module

- Click in the toolbar.

  Only the module selected in the currently selected session is shown.

  NOTE: If you click again, all module sessions are re-displayed.

To view the collection contents

- In the hierarchy, mark the collection.
  - From the context menu, select the **Show collection data** menu item.
  - Double-click the collection name.

    The collection contents is displayed on the right side of the query window.

Table 66: Toolbar functions in the "Queries" view

ICON	FUNCTION
	Sends a query. The entered expression is executed by clicking Web SQL. If more than one such expression has been formulated, mark the expression that is to be executed. Only one expression at a time can be executed.

The expression being executed is evaluated against the module or component currently selected in the tree structure. Use Web SQL syntax for inputting the expression to be executed.

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

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

To reopen a closed query window

- In the toolbar, in the **View** menu, select the **Query** menu item.

Related topics

- Metadata from a web application query on page 141

Metadata from a web application query

A column, `_DebuggerData`, with additional meta data is returned from the web application query.

Table 67: Meaning of the output in the "_DebuggerData" column

Column	Description
EntityState	State of the entity.
IsInteractive	Specifies whether this is an interactive entity (`IsInteractive=True`).
IsReadOnly	Specifies whether the data is read-only (`IsReadOnly=True`) or read/write (`IsReadOnly= False`).
Type	Database table.

For more detailed information about working with entities, see *One Identity Manager Configuration Guide*.

Evaluating the Web SQL expression of a property

This function is a debugging extension and is only available in debug mode. The evaluable Web SQL properties of an XML node for a specific control can easily be evaluated at runtime.

The `Label` node type, for example, has **Text** and **Tooltip text** Web SQL properties. **Identifier** and **Value** are additional Web SQL properties that are stored as HTML attributes and are Web SQL-evaluable.

NOTE: If Web SQL expressions are not available during runtime, these expressions are extracted from the XML definition. If the runtime code no longer corresponds to the XML definition, a different result may be obtained from Web SQL expression evaluation.

In the steps below, the evaluation is described using the `Label` node type as an example.

To evaluate the Web SQL expression of a property

1. In the preview, mark the database object that needs to be displayed and select the properties view.

   The database object, represented as an XML node, is marked in the properties view.

The evaluable Web SQL properties belonging to the XML node are determined in the background and are listed as entries in the context menu. These entries are selected in the next step.

2. Click the marked XML node and select **Evaluate label...**.

   All the corresponding modules are listed in the tree structure in the query window and the definition for the selected Web SQL property is displayed in the SQL input box.

3. In the query view's toolbar, click ▶.

   The results are shown below the SQL input box.

Viewing information specific to just one database object in the query window

In the query window, you can use a filter to display information concerning one database object only, such as a component. This function can be applied to `Module` or `Component` database objects. By default, this filter is not enabled. If the information you require appears in several components or modules, you can open another database object with the filter enabled in the definition tree view. The information concerning the newly selected database object is displayed in the query window.

This function is available in debug mode.

To narrow the display in the query window to a specific database object

NOTE: Before using this function, select the relevant database object and the corresponding property. For more information, see Evaluating the Web SQL expression of a property on page 141.

- In the query view, enable the **Filter on** option.

  The information displayed in the query window concerns the open component or module only.

  NOTE: If there is no information relating to the selected database object or if the selected document is not a component or module, the ⚠ icon is shown next to the filter.

Debugging

If error messages are shown in the preview while the web application is running, the Web Designer provides different forms of help.

NOTE: A WCF connection must be created for this function.

You can set breakpoints in the definition tree view for debugging. For more information, see Setting breakpoints on page 143.

You can use the following buttons in the preview to manage the Web Designer debugging.

Table 68: Buttons for managing debugging in the preview

Icon	Function
	Enable debugger.
	Single step mode.
	Resume execution

For more information, see Working with the preview on page 40.

An external program can also be used to debug the web application (for example, Visual Studio). Debugging steps are created in the debug compilation to help you with subsequent debugging. Debugging with external programs is dependent on the environment used. This approach is therefore not referred to again here.

For internal debugging, called-up methods can be displayed in the **Call Stack** view. This gives you important information on the source of the error.

For more information, see Call stack on page 144.

Setting breakpoints

To set a breakpoint

1. Perform one of the following tasks:
   a. Open the relevant module or component in the navigation view.
   b. Select a corresponding object in the properties view and click .

   For more information, see Properties on page 48.
2. Mark the node where you want to set the breakpoint and select **Set breakpoint**.

   NOTE: Not all nodes in the definition tree view can support a breakpoint. However, as a general rule, you can set a breakpoint on all nodes that can be suspended in single step mode. You can also set a breakpoint for **Container**, **Label**, or **Button** nodes. In these cases, a breakpoint is set as soon as the control is rendered.

   The marked node is highlighted in red.

To remove a breakpoint

- Mark the node that has the breakpoint and select **Remove breakpoint**.

  The marked node is no longer highlighted.

Call stack

Call stack contains a list of internal action calls when the debugger is enabled and a breakpoint activated. For more information, see Setting breakpoints on page 143. All actions executed during the breakpoint are listed. This information can be used to identify the C# methods and related DLLs causing the error.

NOTE: The description of the Call Stack given here only helps you to pinpoint the error source For debugging itself, use external development tools such as Visual Studio.

Call Stack provides the following information:

- Method

  Name of the method called up

- DLL

  DLL name. The DLL provides you with the origin of the method.

9

Monitoring

A installed web application can be monitored from your own web site.

The user groups authorized to call the monitor page are specified in the `Monitor.config` configuration file. By default, these are all the users that are members of the role `Builtin\Administrators`.

The monitor page currently has five tabs. The information shown on the monitor page or the available functions are described below. This page is currently available in the languages configured in your web browser.

To open the monitor page

1. Perform one of the following tasks:
   - On the start page, click .
   - In the menu bar, select **Debug view | Monitor page**.

   The monitor page opens in the standard browser.

Detailed information about this topic

- Status on page 145
- Sessions on page 146
- Assemblies on page 146
- Log files on page 147
- Exceptions on page 147

Status

When you open the monitoring page, the contents of the **Status** tab are displayed. Different information about the web application status is shown.

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

You can update the web application, assuming new assembly files are stored in the database. If there are no new connections pending, the update can start immediately. Otherwise, updating can be delayed until there are no more active connections. This ensures that no one looses their session or any data they have entered.

The update functions of the monitor page do not depend on the settings made in the configuration file.

To put a web application into maintenance mode

- On the **Status** tab, click the **Start maintenance mode** button.

  The web application does not accept any new connections. A warning page is shown when someone tries to connect.

To update the web application immediately

NOTE: This update only functions if there are no connections.

- On the **Status** tab, click the **Update now** button.

  The web application is updated and the new assemblies are loaded from the database.

To update the web application at a later date

NOTE: The update can be delayed until there are no more active connections.

- On the **Status** tab, click the **Update when all user sessions are closed** button.

  The web application is updated and the new assemblies are loaded from the database as soon as there are no more active connections.

Sessions

The **Sessions** tab shows information about every individual active session.

To display more details about individual sessions

- Click the **Details** link next to the active session.

  You can see more detailed the information about the session listed under the session.

To end a session

- Click **Terminate this session** below the active session.

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. Using different file versions in different environments or on different servers can sometimes make it difficult to identify the source of an error.

To search for assemblies

- Enter in the **Search** field, part of all of the assembly file name you want to find.

Log files

The contents of the log files of a web application are shown in the **Log Files** tab. The path for the log files stored in the configuration file (`NLog.config`) is used for this.

NOTE: Configuration setting for logging messages are made in `NLog.config` by NLog. For an exact description and functionality of NLog, see the online help (http://nlog-project.org/).

To improve performance, the most recent 500 log entries are displayed as a maximum. The use is notified when this limit is reached. 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 **Log files** tag 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, if an expected entry is not listed on this page, this only means that a corresponding log entry was not written. For more information about log files, see the *One Identity Manager Process Monitoring and Troubleshooting Guide*.

Exceptions

Unhandled errors are displayed in **Exceptions** if they occur during the execution of the web application. This is a view from the web application log. For more information, see Log files on page 147.

Web Portal performance indicators in Windows performance monitoring

When you install a web application, performance counters are registered, which provide information about the state of the application.

Performance indicators can be installed later.

NOTE: Prerequisites for this are that the web application is installed on a Windows Server

and has sufficient permissions to offer performance indicators. It may be necessary to add the application pool user account to the local group **Performance monitoring user** for this. Apart from this, the web application must be running in order to select the performance indicators.

For detailed information about monitoring with help from performance indicators, see the One Identity Manager Installation Guide.

10

Frequent tasks in the Web Portal

In this section you will discover which tasks you can expect with your successfully installed web application. You also obtain a detailed set of instructions and useful information about each task.

Detailed information about this topic

- Editing captions on page 149
- Adding functions on page 150
- Pasting in texts / captions on page 151
- Inserting grids on page 152
- Presets for grouping grids on page 153
- Visualizing exceptions in the Web Portal on page 154
- Replacing images in resource files on page 154

Editing captions

In order to customize text that is displayed in web applications, you must caption objects. Like all database objects, these are labeled either as default or custom objects. Caption objects are the only default objects that can be modified in a custom installation.

The advantage of being able to change these objects is that you do not need copy any modules. Moreover, it makes no difference whether a pre-defined caption object is referenced in a standard or custom module. The caption can be edited in both cases. Custom caption objects only occur in custom modules. For more information, see Multilingual captions on page 49.

The following caption editing rules and procedures apply.

- Caption objects can be referenced by more than one node. In this case, the changes also effect these nodes. To find the corresponding nodes, you need the search function. To do this, enter the key and search throughout the whole project.

- In the case of multilingual web applications, you may have to modify the translations for these objects. That means you must change several caption objects. You can find these objects with the search function in **Captions...**.

To change the caption of a caption object

1. In the preview, select the text that you want to modify.

   For example "Welcome" on the start page.
2. Mark the **Label** control you want in the `Properties` window and select **Show definition object**.

   The **Label** node for the marked text "Welcome" is highlighted in `Object definition`.
3. Mark the highlighted node and select **Move to extension | Add to configuration**.

   The definition tree view switches to **Configuration (custom)**.
4. Select the **Node editor** view.
5. In the **Text** field, enter your identifier for the place holder `Caption required for the label` as given in the following example.

   ```
   translate('#LDS#Caption required for the label')
   ```
6. Compile the web application.

   You will see the modified `Label` control in the preview.

   NOTE: You must log in again if required.

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. For more information, see Creating new modules on page 92. If an existing operating process is being dealt with the default module being used can be extended. To do this, the module or component in which the new function will be inserted has to be found first.

The following example shows you how to add a function. You want to add the name of the current user to the "Welcome" text on your start page. Before you add a new function, add a new object copy in the example.

To add a function

1. Update the preview in the Web Designer by clicking on and log on in the preview on the login page.

   NOTE: Once you have logged in to the Web Designer, the preview with the login page is loaded automatically.

   For more information, see Working with the preview on page 40.

2. On the preview's start page, click the "Welcome" link and select the **Properties** window in the lower pane of the Web Designer.

   The `Label` control for the "Welcome" text is gray.

3. Mark the `Label` control and, in the context menu, select the **Show definition object** menu item.

   The file belonging to the control is opened as a tab in the definition tree view. In this case, it is the default object `VI_Start`.

4. Mark the `Label` node and swap to the **Create object copy** view.

5. Specify the settings for the object copy and click **Next**.

   This adds new tab for the object copy in the definition tree view. In addition, your web project is given a substitution rule.

6. In the definition tree view, select the object copy tab and search for the **Label** node containing the "Welcome" text.

7. Mark the `Label` node and swap to the **Node editor** view.

8. Enter this text `+" "+from user select top 1 display()` in the **Text** field after `translate("#LDS#Welcome")` and click in the toolbar.

   NOTE: It is important not only to save the object copy with the modified function but also the default object with the automatically added substitution rule.

9. On the **Start page** tab, select the definition tree view and click **Refresh preview**.

   The changes you have made are shown in the preview of your web project.

You must create an object copy to retain the predefined default object in its original state. Customer-specific modules can, of course, be edited without making a copy. For more information, see Creating object copies with the wizard on page 97.

Pasting in texts / captions

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

To paste text or captions

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

   The procedure is the same as creating a new module. For more information, see Creating new modules on page 92.

2. Mark a node and, from the context menu, select the **Display nodes | Label** menu item.

   A new `Label`-type node is inserted in the definition tree view and automatically marked.

3. Switch to **Node editor** view and click next to the **Text** field.

4. Enter a SQL expression with the relevant caption.
5. In the **Action to execute*** menu, select the relevant action to be carried out when text is clicked in the browser.

   The following settings are available.

   - Display only
   - Execute action

NOTE: With **Execute action**, a subordinate `Server action` node is added to the `Label` node. To execute the action defined in the browser, click the label. Normally, `Label` controls, which are dependent on actions, are displayed differently in the browser (depending on the template file entered in the web project).

6. To obtain a valid node definition, the `Label` node must contain at least one text.

   The new text is shown in the preview window after compiling.

Detailed information about this topic

- Editing captions on page 149
- Multilingual captions on page 49

Inserting 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.

NOTE: Because grids can only be inserted under `Container`-type nodes, select, or insert a node of this type at the desired position.

To insert grids

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

   The procedure is the same as creating a new module. For more information, see Creating new modules on page 92.
2. Mark the required `Container` type node and, from the context menu, select the **Wizards | Data display** menu item.

For more information, see Creating a grid display for collection data on page 105.

Table 69: Overview of grid configuration options

Setting	Description
Enable the user to sort columns	If this option is enabled, the user can sort the columns in ascending or descending order by double-clicking the column header. This option is generally set for all columns in this dialog but you can also set it separately for each column in the table.
Sort collection	Use this option to specify an initial sort order in the layer. If this option is not enabled, the data sets are shown in the order that they are in the data set. For example, data sets have a logical sort order if you entered appropriate data when the collection was loaded.
Dynamic column width	Use this option to automatically divide up the column widths based on the length of the entries in the rows by applying an internal optimization algorithm. NOTE: These options relate to how the column width is fixed and can only be used alternatively.
Fixed column width	This option requires input in pixels. All columns in the grid are given the same fixed width.
Custom column width	This option enables the width and the unit (pixel or %) to be edited individually for each column.

3. Configure your settings.

Presets for grouping grids

You can use a simple code snippet to preset grouping of grids.

To preset grouping of grids

1. In the Web Designer, activate the **Extended properties** property on the grid node.
2. On the grid node, under the **Extended properties** entry, insert a new node of the type **Code snippet**.
3. For the new node, select the **Code** property and edit the value.
4. Enter the following source text:

```
var dataSource = GridBand1.DataSource;
var col = dataSource.Table.GetColumn("UID_Department");
```

```
if (!dataSource.GroupingColumns.Contains(col))
    dataSource.GroupingColumns.Add(col);
```

NOTE: Ensure that you select the identifier for `GridBand1`, the grid level and the column name for `UID_Department` that you want to group by.

Visualizing exceptions in the Web Portal

Error messages or exceptions triggered in the database layer (for example, format scripts on database columns) are displayed in a general error message in the Web Portal because exceptions normally contain too much technical detail for the end user.

If a custom error text is displayed in the Web Portal, the script must trigger an exception with exactly the same relevance for the end user. You have the following options.

- End user
- Default
- Intermediate
- Technical

NOTE: However, exceptions are always saved in full in the Web application log, regardless of the relevance for the end user.

Example of format script on a database column that cannot end with a period

```
If Value.ToString().EndsWith(".") Then
    Throw New ViException(#LD("This property may not end in a period sign.")#,
    ExceptionRelevance.EndUser)
End If
```

Replacing images in resource files

You can replace images that are in kept as compiled resources in one of the files `VI.ImageLibrary.dll` or `WebDesigner.ImageLibrary.dll` with project files.

NOTE: It is not possible to use replaced images on the login page if the web application is executing against an application server.

NOTE: Before you can replace an image, you must determine the image to be replaced and its associated information, such as, source, file name, and, if necessary, size, and status.

Find source

There are different ways to obtain more information about an image's properties. You can, for example, find the required information about the image URL. Use the property view to determine the correct source. You can recognize the source `VI.ImageLibrary` by the name `stockImg`. The name `Img` stands for the source, `WebDesigner.ImageLibrary`.

Find file

You can also find the name of the image file through the image URL. The file name is directly after the source and could look like:

```
stockImg=AssignedDirect
```

Find size

This value is only required for file from the `VI.ImageLibrary.Images` source. You obtain the value through the image URL. The value could look like this:

```
size=Small
```

find state

The state is only relevant for image files from the `VI.ImageLibrary.Images` source. This value can also be found from the image URL. The state could look like this:

```
state=Normal
```

There are different rules for the file names of both types of resource file, which are explained separately in the following:

Rules for file names of images from a VI.ImageLibrary.dll.

Resources are stored with the following name format in the `VI.ImageLibrary.dll` file:

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

For example:

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

The values for `<Size>` and `<Pixel>` of a resource file must match.

Table 70: Possible values for size and pixels

<size>	<pixel>	Description
Small	16	Image in 16x16 pixels
Medium	24	Image in 24x24 pixels
Large	32	Image in 32x32 pixels

This resource state can have the following values:

Table 71: Resource state values

<state>	Description
d	Not set
n	Normal
h	"Highlighted", emphasized color

IMPORTANT: This URL is case-sensitive.

Rules for file names of images from a WebDesigner.ImageLibrary.dll.

Resources are stored with the following name format in the `WebDesigner.ImageLibrary.dll` file:

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

For example:

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

To replace an image from a resource file

1. With suitable software if necessary, identify the name of the image you want in the resource file.

   There are different name format depending on which resource file is being dealt with.
2. Create a project file with the same name in the Web Designer and save it.

NOTE: Ensure that images from resource files can be cached in the browser if necessary.

Settings for improved accessibility

Web Designer offers setting options for barrier-free access to the Web Portal. To protect the compatibility of existing installations, these settings must be applied explicitly. You can use the `VI_GetWebSettings` script for this. The `VI_GetWebSettings` script is called up after each successful login, as long as this script is defined in the system. The script returns several key/value pairs that can enable specific settings. These settings only apply to the current session for which you have been authenticated. Different configurations can be implemented depending on the current user.

The following keys and values can be used.

Table 72: Possible settings

Key	Value type	Description
BaseCssClass	String	Values:

Key	Value type	Description
		• imx-highaccessibility: Enables CSS rules for improving accessibility. • imx-highcontrast: Enables high contrast mode. TIP: You can enter several values delimited by a space character.
GridGroupByDisabled	Bool	If the value is true, no grouping will be offered in the grid.
GridResizeDisabled	Bool	If the value is true, no column width modification will be offered in the grid control.
EnableHighAccessibility	Bool	If the value is true, the controls are displayed with increased accessibility.

The following example scripts tries to show how the settings should be made.

```
Public Function VI_GetWebSettings() As System.Collections.Generic.Dictionary(Of String, Object)
    Dim settings As New System.Collections.Generic.Dictionary(Of String, Object)

    settings.Add("GridGroupByDisabled", True)
    settings.Add("GridResizeDisabled", True)
    settings.Add("BaseCssClass", "imx-highaccessibility")

    ' return settings
    VI_GetWebSettings = settings

End Function
```

About us

One Identity solutions eliminate the complexities and time-consuming processes often required to govern identities, manage privileged accounts and control access. Our solutions enhance business agility while addressing your IAM challenges with on-premises, cloud and hybrid environments.

Contacting us

For sales and other inquiries, such as licensing, support, and renewals, visit https://www.oneidentity.com/company/contact-us.aspx.

Technical support resources

Technical support is available to One Identity customers with a valid maintenance contract and customers who have trial versions. You can access the Support Portal at https://support.oneidentity.com/.

The Support Portal provides self-help tools you can use to solve problems quickly and independently, 24 hours a day, 365 days a year. The Support Portal enables you to:

- Submit and manage a Service Request
- View Knowledge Base articles
- Sign up for product notifications
- Download software and technical documentation
- View how-to videos at www.YouTube.com/OneIdentity
- Engage in community discussions
- Chat with support engineers online
- View services to assist you with your product