



One Identity Manager 8.0

Configuration Guide

Copyright 2017 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 HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. One Identity make 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 do 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 indicates a potential for property damage, personal injury, or death.
-  **CAUTION:** A CAUTION icon indicates potential damage to hardware or loss of data if instructions are not followed.
-  **IMPORTANT, NOTE, TIP, MOBILE, or VIDEO:** An information icon indicates supporting information.

One Identity Manager Configuration Guide

Updated - November 2017

Version - 8.0

Contents

One Identity Manager Software Architecture	19
Working with Objects in One Identity Manager	22
Inserting, Modifying and Deleting Objects in One Identity Manager	25
Working with the Designer	27
General Information about the Program	27
Menu Items	28
Views in the Designer	30
Customizing Program Settings	31
Using Help	34
Limiting List Sizes	34
Filtering Lists	35
Searching for List Entries	36
Using the Designer's Own Full-Text Search	37
Using User Defined Filters for Searching	38
Using an Ad Hoc Filter	39
Using a Permanent Filter	39
Making and Committing Changes to Objects	41
The Designer Editors	42
Working with the Object Editor	44
Menu Items	45
Multiple Object Edit	46
Working with the List Editor	46
Menu Items	46
Functions in the Result List	47
Result List Configuration	48
Multiple Editing of List entries	48
Displaying Object Relations	49
Working with the SQL Editor	49
Menu Items	50
Support for Scripting	51
Customizing the One Identity Manager Default Configuration	55

Checking Data Consistency	57
Working with the Consistency Editor	57
Menu Items	57
Starting the Consistency Check	58
Displaying Test Objects and the Test Status	59
Specifying Test Options	60
Recording Test Results	62
Repairing Errors	62
Compiling a One Identity Manager Database	64
Compiling a Database with the Database Compiler	64
Logging Compiler Errors and Warnings	66
Working with Change Labels	69
Creating Change Labels	70
Displaying Contents of a Change Label	71
Booking Changes to a Change Label Retrospectively	72
Basic System Configuration Data	74
One Identity Manager Authentication Module	74
Editing Authentication Modules	103
Authentication Module Properties	104
Initial Data for Authentication Modules	104
Configuration Data for System User Dynamic Authentication	108
Database Connection Data	112
Database Connection Properties	113
Configuring a One Identity Manager Database for Testing, Development or Production	116
Configuration Parameters for System Configuration	117
Working with the Configuration Parameter Editor	117
Menu Items	117
Views in the Configuration Parameter Editor	118
Editing Configuration Parameters	119
Configuration Parameter Properties	120
Configuration Parameter Options	121
Setting up the Mail Notification System	122
Enabling More Languages for Displaying and Maintaining Data	126

Displaying Country Information	127
Working in Different Time Zones	128
Determining Working Hours	128
Editing Countries	129
Editing States	131
Modifying Public Holidays	133
Setting Up and Configuring Schedules	134
Calculating the Execution Time	135
Password Policies in One Identity Manager	138
Predefined Password Policies	139
Editing Password Policies	139
General Master Data for a Password Policy	140
Policy Settings	140
Character Sets for Passwords	141
Custom Scripts for Password Requirements	142
Script for Checking a Password	142
Script for Generating a Password	144
Restricted Passwords	145
Testing a Password	145
Testing Generating a Password	146
Assigning a Password Policy	146
Reloading Changes Dynamically	147
TimeTrace Databases	149
Machine Roles and Server Functions	150
Files for Software Update	151
Importing New Files into a One Identity Manager Database	152
Exporting Files from a One Identity Manager Database	154
Configuring Files	155
Operating Systems in Use	156
System Configuration Reports	157
Using Predefined Database Queries	158
Managing Custom Database Objects within a Database	159
The One Identity Manager Data Model	161
Basics of the Data Model	162
General Advice for Editing Table and Column Definitions	165

Working with the Schema Editor	165
Menu Items	166
Views in the Schema Editor	167
Editing Multiple Properties	168
Working with the Schema View	168
Standard Functions in the Schema View	169
Special Functions for Displaying the Data Model	171
Special Functions for Displaying Dependencies	172
Mapping Table Definitions	172
Table Properties	173
Working with Module Globally Unique Modifiers	179
Table scripts	180
Database Views of Type "View"	181
Database Views of Type "Union"	183
Database Views of Type "Proxy"	185
Database Views of Type "Read-only"	187
Mapping Column Definitions	188
Column Properties	189
Templates	195
Editing Value Templates	197
Example of Local Value Templates within an Object	199
Example of Value Templates across Objects	199
Formats for Checking Values	200
Formatting Types	200
Formatting Scripts	201
Permitted Column Values	202
Mapping Dynamic Foreign Keys	204
Column Dependencies for Setting Values	205
Configuring Columns for Full-Text Search	206
Mapping Table Relations	208
Supporting File Groups	212
Granting One Identity Manager Schema Permissions	215
Predefined Permissions Groups and System Users	216
Editing Permissions Groups and System Users	219
Working with the User & Permissions Group Editor	219

Menu Items	219
Views in the User & Permissions Group Editor	221
Permissions Group Dependencies	223
Copying Permissions Groups	224
Manually Creating Permissions Groups	226
Permissions Groups Properties	226
Editing System Users	227
Adding a System User to Permissions Groups	229
Dynamic system user	230
Which Employees Use System Users?	230
Editing Permissions for One Identity Manager Schema Tables and Columns	231
Rules for Finding Valid Permissions for Tables and Columns	233
Working with the Permissions Editor	235
Menu Items	235
Views in the Permissions Editor	237
Functions in the Permissions Edit View	237
Filtering Entries in the Permissions Editor	238
Granting Permissions for Tables	239
Granting Column Permissions	240
Availability of Certain Functionality	241
Displaying Permissions for an Object	243
Displaying the Current User's Permissions	244
Working with the User Interface	246
Object definitions for the User Interface	247
Selection Criteria for Object Definitions	248
Using Display Text for Object Definitions	248
Working with Object Definitions	249
Object Definition Properties	250
User Interface Navigation	251
Navigation View Elements	252
Recommendations for Editing Menu Navigation	253
Working with the User Interface Editor	254
Menu Items	254
Views in the User Interface Editor	256
Editing Menu Items	258

Selecting the Type of Navigation View to Edit	259
Copy an existing user interface navigation	260
General Menu Item Properties	262
Creating Database Queries for Data Dependent Menu Items	264
Data Dependent Menu Item Uniqueness	266
Recursively Data Dependent Menu Items	266
Editing Lists	267
Display Template for Displaying a List	268
Defining Insert Values	269
Using Links	270
Using Variables	271
Editing and Displaying Variables	274
Forms for the User Interface	274
Recommendations for Editing Forms	275
Working with the Form Editor	276
Menu Items	276
Views in the Form Editor	277
Editing Interface Forms	279
User Interface Form Properties	281
Form Definitions and Form Templates	282
Effects of Object Definitions when Displaying Interface Forms	286
Features of the Assignment Form	287
Forms for Custom Extensions	287
Custom Master Data Forms	289
Configuration Data for Displaying Many-to-Many and Object Relations on Forms	290
Replacing Default Forms with Custom Forms	296
Working with Overview Forms	297
Working with the Overview Form Editor	298
Creating Overview Forms	300
Designing an Overview Form	303
Disabling Overview Forms	305
Statistics in the One Identity Manager	306
Working with Statistic Definitions	306
General Properties of a Statistic Definition	307

Querying Statistic Measurements	308
Linking Statistics into the User Interface	310
Diagram Types for Visualizing Statistics	311
Examples of Statistic Definitions	315
Extending the Launchpad	319
Launchpad Actions	321
Task Definitions for the User Interface	322
Task Definition Properties	324
Applications for Configuring the User Interface	325
Program Properties	326
Icons and Images for Configuring the User Interface	327
Language Dependent Data Representation	328
Basic Rules for Using Language Dependent Data	329
Labeling Columns for Translation	330
Working with the Language Editor	332
Menu Items	332
Editing Translations in the Language Editor	333
Importing Translations	334
Process Orchestration in One Identity Manager	336
Declaring the Job Server	337
Working with the Job Server Editor	338
Menu Items	338
Views in the Job Server Editor	339
Editing a Job Server	340
Job Server Properties	341
Server Functions of a Job Server	343
Job Server Machine Roles	344
Job Server Statistic Information	345
Installing the One Identity Manager Service on the Job Server	346
Customizing the One Identity Manager Service Configuration for a Job Server ...	349
Template for the Configuration File	350
Selecting Modules Types and Editing the Parameters	350
Configuration File Verification Test	352
One Identity Manager Service Configuration	353
One Identity Manager Service Configuration Files	356

Process Collection Module	357
MSSQLJobProvider	358
OracleJobProvider	359
FileJobProvider	359
FTPJobProvider	361
HTTPJobProvider	364
AppServerJobProvider	364
The Jobdestination Module	365
JobServiceDestination	365
FileJobDestination	368
FTPJobDestination	370
HTTPJobDestination	372
The Configuration Module	373
The Logwriter Module	374
EventLogLogWriter	374
FileLogWriter	375
The Dispatcher Module	377
The Connection Module	378
The HTTP Authentication Module	379
The Plug-ins Module	379
HTTPLogPlugin	380
ScheduleCommandPlugin	380
DBSchedulerWatchDogPlugin	380
RequestWatchDogPlugin	381
PerformanceCounterPlugin	381
DebugMailPlugin	382
ShareInfoPlugin	382
RemoteConnectPlugin	382
Module File with Private Key	383
Handling Processes in the One Identity Manager	384
Working with the Process Editor	385
Menu Items	385
Views in the Process Editor	388
Working with the Process Document	389
Defining Processes	391

Creating and Editing Processes	392
Creating and Editing Process Steps	399
Process Step Parameters	406
Searching for an Entry within a Process	410
Comparing Processes	410
Exporting and Importing Processes	411
Simulating Process Generation	411
Checking the Validity of a Process	413
Compiling a Process	415
Executing Processes Automatically	416
Working with the Process Plan Editor	416
Creating a Process Plan	418
Process Components	419
Properties of Process Components, Process Tasks and Parameter Templates	422
Tracking Changes with Process Monitoring	425
Basics for Process Monitoring	426
Logging Data Changes	427
Logging Process Information during Process Handling	429
Setting Up Process Information for Process Handling	431
Recording Messages in the Process History	433
Process Tracking for DBQueue Processor Operations	434
Example for Replacing the GenProcID	434
Archiving and Deleting Recordings	439
Selecting a Method	441
Specifying Data Retention Periods	442
Configuring the One Identity Manager History Database for Archiving	444
Configuring Archiving with XML Files	444
Direct Deletion of Records in the One Identity Manager Database	445
Conditional Compilation using Preprocessor Conditions	448
Preprocessor Relevant Configuration Parameters	449
Preprocessor Conditions in Objects	449
Preprocessor Conditions in VB.Net Expressions	451
Evaluation of Preprocessor Conditions during Compilation	452
One Identity Manager Scripts	453

Using Scripts	453
Message Output	454
Implementing Dates	454
Using Dollar (\$) Notation	455
Accessing Local Object Columns	456
Accessing Object Columns Connected by a Relation	457
Accessing the Old Column Value	457
Accessing Column Display Values	458
Accessing References in Comments	459
Accessing Local Object Meta Values	459
Using base. Object	460
Calling Functions	460
Pre-scripts for using in Processes and Process Steps	461
Using Session Services	461
Querying Configuration Parameters	462
Testing the Existence of Certain Database Entries	462
Querying Session Object Global Variables	463
Using #LD Notation	464
Example for Use in Process Tracking	465
Example for Specifying Language	466
Scripts in the Script Library	467
Working with the Script Editor	467
Menu Items	467
Support for Scripting	469
Editing Scripts	470
Overriding Scripts	472
Testing Script Compilation	473
Testing a Script with the Script Editor	473
Testing Scripts with the System Debugger	474
Loading the System Library	475
Testing and Editing Scripts	476
Testing a Script	477
Editing a Script in the System Debugger	478
Saving Modified Scripts	478
Testing Templates and Formatting Scripts	479

Testing Methods	480
Testing Table Scripts	480
Logging Database Queries and Object Actions	481
Maintaining Mail Templates	483
Working with the Mail Template Editor	483
Menu Items	483
Views in the Mail Template Editor	484
Creating and Editing Mail Templates	485
General Properties of a Mail Template	486
Creating and Editing an Email Definition	487
Using Base Object Properties	488
Use of Hyperlinks in the Web Portal	489
Customizing Email Signatures	493
Reports in the One Identity Manager	494
Working with the Report Editor	494
Menu Items	495
Views in the Report Editor	496
Customizing Program Settings	497
Logging Database Queries	498
Creating and Editing Reports	498
General Report Properties	499
Creating a Data Source	500
Data Retrieval using an SQL Query	502
Data Retrieval using a Database View	502
Data Retrieval using an Object	503
Data Retrieval using Single Object History	504
Data Retrieval using Multiple Object History	506
Data Retrieval using Historical Assignments	507
Data Query for Simulation Data	509
Editing Report Parameters	510
General Parameter Settings	512
Defining Parameter Values	513
Settings for Calculating Values	515
Using Virtual Data Sources	515

Editing the Report Form	516
Adding Data Fields to a Report Form	516
Example of a Simple Report with Data Grouping	518
Translating Reports	522
Linking Reports into the User Interfaces	524
Custom Schema Extensions	526
Creating a New Table	527
Extending a Table	529
Defining Columns	529
Creating a Simple Column	530
Creating a Foreign Key Column	531
Creating New Columns for Database Views	531
Configuring Column Properties	532
Creating a Read-Only Database View	534
Creating Foreign Key Relations for Views	535
Creating a Union View	536
Creating New Assignments Tables	537
Creating Indexes	538
Permissions for Schema Extensions	539
Specifying Change Labels for Schema Extensions	540
Adding Schema Extensions	541
Adding in Custom Extensions to the User Interface	542
Transporting One Identity Manager Schema Customizations	544
Basics for Transport of One Identity Manager Schema Modifications	545
Creating a Transport Package with the Database Transporter	546
Transporting SQL Statements	549
Transporting Favorite Objects	549
Transport by Change Label	550
Transport by Change Information	551
Transporting Schema Extensions	552
Transporting Selected Objects and their Dependencies	553
Transporting System Configuration	554
Transporting System Files	555
Displaying Contents of a Transport Package	555

Importing a Transport Package with the Database Transporter	556
Advice on Importing the Entire System Configuration	557
Displaying Transport History	558
Importing Data	560
Importing Data from CSV Files	560
Loading Import Files	562
Specifying the File Structure	562
Specifying Line Format for Data with Delimiters	563
Specifying Line Format for Data with Fixed Width	564
Specifying a Line Condition	565
Importing Data from a Database	565
Selecting an External Database	566
Source Data Query	567
Import Configuration	568
Assigning Target Tables and Columns	568
Assigning Target Columns with the Assignment Wizard	569
Specifying the Hierarchy	570
Options for Handling Records	571
Specifying Connection Variables	572
Importing Data	573
Using Import Definition Files	573
Web Service Integration	575
Binding a Web Service	576
Generic Web Service Call	576
Direct Web Service Call	578
Self-Defined Web Service Call	578
Creating a Web Service Solution with the Web Service Integration Wizard	579
Changing a Web Service Solution	582
Deleting a Web Service Solution	583
SOAP Web Service	584
Installing and Configuring the SOAP Web Service	585
Installing the SOAP Web Service	586
Configuring the SOAP Web Service	589
Displaying the Status of a SOAP Web Service	591

Uninstalling the SOAP Web Service	592
Examples of Calls	592
One Identity Manager as SPML Provisioning Service Provider	598
SPML Web Service	598
Installing and Configuring SPML Web Services	599
Installing the SPML Web Service	600
Configuring the SPML Web Service	603
Uninstalling the SPML Web Service	606
Configuring the One Identity Manager Schema	606
Preparing the One Identity Manager Schema for Exporting to the SPML Schema ..	607
Creating the Schema File	608
Testing SPML Web Service Functionality	608
SPML Test Front-end Configuration	609
Using the SPML Test Front-end	610
Searching for Errors in the One Identity Manager	611
Monitoring Process Handling with Job Queue Info	611
Working with Job Queue Info	612
Menu Items	612
Customizing Program Settings	614
Updating the Views	615
Column Configuration	615
Job Queue Info Views	615
Monitoring Process Execution	617
Displaying Details of Process Handling	618
Displaying Details of Process Step Handling	619
Displaying Details of a Process Step Parameter	620
Reinstating Process Steps and Processes	621
Enabling Extended Logging of Process Steps	622
Determining the State of the Server	623
Displaying Entries in the DBQueue	624
Displaying Messages in the Database Journal	624
Displaying the Job Queue Sequence	626
Stopping the System (Emergency Stop)	626
Error Messaging in the Error Message Window	627

Displaying Messages in the Error Log View	629
Logging Messages in the Database Journal	632
Writing the One Identity Manager Log	634
Logging Messages using NLog	634
Enabling the Crash Recorder	635
One Identity Manager Service Logging	636
Configuring the One Identity Manager Service Log File	636
Advanced Logging in the One Identity Manager Service	638
Displaying the One Identity Manager Service Log File	638
Extended Debugging in the One Identity Manager Service	640
Output of Extended Return Values from Individual Process Components	641
Outputting Custom Messages in the One Identity Manager Service Log File	641
Logging Messages in the Event View	642
HTTPLogPlugins Log File	643
Displaying One Identity Manager Application Server Status	644
Processing DBQueue Tasks	645
Configuring the DBQueue Processor for Test and Development	645
Initializing the DBQueue Processor	646
DBQueue Processor Reinitialization	648
Controlling Processing of DBQueue Processor Tasks	648
Processing DBQueue Processor Tasks	649
Reactivating DBQueue Processor Tasks	650
Bulk Processing in DBQueue Processor	651
How the Central Dispatcher Communicates with Individual Slots	651
Communication during Processing	652
Scheduled Maintenance Tasks	653
One Identity Manager Configuration Files	655
Globallog.config	655
*.exe.config	658
Global.cfg	659
One Identity Manager Service Configuration Files	660
Jobservice.cfg	660
viNetworkService.exe.config	662
About us	664

Contacting us	664
Technical support resources	664
Index	665

One Identity Manager Software Architecture

The basis for the One Identity Manager structure is classic 3-tier architecture. However, in One Identity Manager the object layer (business logic) is shared. This allows high performance gain due to separate time and location processing.

Database Layer

The database represents the One Identity Manager kernel. It fulfills the main tasks, which are managing data and calculating inheritance. Object properties can be inherited along the hierarchical structures, such as, departments, cost centers, location or business roles. In the case of data management, the database maps the managed target systems, ERP structures as well as the compliance rules and access permissions.

The database is separated into two logical parts, payload and metadata. The payload contains all the information required to maintaining data, such as information about employees, user accounts, groups, memberships and operating data, approval workflows, attestation, recertification and compliance rules.

The metadata contains descriptions for the payload, such as, scripts for formatting rules and value templates or specific interaction. One Identity Manager's entire system configuration, all the front-end control settings and the queues for asynchronous processing of data and processes are also part of the metadata.

Recalculation of inheritance is started by the database trigger logic. The triggers queue processing tasks in a task list called the "DBQueue". The DBQueue Processor processes these tasks and recalculates inheritance of the respective database objects. The table "Jobqueue" is used for storing processing tasks that are run from the object layer.

The database systems SQL Server or Oracle Database can be implemented.

Object Layer

The object layer enables object oriented access to the database data. The VI.DB.DLL creates entities for objects and collections. Entities use external session services to load (EntitySource) and save (UnitOfWork) data objects. Save operations are grouped so that several data objects can be saved in bulk. There are default events (Insert, Update, Delete) available for each object which can be generated after saving.

Each entity is assigned one or more processing logic routines (EntityLogic). These group together operations that can be carried out on an entity. Unique Customizers have been developed for different entities. A Customizer is one EntityLogic, which provides defined behavior for one entity. Customizers execute processing, logic which is normally implemented in the object code, such as mutual exclusion of properties.

A value template can be assigned to each of the generated object's properties. Templates are implemented for generating user data or for transforming values. You can use templates to fill object properties with default values or to form property values from other properties of the same or other objects.

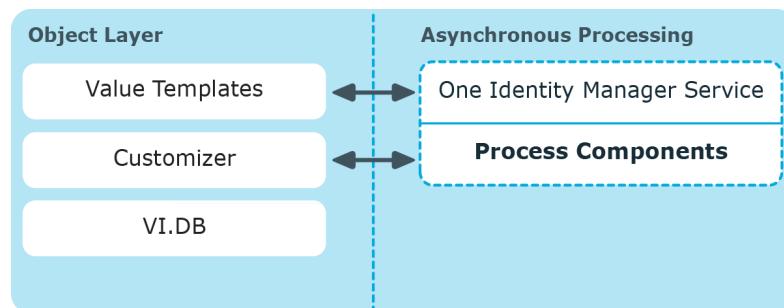
One Identity Manager uses so called 'processes' for mapping business processes. A process consists of process steps, which represent processing tasks and are joined by predecessor/successor relations. This functionality allows flexibility when linking up actions and sequences on object events. Processes are modeled using process templates. A process generator (Jobgenerator) is responsible for converting script templates in processes and process steps into a concrete process in the 'Job queue'.

The server service "One Identity Manager Service" ensures distribution in the network of data managed in the One Identity Manager database. The One Identity Manager Service performs data synchronization between the database and any connected target systems and executes actions at the database and file level. The One Identity Manager Service retrieves process steps from the JobQueue. Process steps are executed by process components. One Identity Manager Service also creates an instance of the required process component and passes the parameters to the process step. Decision logic monitors the execution of the process steps and determines how processing should continue depending on the results of the executed process components. The One Identity Manager Service enables parallel processing of process steps because it can create several instances of process components.

The One Identity Manager Service is the only One Identity Manager component authorized to make changes in the target system.

Strictly speaking, the One Identity Manager Service is part of the object layer because it does not contain any business logic. The One Identity Manager Service provides help for realizing asynchronous processing.

Figure 1: One Identity Manager Object Layer



Presentation Layer

The presentation layer consists of front-ends that are used for data input and output. There are different front-ends for different tasks. For example, a different front-end is used to configure One Identity Manager as that for managing employee data. The contents to be displayed and the extent to which it can be altered is determined in conjunction with the access rights of the respective user through the object layer. Available front-end solutions are client and browser based.

Clients connect to an application server storing business logic. The application server provides a connection pool for accessing the database and ensures a secure connection to the database. Clients send their queries to the application server, which processes the objects, for example, by determining values using templates and sending the results back to the clients. The data from the application is sent to the database when an object is saved.

Clients can alternatively work without external application servers, by keeping the object layer themselves and accessing the database layer directly. In this case, only the part of the object layer required for the acquisition process is mapped in the clients.

There is an application running on a web server based on a web page render engine for implemented browser-based user interfaces. Users use a web browser to access the website that has been dynamically set up and customized for them. Data exchange between database and web server can take place directly or through the application server.

Figure 2: Layer Distribution with Application Server

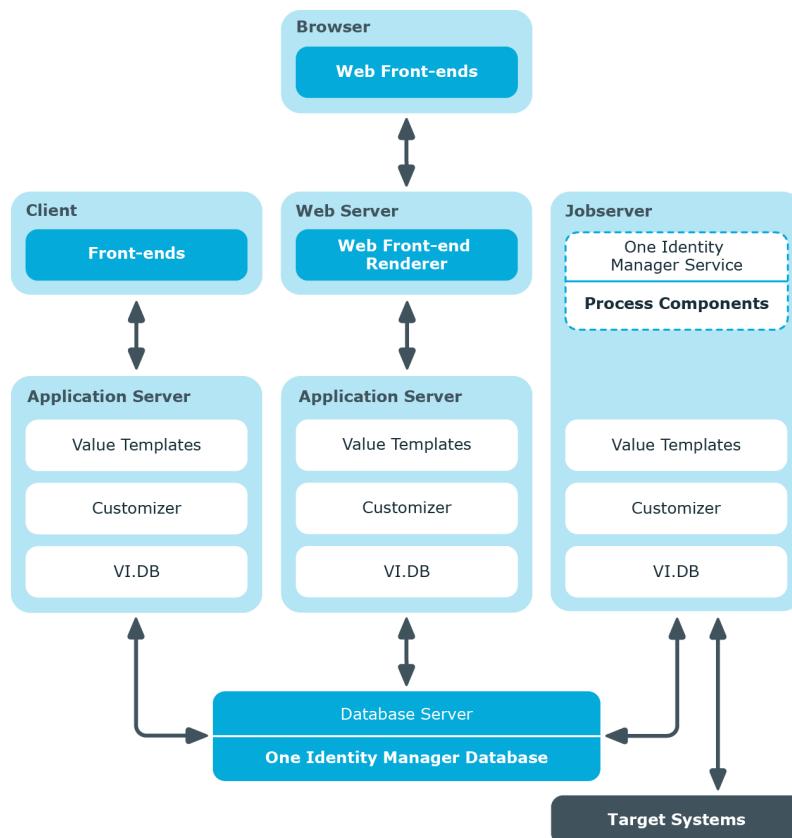
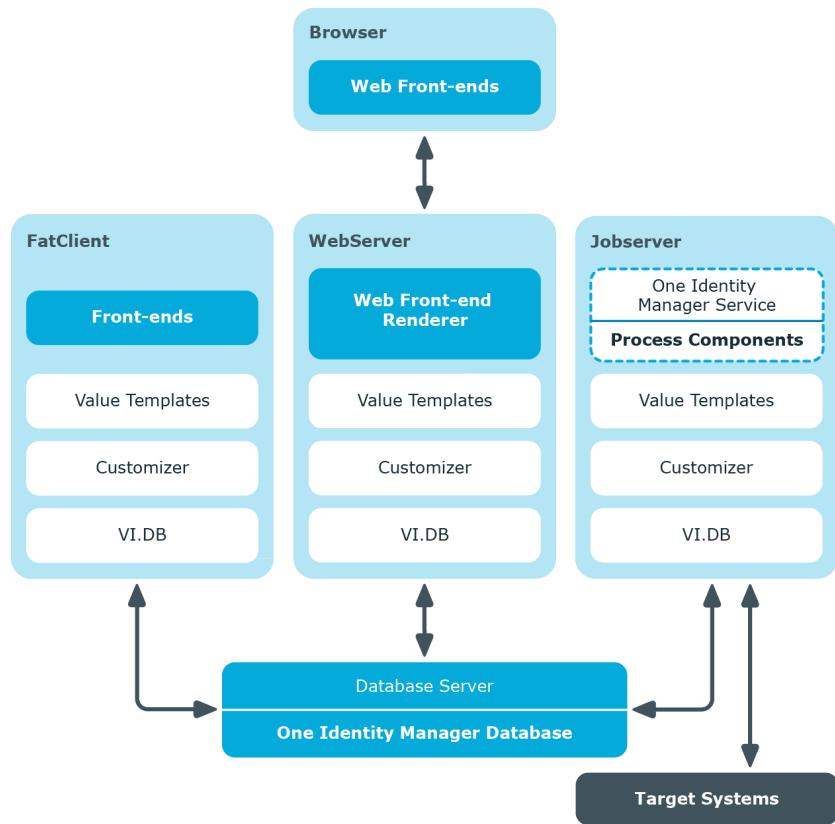


Figure 3: Layer Distribution without Application Server



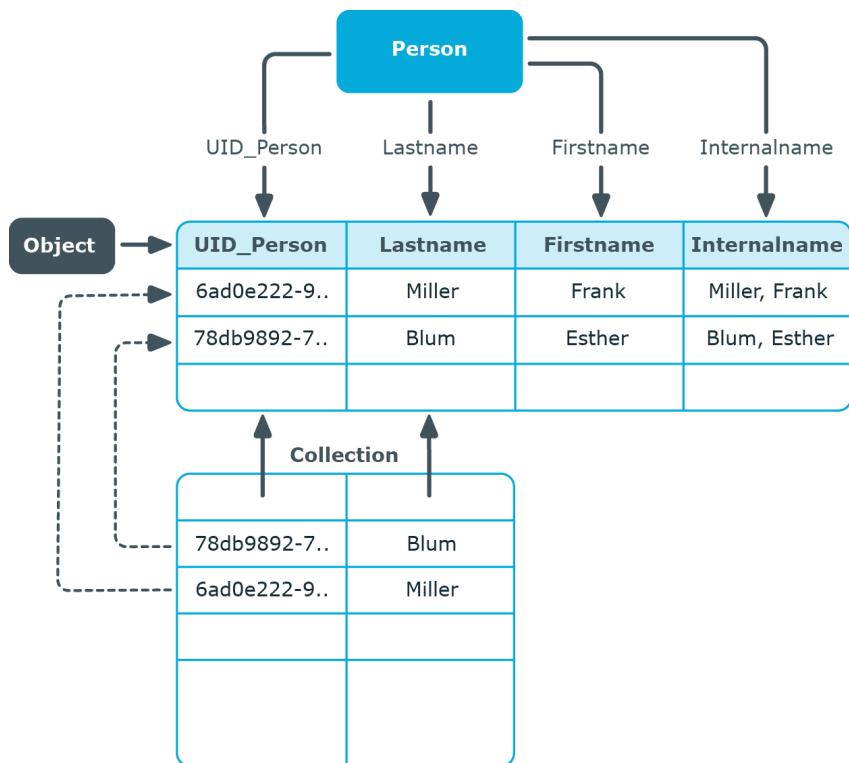
Related Topics

- [Working with Objects in One Identity Manager on page 22](#)
- [Inserting, Modifying and Deleting Objects in One Identity Manager on page 25](#)

Working with Objects in One Identity Manager

The object oriented access to tables and data sets takes place through the One Identity Manager object layer.

Figure 4: Access to Tables and Data Sets



The following applies to this:

- Object class - table
- Properties - columns
- Object - target
- Collection - number (1-n) of columns in a table with several lines.

Objects and collections are mapped through entities. Entities execute database operations using the following default methods:

- "EntitySource" - creates new objects and collection or loads objects and collections.
- "UnitOfWork" - groups save operations of several objects and collections.
- "Discard" - discards objects.
- "MarkForDeletion" - objects are marked for deletion. Not deleted until saved.

When an object is loaded, all the columns are loaded. When a collection is loaded not all the columns are loaded, on the grounds of performance, only the primary key, all columns in the display template plus the details of whether an object is marked for deletion. Defined display templates specify how each collection object is displayed in the front-end. Defaults for the each table's display template are stored in the One Identity Manager schema and can be customized.

Furthermore, each object knows the following default events that can be generated as a result of saving:

- Insert
- Update
- Delete
- Assign (adds M:N assignments)
- Remove (removes M:N assignments)

Processes can be linked to these events that execute actions in different target systems, for example, to add user accounts, add a home directory on a server or write data to the One Identity Manager database.

Table 1: Lifecycle of an Object

Front-end Action	Object State	Event on Saving	Database Action
Insert an object.	Object does not exist.	Insert	UID is created, Object is added to the database.
Change properties.	Object exists in the database and is loaded.	Update	Object properties are changed.
Delete object.	Object exists in the database and is loaded.	Delete	<p>For objects with the property "Marked for deletion" (XMarkedForDeletion):</p> <ul style="list-style-type: none"> The method "MarkForDeletion" is executed. Objects are locked and cannot not be modified. If deferred deletion > 0 days is configured, a deferred operation is created for deletion. The objects are initially disabled. During the retention period, you have the option to restore the objects. If a deleted object is restored, the object properties are reset to their state before deletion. The objects are finally deleted when the deferred deletion time period has expired. Object with deferred deletion on 0 days are deleted immediately. <p>Objects without the property "Marked for deletion" are deleted immediately.</p>

Related Topics

- [Inserting, Modifying and Deleting Objects in One Identity Manager on page 25](#)

Inserting, Modifying and Deleting Objects in One Identity Manager

All actions in One Identity Manager are executed over the object layer and saved in the One Identity Manager database. Each change to an object (insert, change, delete) is executed within a transaction. Another fixed item in a transaction of this type is creating the processes themselves. The transaction can only be successfully completed if the changes are saved and the processes have been successfully generated. If errors occur within the transaction the entire transaction is rolled back.

The following is an example of how to insert a object in One Identity Manager.

Take the following steps in the front-end:

- Insert a new object
- Enter the object properties

Properties dependent on this object are created with template. Side-effects implemented in the Customizer are used, such as, exclusion of certain properties.

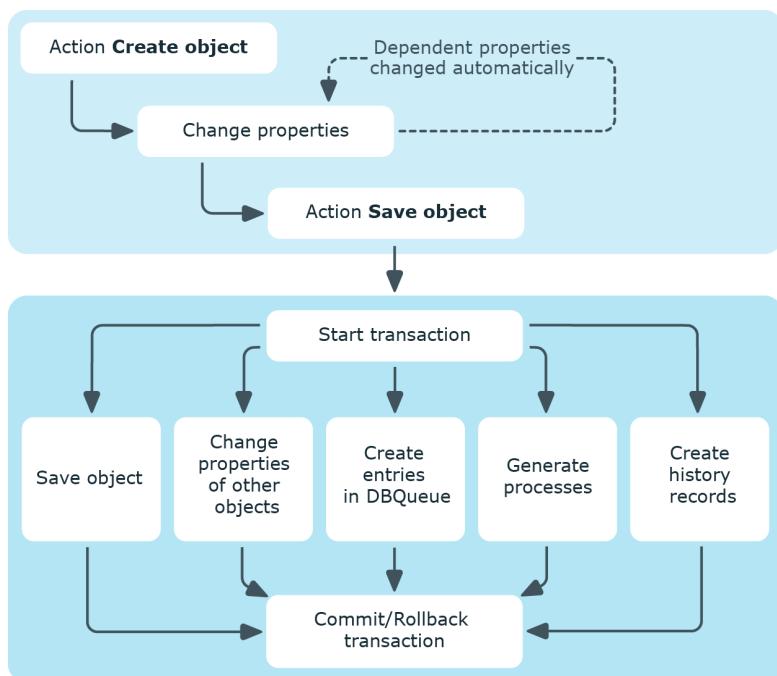
- Save the object

After saving the object in the front-end, the following step are executed in the object layer:

- A transaction is started (Begin Transaction)
- The following steps are processed in parallel:
 - Saving objects in the database
 - Applying templates and formatting scripts to dependent objects
 - Generating processing tasks in the Job queue for the One Identity Manager Service
 - Generating processing tasks in the DBqueue for the DBQueue Processor
 - Generating entries for logging changes in a history
- Transaction ends with success (Commit Transaction) or changes are rolled back if an error occurs (Rollback Transaction)

The following visual helps to show the flow of data when an object is inserted.

Figure 5: Dataflow Inserting a Object



Working with the Designer

The Designer is the main configuration component in One Identity Manager. The program offers an overview of the entire One Identity Manager data model. It enables the configuration of global system settings, for example, languages or configuration parameters such as customizing user interfaces for the various administration tools. It also allows the permissions structure to consolidate the various administrative tasks of each user and user groups. Another important task is the definition of workflows for technically illustrating the administration procedures in the company.

- **NOTE:** The general functionality of One Identity Manager tools is described in the One Identity Manager User Guide for One Identity Manager Tools User Interface and Default Functions. Only additional Designer functions are described in the following.

General Information about the Program

During the startup process the Designer fills an internal database. SQLite is used for the database system. This internal database contains the schema and files from the system part of the connected One Identity Manager database.

Depending on the program's configuration, the internal database is either loaded into main memory or copied to the hard drive of the workstation. To avoid data inconsistencies, only one instance of the program should be started per database. If the database from the first instance is copied to hard disk then the databases for all other instances are loaded into main memory.

All changes to objects in the program are made in the internal database. Rights, formatting rules and side-effects of the Customizer are taken into consideration. Changes made by the user are recorded in a change log. If the internal database is stored on the hard drive, all additional changes are also logged to this database. This means that you can restore to the last working state after connecting to the database if the program crashes.

When the data is transferred all recorded changes are made to the One Identity Manager database. This is done at object level so that, for example, processes are generated and formatting rules are observed. The principle "last writer wins" applies here as opposed to the previous object processing. That means that changes have been made by a user to object properties in the mean time are overwritten.

Depending on the program settings, the internal database is deleted from the hard drive when the program has finished. This means that all the data has to be loaded from the One Identity Manager database when the program is restarted. If the internal database is not deleted when the program finishes, the next program start up can be accelerated because only changes from the One Identity Manager that is connected have to be loaded.

Menu Items

Table 2: Meaning of Items in the Menu Bar

Menu	Menu Item	Meaning	Key Combination
Database	New connection...	Creates a new database connection.	Ctrl + Shift + N
	Save to database...	Displays the change log. Changes to the data can be saved to the One Identity Manager database.	CTRL + SHIFT + S
	Reload data	Data is reloaded from the One Identity Manager database.	
	Compile database...	Starts the Database Compiler.	CTRL + SHIFT + B
	Change management...	Opens a dialog box for editing change labels.	
	Check data consistency...	Opens the Consistency Editor. This item is available when the logged in user is authorized to use this functionality.	
	Run SQL editor...	Starts the SQL editor. This item is available when the logged in user is authorized to use this functionality.	
	Change password...	Changes current user's password.	
	Settings...	For configuring program settings.	
Exit		Exits the program.	Alt + F4

Menu	Menu Item	Meaning	Key Combination
View	Navigation	Activates the navigation view.	CTRL + Q
	Task	Shows/hides task field.	Ctrl + T
	Error log	Shows/hides the error log.	Ctrl + E
	Object import	Shows/hides the view for objects to be imported.	CTRL + I
	Search	Shows/hides search dialog box.	CTRL + SHIFT + F
	Close current document	Closes the current document.	
	Close all documents	Closes all documents that are open in the document view.	
	Activate document	Shows all open documents in a selection list.	
	Layout	Restores the default layout of the program's graphical interface. You can save layouts and load them again.	
	Enable quick edit mode	Activates/deactivates quick edit mode.	
Help	Community	Opens the One Identity Manager community website.	
	Support Portal	Opens the One Identity Manager product support website.	
	Training	Opens the One Identity Manager training portal website.	
	Online documentation	Opens the One Identity Manager documentation website.	
	Search...	Opens the search dialog box.	
	Help for the Designer	Opens program help.	F1
		 NOTE: Specific help information is available for each editor.	
	transport history	Display a chronological list of migration, imports and exports of transport packages	
	Info...	Shows the version information for program.	

Table 3: Functions in the Standard Toolbar

Icon	Meaning
	Saves changes to the One Identity Manager database.
	Shows the previous object in the order that the forms were viewed (object history).
	Shows next object in the order they have already been viewed (object history).
	Database column help. Clicking on the help icon changes the cursor into the help icon. Then when you click on a column description, tips for using the column are displayed in the form of tooltips.
	Prints the edit view. Printer settings are configured in the configuration menus.

Views in the Designer

The following views are displayed in the Designer: There are several editors used for editing data. Their functionality and methods are tailored to the different configuration tasks.

Table 4: Designer Views

View	Description
Navigation Overview	The navigation structure of the user interface is hierarchical and allows users to drill down to the selection of an object definition. The top level of the hierarchy is used to classify One Identity Manager data into specified categories.
Document view	Overview forms and selected editors are displayed in the document view. When an object is selected in the navigation view, the corresponding overview form is displayed. TIP: If quick edit is enabled, you skip the overview and go straight to the first editor available for the object.
Task view	When an object is selected in the navigation view, the available editors are displayed together with their executable tasks. The relevant editor is opened in the document view when the object is selected.
Error log	The program's error log displays all warnings and error messages that have occurred since the program started up. When the Designer restarts, the error log is reinstalled.
Change log	Changes made by the user are recorded in the change log.
Change label	In this view, change labels are created and edited.

Related Topics

- [The Designer Editors on page 42](#)
- [Making and Committing Changes to Objects on page 41](#)
- [Working with Change Labels on page 69](#)

Customizing Program Settings

To change the program settings

- Select **Database | Settings...** from the menu.

NOTE: General configuration setting are preset in the configuration file `Designer.exe.config`. In addition, globally valid configuration settings are defined using a configuration file in One Identity's own format. The configuration files are stored in the program directory.

User settings

Enter the program settings for the user on the **User** tab. These settings are stored in the One Identity Manager database user configuration.

Table 5: User Settings

Setting	Meaning
Clear local cache	Click this button to empty the local cache directory <code>%LocalAppData%\One Identity\One Identity Manager\Cache</code> .
Show balloon help	Specifies whether speech bubbles, which provide information about program functionality, are shown in the program.
Show large images in the navigation	Specifies whether large or small icons are shown next to the categories in navigation.
Show additional icons	Specifies whether icons are shown in the task list in addition to the descriptions.
Use single clicks	Specifies whether objects are selected from the result list with a double click or a single click.
Visible	Number of categories shown in the navigation at start up. Changes become

Setting	Meaning
root nodes	effective after a restart.
Enable quick edit mode at program start up	Specifies whether quick editing is enabled. By default, an object's overview form is displayed first. You can, however, configure the program to show the edit form for the object first, which allows faster editing. In order to do this quick edit mode has to be enabled. Quick edit mode is indicated by an additional icon in the program's status bar.
Show "Getting Started"	Depending on the setting, the category Getting Started is shown or hidden.
Show system data	By setting this option you display comprehensive system information such as system tables, script occurrences, preprocessor dependencies.
Enable list limit	Specifies whether the number of elements shown in the result list and list items in controls should be limited.
Use system settings	If the option Enable list limit is set, the number of elements has to be entered. There is a choice between the global system setting or the users own setting.
Objects	Number of object for your personal list limit. If the number of results is greater than the defined number a filter dialog is opened.
Form history length	Number of forms available to browse through in the form history. You find the form history in the menus attached to the <Back> and <Forward> buttons in the main toolbar.
Search history	Number of entries available in the search history.

General Program Settings

Enter general program settings on the **Application** tab. These settings are saved in the registry database on the workstation.

Table 6: General Program Settings

Setting	Meaning
Culture	Language selection. The initial program login uses the system language for the user interface. Changes to the language settings take effect after the program has been restarted. The language is set globally for all One Identity Manager programs which means that the language setting does not have to be configured for each program individually.

Setting	Meaning
Show additional navigation information	<p>If this option is set, additional navigation information for separate interface components is shown.</p> <p>NOTE: The option is not saved permanently. It has to be reset each time the program is started.</p>
Load all system data at program startup	<p>If this option is not enable, only the tables that are absolutely necessary are loaded when the Designer starts. The rest of the table are loaded in background and the user can already start using the program. The progress of the filling procedure is shown in the program's status bar. If this option is enabled, all tables are loaded when the program starts. The user cannot start using the program until all the tables have been loaded. The changes take effect once the Designer has restarted.</p>
Load BLOB fields from database at program start	<p>If this option is not enabled, the contents of the binary fields is not loaded until needed. If the option is enable, this data is already loaded at program startup. The means that program startup takes longer. The changes take effect once the Designer has restarted.</p>
Save database locally	<p>If this option is enable the internal database is not deleted when the program ends. This accelerates restarting the program the next time since only the changes connected with the One Identity Manager database need to be reloaded. If the option is not enabled the internal database is deleted from the hard disk when the program ends. This means that all the data has to be loaded from the One Identity Manager database when the program is restarted.</p>
Use RAM to store system data (no crash recovery)	<p>If this option is not enable, the internal database is saved on the workstation hard disk. If the option is enabled, the internal database is loaded into the workstation's RAM. In this case the database cannot be restore if the program crashes. The changes take effect once the Designer has restarted.</p>
Database directory	<p>If the option Save database locally is set, the database is stored in the directory %LocalAppData%\One Identity\One Identity Manager\Designer\Cache. You save the database somewhere else by selecting another database directory.</p>

Related Topics

- [General Information about the Program on page 27](#)
- [One Identity Manager Configuration Files on page 655](#)

Using Help

You can open the general help for Designer by select **Help** from the menu or with the **F1** key. The **Help** menu offers specific help information for each editor.

In addition, a Designer help is available for individual properties of the currently displayed object. Clicking on the help icon  changes the cursor into the help icon. Then when you click on a column description, tips for using the column are displayed in the form of tooltips.

Limiting List Sizes

Table 7: Configuration Parameter for Limiting Results

Configuration Parameter	Effect
Common\DBConnection\ListLimit	This configuration parameter specifies the number of list entries above which the filter request becomes effective
Common\DBConnection\WebListLimit	This configuration parameter specifies the number of list entries above which the filter request in the web front becomes effective.

A filter dialog box is implemented in the to limit the number of elements displayed in the result list or in control elements with list values (for example, in menus with lists). If the number of results exceeds the limit, a filter dialog box opens.

You can specify the number of entries above which the filter dialog box comes into effect in the configuration parameter "Common\DBConnection\ListLimit". In addition, the current user can use the system setting or enter their own limit.

To use the system settings

1. Select **Database | Settings...** from the menu.
2. Select the option **Enable list limit** on the **User** tab.
3. On the **User** tab, enable the option **Use system settings**
This setting applies the global value specified in the configuration parameter "Common\DBConnection\ListLimit".
4. Click **OK**.

To use a personal limit

1. Select **Database | Settings...** from the menu.
2. Select the option **Enable list limit** on the **User** tab.

3. Disable the option **Use system settings** on the **User** tab and enter the number of object you want to list in the **Object** box.

The personal limit overwrites the global value in the configuration parameter. Your personal limit is saved in the user configuration.

4. Click **OK**.

Related Topics

- [Customizing Program Settings on page 31](#)
- [Filtering Lists on page 35](#)
- [Using User Defined Filters for Searching on page 38](#)
- [Searching for List Entries on page 36](#)
- [Using the Designer's Own Full-Text Search on page 37](#)

Filtering Lists

You can use the filter dialog box to limit the entries in a list though defined filter criteria. A filter remains in use until you reset it. The settings in the filter dialog are saved in the user configuration.

If you want start particular search inquiries more often, use the database search function.

To limit a result set

1. Enter a **Filter criteria**.

You can use * (asterisk) as a wildcard. The filter is not case-sensitive.

2. Enable the properties to use as filter conditions under **Apply to**.
3. Enable the option **Only show current assignment**, to limit the list to objects that fulfill the filter condition and are already assigned to the base object.

This option is only available for lists with object assignments.

4. Select one of the following actions:

- Press **Apply** to filter the results list.
- Press **Show all** to show all the objects regardless of the set filters.
- Press **Cancel** to exit the query. No objects are shown.

Example

This filter condition is used to search for all parts of an employee's full name. This is put together in the following manner:

Full name

Last name, first name

Example:

Miller, Max
 Miller-Mayer, Max
 Macmiller, Max

Filter condition	Description	Find, according to example
Miller or *Miller*	Finds all employees whose full name contains the string 'Miller'.	Miller, Max Miller-Mayer, Max Macmiller, Max
Miller*	Finds all employees whose full name begins with the 'Miller'.	Miller, Max Miller-Mayer, Max
*Miller	Finds all employees whose full name ends with the 'Miller'.	No entry

Related Topics

- [Using User Defined Filters for Searching on page 38](#)
- [Searching for List Entries on page 36](#)
- [Using the Designer's Own Full-Text Search on page 37](#)

Searching for List Entries

Use the search dialog to search for entries within a list.

To search in a list

1. Open the search dialog with **Search...** in the context menu, with  or **CTRL + F**.
2. Enter the **Search term** or select a previous one from the list using the arrow button.
3. Set the option **Case sensitive** if required.
4. Start the search with **Search** or **Enter**.
5. Use **F3** to continue searching.
6. End the search with **Esc**.

Table 8: Shortcuts for the Search Dialog Box

Shortcut	Action
Ctrl + F	Open search dialog box.

Shortcut	Action
Enter	Start search.
Esc	End search.
F3	Search next.

For a wider ranging search, use the database search function.

Related Topics

- [Using the Designer's Own Full-Text Search on page 37](#)
- [Using User Defined Filters for Searching on page 38](#)

Using the Designer's Own Full-Text Search

Use full-text search to look for entries within the internal Designer database. You will find the full text search on its own menu bar in the Designer.

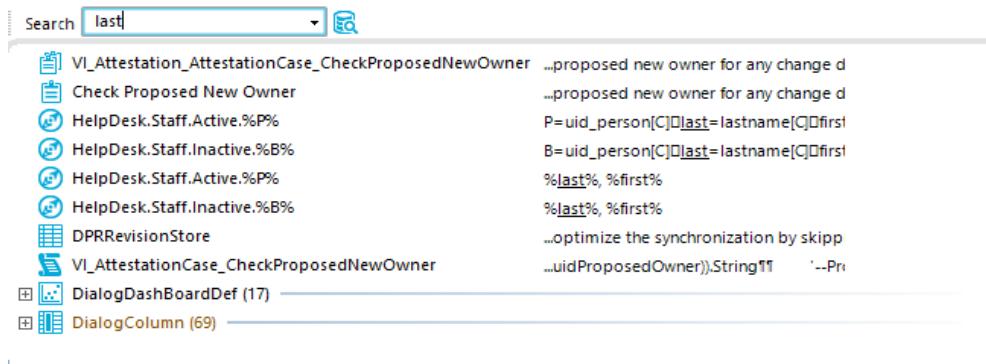
To search for a term

- Enter your search term in the **Search** text box.

You can enter more than one partial terms. Use of wild cards (*) is permitted. Case sensitivity is not taken into account. Entries are searched for that contain all the partial terms given.

Entries (objects) are already displayed while the search term is being entered.

Figure 6: Displaying the Source

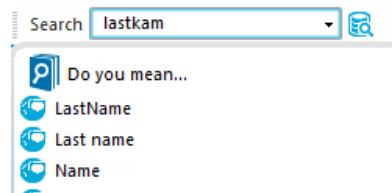


- The icon to the left of the entry shows the entry's object type (table), e.g. a process, a table or a menu item.

- The source that the object is extracted from is shown to the right of the entry. The search term is highlighted by an underline.
- Database tables, object relations and the exact source are also displayed in a tooltip.
- Double click on an entry to jump to the corresponding object.

If there is no entry found for a search term, suggestions are made that you can accept by double clicking with the mouse.

Figure 7: Suggestions for Search Terms



If you have selected an entry, the search term is added to the search history and is therefore available for further searches.

- Open the search history with the arrow in the **search** text box.
- When you select an entry, all available sources are shown.

The number of entries in the search history depends on your program settings.

Figure 8: Search History



TIP: Update the full text catalog of the Designer database if you need to include objects in the search that have been edited after the program started, like new processes or column names. You do this with the **Update index** item, in the search history.

Related Topics

- [Searching for List Entries on page 36](#)
- [Using User Defined Filters for Searching on page 38](#)

Using User Defined Filters for Searching

It is possible to limit the number of list entries in certain Designer editors by using a filter. After creating a filter, the filter conditions are immediately implemented on the set of result currently displayed. A filter remains in use until you reset it.

- Ad hoc filter

Ad hoc filters are used for a one-off reduction of list entries. These filters are not saved and are applied to the data immediately.

- Permanent filter

A permanent filter is recommended if you want to reuse it more frequently. Permanent filters are saved in the user configuration and therefore are always available for use.

Detailed information about this topic

- [Using an Ad Hoc Filter on page 39](#)
- [Using a Permanent Filter on page 39](#)

Using an Ad Hoc Filter

Ad hoc filters are used for a one-off reduction of list entries. These filters are not saved and are applied to the data immediately.

To use an ad hoc filter

- Select **Filter | Define filter...** in the menu or use the icon .

This opens the wizard for creating database queries. The wizard helps you to formulate a condition (where clause) for database queries. The complete database query is composed internally. It always refers to the database table that is specified when you start the wizard.

IMPORTANT: Enter the condition for limiting the result set in SQLite notation. The condition is defined as a valid where clause for database queries. The condition relates to the selected database table found that is determined when the editor starts.

For more information, see the One Identity Manager User Guide for One Identity Manager Tools User Interface and Default Functions.

Related Topics

- [Using a Permanent Filter on page 39](#)

Using a Permanent Filter

Permanent filters are saved and are therefore always available for use. There is a wizard to help you maintain permanent filters.

Table 9: Meaning of the Icons in the Toolbar

Icon	Meaning
	Adds a new filter.
	Edits filter name.
	Deletes filter.
	Switches between displaying database query input in SQLite notation or in the wizard.
	Tests filter.

To create a permanent filter

1. Select the menu **Filter | Manage filters...** or use the arrow next to . This opens the wizard for managing filters.
2. Select the icon and enter the name of the filter on the left-hand side of the wizard. These names are also used for the entries in the editor menus and can therefore always be selected for use.
3. Enter the conditions for limiting the number of results on the right-hand side. The condition is defined as a valid where clause for database queries. You can enter the condition directly in SQLite notation or by using the database query wizard. Use the icon to change between input methods.

IMPORTANT: Enter the condition for limiting the result set in SQLite notation. The condition is defined as a valid where clause for database queries. The condition relates to the selected database table found that is determined when the editor starts.
4. Test the condition using the icon. After the test is completed, a test report is displayed. All the list items are shown that meet the condition. A summary of the test status is also shown.
5. Click **Ok** to save the filter.

For more information, see the One Identity Manager User Guide for One Identity Manager Tools User Interface and Default Functions.

Related Topics

- [Using an Ad Hoc Filter on page 39](#)

Making and Committing Changes to Objects

All changes to objects in the program are made in the internal database. Rights, formatting rules and side-effects of the Customizer are taken into consideration. Changes made by the user are also recorded in a change log.

To display change data

- Select **Database | Commit to database....**

The entries in the change log are grouped by editor. Actions (add, change and delete) that have been executed and changes to the object's properties with old and new values are displayed in the log.

Table 10: Icons in the Change Log

Icon	Description
	Adds the object.
	Deletes the object.
	The object has been changed.

The following functions are available in the change log:

- Enable or disable changes

You can disable individual changes in the change log. These changes are not transferred to the One Identity Manager database when the data is saved. You can however, re-enable changes at a later date. You can enable or disable each change using the corresponding icon in the change log toolbar.

- Assign change labels

It is also possible to create a group of all the changes in the change log, under one label.

- Select the change label from the **Change label** pop-up menu before you save the changes.
- Use the context menu **Special change label** in the change log to assign single changes to a specific change label.

Once the changes are saved in the One Identity Manager database, the objects that are effected are given a change label.

- Save changes

All changes are saved to the One Identity Manager database as specified. This is done at object level so that, for example, processes are generated and formatting rules are observed. The principle "last writer wins" applies here as opposed to the

previous object processing. That means that changes made by a user to object properties in the mean time are overwritten.

Table 11: Functions in the Change Log Toolbar

Icon	Description
	Enables change/section.
	Ignores change/section.
	Edit a change label.
	Specifies a default change label. This change label is used for all subsequent changes.

Related Topics

- [Working with Change Labels on page 69](#)

The Designer Editors

The Designer provides various editors for the One Identity Manager system configuration. The functionality and the mode of operation of the editors depends on the different configuration requirements. When an object is selected in the navigation view the editors that can be used are displayed together with the executable tasks.

Table 12: Designer Editors

Editor	Description
Object Editor	The Object Editor is provided for editing single objects. All properties of an object are represented in table form and can be edited depending on the permissions situation. In addition, properties such as edit permissions and column definitions are shown. For more information, see Working with the Object Editor on page 44.
List Editor	The List Editor is used to display result lists and to quickly edit objects and object relations. Object properties are displayed in tabular form but there is no additional information about each property as in the Object Editor. For more information, see Working with the List Editor on page 46.
User & Permissions Group Editor	The User & Permissions Group Editor is the tool that creates and edits permissions groups and system users. Each employee that logs onto One Identity Manager tools requires a system user ID. More than one employee can work with one system user. The user interface and the system user permissions are loaded during the One Identity Manager tools login. For ease of administration, system users are grouped into permissions groups. Thus the user interface and the permissions structure

Editor	Description
	is controlled over the system user's memberships in permissions groups. For more information, see Working with the User & Permissions Group Editor on page 219.
User Interface Editor	The One Identity Manager is used for editing User Interface Editor tools navigation. The navigation defines specific entry points into the tool's user interface and controls the user oriented navigation down to the selection of an object in the result list. You can set up the structure of the user interface navigation through a menu. There are different types of menu items with specific uses. You can design a multifaceted navigation by combining different types of menu items. For more information, see Working with the User Interface Editor on page 254.
Overview Form Editor	Use the Overview Form Editor to set up overview forms for One Identity Manager tools. For more information, see Working with the Overview Form Editor on page 298.
Permissions Editor	The Permissions Editor is used to edit the permission or rights structure for permissions groups and system users. Each permissions group and even each system user can be given permissions for the table and columns in the One Identity Manager database. The permissions for the One Identity Manager database tables are issued depending on their membership in permissions groups when a system user logs into One Identity Manager tools. In this way, system user's access to tables and individual columns in the data can be controlled. For more information, see Working with the Permissions Editor on page 235.
Process Editor	The principle of the One Identity Manager's functionality allows the assignment of flexible actions and sequences of specific events. For example, the steps that need to be executed in order to add a user account to the database can be described in the form of a workflow. In this case, each action is represented by a process step and workflows are transformed into processes by linking the process steps together. The Process Editor is the tool with which the process sequence are defined and modified in the One Identity Manager, events assigned and linked. For more information, see Working with the Process Editor on page 385.
Job Server Editor	You use the Job Server Editor to edit the Job server attributes and the One Identity Manager Service configuration file. For more information, see Working with the Job Server Editor on page 338.
Language Editor	The One Identity Manager displays information on a language dependent basis. You can use this to edit captions for the One Identity Manager tool user interfaces in different languages. The One Identity Manager default installation is supplied in the languages "English - United States [en-US]" and "German - Germany [de-DE]". You can use other languages if required. To do this it is advisable to translate the required text before starting to use the One Identity Manager. There is a Designer in the

Editor	Description
	Language Editor to help you do this. For more information, see Working with the Language Editor on page 332.
Schema Editor	The Schema Editor displays an overview of the entire One Identity Manager database model. Schema Editor is used to customize table and column definitions to suit the customer. For example, you can define formatting rules or formatting scripts with the Schema Editor. For more information, see Working with the Schema Editor on page 165.
Configuration Parameter Editor	The Configuration Parameter Editor provides an overview of all the configuration parameters in the One Identity Manager and their current values. You should check and if necessary adjust the configuration parameters before the systems goes live. For more information, see Working with the Schema Editor on page 165.
Script Editor	Scripts are used in the One Identity Manager to check values in columns or to start events such as create, modify or delete objects. You can use the Script Editor to create, edit and test customer specific scripts. For more information, see Working with the Schema Editor on page 165.
SQL Editor	You can use the SQL Editor to run database queries against the internal Designer's SQLite database. For more information, see Working with the SQL Editor on page 49.
Consistency Editor	Use the Consistency Editor to analyze database object for data consistency. A number of tests are offered to test the database and if necessary run a repair. For more information, see Working with the SQL Editor on page 49.
Mail Template Editor	Use the Mail Template Editor to create and edit mail templates. Mail templates are used to generate email notifications, for example, notifications from process handling, attestation or IT Shop request statuses. Mail text is defined in several languages in the mail template. This ensures that the language of the recipient is taken into account when the email is generated. For more information, see Working with the SQL Editor on page 49.

Working with the Object Editor

The Object Editor is provided for editing single objects. All properties of an object are represented in table form and can be edited depending on the permissions situation. In addition, properties such as edit permissions and column definitions are shown.

Menu Items

The following items are added to the menu bar when the editor starts.

Table 13: Menu Items Added by the Editor

Menu	Menu Item	Meaning
Object	New	Creates a new object.
	Save	Saves all changes to an object.
	Delete	Deletes current object.
	Discard	Returns object to previous state.
	Reload object	Updates the object view.
Options	Group view	The object properties are displayed as groups.
	Column names	Column display text is shown. If the option is not enabled, the technical names according to the One Identity Manager schema are shown.
	Primary key	Show/Hide primary keys.
	Extended	Show/hide the extended column properties.
Help	Object Editor help	Opens the editor help.

Table 14: Meaning of Toolbar Icons

Icon	Meaning
	The object properties are displayed as groups.
	Displays caption/technical name depending on the One Identity Manager schema.
	Shows/Hides primary keys.
	Shows/hides the extended column properties.
	Creates object.
	Saves changes.
	Deletes object.
	Resets object to previous state (lose changes).
	Updates object view.

Multiple Object Edit

It is possible to edit more than one object of the same object type at the same time in the Object Editor.

1. Select entries in the result list with **SHIFT + CLICK** or **CTRL + CLICK**.
2. Open the editor from the task view,

The task name contains the number of selected objects. Input fields that have different values are marked in color. The values entered and saved in the fields are stored for all objects.

Working with the List Editor

The List Editor is used to display result lists and to quickly edit objects and object relations. Object properties are displayed in tabular form but there is no additional information about each property as in the Object Editor.

Menu Items

The following items are added to the menu bar when the editor starts.

Table 15: Menu Items Added by the Editor

Menu	Menu Item	Meaning
Object	New	Creates a new object.
	Delete	Deletes current object.
	Reload object	Updates the object view.
Filter	Define filter...	Opens a dialog window for creating an ad hoc filter.
	Delete filter	Deletes the filter.
	Manage filters...	Opens a dialog window for creating permanent filters.
View	Properties	Shows/hides the edit view.
	Select columns...	Opens a dialog window for selecting columns to be displayed in the list.
	Select table relations...	Opens a dialog box for selecting object relation to view.
Help	List Editor help	Opens the editor help.

Table 16: Meaning of Toolbar Icons

Icon	Meaning
	Creates object.
	Deletes object.
	Copies object to the clipboard.
	Inserts object or relation from the clipboard. Next to the icon there is a menu to select the type from.
	Updates object view.
	User defined filter for displaying objects.
	Resets custom filter.

Functions in the Result List

The entries in the result list of the List Editor are displayed with a valid name tag. If an object is selected in the result list, the object properties and object relations are shown in the edit view. You can limit the number of entries in the result list by setting a filter. For this you can use an ad hoc filter or a permanent filter.

Table 17: Entries in the Result List Context Menu

Context Menu Item	Meaning
New	Inserts a new object.
Delete	Deletes selected object.
Copy	Copies the selected object into the clipboard.
Paste	Inserts the copied object from the clipboard.
Properties	Displays the object properties of the selected entry.
Select columns...	Opens a dialog window for selecting columns to be displayed in the list.
Navigation	Shows all other editors that can be used with the selected object.

TIP: Click on a column header in the result list to sort the table entries by that column.

TIP: You have the possibility to show other properties in the result list.

Related Topics

- [Using User Defined Filters for Searching on page 38](#)
- [Result List Configuration on page 48](#)

Result List Configuration

To display object properties

1. Select **Select columns....**
2. You specify which object properties should be additionally shown in which order in the result list. You can also enter the column width and alignment of the column description.
3. Click **OK**.

Table 18: Meaning of Toolbar Icons

Icon	Meaning
<input checked="" type="checkbox"/>	Inserts column in view.
<input type="checkbox"/>	Removes column from view.
	Moves column up.
	Moves column down.
	Specifies how columns are labeled. If selected the column names are displayed. If this option is not active, the identifiers from the data model are shown.

Multiple Editing of List entries

It is possible to edit more than one object of the same object type at the same time in the List Editor.

- Select entries in the result list with **SHIFT + CLICK** or **CTRL + CLICK**.

Entries in the list that have different input are specially labeled in the edit view. When one field is edited and saved, all the other objects are given the same value.

Displaying Object Relations

To display object relations

1. Select **View | Select table relations...** in the menu.
2. You can specify which object relations are displayed in the List Editor.

Table 19: Items in the Dialog Window Context Menu

Context Menu Item	Meaning
Show relation	Shows the object relation edit window in the edit view..
Hide relation	Hides the object relation edit window in the edit view.

3. Click **OK**.

Each relation is shown in its own edit window. You can change the assignment with the mouse (double click on icon) or the context menu

Table 20: Meaning of Icons in the View

Icon	Meaning
	Object is assigned to the selected base object.
	Object is not assigned to all selected base objects (multiple entry editing).
	Object is not assigned to an object.

Table 21: Items in the View's Context Menu

Context Menu Item	Meaning
Assign	Assign object to the selected base object.
Remove	Remove assignment of object to base object.
Properties	Displays the object properties of the selected entry.
Navigation	Shows all other editors that can be used with the selected object.

Working with the SQL Editor

You can use the SQL Editor to run database queries against the internal Designer's SQLite database. This internal database contains the schema and the files from the system components of the connected One Identity Manager database.

NOTE: The SQL Editor is only available if the current user is authorized to use the program function "Option to call the SQL editor in Designer to run SQL statements in the SQL database. (Designer_SQLEditor)".

To execute a database query

- Start the SQL Editor from the **Database | Run SQL Editor...** menu.

This opens in the SQL Editor in the Designer's document view.

- Enter your database query in SQLite notation in the query window. The editor supports highlighting.

- Run the database query using **F5**.

The result are displayed in the results window.

TIP: Click with the mouse in a column header to sort by the selected column.

TIP: Use the shortcut **CTRL + C** to copy lines or single entries into the clipboard. Use **SHIFT+ SELECT** or **CTRL + SELECT** to select more than one line in the table.

Menu Items

The following items are added to the menu bar when the editor starts.

Table 22: Menu Items Added by the Editor

Menu	Menu Item	Meaning
Edit	Undo	Restores state before last change.
	Redo	Restores state after last change.
	Cut	Deletes marked code but save in clipboard.
	Copy	Copies selected code to the clipboard.
	Paste	Inserts data stored in the clipboard from copy or cut.
	Delete	Deletes selected code.
	Reduce text indent	Reduces the indent of the selected code in the query window.
	Increase text indent	Increases the indent of the selected code in the query window.
SQL	Execute (F5)	Executes the query.
	Show as table	Switches output between text and table.
Help	SQL Editor help	Opens the editor help.

Table 23: Meaning of Toolbar Icons

Icon	Meaning
undo	Undoes last change.
redo	Redoes last change.
cut	Cuts selected code.
copy	Copies selected code into clipboard.
paste	Inserts code from clipboard.
delete	Deletes selected code.
reduce indent	Reduces the indent of the selected code in the query window.
increase indent	Increases the indent of the selected code in the query window.
text-table	Switches output between text and table.
execute	Executes the query.

Support for Scripting

This input field is used in the editors when the input data needs to have a specified syntax (for example SQL, XML, Visual Basic .NET). It has an advanced edit mode which provides additional actions.

To switch to advanced mode

- Use the key combination **CTRL + ALT + Enter** or a the button in the right bottom corner.

Figure 9: Directly Entering a Database Query

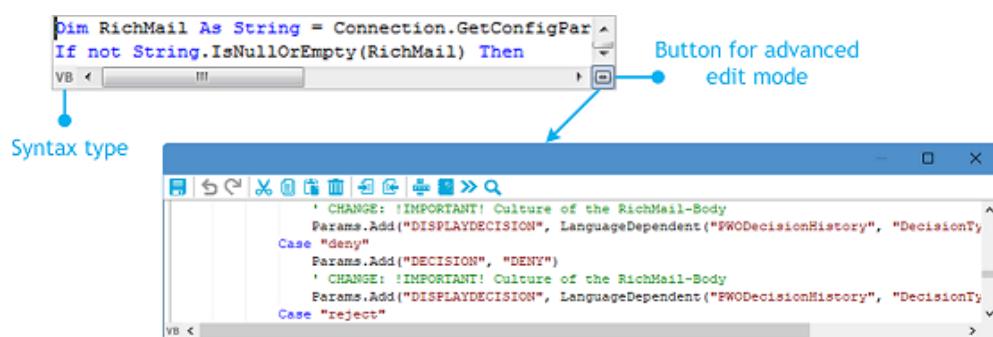


Table 24: Meaning of Icon in Advanced Edit Mode

Icon	Meaning
	Quitting advanced edit mode.
	Undoes last change.
	Redoes last change.
	Cuts selected code.
	Copies selected code into clipboard.
	Inserts code from clipboard.
	Deletes selected code.
	Decreases insert.
	Increases insert.
	Shows/hides line numbers.
	Inserts code snippet.
	Word wrap automatically.
	Search within code.

Additional input aids are provided for creating script code.

Syntax Highlighting

The input fields support syntax highlighting depending on the syntax type.

Auto-Completion

Auto-completion can be used when creating script code. The amount of scripted code to enter is reduced by displaying the names of properties or functions that can be used. Automatic completion is called with the key combination **CTRL + SPACE** at the appropriate point in the editor. The contents of the list is determined by the key words in the code.

Entering Code Snippets

Input fields that required data in VB.Net syntax support code snippets. Standard code snippets are available in the option "Visual Basic". The option "Object layer" contains special code snippets for the One Identity Manager object layer.

You can insert code snippets using the following options:

1. Using the icon

- Select the  in the menu bar.
- Select the option "Object layer" or "Visual Basic".
- Select the code snippet.

2. Using a shortcut

- Press the **F2** key.
- Select the option "Object layer" or "Visual Basic".
- Select the code snippet.

3. Using an aliases

- Enter an alias.
- Use **Tab** to insert the code snippet.

 **NOTE:** Remember that the connection name is case sensitive.

 **NOTE:** If you select the code snippet using a shortcut or the  icon, a short description and the shortcut name is displayed in a tooltip.

 **TIP:** You can use custom code snippets. To do this, create a directory `CustomSnippets` in the One Identity Manager installation directory to store the code snippets. Use Visual Studio documentation to develop your own code snippets.

Inputting Values using Dollar (\$) Notation

If you enter a "\$" in fields which expect VB.Net expressions, a wizard opens. All properties of the current object are displayed. You can also see a tooltip with a detailed description of the property. If you select an FK column, you can navigate to the columns of the associated table using the arrows keys. Exit the selection on the target column with **Enter** or by double-clicking. The complete \$ notation for your selection should now be shown in the input column. Use **Esc** or exit the input field to close the list without accepting any data.

Figure 10: Help List for Dollar Notation

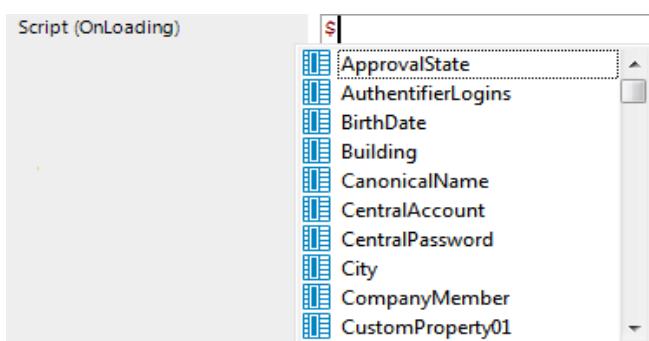


Table 25: Meaning of the Symbols used in the Help List.

Icon	Meaning
	Property of current object.
	Primary key (PK).
	Foreign key (FK).
	Dynamic foreign key
	Table
	Special properties
	Script

Table 26: Help List Functions

Shortcut	Action
Down arrow	Opens the help list.
Up arrow, down arrow	Swaps to next or previous entry respectively.
Left arrow, right arrow	Swaps from FK to parent object or back to the child object respectively.
Enter	Accepts the value in dollar notation.

Customizing the One Identity Manager Default Configuration

You can customize large parts of the One Identity Manager default configuration. For example, you can specify your own display names for columns or menu items or define your own templates and formatting rule for column values.

- Changes to data are labeled with the  icon in front of the modified value. As long as the changes have not been saved, you can restore them by clicking on the icon.
- Changes to the default configuration are labeled with the  icon in the Designer. To restore the default configuration, click on the icon.

If you customize a default configuration, the change is captured by a trigger and the default configuration is copied into a configuration buffer. You can retrieve changes from the configuration buffer and restore the default configuration in this way.

If the default configuration is changed by a service pack, a complete version upgrade or by loading a hotfix package during a One Identity Manager version upgrade, a check is made to see if it has already been customized. In this case, the modified default configuration is copied to the configuration buffer. This ensures that customizations do not go missing.

You can prevent individual properties from being overwritten by transports or normal editing using a lock.

For example, you might want to block processing:

- Configuration parameters and their values should not be overwritten when a test environment is transported to a productive system.
- Server configurations should neither be overwritten in the test environment nor the productive system during a transport.
- Passwords of administrative user accounts should neither be overwritten in the test environment nor the productive system during a transport.

To lock single properties for editing

1. Open the object you want to lock, in the Designer or Manager.
2. Click on the property name and select the context menu **Prohibit modification**.
The input field is locked and grayed-out.

3. To unlock the property again, click the property's name and select **Permit modification**.

Checking Data Consistency

The consistency check provides different tests for analyzing data objects and to ascertain the current state of their data. You can use your own tests as well as the predefined one and repair the data if necessary.

NOTE: You should run a consistency check at regular intervals or after major changes to the system configuration.

You can run consistency checks in the Manager and in the Designer. Database tests are run in their entirety in the Manager and the Designer. In the case of table tests and object tests in the Manager, the application model is tested and in the Designer, the system model is tested.

NOTE: The Consistency Editor is only available if the current user is authorized to use the function "Option to call a consistency check for a database (CommonConsistencyCheck)".

Detailed information about this topic

- [Starting the Consistency Check on page 58](#)
- [Recording Test Results on page 62](#)
- [Repairing Errors on page 62](#)

Working with the Consistency Editor

Use the Consistency Editor to test the consistency of your data. The editor is started from the program "Designer" and opens in the document view. Only additional Consistency Editor functions are described in the following.

Menu Items

The following items are added to the menu bar when the editor starts.

Table 27: Meaning of Items in the Menu Bar

Menu	Menu Item	Meaning
Consistency check	Run	Starts the consistency check.
	Cancel	Aborts the consistency check.
	Options...	Several test options are available to be set.
View	Error log	Shows/hides the error log.
Help	Consistency Editor	Opens the editor help.

Table 28: Meaning of the Icons in the Toolbar

Icon	Meaning
	Test options. Configure settings for the test.
	Stop. Stops the consistency check.
	Start. Starts the consistency check.

Starting the Consistency Check

NOTE: It is recommended to run consistency checks with an administrative system user.

Consistency checks of type "Object test" are always run in user context. If the user does not have any permissions for a certain object, errors may not be identified or repairing errors may fail.

To run a consistency check

1. Start the Consistency Editor in the using the menu item **Database | Check data consistency.....DesignerManager**
During start up, One Identity Manager schema table definitions are loaded from the related main database and database objects are made available for testing.
2. Specify the test options.
The test configuration dialog opens when the Consistency Editor is started. Enable the icon in front of the test name to include a test in the test run.
3. Start the consistency check. The following test procedures are available in the Consistency Editor for this:
 - Check all test objects
Start this test from the menu **Consistency check | Start**.

- Checking single test objects
Select the test objects you want from the list and start the test by right-clicking and pressing **Test**.
 - **TIP:** Use **SHIFT + CLICK** or **CTRL + CLICK** to select multiple items for testing.
- 4. Verify error output.
- 5. Repair errors if necessary.

Related Topics

- [Displaying Test Objects and the Test Status on page 59](#)
- [Specifying Test Options on page 60](#)
- [Recording Test Results on page 62](#)
- [Repairing Errors on page 62](#)

Displaying Test Objects and the Test Status

During Consistency Editor startup, the table definitions from the One Identity Manager schema are loaded from the connected database and the database objects are made available for testing. The database tables, the number of objects per table and the test status are displayed in the Consistency Editor's list view.

- **TIP:** Click with the mouse in a column header to sort by the selected column.

Table 29: List View Information

Column	Meaning
Target	Test object name.
Count	Total number of objects in the database table.
Verified	Test progress in percent.
error	The number of error that occurred during a consistency check.
Status	Current test status. The status is updated during the consistency check.

Table 30: Meaning of List View Icons

Icon	Meaning
	Test object is currently being test.
	Consistency check was successful for this Test object.
	Consistency check for this test object is complete but errors occurred.

Table 31: List View Context Menu Items

Context Menu Item	Meaning
Enable	Enables selected test object(s) for the period of the consistency check.
Disable	Disables selected test object(s) for the period of the consistency check.
Test	Starts execution of the consistency check for the selected test object(s).
Skip	Skip the test object during the consistency check.

Specifying Test Options

Before you run a consistency check you should set the test options you require. The test configuration dialog opens when the Consistency Editor is started.

- Enable the icon in front of the test name to include a test in the test run.

NOTE: You can change the test options at anytime using the menu item **Consistence check | Options....**

Figure 11: Specifying Test Options

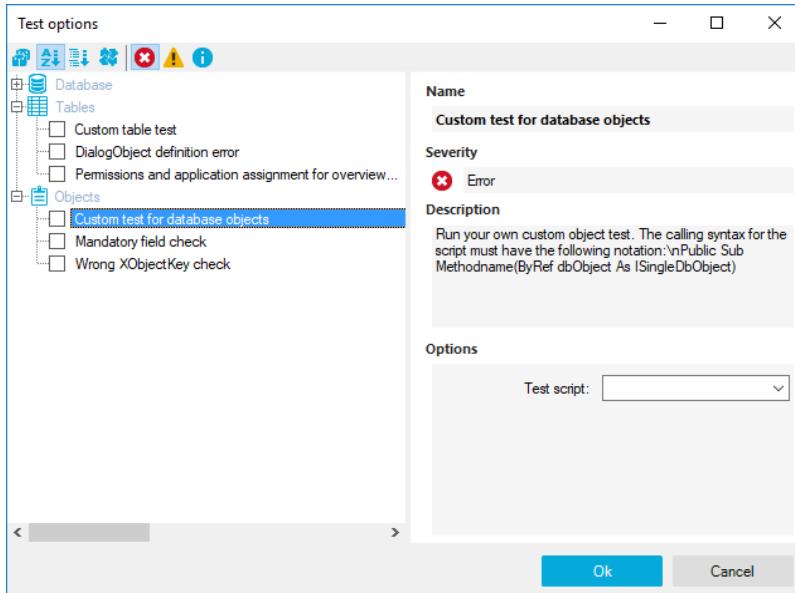


Table 32: Meaning of the Icons for the Test Options

Icon	Meaning
Database icon	Tests are grouped by themes.
Tables icon	Tests are grouped by types (database, tables , objects).
List icon	Tests are displayed as a list.
Module icon	Tests are grouped by module association.
Error icon	Tests are displayed with severity level "Error".
Warning icon	Tests are displayed with severity level "Warning".
Information icon	Tests are displayed with severity level "Information".

Test options are separated into test at database level, table level and object level. There are already predefined tests available. Use user defined tests to run your own tests. You can use the scripts from the script library for these tests. All the scripts from the library that use the following calling syntax are available:

Database test:

```
Public Sub Methodname (ByRef con As IConnection)
```

Table test:

```
Public Sub Methodname (ByRef dbTable As ITableDef)
```

Object test:

```
Public Sub Methodname (ByRef dbObject As ISingleDBObject)
```

Recording Test Results

During the consistency check, the number of tested objects and the test status is updated in the editor's list view. Once the test has completed, any error messages are outputted to the Consistency Editor error log.

Table 33: Meaning of Icons in the Error Log

Icon	Meaning
	Shows all error messages.
	Only shows errors in the selected objects list view.
	A full description of the error is shown in a separate window.
	Fixes the error.
	Saves the error messages in a log file.
	Deletes the error messages.

TIP: Double-click an entry and a dialog box appears with a detailed description of the error.

Related Topics

- [Repairing Errors on page 62](#)

Repairing Errors

NOTE: This Consistency Editor repair function is only available if the current user is authorized to use the function "Option to start automatic consistency check repair function (Common_ConsistencyCheck_Repair)".

The button **Repair** appears in the Consistency Editor error log if it is possible to repair the error automatically.

To repair incorrect data

- Select the error in the Consistency Editor log.

TIP: Use **SHIFT + CLICK** or **CTRL + CLICK** to select multiple entries for repair.

- Start the error correction with **Repair**.

The error is corrected directly in the main database connected to the . Subsequent data changes are made with the One Identity Manager Service.

NOTE: When you repair templates, dependent objects may also be changed when the template is rerun. In certain circumstances, this might result in a large number of objects being changed, saved and even processes being generated.

Compiling a One Identity Manager Database

The One Identity Manager database must be compiled after changes to configuration data. The database compilation is started immediately from the Configuration Wizard or the Database Transporter after a migration package or a customer's complete configuration package has been imported. Use the Database Compiler to compile the One Identity Manager database after importing hotfixes or when changes have been made to processes, scripts, formatting rules, object definitions, task definitions, and preprocessor relevant configuration parameters.

Detailed information about this topic

- [Compiling a Database with the Database Compiler on page 64](#)
- [Logging Compiler Errors and Warnings on page 66](#)

Compiling a Database with the Database Compiler

To compile a database, use the Database Compiler. You start the Database Compiler in the Designer.

NOTE: Before you begin the compilation, all the DBQueue Processor tasks have to be processed. If there are still outstanding tasks on the database, you are notified by the Database Compiler. In this case the compilation cannot go ahead.

To compile a database

1. Select **Database | Compile database...** in the Designer.
2. Enter which parts of the database should be recompiled.

TIP: Hide the other options using the arrow next to the option.

- a. To compile web services, select the **Web Services** option.

The One Identity Manager offers the option of linking in data that comes from different web service interfaces. The web service proxy code is stored in the database. The Database Compiler compiles the proxy code for all web services of a DLL and saves it in the database. When changes are made to proxy code the database needs to be compiled.

- b. To create type-safe classes for the data model, select the option **Typed wrappers**.

Type-safe classes are created from table and column definition that you can use in scripts. This checks whether the classes used in writing and compiling scripts are correct.

 **TIP:** Use this option to compile the database after extending the schema.

- c. To compile scripts from the script library, select the following items:

Table 34: Options for Compiling Scripts

Option	Description
Do not compile scripts	Scripts from the script library are not compiled.
Script without dependencies	Changes to scripts do not become active until the One Identity Manager tool is restarted.
Scripts including all dependencies	Recompiles the scripts and all dependencies (templates, methods, processes). This guarantees that the script changes are loaded and become effective immediately. One Identity Manager tools do not need to be restarted.

- d. To compile additional script expressions, select the options **Templates, methods, etc.** and specify which script expressions to compile in the options underneath. You can select from templates and formatting scripts, scripts for selecting tables, view and objects, task definitions, insert values and parameter scripts.
- e. To compile processes, select the option **Processes** and specify which processes to compile in the options underneath.

Table 35: Options for Compiling Scripts

Option	Description
All processes	All processes are compiled.

Option	Description
Changed processes	All processes that have changed since the previous compilation are compiled.
Selected processes...	Only processes for the selected objects are compiled.
To select objects separately	
<ol style="list-style-type: none"> 1. Click [...]. 2. Select changed processes, all processes or only selected processes (user defined). You can limit the preselection more. 3. Click OK. 	
<ol style="list-style-type: none"> <li value="4">f. Set the option Extract language dependent strings if you want text from scripts to be translated. <p>This creates templates for translation. Use the Language Editor to translate.</p>	
<ol style="list-style-type: none"> 3. To start compiling, click Next. Compiling may take some time. 4. Click Next. 5. Click Finish to quit the program. 	

Logging Compiler Errors and Warnings



NOTE: Any compiler errors and warnings are recorded during compilation. You can view compiler errors and warnings after compilation is complete.

- Use **Save log to file...** in the context menu to save the compilation log.
- Correct the error after compilation is finished.
- Compile the database again after correcting the errors. To do this, start compilation in Designer using **Database | Compile database....**

To display messaging during compilation and saving

- Select **Display** to display a message. This opens an error message window.

In addition, a more detailed description of the error is displayed. Configure the amount of information to be displayed using the options in the error message window. Open the configuration with the button and enable or disable the options you want.

Table 36: Options for Displaying Error Messages

Option	Meaning
Show previous errors	Specifies whether all previous errors that lead to the current error, should also be shown.
Show One Identity error numbers	Specifies whether internal error numbers are shown.
Show error positions	Specifies whether error position are also shown in the program code.
Wrap long lines	Specifies whether long error messages are wrapped.
Show only user relevant	Specifies whether all error messages or only message classified as "user relevant" are shown.
Show asynchronous calls	Specifies whether error messages in asynchronous method calls are shown.
Show crash report	Specifies whether error messages from the crash recorder are shown.

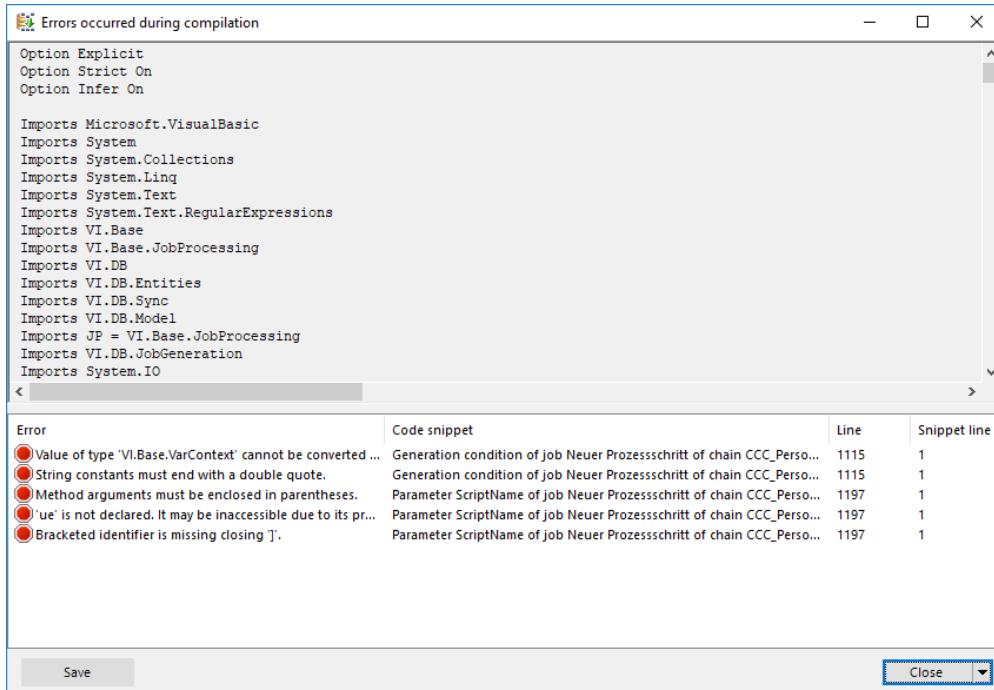
i | **NOTE:** The button **Send as mail** creates a new email message in the default mail program and copies over the error text.

- To save all the messages in a file, select the entry and select **Save log to file...** from the context menu.
- To add a message to the clipboard, select the entry and user **CTRL + C**.

If errors occur, these are shown immediately in a separate log window during compilation.

- If you double click on the error message with the mouse you jump to the corresponding line in the source code view (upper part of the window). You can only view the source code you cannot edit it.
- Use the **Save** button to save the error messages to file.
- Use the **Close** button to close the error log. Then the compilation continues.

Figure 12: Error Message Log



Working with Change Labels

Define change labels under which changes are grouped together in order to swap data between development and test databases as well as the productive database.

You can create and edit change labels in different One Identity Manager tools. The method is similar in all tools. Change labels are allocated using different methods depending on the One Identity Manager tool. Changes are normally allocated before or on saving the changes in the database.

Change labels are available in the program "Database Transporter" as export criteria for creating custom configuration packages.

Change labels can contain:

- Changes to single object properties at a specific point in time

When you create a custom configuration package, single object properties are added to the transport package. The properties contain the values given at the time they were added.

IMPORTANT: This is the default process when applying changes in the Designer. If you apply this method, be consistent in booking all changes to the change label when the object is saved. It is not possible to add changes of individual properties to the change label at a later date.

- Snapshots of an object at a specific point in time, optionally with dependent objects

When you create a custom configuration package, the object is added to the transport package with all its properties. The properties contain the values given at the time they were added.

- Reference to an object, optionally with dependent objects

When you create a custom configuration package, the object is added to the transport package with all its properties. The properties are determined at the time of export.

IMPORTANT: Object references cannot be grouped together with changes to single properties and snapshots of objects in the same change label. Book object references to their own change labels.

Detailed information about this topic

- [Creating Change Labels on page 70](#)
- [Displaying Contents of a Change Label on page 71](#)
- [Booking Changes to a Change Label Retrospectively on page 72](#)
- [Making and Committing Changes to Objects on page 41](#)
- [Transport by Change Label on page 550](#)

Creating Change Labels

You can create and edit change labels in different One Identity Manager tools. The method is similar in all tools and is explained in the following example from the Designer.

To create or edit change labels

1. in the Designer menu, click **Database | Change management....**
2. In the dialog box "Edit change labels...", select the icon  next to the list **Change labels**.
A dialog box for editing change labels is opened.
3. Create a new change label with the  button.
- OR -
Select a change label and open the edit view with the  button.
4. Enter the change label data.

Table 37: Properties of change labels

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	Option to enter a specific parent change label.
Status	Status of object changes, for example: development, test, production.
Status comments	Additional status comments
Comment	Additional information about changes made to a change label.

Property	Meaning
Label type	Label type for further classification. Label type "Change" is used by default.
Locked	Indicates if the change label is locked. If a change label is locked, you can no longer make changes to it.

5. Click the  button.

6. Click **OK**.

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

To delete a change label

1. In the Manager menu, select **Database | Edit change labels...**
2. Click the  button next to the **Change labels** list.
A dialog box for creating and editing change labels opens.
3. Select the change label you want to delete and click the  button.
4. Confirm the security prompt with **Yes**.
5. To close the dialog box, click **Cancel**.
6. To close the dialog box, "Edit change labels..." click **OK**.

Related Topics

- [Displaying Contents of a Change Label on page 71](#)
- [Booking Changes to a Change Label Retrospectively on page 72](#)

Displaying Contents of a Change Label

To display the contents of a change label

1. Select **Database | Change management...** from the menu in the Designer.
2. In the dialog box "Edit change labels...", select the change label you want from the **Change labels** list.
 - Under "Assigned changes" you can see all the changes made to individual object settings as well as all snapshots of objects.
 - All references to objects are shown under "Assigned objects".

Related Topics

- [Creating Change Labels on page 70](#)
- [Booking Changes to a Change Label Retrospectively on page 72](#)

Booking Changes to a Change Label Retrospectively

You can select individual objects and their dependencies from any objects in the database and book them to a change label. You can book objects as snapshots or references depending on which One Identity Manager tool it applies to.

- IMPORTANT:** Object references cannot be grouped together with changes to single properties and snapshots of objects in the same change label. Book object references to their own change labels.
- NOTE:** It is not possible to add changes of single properties to the change label at a later date.

In certain cases, it is necessary to add the dependent objects to the change label as well. For example, if processes are being transported, the dependent process steps, process parameters and events should also be transported. This is also true for approval policies, approval workflows, approval steps and approval procedures.

To book objects to a change label retrospectively

1. Select **Database | Change management...** from the menu in the Designer.
2. In the dialog box "Edit change labels...", select the change label using the **Change label** list.
3. Select the database table from the **Table** list from which you want to copy objects to the change label.
4. To limit the number of objects found
 - a. Click  next to the **Table** menu.
 - b. Enter a condition in the "Filter".
Enter the condition as a WHERE clause for a database query. You can enter the database query directly as in SQL or use the wizard, which you open by clicking on the  button next to the text box.
 - c. Click **Apply**.
5. To map dependent objects
 - a. Click  next to the **Table** menu.
This opens a separate selection box that displays the ChildRelation (CR), ForeignKey (FK) and many-to-many relations for the selected database table.
 - b. Under "Table relations" enable the desired table relations.
When you select and assign an object, the objects linked by these table relationships are marked with the same change label.

6. Select the desired objects in the "Objects list".
 - Click  to add all the selected objects to the change label as a reference.
 - Click  to add all the selected objects to the change label as a snapshot.
-  **TIP:** Multi-select objects using **SHIFT + CLICK** or **CTRL + CLICK**.

To remove objects from a change label

1. Select **Database | Change management...** from the menu.
 2. In the dialog box "Edit change labels...", select the change label using the **Change label** list.

All objects assigned to this change label are displayed.
 3. Select the objects you want to remove from the change label in the "Assigned changes" or "Assigned objects" list.
-  **TIP:** Multi-select objects using **SHIFT + CLICK** or **CTRL + CLICK**.
4. Click the  button to remove the objects from the change label.

Related Topics

- [Creating Change Labels on page 70](#)
- [Displaying Contents of a Change Label on page 71](#)

Basic System Configuration Data

The base data includes the main settings for configuring the . They are usually checked and customized on a one-off basis before the One Identity Manager goes into operation. The base data contains the database connection data, authentication module usage, languages used or the configuration parameter settings.

One Identity Manager Authentication Module

One Identity Manager uses different authentication modules for logging in to administration tools. Authentication modules identify the system users to be used and load the user interface and database resource editing permissions depending on their permission group memberships.

- **NOTE:** After initial schema installation, only the authentication modules "system user" and "ComponentAuthenticator" and role-based authentication modules are enabled in the One Identity Manager.
- **NOTE:** Authentication modules are defined in the One Identity Manager modules and are not available until the modules are installed.

The following authentication modules are available:

System user

Login Data	The system user's identifier and password.
Prerequisites	The system user with permissions exists in the One Identity Manager database.
Set as default	Yes
Single Sign-On	No

Front-end login allowed	Yes
Web Portal login allowed	No
Remarks	<p>The user interface and the write permissions are loaded through the system user.</p> <p>Data modifications are attributed to the system user.</p>

- **IMPORTANT:** The system user "viadmin" is supplied by default. The system user "viadmin" has a predefined user interface and has access rights to database resources. The interface and access rights for "viadmin" should not be used live or be modified, as it is a template system user and is overwritten by each schema update.
- **TIP:** Create your own system user with the appropriate permissions. This can be done on initial installation of the One Identity Manager database. This system user can compile an initial One Identity Manager database and can be used to log into the administration tools for the first time.

Employee

- **NOTE:** This authentication module is available if the module Identity Management Base Module is installed.

Login Data	Employee's central user account and password.
Prerequisites	<p>The system user with permissions exists in the One Identity Manager database.</p> <p>The employee exists in the One Identity Manager database.</p> <ul style="list-style-type: none"> • The central user account is entered in the employee's master data. • The system user is entered in the employee's master data. • The password is entered in the employee's master data.
Set as default	Yes
Single Sign-On	No
Front-end login allowed	Yes
Web Portal login allowed	Yes
Remarks	If an employee owns more than one identity, the configuration parameter "QER\Person\MasterIdentity\UseMasterForAuthentication" controls which

employee is used for authentication.

- If this configuration parameter is set, the employee's main identity is used for authentication.
- If the parameter is not set, the employee's subidentity is used for authentication.

The user interface and the write permissions are loaded through the system user that is directly assigned to the logged in employee.

Changes to the data are assigned to the logged in employee.

Generic single sign-on (role based)

 **NOTE:** This authentication module is available if the module Identity Management Base Module is installed.

Login Data	The authentication module uses the Active Directory login data of user currently logged in on the workstation.
Prerequisites	<p>The employee exists in the One Identity Manager database.</p> <p>The employee is assigned at least one application role.</p> <p>The user account exists in the One Identity Manager database and the employee is entered in the user account's master data.</p>
Set as default	No
Single Sign-On	Yes
Front-end login allowed	Yes
Web Portal login allowed	Yes
Remarks	<p>One Identity Manager searches for the user account according to the configuration and finds the employee assigned to the user account.</p> <p>If an employee owns more than one identity, the configuration parameter "QER\Person\MasterIdentity\UseMasterForAuthentication" controls which employee is used for authentication.</p> <ul style="list-style-type: none">• If this configuration parameter is set, the employee's main identity is used for authentication.• If the parameter is not set, the employee's subidentity is used for authentication.
	A dynamic system user determined from the employee's application roles. The user interface and the write permissions are loaded through this

system user.

Changes to the data are assigned to the logged in employee.

Modify the following configuration parameters in the Designer to implement the authentication module.

Table 38: Configuration Parameters for the Authentication Module

Configuration parameter	Meaning
QER\Person\OAuthAuthenticator	This configuration parameter specifies whether authentication through single sign-on is supported.
QER\Person\GenericAuthenticator\SearchTable	This configuration parameter contains the table in the One Identity Manager schema in which user information is stored. The table must contain a foreign key with the name UID_Person, which points to the table Person. Example: ADSAccount
QER\Person\GenericAuthenticator\SearchColumn	This configuration parameter contains the column from the One Identity Manager table (SearchTable), which is used to search for the user name of the current user. Example: CN
QER\Person\GenericAuthenticator\EnabledBy	This configuration parameter contains a pipe () delimited list of Boolean columns from the One Identity Manager table (SearchTable) enabled by the user account for the login.
QER\Person\GenericAuthenticator\DisabledBy	This configuration parameter contains a pipe () delimited list of Boolean columns from the One Identity Manager table (SearchTable) disabled by the user account for the login. Example: AccountDisabled

Employee (role based)

 **NOTE:** This authentication module is available if the module Identity Management Base Module is installed.

Login Data Employee's central user account and password.

Prerequisites The employee exists in the One Identity Manager database.

- The central user account is entered in the employee's master data.
- The password is entered in the employee's master data.

	The employee is assigned at least one application role.
Set as default	Yes
Single Sign-On	No
Front-end login allowed	Yes
Web Portal login allowed	Yes
Remarks	<p>If an employee owns more than one identity, the configuration parameter "QER\Person\MasterIdentity\UseMasterForAuthentication" controls which employee is used for authentication.</p> <ul style="list-style-type: none"> • If this configuration parameter is set, the employee's main identity is used for authentication. • If the parameter is not set, the employee's subidentity is used for authentication. <p>A dynamic system user determined from the employee's application roles. The user interface and the write permissions are loaded through this system user.</p> <p>Changes to the data are assigned to the logged in employee.</p>

Employee (dynamic)

 **NOTE:** This authentication module is available if the module Identity Management Base Module is installed.

Login Data	Employee's central user account and password.
Prerequisites	<p>The employee exists in the One Identity Manager database.</p> <ul style="list-style-type: none"> • The central user account is entered in the employee's master data. • The password is entered in the employee's master data. <p>The configuration data for dynamically determining the system user is defined in the application. Thus, an employee can, for example, be assigned a system user dynamically depending on their department membership.</p>
Set as default	Yes
Single Sign-On	No
Front-end	Yes

login allowed	
Web Portal	Yes
login allowed	
Remarks	<p>If an employee owns more than one identity, the configuration parameter "QER\Person\MasterIdentity\UseMasterForAuthentication" controls which employee is used for authentication.</p> <ul style="list-style-type: none"> • If this configuration parameter is set, the employee's main identity is used for authentication. • If the parameter is not set, the employee's subidentity is used for authentication. <p>The application configuration data is used to determine a system user, which is automatically assigned to the employee. The user interface and write permissions are loaded through the system user that is dynamically assigned to the logged in employee.</p> <p>Changes to the data are assigned to the logged in employee.</p>

User account

 **NOTE:** This authentication module is available if the module Identity Management Base Module is installed.

Login Data	The authentication module uses the Active Directory login data of user currently logged in on the workstation.
Prerequisites	<p>The system user with permissions exists in the One Identity Manager database.</p> <p>The employee exists in the One Identity Manager database.</p> <ul style="list-style-type: none"> • Permitted logins are entered in the employee's master data. The logins are expected in the form: domain\user. • The system user is entered in the employee's master data.
Set as default	No
Single Sign-On	Yes
Front-end login allowed	Yes
Web Portal login allowed	Yes
Remarks	All employee logins saved in the One Identity Manager database are found. The employee whose login data matches that of the current user is used for logging in.

If an employee owns more than one identity, the configuration parameter "QER\Person\MasterIdentity\UseMasterForAuthentication" controls which employee is used for authentication.

- If this configuration parameter is set, the employee's main identity is used for authentication.
- If the parameter is not set, the employee's subidentity is used for authentication.

The user interface and access permissions are loaded through the system user that is directly assigned to the employee found.

Data modifications are attributed to the current user account.

User Account (role based)

 **NOTE:** This authentication module is available if the module Identity Management Base Module is installed.

Login Data	The authentication module uses the Active Directory login data of user currently logged in on the workstation.
Prerequisites	<p>The employee exists in the One Identity Manager database.</p> <ul style="list-style-type: none">• Permitted logins are entered in the employee's master data. The logins are expected in the form: domain\user. <p>The employee is assigned at least one application role.</p>
Set as default	No
Single Sign-On	Yes
Front-end login allowed	Yes
Web Portal login allowed	Yes
Remarks	<p>All employee logins saved in the One Identity Manager database are found. The employee whose login data matches that of the current user is used for logging in.</p> <p>If an employee owns more than one identity, the configuration parameter "QER\Person\MasterIdentity\UseMasterForAuthentication" controls which employee is used for authentication.</p> <ul style="list-style-type: none">• If this configuration parameter is set, the employee's main identity is used for authentication.• If the parameter is not set, the employee's subidentity is used for authentication.

A dynamic system user determined from the employee's application roles. The user interface and the write permissions are loaded through this system user.

Data modifications are attributed to the current user account.

Account-based system user.

Login Data	The authentication module uses the Active Directory login data of user currently logged in on the workstation.
Prerequisites	<p>The system user with permissions exists in the One Identity Manager database.</p> <ul style="list-style-type: none">Permitted logins are entered in the system user's master data. The logins are expected in the form: domain\user.
Set as default	No
Single Sign-On	Yes
Front-end login allowed	Yes
Web Portal login allowed	No
Remarks	<p>All system user logins saved in the One Identity Manager database are found. The system user whose login data matches that of the current user is used for logging in.</p> <p>The user interface and the write permissions are loaded through the system user.</p> <p>Data modifications are attributed to the current user account.</p>

Active Directory user account

 **NOTE:** This authentication module is available if the module Active Directory Module is installed.

Login Data	The authentication module uses the Active Directory login data of user currently logged in on the workstation.
Prerequisites	<p>The system user with permissions exists in the One Identity Manager database.</p> <p>The employee exists in the One Identity Manager database and the system user is entered in the employee's master data.</p> <p>The Active Directory user account exists in the One Identity Manager</p>

	database and the employee is entered in the user account's master data.
Set as default	Yes
Single Sign-On	Yes
Front-end login allowed	Yes
Web Portal login allowed	Yes
Remarks	<p>The appropriate user account is found in the One Identity Manager database through the user's SID and the domain given at login. One Identity Manager determines which employee is assigned to the user account.</p> <p>If an employee owns more than one identity, the configuration parameter "QER\Person\MasterIdentity\UseMasterForAuthentication" controls which employee is used for authentication.</p> <ul style="list-style-type: none"> • If this configuration parameter is set, the employee's main identity is used for authentication. • If the parameter is not set, the employee's subidentity is used for authentication. <p>The user interface and access permissions are loaded through the system user that is directly assigned to the employee found. If the employee is not assigned to a system user, the system user is taken from the configuration parameter "SysConfig\Logon\DefaultUser".</p> <p>Data modifications are attributed to the current user account.</p>

NOTE: If the option **Connect automatically** is set, authentication is no longer necessary for subsequent logins.

Active Directory user account (role based)

NOTE: This authentication module is available if the module Active Directory Module is installed.

Login Data	The authentication module uses the Active Directory login data of user currently logged in on the workstation.
Prerequisites	<p>The employee exists in the One Identity Manager database.</p> <p>The employee is assigned at least one application role.</p> <p>The Active Directory user account exists in the One Identity Manager database and the employee is entered in the user account's master data.</p>

Set as default	Yes
Single Sign-On	Yes
Front-end login allowed	Yes
Web Portal login allowed	Yes
Remarks	<p>The appropriate user account is found in the One Identity Manager database through the user's SID and the domain given at login. One Identity Manager determines which employee is assigned to the user account.</p> <p>If an employee owns more than one identity, the configuration parameter "QER\Person\MasterIdentity\UseMasterForAuthentication" controls which employee is used for authentication.</p> <ul style="list-style-type: none"> • If this configuration parameter is set, the employee's main identity is used for authentication. • If the parameter is not set, the employee's subidentity is used for authentication. <p>A dynamic system user determined from the employee's application roles. The user interface and the write permissions are loaded through this system user.</p> <p>Data modifications are attributed to the current user account.</p>

NOTE: If the option **Connect automatically** is set, authentication is no longer necessary for subsequent logins.

Active Directory user account (manual input/role based)

NOTE: This authentication module is available if the module Active Directory Module is installed.

Login Data	Login name and password for registering with Active Directory. You do not have to enter the domain.
Prerequisites	<p>The employee exists in the One Identity Manager database.</p> <p>The employee is assigned at least one application role.</p> <p>The Active Directory user account exists in the One Identity Manager database and the employee is entered in the user account's master data.</p> <p>The domain for logging in are entered in the configuration parameter "TargetSystem\ADS\AuthenticationDomains".</p>

Set as default	Yes
Single Sign-On	No
Front-end login allowed	Yes
Web Portal login allowed	Yes
Remarks	<p>The user's identity is determined from a predefined list of permitted Active Directory domains. The corresponding user account and employee are determined in the One Identity Manager database, which the user account is assigned to.</p> <p>If an employee owns more than one identity, the configuration parameter "QER\Person\MasterIdentity\UseMasterForAuthentication" controls which employee is used for authentication.</p> <ul style="list-style-type: none"> • If this configuration parameter is set, the employee's main identity is used for authentication. • If the parameter is not set, the employee's subidentity is used for authentication. <p>A dynamic system user determined from the employee's application roles. The user interface and the write permissions are loaded through this system user.</p> <p>Data modifications are attributed to the current user account.</p>

Active Directory user account (manual input)

 **NOTE:** This authentication module is available if the module Active Directory Module is installed.

Login Data	Login name and password for registering with Active Directory. You do not have to enter the domain.
Prerequisites	<p>The employee exists in the One Identity Manager database.</p> <p>The Active Directory user account exists in the One Identity Manager database and the employee is entered in the user account's master data.</p> <p>The domain for logging in are entered in the configuration parameter "TargetSystem\ADS\AuthenticationDomains".</p> <p>The configuration data for dynamically determining the system user is defined in the application. Thus, an employee can, for example, be assigned a system user dynamically depending on their department membership.</p>

Set as default	No
Single Sign-On	No
Front-end login allowed	Yes
Web Portal login allowed	Yes
Remarks	<p>The user's identity is determined from a predefined list of permitted Active Directory domains. The corresponding user account and employee are determined in the One Identity Manager database, which the user account is assigned to.</p> <p>If an employee owns more than one identity, the configuration parameter "QER\Person\MasterIdentity\UseMasterForAuthentication" controls which employee is used for authentication.</p> <ul style="list-style-type: none"> • If this configuration parameter is set, the employee's main identity is used for authentication. • If the parameter is not set, the employee's subidentity is used for authentication. <p>The application configuration data is used to determine a system user, which is automatically assigned to the employee. The user interface and write permissions are loaded through the system user that is dynamically assigned to the logged in employee.</p> <p>Data modifications are attributed to the current user account.</p>

Active Directory user account (dynamic)

 **NOTE:** This authentication module is available if the module Active Directory Module is installed.

Login Data	The authentication module uses the Active Directory login data of user currently logged in on the workstation.
Prerequisites	<p>The employee exists in the One Identity Manager database.</p> <p>The Active Directory user account exists in the One Identity Manager database and the employee is entered in the user account's master data.</p> <p>The configuration data for dynamically determining the system user is defined in the application. Thus, an employee can, for example, be assigned a system user dynamically depending on their department membership.</p>
Set as default	No

Single Sign-On	Yes
Front-end login allowed	Yes
Web Portal login allowed	Yes
Remarks	<p>The appropriate user account is found in the One Identity Manager database through the user's SID and the domain given at login. One Identity Manager determines which employee is assigned to the user account.</p> <p>If an employee owns more than one identity, the configuration parameter "QER\Person\MasterIdentity\UseMasterForAuthentication" controls which employee is used for authentication.</p> <ul style="list-style-type: none"> • If this configuration parameter is set, the employee's main identity is used for authentication. • If the parameter is not set, the employee's subidentity is used for authentication. <p>The application configuration data is used to determine a system user, which is automatically assigned to the employee. The user interface and write permissions are loaded through the system user that is dynamically assigned to the logged in employee.</p> <p>Data modifications are attributed to the current user account.</p>

NOTE: If the option **Connect automatically** is set, authentication is no longer necessary for subsequent logins.

LDAP user account (dynamic)

NOTE: This authentication module is available if the module LDAP Module is installed.

Login Data	<p>Login name, identifier, distinguished name or user ID of an LDAP user account.</p> <p>LDAP user account's password.</p>
Prerequisites	<p>The employee exists in the One Identity Manager database.</p> <p>The LDAP user account exists in the One Identity Manager database and the employee is entered in the user account's master data.</p> <p>The configuration data for dynamically determining the system user is defined in the application. Thus, an employee can, for example, be assigned a system user dynamically depending on their department membership.</p>
Set as	No

default	
Single Sign-On	No
Front-end login allowed	Yes
Web Portal login allowed	Yes
Remarks	<p>If you log in using a login name, identifier or user ID, the corresponding user account is determined in the One Identity Manager database through the container's domain. Logging in with a distinguished name is done directly. One Identity Manager determines which employee is assigned to the LDAP user account.</p> <p>If an employee owns more than one identity, the configuration parameter "QER\Person\MasterIdentity\UseMasterForAuthentication" controls which employee is used for authentication.</p> <ul style="list-style-type: none"> • If this configuration parameter is set, the employee's main identity is used for authentication. • If the parameter is not set, the employee's subidentity is used for authentication. <p>The application configuration data is used to determine a system user, which is automatically assigned to the employee. The user interface and write permissions are loaded through the system user that is dynamically assigned to the logged in employee.</p> <p>Data modifications are attributed to the current user account.</p>

Modify the following configuration parameters in the Designer to implement the authentication module.

Table 39: Configuration Parameters for the Authentication Module

Configuration parameter	Meaning
TargetSystem\LDAP\Authentication	The configuration parameter allows configuration of the LDAP authentication module.
TargetSystem\LDAP\Authentication\Authentication	The configuration parameter specified the authentication mechanism. Permitted values are "Secure", "Encryption", "SecureSocketsLayer", "ReadonlyServer", "Anonymous", "FastBind", "Signing", "Sealing", "Delegation" and "ServerBind". The value can be combined with commas

Configuration parameter	Meaning
	(,). For more information about authentication types, see the MSDN Library .
	Default is ServerBind.
TargetSystem\LDAP\Authentication\Port	LDAP server's port. Default is port 389.
TargetSystem\LDAP\Authentication\RootDN	The configuration parameter contains the root domain's distinguished name. Syntax: <code>dc=MyDomain</code>
TargetSystem\LDAP\Authentication\Server	The configuration parameter contains the name of the LDAP server.

LDAP user account (role based)

 **NOTE:** This authentication module is available if the module LDAP Module is installed.

Login Data	Login name, identifier, distinguished name or user ID of an LDAP user account. LDAP user account's password.
Prerequisites	The employee exists in the One Identity Manager database. The employee is assigned at least one application role. The LDAP user account exists in the One Identity Manager database and the employee is entered in the user account's master data. The configuration data for dynamically determining the system user is defined in the application. Thus, an employee can, for example, be assigned a system user dynamically depending on their department membership.
Set as default	No
Single Sign-On	No
Front-end login allowed	Yes
Web Portal login allowed	Yes

Remarks	<p>If you log in using a login name, identifier or user ID, the corresponding user account is determined in the One Identity Manager database through the container's domain. Logging in with a distinguished name is done directly. One Identity Manager determines which employee is assigned to the LDAP user account.</p> <p>If an employee owns more than one identity, the configuration parameter "QER\Person\MasterIdentity\UseMasterForAuthentication" controls which employee is used for authentication.</p> <ul style="list-style-type: none"> • If this configuration parameter is set, the employee's main identity is used for authentication. • If the parameter is not set, the employee's subidentity is used for authentication. <p>A dynamic system user determined from the employee's application roles. The user interface and the write permissions are loaded through this system user.</p> <p>Data modifications are attributed to the current user account.</p>
---------	--

Modify the following configuration parameters in the Designer to implement the authentication module.

Table 40: Configuration Parameters for the Authentication Module

Configuration parameter	Meaning
TargetSystem\LDAP\Authentication	The configuration parameter allows configuration of the LDAP authentication module.
TargetSystem\LDAP\Authentication\Authentication	The configuration parameter specified the authentication mechanism. Permitted values are "Secure", "Encryption", "SecureSocketsLayer", "ReadonlyServer", "Anonymous", "FastBind", "Signing", "Sealing", "Delegation" and "ServerBind". The value can be combined with commas (,). For more information about authentication types, see the MSDN Library .
	Default is ServerBind.
TargetSystem\LDAP\Authentication\Port	LDAP server's port. Default is port 389.
TargetSystem\LDAP\Authentication\RootDN	The configuration parameter contains the root domain's

Configuration parameter	Meaning
	distinguished name. Syntax: dc=MyDomain
TargetSystem\LDAP\Authentication\Server	The configuration parameter contains the name of the LDAP server.

HTTP header (role based)

The authentication module support authentication through Web Single Sign-On solutions that work with proxy-based architecture.

Login Data Employee's central user account or personnel number.

Prerequisites The employee exists in the One Identity Manager database.

- The central user account or personnel number is entered in the employee's master data.

The employee is assigned at least one application role.

Set as default Yes

Single Sign-On Yes

Front-end login allowed No

Web Portal login allowed Yes

Remarks You must pass the user (in the form: UserName =<user name of authenticated user>) in the HTTP header. The employee is found in the One Identity Manager database whose central user account or personnel number matches the user name passed down.

If an employee owns more than one identity, the configuration parameter "QER\Person\MasterIdentity\UseMasterForAuthentication" controls which employee is used for authentication.

- If this configuration parameter is set, the employee's main identity is used for authentication.
- If the parameter is not set, the employee's subidentity is used for authentication.

A dynamic system user determined from the employee's application roles. The user interface and the write permissions are loaded through this system user.

Changes to the data are assigned to the logged in employee.

HTTP header

The authentication module support authentication through Web Single Sign-On solutions that work with proxy-based architecture.

Login Data	Employee's central user account or personnel number.
Prerequisites	<p>The system user with permissions exists in the One Identity Manager database.</p> <p>The employee exists in the One Identity Manager database.</p> <ul style="list-style-type: none">• The central user account or personnel number is entered in the employee's master data.• The system user is entered in the employee's master data.
Set as default	No
Single Sign-On	Yes
Front-end login allowed	No
Web Portal login allowed	Yes
Remarks	<p>You must pass the user (in the form: <code>UserName =<user name of authenticated user></code>) in the HTTP header. The employee is found in the One Identity Manager database whose central user account or personnel number matches the user name passed down.</p> <p>If an employee owns more than one identity, the configuration parameter "<code>QER\Person\MasterIdentity\UseMasterForAuthentication</code>" controls which employee is used for authentication.</p> <ul style="list-style-type: none">• If this configuration parameter is set, the employee's main identity is used for authentication.• If the parameter is not set, the employee's subidentity is used for authentication. <p>The user interface and the write permissions are loaded through the system user that is directly assigned to the logged in employee. If the employee is not assigned to a system user, the system user is taken from the configuration parameter "<code>SysConfig\Logon\DefaultUser</code>".</p>
	Changes to the data are assigned to the logged in employee.

OAuth 2.0/OpenID Connect

 **NOTE:** This authentication module is available if the module Identity Management Base Module is installed.

The authorization module supports the authorization code for OAuth 2.0 and OpenID Connect. For more detailed information about the authorization code flow, see, for example, the [OAuth Specification](#) or the [OpenID Connect Specification](#).

This authentication module uses a Secure Token Service for logging in. This login procedure can be used with every Secure Token Service which can return an OAuth 2.0 token.

Login Data	Dependent on the authentication method of the secure token service.
Prerequisites	<p>The system user with permissions exists in the One Identity Manager database.</p> <p>The employee exists in the One Identity Manager database.</p> <ul style="list-style-type: none">• The system user is entered in the employee's master data. <p>The user account exists in the One Identity Manager database and the employee is entered in the user account's master data.</p>
Set as default	No
Single Sign-On	No
Front-end login allowed	Yes
Web Portal login allowed	Yes
Remarks	<p>One Identity Manager determines which employee is assigned to the user account.</p> <p>If an employee owns more than one identity, the configuration parameter "QER\Person\MasterIdentity\UseMasterForAuthentication" controls which employee is used for authentication.</p> <ul style="list-style-type: none">• If this configuration parameter is set, the employee's main identity is used for authentication.• If the parameter is not set, the employee's subidentity is used for authentication. <p>The user interface and access permissions are loaded through the system user that is directly assigned to the employee found.</p> <p>Data modifications are attributed to the current user account. To do this, the claim type whose value is used for labeling data changes must be declared.</p>

The respective user interface prompts for the authorization code. The configuration parameter "QER\Person\OAuthAuthenticator>LoginEndpoint" is used to open an extra login dialog box for determining the authorization code. The authentication module requires an access token from the token endpoint and the certificate is required to check the security token. In the process, an attempt is made to find the certificate from the web application configuration. If this is not possible, configuration parameters are applied. To find the certificate for testing the token, the certificate stores are queries in the following order:

1. Web application configuration (table QBMWebApplication)
 - a. Certificate text (QBMWebApplication.CertificateText) .
 - b. Subject or finger print from the local store (QBMWebApplication.OAuthCertificateSubject and QBMWebApplication.OAuthCertificateThumbPrint).
 - c. Certificate endpoint (QBMWebApplication.CertificateEndpoint).

In addition, the subject or finger print is used to check certificates from the server if they are given and do not exist locally on the server.

2. Configuration Parameter

- a. Certificate text (configuration parameter "QER\Person\OAuthAuthenticator\CertificateText").
 - b. Subject or finger print from the local store (configuration parameter "QER\Person\OAuthAuthenticator\CertificateSubject" and "QER\Person\OAuthAuthenticator\CertificateThumbPrint").
 - c. Certificate endpoint (configuration parameter "QER\Person\OAuthAuthenticator\CertificateEndpoint").
- In addition, the subject or finger print is used to check certificates from the server if they are given and do not exist locally on the server.
- d. JSON Web Key endpoint (configuration parameter "QER\Person\OAuthAuthenticator\JsonWebKeyEndpoint").

A claim type is required to find the user account from the user information. In addition, it is specified which One Identity Manager schema information should be used to search for the user account.

Authentication through OpenID is built on OAuth. OpenID Connection authentication uses the same mechanisms, but make user claims available either in an ID token or through a UserInfo endpoint. Other configuration settings are required for using OpenID Connect. If the configuration parameter "QER\Person\OAuthAuthenticator\Scope" contains the value "openid", the authentication module uses OpenID Connect.

Modify the following configuration parameters in the Designer to implement the authentication module.

Table 41: Configuration Parameters for the Authentication Module

Configuration Parameter	Meaning
QER\Person\OAuthAuthenticator	This configuration parameter specifies whether authentication is supported through security tokens.
QER\Person\OAuthAuthenticator\CertificateEndpoint	The configuration parameter contain the certificate endpoint's Uniform Resource Locator (URL) on the authorization server. Example: <code>https://localhost/RSTS/SigningCertificate</code>
QER\Person\OAuthAuthenticator\CertificateSubject	The configuration parameter contain the subject of the certificate to use for testing. Either subject or finger print must be set.
QER\Person\OAuthAuthenticator\CertificateThumbPrint	This configuration parameter contains the fingerprint of the certificate used to verify the security token.
QER\Person\OAuthAuthenticator\ClientID	This configuration parameter specifies whether the client application supports this authentication.
QER\Person\OAuthAuthenticator\ClientID\Web	This configuration parameter contains the web application's Uniform Resource Name URN, which supports this authentication. Example: <code>urn:OneIdentityManager/Web</code>
QER\Person\OAuthAuthenticator\ClientID\Windows	This configuration parameter contains the native application's Uniform Resource Name URN, which supports this authentication. Example: <code>urn:OneIdentityManager/WinClient</code>
QER\Person\OAuthAuthenticator\DisabledByColumns	This configuration parameter contains a pipe () delimited list of Boolean columns from the One Identity Manager table (SearchTable) disabled by the user account for the login. Example: <code>AccountDisabled</code>
QER\Person\OAuthAuthenticator\EnabledByColumns	This configuration parameter contains a pipe () delimited list of Boolean columns from the One Identity Manager table (SearchTable) enabled by the user account for the login.
QER\Person\OAuthAuthenticator\IssuerName	This configuration parameter contains the certificate issuer's Uniform Resource Name (URN) for verifying the security token. Example: <code>urn:STS/identity</code>
QER\Per-	This configuration parameter contains the Uniform Resource

Configuration Parameter	Meaning
son\OAuthAuthenticator\LoginEndpoint	Locator (URL) of the Secure Token Service login page. Example: <code>http://localhost/rsts/login</code>
QER\Person\OAuthAuthenticator\Resource	This configuration parameter contains the Uniform Resource Name (URN) of the resource to be queried, for example ADFS.
QER\Person\OAuthAuthenticator\SearchClaim	This configuration parameter contains the claim type's Uniform Resource Identifier (URI) found from the login data. Example: name of an entity <code>http://schemas.xmlsoap.org/ws/2005/05/identity/claims/nameidentifier</code>
QER\Person\OAuthAuthenticator\SearchColumn	This configuration parameter contains the column from the One Identity Manager table (SearchTable), which is used to search for user data. Equivalent to the claim type (SearchClaim) in the One Identity Manager schema. Example: ObjectGUID
QER\Person\OAuthAuthenticator\SearchTable	This configuration parameter contains the table in the One Identity Manager schema in which user information is stored. The table must contain a foreign key with the name UID_Person, which points to the table Person. Example: ADSAccount
QER\Person\OAuthAuthenticator\TokenEndpoint	This configuration parameter contains the token endpoint's Uniform Resource Identifier (URL) of the authorization server for returning the access token to the client for logging in. Example: <code>https://localhost/rsts/oauth2/token</code>
QER\Person\OAuthAuthenticator\UserNameClaim	This configuration parameter contains the claim type's Uniform Resource Identifier (URL) used to label change data (XUserInserted, XUserUpdated).. Example: User Principle Name (UPN) <code>http://schemas.xmlsoap.org/ws/2005/05/identity/claims/upn</code>
QER\Person\OAuthAuthenticator\InstalledRedirectUri	This configuration parameter contains the Uniform Resource Identifier (URL) for forwarding to installed applications. Example: urn:InstalledApplication

Configuration Parameter	Meaning
QER\Person\OAuthAuthenticator\AllowSelfSignedCertsForTLS	The configuration parameter specifies whether self-signed certificates are allowed for connecting to the token and UserInfo endpoint.
QER\Person\OAuthAuthenticator\CertificateText	This configuration parameter contains the contents of the certificate as a Base64 coded string. It is used if no certificate is configured.
QER\Person\OAuthAuthenticator\JsonWebKeyEndpoint	This configuration parameter contains the Uniform Resource Identifier (URL) of the JSON Web Key endpoint, which supplies the signature key. At the moment, only JWK files, which contain the certificate in the x5c field are supported.
QER\Person\OAuthAuthenticator\LogoffEndpoint	This configuration parameter contains the Uniform Resource Identifier (URL) of the log off end point. Example: http://localhost/rsts/login?wa=wsignin1.0
QER\Person\OAuthAuthenticator\SharedSecret	This configuration parameter contains the Share-Secret value used for authenticating at the token endpoint.

Table 42: Additional Configuration Parameters for OpenID Connect

Configuration Parameter	Meaning
QER\Person\OAuthAuthenticator\Scope	This configuration parameter specifies the authentication log. If the configuration parameter has the value "openid", OpenID Connect is used and otherwise OAuth2.
QER\Person\OAuthAuthenticator\UserInfoEndpoint	This configuration parameter contains the Uniform Resource Locator (URL) of the OpenID Connection UserInfo endpoint.

OAuth 2.0/OpenID Connect (role-based)

 **NOTE:** This authentication module is available if the module Identity Management Base Module is installed.

The authorization module supports the authorization code for OAuth 2.0 and OpenID Connect. For more detailed information about the authorization code flow, see, for example, the [OAuth Specification](#) or the [OpenID Connect Specification](#).

This authentication module uses a Secure Token Service for logging in. This login procedure can be used with every Secure Token Service which can return an OAuth 2.0 token.

Login Data	Dependent on the authentication method of the secure token service.
Prerequisites	<p>The employee exists in the One Identity Manager database.</p> <p>The employee is assigned at least one application role.</p> <p>The user account exists in the One Identity Manager database and the employee is entered in the user account's master data.</p>
Set as default	No
Single Sign-On	No
Front-end login allowed	Yes
Web Portal login allowed	Yes
Remarks	<p>One Identity Manager determines which employee is assigned to the user account.</p> <p>If an employee owns more than one identity, the configuration parameter "QER\Person\MasterIdentity\UseMasterForAuthentication" controls which employee is used for authentication.</p> <ul style="list-style-type: none"> • If this configuration parameter is set, the employee's main identity is used for authentication. • If the parameter is not set, the employee's subidentity is used for authentication. <p>A dynamic system user determined from the employee's application roles. The user interface and the write permissions are loaded through this system user.</p> <p>Data modifications are attributed to the current user account. To do this, the claim type whose value is used for labeling data changes must be declared.</p>

The respective user interface prompts for the authorization code. The configuration parameter "QER\Person\OAuthAuthenticator\LoginEndpoint" is used to open an extra login dialog box for determining the authorization code. The authentication module requires an access token from the token endpoint and the certificate is required to check the security token. In the process, an attempt is made to find the certificate from the web application configuration. If this is not possible, configuration parameters are applied. To find the certificate for testing the token, the certificate stores are queries in the following order:

1. Web application configuration (table QBMWebApplication)
 - a. Certificate text (QBMWebApplication.CertificateText) .
 - b. Subject or finger print from the local store (QBMWebApplication OAuthCertificateSubject and QBMWebApplication OAuthCertificateThumbPrint).

- c. Certificate endpoint (`QBMWebApplication.CertificateEndpoint`).

In addition, the subject or finger print is used to check certificates from the server if they are given and do not exist locally on the server.

2. Configuration Parameter

- a. Certificate text (configuration parameter "`QER\Person\OAuthAuthenticator\CertificateText`").
- b. Subject or finger print from the local store (configuration parameter "`QER\Person\OAuthAuthenticator\CertificateSubject`" and "`QER\Person\OAuthAuthenticator\CertificateThumbPrint`").
- c. Certificate endpoint (configuration parameter "`QER\Person\OAuthAuthenticator\CertificateEndpoint`").
In addition, the subject or finger print is used to check certificates from the server if they are given and do not exist locally on the server.
- d. JSON Web Key endpoint (configuration parameter "`QER\Person\OAuthAuthenticator\JsonWebKeyEndpoint`").

A claim type is required to find the user account from the user information. In addition, it is specified which One Identity Manager schema information should be used to search for the user account.

Authentication through OpenID is built on OAuth. OpenID Connection authentication uses the same mechanisms, but make user claims available either in an ID token or through a UserInfo endpoint. Other configuration settings are required for using OpenID Connect. If the configuration parameter "`QER\Person\OAuthAuthenticator\Scope`" contains the value "openid", the authentication module uses OpenID Connect.

Modify the following configuration parameters in the Designer to implement the authentication module.

Table 43: Configuration Parameters for the Authentication Module

Configuration Parameter	Meaning
<code>QER\Person\OAuthAuthenticator</code>	This configuration parameter specifies whether authentication is supported through security tokens.
<code>QER\Person\OAuthAuthenticator\CertificateEndpoint</code>	The configuration parameter contain the certificate endpoint's Uniform Resource Locator (URL) on the authorization server. Example: <code>https://localhost/RSTS/SigningCertificate</code>
<code>QER\Person\OAuthAuthenticator\CertificateSubject</code>	The configuration parameter contain the subject of the certificate to use for testing. Either subject or finger print must be set.
<code>QER\Person\OAuthAuthenticator\CertificateThumbPrint</code>	This configuration parameter contains the fingerprint of the certificate used to verify the security token.

Configuration Parameter	Meaning
QER\Person\OAuthAuthenticator\ClientID	This configuration parameter specifies whether the client application supports this authentication.
QER\Person\OAuthAuthenticator\ClientID\Web	This configuration parameter contains the web application's Uniform Resource Name URN, which supports this authentication. Example: urn:OneIdentityManager/Web
QER\Person\OAuthAuthenticator\ClientID\Windows	This configuration parameter contains the native application's Uniform Resource Name URN, which supports this authentication. Example: urn:OneIdentityManager/WinClient
QER\Person\OAuthAuthenticator\DisabledByColumns	This configuration parameter contains a pipe () delimited list of Boolean columns from the One Identity Manager table (SearchTable) disabled by the user account for the login. Example: AccountDisabled
QER\Person\OAuthAuthenticator\EnabledByColumns	This configuration parameter contains a pipe () delimited list of Boolean columns from the One Identity Manager table (SearchTable) enabled by the user account for the login.
QER\Person\OAuthAuthenticator\IssuerName	This configuration parameter contains the certificate issuer's Uniform Resource Name (URN) for verifying the security token. Example: urn:STS/identity
QER\Person\OAuthAuthenticator>LoginEndpoint	This configuration parameter contains the Uniform Resource Locator (URL) of the Secure Token Service login page. Example: http://localhost/rsts/login
QER\Person\OAuthAuthenticator\Resource	This configuration parameter contains the Uniform Resource Name (URN) of the resource to be queried, for example ADFS.
QER\Person\OAuthAuthenticator\SearchClaim	This configuration parameter contains the claim type's Uniform Resource Identifier (URI) found from the login data. Example: name of an entity <code>http://schemas.xmlsoap.org/ws/2005/05/identity/claims/nameidentifier</code>
QER\Per-	This configuration parameter contains the column from the

Configuration Parameter	Meaning
son\OAuthAuthenticator\SearchColumn	<p>One Identity Manager table (SearchTable), which is used to search for user data. Equivalent to the claim type (SearchClaim) in the One Identity Manager schema.</p> <p>Example: ObjectGUID</p>
QER\Person\OAuthAuthenticator\SearchTable	<p>This configuration parameter contains the table in the One Identity Manager schema in which user information is stored. The table must contain a foreign key with the name UID_Person, which points to the table Person.</p> <p>Example: ADSAccount</p>
QER\Person\OAuthAuthenticator\TokenEndpoint	<p>This configuration parameter contains the token endpoint's Uniform Resource Identifier (URL) of the authorization server for returning the access token to the client for logging in.</p> <p>Example: https://localhost/rsts/oauth2/token</p>
QER\Person\OAuthAuthenticator\UserNameClaim	<p>This configuration parameter contains the claim type's Uniform Resource Identifier (URL) used to label change data (XUserInserted, XUserUpdated)..</p> <p>Example: User Principle Name (UPN) http://schemas.xmlsoap.org/ws/2005/05/identity/claims/upn</p>
QER\Person\OAuthAuthenticator\InstalledRedirectUri	<p>This configuration parameter contains the Uniform Resource Identifier (URL) for forwarding to installed applications.</p> <p>Example: urn:InstalledApplication</p>
QER\Person\OAuthAuthenticator\AllowSelfSignedCertsForTLS	The configuration parameter specifies whether self-signed certificates are allowed for connecting to the token and UserInfo endpoint.
QER\Person\OAuthAuthenticator\CertificateText	This configuration parameter contains the contents of the certificate as a Base64 coded string. It is used if no certificate is configured.
QER\Person\OAuthAuthenticator\JsonWebKeyEndpoint	This configuration parameter contains the Uniform Resource Identifier (URL) of the JSON Web Key endpoint, which supplies the signature key. At the moment, only JWK files, which contain the certificate in the x5c field are supported.
QER\Person\OAuthAuthenticator\LogoutEndpoint	<p>This configuration parameter contains the Uniform Resource Identifier (URL) of the log off end point.</p> <p>Example: http://localhost/rsts/login?wa=wsignout1.0</p>

Configuration Parameter	Meaning
QER\Person\OAuthAuthenticator\SharedSecret	This configuration parameter contains the Share-Secret value used for authenticating at the token endpoint.

Table 44: Additional Configuration Parameters for OpenID Connect

Configuration Parameter	Meaning
QER\Person\OAuthAuthenticator\Scope	This configuration parameter specifies the authentication log. If the configuration parameter has the value "openid", OpenID Connect is used and otherwise OAuth2.
QER\Person\OAuthAuthenticator\UserInfoEndpoint	This configuration parameter contains the Uniform Resource Locator (URL) of the OpenID Connection UserInfo endpoint.

Synchronization authenticator

 **NOTE:** This authentication module is available if the module Target System Synchronization Module is installed.

This authentication module integrates the default method for Synchronization Editor login.

Login Data	Use the system user "sa" to log in.
Prerequisites	
Set as default	Yes
Single Sign-On	No
Front-end login allowed	No
Web Portal login allowed	No
Remarks	The system user "sa" should not be changed, as it overwritten each time the schema is installed.

Web Agent authenticator

The authentication module integrates the default method for Web Designer login, to access the database before the first user login.

Login Data	Use the system user "sa" to log in.
Prerequisites	

Set as default	Yes
Single Sign-On	No
Front-end login allowed	No
Web Portal login allowed	No
Remarks	The system user "sa" should not be changed, as it overwritten each time the schema is installed.

Component authenticator

This authentication module integrates the default method for registering process components.

Login Data	Use the system user "sa" to log in.
Prerequisites	
Set as default	Yes
Single Sign-On	No
Front-end login allowed	No
Web Portal login allowed	No
Remarks	The system user "sa" should not be changed, as it overwritten each time the schema is updated.

Crawler

The authentication module is used by the application server to compile search indexes for full text search over the database.

Login Data	Use the system user "sa" to log in.
Prerequisites	
Set as default	Yes
Single Sign-On	No
Front-end login allowed	No
Web Portal login allowed	No
Remarks	The system user "sa" should not be changed, as it overwritten each time the schema is installed.

Related Topics

- [Editing Authentication Modules on page 103](#)
- [Initial Data for Authentication Modules on page 104](#)

Editing Authentication Modules

To enable other authentication modules

1. Select the category **Base Data | Security settings | Authentication modules** in the Designer.
2. Select the authentication module and set the option **Enabled** to "True".
3. Save the changes to the database using **Database | Commit to database....**
4. Click **Save**.

This allows you to log in to the assigned application using this authentication module. Ensure that users found through the authentication module have the required permissions to use the program.

If create custom authentication modules, assign them to the existing programs. Assignments to predefined authentication modules must not normally be changed.

To assign an authentication module to programs

1. Select the category **Base Data | Security settings | Authentication modules** in the Designer.
2. Select **View | Select table relations...** in the menu.
3. Enable the table DialogProductHasAuthentifier.
This shows the tab **Programs**.
4. Select the authentication module and assign it to programs.

Detailed information about this topic

- [Authentication Module Properties on page 104](#)
- [Initial Data for Authentication Modules on page 104](#)
- [Configuration Data for System User Dynamic Authentication on page 108](#)
- [One Identity Manager Authentication Module on page 74](#)

Authentication Module Properties

Table 45: Authentication Module Properties

Property	Meaning
Enabled	Specifies whether the authentication module can be used.
Display name	This name is used to identify the authentication module in the administration tool's login window.
Authentication Module	Enter name of the authentication module.
Authentication type	Specifies the type of authentication module. The options are "Dynamic" or "Role based".
Processing status	The process state is used for creating custom configuration packages.
Initial data	Initial data for logging in with this authentication module.
Class	Authentication module class.
Assembly name	Name of the assembly file.
Sort order	Specify the order in which the modules are displayed in the login window.
Single sign-on	Specifies whether the authentication module may be authenticated without a password.
Select in front-end	Specifies whether the authentication module can be selected in the login window.

Related Topics

- [Initial Data for Authentication Modules on page 104](#)

Initial Data for Authentication Modules

The authentication string is formatted as follows:

Module=<name>;<property1>=<value1>;<property2>=<value2>,...

Example:

Module=DialogUser;User=viadmin;Password=*****

The initial data is one part of the authentication string (parameter-value pair without module ID). Initial data from the authentication string is preallocated by default for each authentication instance.

To specify initial data

1. Select the category **Base Data | Security settings | Authentication modules** in the Designer.
2. Select the authentication module and enter the data in **Initial data**.

Syntax:

property1=value1;property2=value2

Example:

user=viadmin;password=*****

You can use different initial data depending on the authentication module.

Table 46: Initial Data for Authentication Modules

Module Display Name	Authentication Module	Parameter	Meaning/Comment
System user	DialogUser	User	User name.
		Password	User password.
Active Directory user account	ADSAccount		
Active Directory user account (dynamic)	DynamicADSAccount	Product	Use case. The system user is determined through the use case configuration data.
Active Directory user account (manual input)	DynamicManualADS	Product	Use case. The system user is determined through the use case configuration data.
		User	User name. The user's identity is determined from a predefined list of permitted Active Directory domains. You specify permitted Active Directory domains in the configuration parameter "TargetSystem\ADS\AuthenticationDomains".
		Password	User password.
Active Directory user	RoleBasedADSAcount		No parameters required

Module Display Name	Authentication Module	Parameter	Meaning/Comment
account (role based)			
Active Directory user account (manual input/role based)	RoleBasedManualADS	User	User name. The user's identity is determined from a predefined list of permitted Active Directory domains. You specify permitted Active Directory domains in the configuration parameter "TargetSystem\ADS\AuthenticationDomains".
		Password	User password.
Employee	Employee	User	Employee's central user account.
		Password	User password.
Employee	DynamicPerson (dynamic)	Product	Use case. The system user is determined through the use case configuration data.
		User	User name.
		Password	User password.
Employee	RoleBasedPerson (role based)	User	User name.
		Password	User password.
HTTP header	HTTPHeader	Header	HTTP header to use.
		KeyColumn	Comma delimited list of key columns in the table Person to be searched for user names. Default: CentralAccount, PersonnelNumber
HTTP header (role based)	RoleBasedHTTPHeader	Header	HTTP header to use.
		KeyColumn	Comma delimited list of key columns in the table Person to be searched for user names. Default: CentralAccount, PersonnelNumber
LDAP user	DynamicLdap	User	User name. Default: CN, DistinguishedName, UserID,

Module Display Name	Authentication Module	Parameter	Meaning/Comment
account (dynamic)			UIDLDAP
LDAP user account (role based)	RoleBasedLdap	Password	User password.
		User	User name. Default: CN, DistinguishedName, UserID, UIDLDAP
		Password	User password.
Generic single sign-on (role based)	RoleBasedGeneric	SearchTable	Table in which to search for the user name of the logged in user. This table must contain a FK named UID_Person which points to the table Person.
		SearchColumn	Column from <SearchTable> in which to search for the user name of the logged in user.
		DisabledBy	Pipe () delimited list of Boolean columns which block a user account from logging in.
		EnabledBy	Pipe () delimited list of Boolean columns which release a user account for logging in.
OAuth 2.0/Open-ID Connect	OAuth		Dependent on the authentication method of the secure token service.
OAuth 2.0/Open-ID Connect (role-based)	OAuthRoleBased		Dependent on the authentication method of the secure token service.
Account based system user	DialogUserAccountBased		No parameters required
User	QERAccount		

Module Display Name	Authentication Module	Parameter	Meaning/Comment
account			
User account (role based)	RoleBasedQERACcount		

Related Topics

- [Configuration Data for System User Dynamic Authentication on page 108](#)
- [One Identity Manager Authentication Module on page 74](#)

Configuration Data for System User Dynamic Authentication

In the case of dynamic authentication modules, the system user assigned to the employee is not used for the log in. The system user which is configured using the user interface special configuration data is taken instead.

To specify configuration data

1. Select the category **Base Data | Security settings | Programs** in the Designer.
2. Select the application and adjust the **Configuration data**.

Use XML syntax for entering the configuration data:

```
<DialogUserDetect>
    <Usermappings>
        <Usermapping>
            DialogUser = "System user name"
            Selection = "Selection criterion"
        />
        <Usermapping>
            DialogUser = "System user name"
        />
        ...
    </Usermappings>
</DialogUserDetect>
```

Enter the system user (DialogUser) in Usermappings section. Specify which employee the given system user should use with the selection criterion (Selection). You are not obliged to enter a selection criterion for the assignment. The first system user that has the required assignment is used for the log in.

You can assign function groups to permissions groups on order to deal with complex rights and user interface structures. The function groups allow you to map the functions an employee has in the company, for example, IT controller or branch manager. Assign the function groups to the permissions groups. A function group can refer to several permissions groups and several function groups can refer to one permissions group.

If the section FunctionGroupMapping is in the configuration data, this is evaluated first and the system user that is found is used. The authentication module uses the system user that is the exact member of the permissions group found for the login. If none is found the section Usermapping is evaluated.

```
<DialogUserDetect>
    <FunctionGroupMapping>
        PersonToFunction = "View mapping employee to function group"
        FunctionToGroup = "View mapping function group to permissions group"
    />
    <Usermappings>
        <Usermapping>
            DialogUser = "System user name"
            Selection = "Selection criterion"
        />
        ...
    </Usermappings>
</DialogUserDetect>
```

Related Topics

- [Example of a Simple System User Assignment on page 109](#)
- [Example of a System User Assignment using Selection Criteria on page 110](#)
- [Example of a Function Group Assignment on page 111](#)

Example of a Simple System User Assignment

All employees should be able to see the user interface for an IT Shop in a web front-end, without taking table and column permissions into account.

To do this, you set up a new application, for example, "WebShop_Customer", and change the configuration data as follow:

```
<DialogUserDetect>
    <Usermappings>
```

```

<Usermapping
    DialogUser = "dlg_all"
/>
</Usermappings>
</DialogUserDetect>

```

Add a new permissions group "WebShop_Customer", which contains the user interface for the application, consisting of menu items, user interface forms and task definitions. The user interface could consist of the following menu items:

- Employee contact data
- Request a product
- Cancel a product

Define a new system user "dlg_all" and add it to the permissions groups "vi_CentralPwd", "vi_ITShopOrder" and "WebShop_Customer".

Related Topics

- [Configuration Data for System User Dynamic Authentication on page 108](#)
- [Example of a System User Assignment using Selection Criteria on page 110](#)
- [Example of a Function Group Assignment on page 111](#)

Example of a System User Assignment using Selection Criteria

The scenario described in the previous example is extended such that only the cost center manager can see an employee's leaving date. You need to add the input field **LeavingDate** to the contact data form to do this.

Permissions are used for controlling viewing and editing. Define a new system user "dlg_kst" and add it to the permissions groups "vi_CentralPwd", "vi_ITShopOrder" and "WebShop_Customer". You should also give the system user read and write access to the column Person.Exitdate.

Change the application configuration data such that cost center manager uses the system user "dlg_kst" to log in. All other employees use the system user "dlg_all" to log in.

Change the configuration data as follows:

```

<DialogUserDetect>
    <Usermappings>
        <Usermapping
            DialogUser = "dlg_kst"
            Selection = "select 1 where %uid% in (select uid_personhead from
                profitcenter)"

```

```

    />
<Usermapping
    DialogUser = "dlg_all"
/>
</Usermappings>
</DialogUserDetect>

```

Related Topics

- [Configuration Data for System User Dynamic Authentication on page 108](#)
- [Example of a Simple System User Assignment on page 109](#)
- [Example of a Function Group Assignment on page 111](#)

Example of a Function Group Assignment

To assign function groups to permissions groups you have to define two database views. The first database view shows the assignment of employees to function groups. The database view contains two columns `UID_Person` and `FunctionGroup`.

Example:

```

create view custom_Person2Fu as
    select uid_personHead as UID_Person, 'Cost center manager' as FunctionGroup
    from Profitcenter
    where isnull(uid_personHead, '') > ''
    union all
    select uid_personHead, 'Department manager' as FunctionGroup
    from Department
    where isnull(uid_personHead, '') > ''

```

The second database view assigns function groups to permissions groups. This database view contains two columns `FunctionGroup` and `DialogGroup`.

Example:

```

create view custom_Fu2D as
    select 'Cost center manager' as FunctionGroup, 'Custom_Dialoggroup_ChefP' as
    DialogGroup
    union all select 'Department manager', 'Custom_Dialoggroup_ChefD'

```

Set up the permissions groups with the desired user interface and the necessary permissions. Define the system users corresponding to the number of permissions groups required. The number of system user accounts necessary is $2^{\text{number of permissions groups}} - 1$. Assign the system user directly to the permissions groups.

Table 47: Extracts from the table "DialogUserInGroup":

Group name	UserName
Custom_Dialoggroup_ChefP	CustomUserP
Custom_Dialoggroup_ChefP	CustomUserPD
Custom_Dialoggroup_ChefD	CustomUserPD
Custom_Dialoggroup_ChefD	CustomUserD

Change the configuration data for assigning function groups to permissions groups as follows:

```
<DialogUserDetect>
  <FunctionGroupMapping>
    PersonToFunction = "custom_Person2Fu"
    FunctionToGroup = "custom_Fu2D"
  />
</DialogUserDetect>
```

Related Topics

- [Configuration Data for System User Dynamic Authentication on page 108](#)
- [Example of a Simple System User Assignment on page 109](#)
- [Example of a System User Assignment using Selection Criteria on page 110](#)

Database Connection Data

The One Identity Manager database connection data is set up by the initial schema installation. This information is also accessed when tasks are generated for the One Identity Manager Service.

NOTE: Changes to the data are not usually necessary and should only be made by advanced users.

To change the connection parameter

1. Select the category **Base Data | General | Databases** in the Designer.
2. Select the database in the List Editor.
3. Select **Define connection string for database <database name>** in the task view.

- Enter the database connection data.

Table 48: SQL Server Database Connection Data

Data	Description
Server	Database server.
Windows authentication	Specifies whether Windows authentication is used. This type of authentication is not recommended. If you decide to use it anyway, ensure that your environment supports Windows authentication.
User	Database user.
Password	Database user password.
Database	Database.

Table 49: Oracle Database Connection Data

Data	Description
Direct access (without Oracle client)	Set this option for direct access. Deactivate this option for access via Oracle Clients. Which connection data is required, depends on how this option is set.
Server	Database server.
Port	Oracle instance port.
Service name	Service name.
User	Oracle database user.
Password	Database user password.
Data source	TNS alias name from <code>TNSNames.ora</code> .

- Click **OK**.

Database Connection Properties

NOTE: Changes to the data are not usually necessary and should only be made by advanced users.

To display database information

- Select the category **Base Data | General | Databases** in the Designer.

Table 50: Database information

Property	Meaning
Main database	Identifies the database as the main database. The One Identity Manager database is marked with this option when the schema is installed the first time.
Customer	Name of the customer.
Description	Description of database.
Customer prefix	Customer ID for prefix. The customer prefix is used to create and transfer customized scripts, processes and extensions to the One Identity Manager schema.
Module owner	Module owner ID for prefix. The prefix is used to create and transfer customized scripts, processes and extensions to the One Identity Manager schema.
Staging level	Specifies whether the database is a test, development or production database. Permitted values are: <ul style="list-style-type: none">• Development system• Test environment• Production system
Status bar color	The color of the status bar can be set differently to the layout color depending on the staging level. The color can be defined by template and customized. The following colors are defined as default: <ul style="list-style-type: none">• None - development system database is connected.• Green - test environment database is connected.• Yellow - production system database is connected.
Last compiler relevant configuration date	Time of the last change to the configuration which required the database to be recompiled. If the value is changed the database has to be recompiled.
Simulation started	Time at which the last front-end simulation was started.
Stop DBQueue Processor	If this option is set for the main database, the DBQueue Processor does not process any more tasks. You can make the DBQueue Processor start and stop with the appropriate administrative rights in the program "Job Queue Info".
Stop One Identity Manager Service	If this option is set for the main database, the One Identity Manager Service does not process any more tasks. You can make the service start and stop with the appropriate administrative permissions in the program "Job Queue Info".

Property	Meaning
Provider	"VI.DB.ViSqlFactory,VI.DB" is entered for the Microsoft SQL Server connection. "VI.DB.Oracle.ViOracleFactory, VI.DB.Oracle" is entered for the Oracle server connection.
Connection parameter	Login data for the database user, database server and the database. The data is entered into the database during schema installation.
Authentication Module	The default authentication is entered here.
Language culture	Default language. Fallback alternative for displaying language dependent text.
Edition	Edition ID.
Edition version	Version number of the edition.
Edition description	Detailed description of the edition.
Database ID	Identifier for the database. The database ID is taken from the original database server and database data. The database ID has to be recalculated if a database is created from a database backup on another server. When a database is compiled, the database ID is checked and changed if necessary.
Single-user mode process	Process requiring single-user mode. If the value = 0, single-user mode is not required. ① NOTE: The database schedule QBM_PWatchDog on <database> check regularly whether single-user mode is still required and resets the options if necessary.
Single-user mode start time	Time of the request for single-user mode. ① NOTE: The database schedule QBM_PWatchDog on <database> check regularly whether single-user mode is still required and resets the options if necessary.
Public key for encryption	The public key is entered by the program "Crypto Configuration" and is needed for database encryption. For more information, see the One Identity Manager Installation Guide.

Related Topics

- [Database Connection Data on page 112](#)
- [Configuring a One Identity Manager Database for Testing, Development or Production on page 116](#)
- [One Identity Manager Authentication Module on page 74](#)
- [Stopping the System \(Emergency Stop\) on page 626](#)

- [Compiling a One Identity Manager Database on page 64](#)
- [Enabling More Languages for Displaying and Maintaining Data on page 126](#)
- [Language Dependent Data Representation on page 328](#)

Configuring a One Identity Manager Database for Testing, Development or Production

Use the staging level of the One Identity Manager database to specify whether a test database, development database or a live database is being dealt with. A number of configuration settings are controlled by the staging level. These are set when you modify the staging level.

Table 51: Database Settings for Development, Test and Live Environments

Setting	Database Staging Level		
	Development Environment	Test Environment	Live Environment
Color of the One Identity Manager tools status bar.	none	Green	Yellow
Maximum DBQueue Processor runtime	20 minutes	40 minutes	120 minutes
Maximum number of slots for DBQueue Processor	3	5	Maximum number of slots according to the hardware configuration

To modify a database staging level

1. Select the category **Base Data | General | Databases** in the Designer.
2. Select the database and change the value of the property **Staging level** to "Test environment", "Development system" or "Development system".
3. Select the **Database | Commit to database...** and click **Save**.

Related Topics

- [Configuring the DBQueue Processor for Test and Development on page 645](#)

Configuration Parameters for System Configuration

Use configuration parameters to configure the behavior of the system's basic settings. One Identity Manager provides default settings for different configuration parameters. Check the configuration parameters and modify them as necessary to suit your requirements.

Configuration parameters are defined in the One Identity Manager modules. Each One Identity Manager module can also install configuration parameters. You can find an overview of all configuration parameters in the category **Base data | General | Configuration parameters** in the Designer.

Detailed information about this topic

- [Editing Configuration Parameters on page 119](#)

Working with the Configuration Parameter Editor

The Configuration Parameter Editor enables you to edit configuration parameters and their options. The editor is started from the program "Designer" and opens in the document view. Only additional Configuration Parameter Editor functions are described in the following.

Menu Items

The following items are added to the menu bar when the editor starts.

Table 52: Menu Items Added by the Editor

Menu	Menu Item	Meaning
Configuration parameter	New	Adds a new configuration parameter.
	Delete	Deletes the selected configuration parameter.
	Show captions	Shows the caption for the configuration parameter. If the option is not enabled, the technical names according to the database model are shown.
	Reload parameters	Updates the configuration parameter list.

Menu	Menu Item	Meaning
View	Properties	Shows/hides the edit view.
	Preprocessor definitions	Shows/hides preprocessor definitions view.
Help	Help for all configuration parameters	Opens the help on this topic.
	Editor help	Opens the editor help.

Views in the Configuration Parameter Editor

The Configuration Parameter Editor has several views for displaying and editing the configuration parameters.

Table 53: Configuration Parameter Editor Views

View	Description
Hierarchical view	This view contains a hierarchical representation of the configuration parameters and some selected properties. You can enable and disable the configuration parameters by clicking with the mouse on a check box.
Edit view	Use the edit view to edit the properties of the selected configuration parameter and its options. A default context menu is available for input fields.
Preprocessor definitions view	All preprocessor statements for preprocessor relevant configuration parameters are shown in this view. Preprocessor conditions cannot be edit here.

TIP: If you double click on a entry you change to the corresponding configuration parameter in the hierarchical view.

Table 54: Context Menu Items for the Hierarchical View

Context Menu Item	Meaning
New	Adds a new configuration parameter.
Delete	Deletes the selected configuration parameter.
Select columns...	Opens a dialog window to select columns to be shown additionally in the hierarchical view.

Editing Configuration Parameters

The supplied configuration parameters and the permitted values are maintained by the schema installation. You cannot edit the properties of these parameters. You can enable and disable them and specify the actual value for the parameter. Other properties of predefined configuration parameters cannot be edited. Calculation tasks for the DBQueue Processor can result from changes a configuration parameter.

To edit configuration parameters

1. Select the category **Base data | General | Configuration parameters** in the Designer.
2. Start the Configuration Parameter Editor from **Edit configuration parameters** in the task view.

All configuration parameters and their options are displayed in the hierarchical view. You can also see whether the parameters are preprocessor relevant or if the value should be encrypted before being passed.

3. Select the configuration parameter and configure the settings.

Table 55: Configuration Parameter Settings

Property	Meaning
Enabled	Specifies whether the configuration parameter is enabled. Check the box to enable the configuration parameter. Uncheck the box to disable the configuration parameter.
Value	Value of the configuration parameter.

Property	Meaning
Enabled	Specifies whether the configuration parameter is enabled. Check the box to enable the configuration parameter. Uncheck the box to disable the configuration parameter.
Value	Value of the configuration parameter.

IMPORTANT: Compile the database if the configuration parameter is preprocessor relevant.

If it is necessary to define more custom configuration parameters, you can add them in the configuration parameter list item "Custom".

To set up a new configuration parameter

1. Select the category **Base data | General | Configuration parameters** in the Designer.
2. Start the Configuration Parameter Editor from **Edit configuration parameters** in the task view.
3. Select the configuration parameter "Custom" and enter a new configuration parameter using the **New** in the context menu.
4. Edit the configuration parameter's master data.

- Specify permitted values on the **options** tab (optional).
 - Click **Add** to add a new option.
 - Click **Remove** to remove an option.

Detailed information about this topic

- [Configuration Parameter Properties on page 120](#)
- [Configuration Parameter Options on page 121](#)

Related Topics

- [Preprocessor Relevant Configuration Parameters on page 449](#)
- [Compiling a One Identity Manager Database on page 64](#)

Configuration Parameter Properties

Table 56: Configuration Parameter Properties

Property	Description
Full name	Full name of the configuration parameter. This consists of the name of the parameter and the name of the parent parameter.
Parameter	Technical name of the configuration parameter.
Display name	The display name supplies the caption for the configuration parameter. The display names can be stored as language dependent. Configuration parameters that do not have a display name are displayed in brackets (<>). In addition, a tooltip with the technical name is displayed.
Sort order	The sort order affects how the configuration parameters are ordered in the Configuration Parameter Editor. ① NOTE: The sort order is only effective if the display text is shown in the editor (menu Configuration parameters Show captions).
Value	Value of the configuration parameter. You must enter a value for every configuration parameter. Even the parent parameters which serve no purpose other than providing a structure, may not be empty, otherwise the child configuration parameters cannot be accessed. Some configuration parameters have several permitted values. These are specified using the configuration parameter options and can be selected here. A description of the selected option is also shown.
Description	Description of the configuration parameter. Click Edit to edit the description.
Enabled	Specifies whether the configuration parameter is enabled. If this option is

Property	Description
	set, the configuration parameter is enabled. If this option is not set, then the whole tree from this point on is considered disabled and the configuration parameter and its child parameters are considered not to exist.
Encrypted	Configuration parameters are marked with this option when they contain encrypted data, for example, passwords. When a new value is entered it is therefore encrypted immediately.
Preprocessor relevant parameter	This option marked configuration parameters as preprocessor relevant. A preprocessor statement is entered in the associated option field that is used for conditional compiling. i NOTE: When a preprocessor relevant configuration parameter is set it is valid globally across the system. The preprocessor condition does not come into effect until the database has been compiled. Every time a preprocessor relevant configuration parameter or its option is changed the database needs to be recompiled.

Related Topics

- [Configuration Parameter Options on page 121](#)
- [Preprocessor Relevant Configuration Parameters on page 449](#)
- [Compiling a One Identity Manager Database on page 64](#)

Configuration Parameter Options

If a configuration parameter only permits certain values, these values are defined in the configuration parameter options.

Table 57: Option Properties

Property	Description
Value	Value permitted for the configuration parameter.
Description	Description of the configuration parameter option.
Preprocessor expression	Preprocessor relevant configuration parameters as assigned a valid preprocessor expression in the options. This can be used as a preprocessor condition for conditional compiling.

Related Topics

- [Configuration Parameter Properties on page 120](#)
- [Preprocessor Relevant Configuration Parameters on page 449](#)

Setting up the Mail Notification System

The One Identity Manager sends email notifications about various actions taken within the system. Thus, various notifications are sent to requester and approver within the request process. In the same way, notifications about attestation cases are sent or reports delivered by email. Notifications are sent when an action is successfully or unsuccessfully executed during process handling.

NOTE: In addition to the configuration parameters listed in the following, other configuration parameters may be necessary for different notification processes. Some configuration parameters are only available when the module is installed.

You can implement custom notifications in addition to predefined notification processes.

Prerequisites for using notification systems are:

1. Declaring a Job server as SMTP host for sending mail.
2. Enabling configuration parameters for mail notification

NOTE: In addition to the configuration parameters listed in the following, other configuration parameters may be necessary for different notification processes.

Table 58: General Configuration Parameters for Mail Notification

Configuration Parameter	Meaning
Common\InternationalEMail	This parameter specifies whether international domain names and unicode characters are supported in email addresses. IMPORTANT: The mail server must also support this function. If necessary, you must override the script VID_IsSMTPAddress
Common\MailNotification	Notification data.
Common\MailNotification\DefaultAddress	Default email address (recipient) for sending notifications.
Common\MailNotification\DefaultCulture	Default language that emails are sent in if no language can be determined for a recipient.
Common\MailNotification\DefaultLanguage	Default language for sending messages.
Common\MailNotification\DefaultSender	Default email address (sender) for

Configuration Parameter	Meaning
	sending notifications.
Common\MailNotification\Encrypt	Specifies whether emails are encrypted.
Common\MailNotification\Encrypt\ConnectDC	Domain controller to use.
Common\MailNotification\Encrypt\ConnectPassword	User password. This is optional.
Common\MailNotification\Encrypt\ConnectUser	User account for querying Active Directory. This is optional.
Common\MailNotification\Encrypt\DomainDN	Distinguished name of the domain to search through.
Common\MailNotification\Encrypt\EncryptionCertificateScript	Script, which supplies a list of encrypted certificates (default: QBM_GetCertificates).
Common\MailNotification\NotifyAboutWaitingJobs	Specifies whether a message should be sent if the process steps have a particular execution state in the job queue.
Common\MailNotification\SignCertificateThumbprint	SHA1 thumbprint of the certificate to use for the signature. This can be in the computer's or the user's My Store.
Common\MailNotification\SMTPAccount	User account name for authentication on an SMTP server.
Common\MailNotification\SMTPDomain	User account domain for authentication on the SMTP server.
Common\MailNotification\SMTPPassword	User account password for authentication on the SMTP server.
Common\MailNotification\SMTPPort	Port for SMTP services on the SMTP server (default: 25).
Common\MailNotification\SMTPRelay	SMTP server for sending notifications.
Common\MailNotification\SMTPUseDefaultCredentials	If this parameter is set, the One Identity Manager Service login credentials are used for authentication on the SMTP server. If the configuration parameter is not set, the login data stored in the

Configuration Parameter	Meaning
	parameters "Common\MailNotification\SMTPDo main", "Common\MailNotification\SMTPAc count" and "Common\MailNotification\SMTPPa ssword" is used.
Common\MailNotification\TransportSecurity	This configuration parameter defined the encryption method for sending notification by email. If none of the following options are given, the port is used to define the behavior (port: 25 = no encryption, port: 465 = with SSL/TLS encryption).

NOTE: The parameters for encryption method to be used are not set in the processes for sending notification by email. If you use the configuration parameter, change the processes accordingly.

Table 59: Permitted Values

Value	Meaning
Auto	Identifies the encryption method automatically.
SSL	Encrypts the entire session with SSL/TLS.
STARTTLS	Uses the STARTTLS mail server

Configuration Parameter	Meaning	
	Value	Meaning
		extension. Switches TLS encryption after the 'greeting' and loading the server capabilitie s. The connection fails if the server does not support the STARTTLS extension.
	STARTTLSWhenAvai lable	Uses the STARTTLS mail server extension if available. Switches on TLS encryption after the 'greeting' and loading the server capabilitie s, however, only if it supports the STARTTLS extension.
Common\MailNotification\VendorNotification		Enables the email address of your company's contact person. The email address is used as the return address for notifying vendors.

Configuration Parameter	Meaning
	<p>If the configuration parameter is set, One Identity Manager generates a list of system settings once a month and sends the list to One Identity. This list does not contain any personal data. You may review the most recent list at any time from in the Help Info... menu. The list will be reviewed by our customer support team who will look for material changes in a proactive effort to identify potential issues before they materialize on your system. The lists may be used by our R&D staff for analysis, diagnosis, and replication for testing purposes. We will keep and refer to this information for as long as your company remains on support for this product.</p>

Related Topics

- [Declaring the Job Server on page 337](#)
- [Overriding Scripts on page 472](#)

Enabling More Languages for Displaying and Maintaining Data

The default One Identity Manager installation is supplied in the languages "English - United States [en-US]" and "German - Germany [de-DE]". You can add other languages to the user interface and display text if required. In this case, you should translate the text before One Identity Manager goes live. There is a Language Editor in the Designer to help you do this. A special control element is provided in the One Identity Manager tools which aids multi-language input.

User Login Language

The display text appears in the language that the user logged on to the tool with. When you log in the first time, the system language is used for displaying the user interface. The user can change the login language in any administration tool. This sets the language globally for all the tools that the user uses. Therefore the user does not have to set the login

language in every tool separately. Changes to the login language take effect after the tool has been restarted.

All languages set with the option **Select in front-end** are available as login language.

To make other login language available

1. Select the category **Base Data | Localization | Cultures** in the Designer.
2. Select the language.
3. Set the option **Select in front-end**.

One Identity Manager Default Language

Maintenance of default data takes place in the default language. The default language for a One Identity Manager installation is "English - United States [en-US]". The default language is valid across the system. It is not recommended to change the default language during working hours

The ideal case is when the One Identity Manager language matches the user's administration tool login language. If these two settings are different, then the default language is used if no captions are found in the requested login language for a set of language dependent data.

Related Topics

- [Language Dependent Data Representation on page 328](#)

Displaying Country Information

The One Identity Manager requires country information at different stages, for example, employee country and state assignments are accessed when email notifications are created or IT Shop workflows are being determined.

Language, time zones, public holidays and working hours are mapped as well as countries and states. The basis data is loaded into the database during schema installation.

- NOTE:** Enable countries that are used cross-system during the database schema installation.
- NOTE:** All countries in which the Web interface is active must be enabled for working in different time zones.

To enable a country

1. Select the category **Base Data | Localization | Country** in the Designer.
2. Select a country

3. Set the option **Enabled**.
4. Save the changes.

Detailed information about this topic

- [Working in Different Time Zones on page 128](#)
- [Editing Countries on page 129](#)
- [Editing States on page 131](#)
- [Modifying Public Holidays on page 133](#)

Working in Different Time Zones

Time stamps, for example, insert dates or modification dates, are stored in the One Identity Manager with the respective UTC. The object layer transforms this time data into the currently valid time zone data when an object is loaded. The user, therefore, sees all the values in local time. When an object is saved the current time zone data is transformed into UTC data.

Countries and time zones are linked to another in the One Identity Manager schema. This makes it easier to find out the time zones when web fronts such as Web Portal are in use. This has to be done at start up when all countries have to enabled that have an active web front.

Related Topics

- [Editing Countries on page 129](#)
- [Editing States on page 131](#)
- [Modifying Public Holidays on page 133](#)

Determining Working Hours

Working hours are calculated for various function in One Identity Manager. For example, to determine working hours in the IT Shop or for determining reaction and solution times for calls in the Helpdesk Module.

In order to determined the correct working hours, ensure that a state is entered in the employee's master data. Weekends and public holidays are taken into account when calculating working hours. Public holiday are entered by state (county) in the One Identity Manager. You can add separate public holidays for states.

The settings of the following configuration parameters are also taken into when calculating working hours:

- QBM\WorkingHours\IgnoreHoliday
If this configuration parameter is set, public holidays are not included in the calculation of working hours.
- QBM\WorkingHours\IgnoreWeekend
If this configuration parameter is set, weekends are not included in the calculation of working hours.

Related Topics

- For more information, see [Modifying Public Holidays on page 133](#).

Editing Countries

The countries are loaded into the database during the schema installation and do not normally have to be customized.

NOTE: A new submenu item with the country name is shown for enabled countries in the category **Base Data | Localization | Country** in the Designer. Disabled countries are displayed under **Disabled | <country name>**.

To edit a country

1. Select the category **Base Data | Localization | Country** in the Designer.
2. Select a country.
3. Edit the master data.

To assign a culture to a country

- Select **View | Select table relations...** in the menu and enable the table **DialogCountryHasCulture**.
The **Culture** tab is displayed in the edit view.

To assign time zones to a country

- Select **View | Select table relations...** in the menu and enable the table **DialogCountryHasTimeZone**.
The **Time zones** tab is displayed in the edit view.

To assign public holidays to a country

- Select **View | Select table relations...** in the menu and enable the table **DialogCountryHoliday**.
The **Holidays** tab is displayed in the edit view.

To assign states to a country

- Select **View | Select table relations...** in the menu and enable the table **DialogState**.

The **States** tab is displayed in the edit view.

Table 60: Country Properties

Property	Description
Country name	Name of the country.
Description	Description of the country.
Enabled	If this option is set, this country is can be selected from the list in the administration tools. This helps to limit the selection of time zones and cultures.
Daylight saving time	This option specifies whether daylight saving time is taken into account when the difference to UTC time is calculated.
Hours (default)	Specify the working hours which apply across the country. Working hours are taken into account when calculating time periods, for example in the IT Shop. Default is Monday to Friday, 9am until 5pm respectively.
Country name (national language)	Name of the country in the national language using the national script.
Capital city (national language)	Name of the capital city in the national language using the national script.
Country code	International telephone code for the country.
International vehicle reg. ID	International identifier for vehicle license plates.
ISO code (2-letter)	Two letter country code for this country. This data has to comply with ISO 3166, a standard for coding geographical units.
ISO code (3-letter)	Three letter country code for this country. This data has to comply with ISO 3166, a standard for coding geographical units.
ISO code (numeric)	Numeric country code for this country. This data has to comply with ISO 3166, a standard for coding geographical units.
Object class	Object class for mapping country data in an LDAP schema.
Search mask	Search mask for mapping country data in an LDAP schema.

Property	Description
UTC Offset (average)	Average time difference between country and UTC time. This value is calculated by the DBQueue Processor based on the country's time zones.
Cultures	Language and language code of the country. The language specifies the language for email notification sent to users.
Time zones	The country's time zone. The calculation of processes that are time dependent, such as in the IT Shop, is taken in account by specifying a time zone.
Holidays	National holidays. National holidays are displayed in the category Base Data Localization Country <country name> Public holidays.
States	States within this country. States are displayed in the category Base Data Localization Country <country name> States.

Related Topics

- [Working in Different Time Zones on page 128](#)
- [Editing States on page 131](#)
- [Modifying Public Holidays on page 133](#)

Editing States

The states are loaded into the database during the schema installation and do not normally have to be customized.

To edit a state

1. Select the category **Base Data | Localization | Country | <country name> | States | <state name>** in the Designer.
2. Select a state.
3. Edit the master data.

To assign a culture to a state

- Select **View | Select table relations...** in the menu and enable the table **DialogStateHasCulture**.
The **Culture** tab is displayed in the edit view.

To assign time zones to a state

- Select **View | Select table relations...** in the menu and enable the table **DialogStateHasTimeZone**.
The **Time zones** tab is displayed in the edit view.

To assign public holidays to a state

- Select **View | Select table relations...** in the menu and enable the table **DialogStateHoliday**.
The **Holidays** tab is displayed in the edit view.

Table 61: State Properties

Property	Description
State	Name of the state.
State name (national language)	Name of the state in the national language using the national script.
Country	Enter the country that the state belongs to.
Enabled	Use this option to mark the states that your system uses.
Daylight saving time	Specifies whether daylight saving time is taken into account when the difference to UTC time is calculated.
Hours (default)	Specify the working hours which apply across the state. Working hours are taken into account when calculating time periods, for example in the IT Shop. Default is Monday to Friday, 9am until 5pm respectively.
Capital city	Name of the state's capital.
Capital city (national language)	Name of the capital city in the national language using the national script.
Short name	Abbreviation for the state according to ISO 3166-2 standard, for example "CA" for California or "SN" for Saxony.
UTC Offset (average)	Average time difference between country and UTC time. This value is calculated by the DBQueue Processor based on the state's time zones.
Language culture	Language and language code of the country. The language specifies the language for email notification sent to users.
Time zones	The country's time zone. The calculation of processes that are time dependent, such as in the IT Shop, is taken in account by specifying a time zone.
Holidays	National holidays. State holidays are displayed in the category Base Data Localization Country <country name> States <state name> Public holidays .

Related Topics

- [Working in Different Time Zones on page 128](#)
- [Editing Countries on page 129](#)
- [Modifying Public Holidays on page 133](#)

Modifying Public Holidays

The holidays are loaded into the database during the schema installation and do not normally have to be customized.

To display a holiday

1. To display a country's holidays, select the category **Base Data | Localization | Country | <country name> | Public holidays** in the Designer.
- OR -
To display a state's holidays, select the category **Base Data | Localization | Country | <country name> | States | <state name> | Public holidays** in the Designer.

Table 62: Public Holiday Properties

Property	Description
Date (ISO Format)	The date of the public holiday is entered in ISO format, for example "yyyy-mm-dd" where: yyyy - year, 4-digit mm - month, 2-digit dd - day, 2-digit
Public holiday name	Name of the holiday.
Public holiday name (national language)	Name of the holiday in the national language using the national script.
Country/State	Name of the country or state of the holiday.
Processing status	The process state is used for creating custom configuration packages.

Related Topics

- [Editing Countries on page 129](#)
- [Editing States on page 131](#)

Setting Up and Configuring Schedules

Frequently, you need to run processes and calculation tasks at specified time intervals. To make this possible, you can define schedules in the One Identity Manager. Schedules are required, for example, for scheduled execution of processes within process handling or for different calculation tasks within One Identity Manager. A schedule can be in control of several tasks. Execution times are configured in a schedule for the tasks to be executed.

- NOTE:** There are already schedules defined in the default One Identity Manager installation. Configure these according to your custom requirements.
- NOTE:** When a schedule is executed, all tasks assigned to the schedule are executed. Before you use a schedule on a repeated basis, check the effects of the process handling.

To edit schedules

1. Select the category **Base Data | General | Schedules** in the Designer.
 2. Select the schedule.
 - OR -
- Create a new schedule with the menu item **Object | New**.
3. Edit the schedule's master data.

Table 63: Schedule Properties

Property	Meaning
Name	Schedule ID. Translate the given text using the  button.
Description	Detailed description of the schedule. Translate the given text using the  button.
Table	Table whose data can be used by the schedule.
Enabled	Specifies whether the schedule is enabled or not. NOTE: Only active schedules are executed.
Time zones	Unique identifier for the time zone that is used for executing the schedule. Select either "Universal Time Code" or one of the time zones. NOTE: When you add a new schedule, the time zone is preset to that of the client from which you started the Designer.
Start (date)	The day on which the schedule should be run for the first time.
Validity period	Period within which the schedule is executed. <ul style="list-style-type: none">If the schedule will be run for an unlimited period, select the option Unlimited duration.

Property	Meaning
	<ul style="list-style-type: none"> To set a validity period, select the option Limited duration and enter the day the schedule will be run for the last time in End (date).
Occurs	<p>Interval in which the task is executed. Valid interval types are "Every minute", "Hourly", "Daily", "Weekly", "Monthly" and "Yearly".</p> <p>Specify the exact weekday for the interval type "Weekly". Specify the day of the month (1st - 31st) for the interval type "Monthly". Specify the day of the year (1 - 366) for the interval type "Yearly".</p> <p>NOTE: Schedules that have the sub-interval "31" and interval type "monthly" are run on the "31st of the month". The task is, therefore, only run in months with 31 days. The same is true of the interval type "yearly" and the sub-interval "366".</p>
Start time	<p>Fixed start time for the interval types "daily", "weekly", "monthly" and "yearly". Enter the time in local format for the chosen time zone.</p> <p>The start time for interval types "Every minute" and "Hourly" is calculated from the rate of occurrence and the interval type.</p>
Repeat every	Rate of occurrence for executing the schedule within the selected time interval. Select at least one weekday for the interval type "Weekly".
Last planned run/Next planned run	<p>Execution time calculated by the DBQueue Processor. They are recalculated each time a schedule is run. The time of the next run is calculated from the interval type, rate of occurrence and the start time.</p> <p>NOTE: The One Identity Manager provides the start information in the time zone of the client where the program was started. Changes due to daylight saving are taken into account.</p>

- NOTE:** Use the **Start** button to execute the schedule immediately. Take note:
- The last execution time is not updated when the schedule is started manually.
 - Before you start a schedule manually, check whether other processes will be executed as a result, that also need to be preprocessed by the One Identity Manager.

Related Topics

- [Calculating the Execution Time on page 135](#)

Calculating the Execution Time

The database schedule QBM_PWatchDog on <database> verifies the schedules that need to be run and their start times, at regular intervals. When the database scheduler is run, all tasks

are found that are within the valid time period and are enabled. A task is queued in the DBQueue for each schedule to be run. Then the time for the next scheduled run is calculated through the database schedule and entered in the schedule.

For tasks with interval types "minutely" or "hourly", the next schedule start time is calculated from the database schedule run time, the time zone and the rate of execution. The next execution time for tasks with the interval types "Daily", "Weekly", "Monthly" and "Yearly" are calculated from the given sub-interval and the start time.

Behavior of New Schedules

- The execution times for a new schedule are initially empty.
- Only the last scheduled run (date "01/02/1900") and the next scheduled run (date "12/30/1899") are entered the first time the database schedule is executed after a new schedule has been set up. The task is not executed.
- The next time the database schedule is executed, the next possible execution time (without taking the interval into account) is found. The task is not executed.
- The task is run if the execution time has been reached. When the next scheduled run is calculated, this time the interval will be taken into account.

Behavior of Modified Schedules

- When a schedule is modified, the next scheduled run field is cleared.
- The first time the database schedule is executed after the modifications, the next possible execution time (without taking the interval into account) is found. The task is not executed.
- The task is run if the execution time has been reached. When the next scheduled run is calculated, this time the interval will be taken into account.

Example 1

Run a schedule every 15 minutes.

Current time	Monday, 7/14/2014, 8:59 AM
Interval type	Every minute
Interval	15

The next time the database schedule is run at 9 am the schedule is only preset because the execution time is empty. The following run of the database schedule at 9:01 am determines the next scheduled run.

This results in the follow scenario:

Last scheduled run	1/2/1900, 00:00:00 AM
Next scheduled run	7/14/2014, 09:16:00 AM

This task is run for the first time the next time the database schedule runs at 9:16 am. The next scheduled run is calculated as follows:

Last scheduled run	7/14/2014, 09:16:00 AM
Next scheduled run	7/14/2014, 09:31:00 AM

Example 2

Run a schedule once a week on Wednesdays at 12 pm.

Current time	Monday, 7/14/2014, 8:59 AM
Interval type	Weekly
Interval	1
Sub interval	Wednesday
Start time	12:00 PM

The next time the database schedule is run at 9 am the schedule is only preset because the execution time is empty. The following run of the database schedule at 9:01 am determines the next scheduled run. The next possible execution time is determined based on the current date (07/14/2014) without taking the interval into account.

This results in the follow scenario after the database schedule has run:

Last scheduled run	1/2/1900, 00:00:00 AM
Next scheduled run	7/16/2014, 12:00:00 PM

The task is run for the first time when the database schedule is executed on 07/16/2014 at 12 pm. The next scheduled run is calculated as follows:

Last scheduled run	7/16/2014, 12:00:00 PM
Next scheduled run	7/23/2014, 12:00:00 PM

Example 3

Run a schedule every quarter on the 15th day of the month at 6pm.

Current time	Monday, 7/14/2014, 8:59 AM
Interval type	Monthly
Interval	3

Sub interval	15
Start time	6 PM

The next time the database schedule is run at 9 am the schedule is only preset because the execution time is empty. The following run of the database schedule at 9:01 am determines the next scheduled run. The next possible execution time is determined based on the current date (07/14/2014) without taking the interval into account.

This results in the follow scenario after the database schedule has run:

Last scheduled run	1/2/1900. 00:00:00 AM
Next scheduled run	8/15/2014, 18:00:00 PM

The task is run for the first time when the database schedule is executed on 08/15/2014 at 6 pm. The next scheduled run is calculated as follows:

Last scheduled run	8/15/2014, 18:00:00 PM
Next scheduled run	11/15/2014, 18:00:00 PM

Password Policies in One Identity Manager

One Identity Manager provides you with support for creating complex password policies, for example, for system user passwords, the employees' central password as well as passwords for individual target systems. Password polices apply not only when the user enters a password but also when random passwords are generated.

Predefined password policies are supplied with the default installation that you can user or customize if required. You can also define your own password policies.

Detailed information about this topic

- [Predefined Password Policies on page 139](#)
- [Editing Password Policies on page 139](#)
- [Custom Scripts for Password Requirements on page 142](#)
- [Restricted Passwords on page 145](#)
- [Testing a Password on page 145](#)
- [Testing Generating a Password on page 146](#)
- [Assigning a Password Policy on page 146](#)

Predefined Password Policies

You can customize predefined password policies to meet your own requirements, if necessary.

Password for logging into One Identity Manager

The password policy "One Identity Manager password policy" is used for logging into One Identity Manager. This password policy defined the settings for the system user passwords (DialogUser.Password and Person.DialogUserPassword) as well as the access code for a one off log in on the Web Portal (Person.Passcode).

The password policy "One Identity Manager password policy" is also labeled as the default and is used when no other password policy is found.

Password policy for forming employees' central passwords

An employee's central password is formed from the target system specific user accounts by respective configuration. The password policy "Employee central password policy" defines the settings for the central password (Person.CentralPassword).

IMPORTANT: Ensure that the password policy "Employee central password policy" does not violate the target system specific password requirements.

Password policies for target systems

A predefined password that you can apply to the user account password columns, is provided for every target system.

- NOTE:** When you update One Identity Manager version 7.x to One Identity Manager version 8.0, the configuration parameter settings for forming passwords are passed on to the target system specific password policies.
- IMPORTANT:** If you are not working with target system specific password policies, the default policy applies. In this case, ensure that the password policy "One Identity Manager password policy" does not violate the target system requirements.

Editing Password Policies

To edit a password policy

1. Select the category **Base Data | Security Settings | Password policies** in the Designer.
2. Select the password policy in the List Editor.
- OR -

- Create a new password policy from the menu **Object | New**.
3. Edit the password policy's master data.
 4. Save the changes.

Detailed information about this topic

- [General Master Data for a Password Policy on page 140](#)
- [Policy Settings on page 140](#)
- [Character Sets for Passwords on page 141](#)
- [Custom Scripts for Password Requirements on page 142](#)

General Master Data for a Password Policy

Enter the following master data for a password policy.

Table 64: Master Data for a Password Policy

Property	Meaning
Display name	Password policy name. Translate the given text using the  button.
Description	Spare text box for additional explanation. Translate the given text using the  button.
Error Message	Custom error message outputted if the policy is not fulfilled. Translate the given text using the  button.
Owner (Application Role)	Application roles whose members can configure the password policies.
Default policy	Mark as default policy for passwords.  NOTE: The password policy "One Identity Manager password policy" is marked as the default policy. This password policy is applied if no other password policies can be found.

Related Topics

- [Editing Password Policies on page 139](#)

Policy Settings

Define the following settings for a password policy on the **Password** tab.

Table 65: Policy Settings

Property	Meaning
Initial password	Initial password for new user accounts. If no password is given when the user account is added or a random password is generated, the initial password is used.
Password confirmation	Reconfirm password.
Min. Length	Minimum length of the password. Specify the number of characters a password must have.
Max. length	Maximum length of the password. Specify the number of characters a password can have.
Max. errors	Maximum number of errors. Set the number of invalid passwords. If the user has reached this number the user account is blocked.
Validity period	Maximum age of the password. Enter the length of time a password can be used before it expires.
Password history	Enter the number of passwords to be saved. If the value '5' is entered, for example, the last 5 passwords of the user are saved.
Min. password strength	Specifies how secure the password must be. The higher the password strength, the more secure it is. The password strength is not tested if the value is '0'. The values '1', '2', '3' and '4' gauge the required complexity of the password. The value '1' demands the least complex password. The value '4' demands the highest complexity.
Name properties denied	Specifies whether name properties are permitted in the password.

Related Topics

- [Editing Password Policies on page 139](#)

Character Sets for Passwords

Use the **Character classes** tab to specify which characters are permitted for a password.

Table 66: Character Classes for Passwords

Property	Meaning
Min. letters	Specifies the minimum number of alphabetical characters the password must contain.

Property	Meaning
Min. number lower case	Specifies the minimum number of lowercase letters the password must contain.
Min. number uppercase	Specifies the minimum number of uppercase letters the password must contain.
Min. number digits	Specifies the minimum number of digits the password must contain.
Min. number special characters	Specifies the minimum number of special characters the password must contain.
Permitted special characters	List of permitted characters.
Denied special characters	List of characters, which are not permitted.
Max. identical characters in total	Maximum number of identical characters that can be present in the password in total.
Max. identical characters in succession	Maximum number of identical character that can be repeated after each other.

Related Topics

- [Editing Password Policies on page 139](#)

Custom Scripts for Password Requirements

You can implement custom scripts for testing and generating password if the password requirements cannot be mapped with the existing settings options. Scripts are applied in addition to the other settings.

Detailed information about this topic

- [Script for Checking a Password on page 142](#)
- [Script for Generating a Password on page 144](#)

Script for Checking a Password

You can implement a check script if additional policies need to be used for checking a password, which cannot be mapped with the available settings.

Syntax for Check Scripts

```
Public Sub CCC_CustomPwdValidate( policy As VI.DB.Passwords.PasswordPolicy, spwd As System.Security.SecureString)
```

With parameters:

policy = password policy object

spwd = password to test

 **TIP:** To use a base object, take the property Entity of the PasswordPolicy class.

Example for a script for testing a password

A password cannot have '?' or '!' at the beginning. The script checks a given password for validity.

```
Public Sub CCC_PwdValidate( policy As VI.DB.Passwords.PasswordPolicy, spwd As System.Security.SecureString)

    Dim pwd = spwd.ToInsecureArray()

    If pwd.Length>0

        If pwd(0)="?" Or pwd(0)!="!"

            Throw New Exception(#LD("Password can't start with '?' or '!')#)

        End If

    End If

    If pwd.Length>2

        If pwd(0) = pwd(1) AndAlso pwd(1) = pwd(2)

            Throw New Exception(#LD("Invalid character sequence in password")#)

        End If

    End If

End Sub
```

To use a custom script for checking a password

1. Create your script in the category **Script Library** in the Designer.
2. Edit the password policy.
 - a. Select the category **Base Data | Security Settings | Password policies** in the Designer.
 - b. Select the password policy in the List Editor.
 - c. Enter the name of the script to test the password in **Check script** on the **Scripts** tab.
 - d. Save the changes.

Related Topics

- [Script for Generating a Password on page 144](#)
- [Editing Password Policies on page 139](#)
- [Scripts in the Script Library on page 467](#)

Script for Generating a Password

You can implement a generating script if additional policies need to be used for generating a random password, which cannot be mapped with the available settings.

Syntax for Generating Script

```
Public Sub CCC_PwdGenerate( policy As VI.DB.Passwords.PasswordPolicy, spwd As System.Security.SecureString)
```

With parameters:

```
policy = password policy object  
spwd = generated password
```

 **TIP:** To use a base object, take the property Entity of the PasswordPolicy class.

Example for a script to generate a password

The script replaces the invalid characters '?' and '!' in random passwords.

```
Public Sub CCC_PwdGenerate( policy As VI.DB.Passwords.PasswordPolicy, spwd As System.Security.SecureString)  
  
    Dim pwd = spwd.ToInsecureArray()  
    ' replace invalid characters at first position  
    If pwd.Length>0  
        If pwd(0)="?" Or pwd(0)!=""  
            spwd.SetAt(0, CChar("_"))  
        End If  
    End If  
End Sub
```

To use a custom script for generating a password

1. Create your script in the category **Script Library** in the Designer.
2. Edit the password policy.
 - a. Select the category **Base Data | Security Settings | Password policies** in the Designer.

- b. Select the password policy in the List Editor.
- c. Enter the name of the script to generate a password in **Generation script** on the **Scripts** tab.
- d. Save the changes.

Related Topics

- [Script for Checking a Password on page 142](#)
- [Editing Password Policies on page 139](#)
- [Scripts in the Script Library on page 467](#)

Restricted Passwords

You can add words to a list of restricted terms to prohibit them from being used in passwords.

 **NOTE:** The restricted list applies globally to all password policies.

To add a term to the restricted list

1. Select the category **Base Data | Security Settings | Restricted passwords** in the Designer.
2. Create a new entry with the menu item **Object | New** and enter the term to excluded to the list.
3. Save the changes.

Testing a Password

When you test a password, all the password policy settings, custom scripts and the restricted passwords are taken into account.

To test whether a password conforms to the password policy

1. Select the category **Base Data | Security Settings | Password policies** in the Designer.
2. Select the password policy in the List Editor.
3. Select the **Test** tab.
4. Select the table and object to be tested in **Base object for test**.
5. Enter a password in **Enter password to test**.

A display next to the password shows whether it is valid or not.

Testing Generating a Password

When you generate a password, all the password policy settings, custom scripts and the restricted passwords are taken into account.

To generate a password that conforms to the password policy

1. Select the category **Base Data | Security Settings | Password policies** in the Designer.
2. Select the password policy in the List Editor.
3. Select the **Test** tab.
4. Click **Generate**.

This generates and displays a password.

Assigning a Password Policy

You can assign password policies to system user passwords, the employees' central password as well as passwords for individual target systems. Assign a password policy to the base object to which it should apply.

- The predefined password policy "One Identity Manager password policy" is assigned to the system user passwords (`DialogUser.Password` and `Person.DialogUserPassword`) as well as the employee's access code (`Person.Passcode`).
- The predefined password policy "Employee central password policy" is assigned to the employee's central password (`Person.CentralPassword`).

If you want to apply another password policy to the password column, change the password policy assignment in the Manager.

To change a password policy's assignment

1. Select the category **Employees | Basic configuration data | Password policies** in the Manager.
2. Select the password policy in the result list.
3. Select **Assign objects** in the task view.
4. Select the assignment you want to change in **Assignments**.
5. Select the new password policy to apply from the **Password Policies** menu.
6. Save the changes.

To reassign a password policy

1. Select the category **Base Data | Security Settings | Password policies** in the Designer.
2. Select the password policy in the result list.

- Click **Add** in the **Assignments** section and enter the following data.

Table 67: Assigning a Password Policy

Property	Description
Password column	The password column's identifier.
Apply to	<p>Application scope of the password policy.</p> <p>To specify an application scope</p> <ol style="list-style-type: none"> Click ... next to the text box. Select the table which contains the password column under Table. Select the specific base objects under Apply to. Click OK.

- Save the changes.

NOTE: Permitted base objects and their password columns are defined in the view QBMVPwdPolicyColumns. You can customize this view, if required, by extending it with the type "Union". [For more information, see Database Views of Type "Union" on page 183.](#)

If you create new custom tables, add the customizer "VI.Common.Customizer.PwdPolicyColumnEntityLogic" to the table definition.

Reloading Changes Dynamically

Table 68: Configuration Parameter for Reloading Changes

Configuration parameter	Meaning
Common\CacheReload	Checks for values to be reloaded in the DialogSemaphore table.
Common\CacheReload\Interval	This parameter defines a time interval in seconds after which the values in the table DialogSemaphore are checked during access. This parameter is only evaluated when the parameter "Common\CacheReload\Type" is set to "TIMER".

Configuration parameter	Meaning
Common\CacheReload\Type	<p>This parameter defines which method is used to check the validity of the cached data.</p> <p>Permitted values are:</p> <ul style="list-style-type: none"> • ALWAYS (check every access) • NEVER (never check) • TIMER (check after time interval)

Cached system data can be dynamically reloaded if it has changed. The changes are reloaded automatically in background. An exception to this are changes that effect the character of the user interface. These changes are only reloaded after requesting confirmation from the user. The semaphore is incremented when changes are made. The semaphore is calculated when the DBQueue Processor is run.

Which columns are reloaded is defined in the data model . You can find an overview of the semaphore in the category **Base data | Advanced | Semaphore** in the Designer.

- To reload data after changes to a column, the column must be assigned to the semaphore.
- To reload data after inserting or deleting in a table, the primary column key must be assigned to the semaphore.

Table 69: Changes to Reload

Changes	Semaphore
Script assembly and Customizer	Assembly
Calculate column dependencies	BulkdDependencies
Names, such as column headings or display text	Caption
Configuration parameter	Config
Countries and time zones	Country
Parts of user interface	Dialog
Use of special program functions	Feature
Icons	Image
Tables, columns, table and column identifiers, objects, tasks	Model
Notification	Notification
Rights and group memberships	Right
Software revisions status (for software update)	SoftwareRevision
Statistic definitions	DashBoardDef

Changes	Semaphore
Statistical content	DashBoardContent
Module dependencies	ModuleDepend
User data stored in memory.	UserDataResident
Changes to synchronization configuration	DPRConfiguration
Changes to module dependencies	ModuleDepend
Changes to the Web Portal configuration	AEDS
Changes to predefined SQL queries	LimitedSQL

TimeTrace Databases

The history databases must be declared in the One Identity Manager database if archive data is to be included in the One Identity Manager TimeTrace function. Historical data is displayed in the TimeTrace view in the Manager.

To link a History Database into a TimeTrace

1. Select the category **Base Data | General | TimeTrace databases** in the Designer.
2. Select the menu item **Object | New**.
3. Enter the One Identity Manager History Database's name.
4. Declare the **Connection parameter** .
 - a. Open the connection data dialog box using the [...] button next to the text box.
 - b. Enter the database connection data for the One Identity Manager History Database.

Table 70: SQL Server Database Connection Data

Data	Description
Server	Database server.
Windows authentication	Specifies whether Windows authentication is used. This type of authentication is not recommended. If you decide to use it anyway, ensure that your environment supports Windows authentication.
User	Database user.
Password	Database user password.
Database	Database.

Table 71: Oracle Database Connection Data

Data	Description
Direct access (without Oracle client)	Set this option for direct access. Deactivate this option for access via Oracle Clients. Which connection data is required, depends on how this option is set.
Server	Database server.
Port	Oracle instance port.
Service name	Service name.
User	Oracle database user.
Password	Database user password.
Data source	TNS alias name from TNSNames.ora.

- c. Click **OK**.
 5. Save the changes.
- NOTE:** Set the option **Disabled** to disable the connection at a later time. If a One Identity Manager History Database is disabled, it is not taken into account when determining change data in the TimeTrace.

Machine Roles and Server Functions

A machine role describes the role a computer or server assumes in a One Identity Manager system. You can give each computer or server several roles. This means, one or more machine roles can be assigned. You select machine roles when One Identity Manager components are installed.

Machine roles are structured hierarchically. If you select a machine role at installation, all parent machine are also assigned.

An example of machine role structure

```
Server
  Job server
    Active Directory
```

If you select the machine role "Active Directory" at installation, the additional machine roles "Job Server" and "Server" are assigned.

Some machine roles, for example, "Web" cannot be actively selected during installation. These machine roles are automatically assigned when different web applications are installed with the Web Installer.

Machine roles for installing the One Identity Manager Service are linked with server functions. The server function defines the functionality of a server in One Identity Manager. One Identity Manager processes are handled depending on the server function. The server functions available are predefined when a server is installed, based on the selected machine role.

Example for the connection between machine roles and server functions.

The machine "Active Directory" is linked to the server function "Active Directory connector". Therefore, when you set up a Active Directory synchronization project after the machine role is installed, the server is available as synchronization server in One Identity Manager.

The installation packages and files to be installed on the computer or server are specified in a machine role. The information about the machine role, the installation package and the files is saved in the file `InstallState.config` during installation and are thus available for automatic software update.

If you import new files into the One Identity Manager database with the Software Loader, you should assign the files to a machine role. This ensures that the files are distributed by automatic software update.

To display information about machine roles

- Select the category **Base Data | Installation | Machine roles** in the Designer.

To display information about server functions

- Select the category **Base Data | Installation | Server functions** in the Designer.

Related Topics

- [Files for Software Update on page 151](#)
- [Editing a Job Server on page 340](#)

Files for Software Update

All the files included in an installation of the One Identity Manager are stored in the One Identity Manager database with name, repository and executable code. The One Identity Manager tool to which each file belongs, for example the Manager or the One Identity Manager Service, is entered in the database. In order to distribute new or changed custom files, such as custom form archives, through automatic software updating, the files are loaded into the Software Loader database with the program "One Identity Manager".

NOTE: You will find detailed information about updating One Identity Manager and about how automatic software update works in the One Identity Manager Installation Guide.

Detailed information about this topic

- Importing New Files into a One Identity Manager Database on page 152
- Exporting Files from a One Identity Manager Database on page 154
- Configuring Files on page 155

Importing New Files into a One Identity Manager Database

In order to distribute new or changed custom files, such as custom form archives, through automatic software updating, the files are loaded into the One Identity Manager database with the program "Software Loader".

To import files

1. Open the Launchpad and select **Files for software update**. This starts the program "Software Loader".
2. Select **Import into database** on the start page.
3. Enter the One Identity Manager database connection credentials on the **Connect to database** page.
4. Specify the file to be imported on the **Select files** page.
 - a. Select the base directory where the files can be found.

The status and file size of all the files in the selected directory are displayed in the file list. The status is determined from the file information in the database. To test the file version, the file size and the hash value are determined and compared to the entry in the database.

NOTE: Take note when selecting the base directory that a directory tree is not created accidentally.

Table 72: Status Meaning

Status	Meaning
Version unknown	The file belongs to the known files but has not been loaded into the database yet. There is no version information in the database.
File unknown	This file is new. The file is in the list of known files but has not been loaded in the database yet. There is no version information in

Status	Meaning
	the database.
Version OK	The file version matches the version in the database.
Version changed	This version of the file has change with respect to the version in the database.
b.	Mark the files to be loaded into the One Identity Manager database. You can multi-select files with SHIFT + SELECT or CTRL + SELECT .
TIP:	Click with the mouse in a column header to sort by the selected column.
TIP:	Modified files can be preselected in the context menu.

Table 73: Meaning of Context Menu Items

Context Menu Item	Meaning
Open all directories	All the directories are opened.
Open all modified files	All the files with the status "Version changed" are selected. Files in the subdirectories are only selected if the directories are opened beforehand.

5. Apply a change label on the **Select change label** page.

Issue a change label to mark files in order to simplify the transfer of new files between various databases (test database, development database, operational database). Change labels are offered in the program "Database Transporter" as export criteria when a customer transport package is created.

- a. Select **Assign files to following change label**.
 - b. Select the change label using the button next to this option.
6. The files are loaded straight from the One Identity Manager database. After successfully loading the files into the database, the semaphore value "Softwarerevision" is updated in the database by the DBQueue Processor. In this way, the files to be updated are added to the update file list at the next semaphore test and distributed to the workstations and servers.

7. Specify other file settings on the **Assign machine roles** page.
 - a. Assign a computer role to the files.
 - b. To specify other settings, click ... next to the file name.

Table 74: Other File Settings

Setting	Description
Source directory	Path in the installation source.
Create backup	A copy of the file is made if the software is updated automatically.
No update	The file is not updated automatic software update.

8. Click **Finish** on the last page to end the program.

Related Topics

- [Exporting Files from a One Identity Manager Database on page 154](#)
- [Configuring Files on page 155](#)
- [Machine Roles and Server Functions on page 150](#)

Exporting Files from a One Identity Manager Database

In order to equip individual Job servers with the newest software version, you have to export the files from the One Identity Manager database. The program "Software Loader" exports the files from the database.

To export files

1. Open the Launchpad and select **Files for software update**. This starts the program "Software Loader".
2. Select **Export from database** on the start page and click **Next**.
3. Enter the One Identity Manager database connection data on the **Connect to database** page and click **Next**.
4. Specify which data to export on the **Select files** page.
 - a. Specify the destination directory to which to export the data.
 - b. Mark the files to export. You can multi-select files with **SHIFT + CLICK** or **CTRL + CLICK**.

TIP: Click with the mouse in a column header to sort by the selected column.

Exportable files are displayed with their status and file size. The program checks whether One Identity Manager files already exist in the given directory in order to determine the status. If this is the case, the files are updated, otherwise the files are recreated.

Table 75: Status Meaning

Status	Meaning
Unknown file	The file has not been exported from the database into the given directory yet.
Version OK	The file version matches the version in the database.
Version changed	This version of the file has change with respect to the version in the database.

- c. Click **Next**.
5. The marked files are export to the given directory. This may take some time depending on the number of files selected. The export steps are displayed on the page **Uploading files**. Any export errors are displayed.
 - After exporting is complete, click **Next**.
6. Click **Finish** on the last page to end the program.

Related Topics

- [Importing New Files into a One Identity Manager Database on page 152](#)
- [Configuring Files on page 155](#)

Configuring Files

Specify whether a backup should be made of an existing file when the software is updated.

NOTE: You specify these properties when you import new files with the "Software Loader" program.

To configure the file properties

1. Select the category **Base data | Installation | One Identity Manager software** in the Designer.

2. Select a file.
3. Edit the following master data.

Table 76: File Properties

Property	Description
Create backup	A backup of the existing file is made during the software update for files that are labeled with this option.
No update	The file is not updated automatic software update.

⚠ CAUTION: Do not change any other properties as it may result in errors during automatic software update.

Related Topics

- [Importing New Files into a One Identity Manager Database on page 152](#)
- [Exporting Files from a One Identity Manager Database on page 154](#)

Operating Systems in Use

References to operating systems are required at various places in the One Identity Manager schema. The One Identity Manager supplies predefined entries for operating systems. If other operating systems are required, you can create the necessary entries.

To set up an operating system

1. Select the category **Base Data | Advanced | Operating systems** in the Designer.
2. Select the menu item **Object | New**.
3. Edit the master data.

Table 77: Operating System Properties

Property	Description
Processing Status	Object processing status. The process state is used for creating customer configuration packages.
Operating system	Short name of the operating system.
Name	Long name of the operating system.
Client operating system	Specifies whether the operating system can be used for workstations.

Property	Description
Server operating system	Specifies whether the operating system can be used for servers.
Version	Version number of the operating system.

System Configuration Reports

The category **Documentation** contains different reports about system configuration and customizations. When you select an entry in this category the corresponding report is generated. Generating the report may take some time depending on its size.

Table 78: System Configuration Reports

Report	Contents
System configuration	This report contains the description and settings of enabled configuration parameters.
Processes	This report contains the description of all enabled default processes. The process steps and their parameters as well as the scripts used and configuration parameters for a process are listed.
Process Components	The report contains the description of all process components with their tasks and parameters.
Templates	This report contains the descriptions of all default templates including affected columns, scripts used and configuration parameters.
Formatting rules	This report contains the description of all default formatting rules including scripts used and configuration parameters.
Scripts	This report contains the description of all default scripts including configuration parameters used. Process usage, process steps, templates, formatting rules and scripts are listed for each script.
Full report	Full report about system configuration. This groups together all the information in the separate reports.

Table 79: Reports Available for Customizing

Report	Contents
System configuration	This report contains the description and settings of enabled configuration parameters.
Processes	This report contains the description of all enabled default processes. The process steps and their parameters as well as the scripts used and configuration parameters for a process are listed.

Report	Contents
Templates	This report contains the descriptions of all default templates including affected columns, scripts used and configuration parameters.
Formatting rules	This report contains the description of all default formatting rules including scripts used and configuration parameters.
Scripts	This report contains the description of all default scripts including the configuration parameters used. Process usage, process steps, templates, formatting rules and scripts are listed for each script.
One Identity Manager schema	This report contains the description of custom One Identity Manager schema extensions (tables and columns).
Full report	Full report about system configuration. This groups together all the information in the separate reports. In addition, information about customized database objects is also listed, such as database procedures, functions, triggers or view definitions.

Using Predefined Database Queries

Direct database queries cannot be carried out from front-ends and web application when an application server is implemented due to security issues. Database queries, those required on forms, for example, must be formulated as "predefined database queries" in One Identity Manager. Database queries are always executed with the permissions of the current user. Prefined database queries must be assigned to a permissions group.

A wizard in the Web Designer helps you to create database queries for the Web Portal and to link it with at least one permissions group. You can enter more predefined database queries in the Designer.

To create predefined database queries

1. Select the category **Base Data | Advanced | Predefined SQL** in the Designer.
2. Select the menu item **Object | New**.
3. Edit the master data.

Table 80: Properties of Predefined Database Queries

Property	Description
Processing status	Object processing status. The process state is used for creating customer configuration packages.
Description	Spare text box for additional explanation.

Property	Description
Identifier for SQL code	Unique identifier used to identify the query
Code	Full database query SQL syntax. You can also use SQL parameters in the query.

4. Assign permissions groups.
 - a. Select **View | Select table relations...** in the menu and enable the table `QBMGroupHasLimitedSQL`.
 - b. Select one or more permissions groups on the **Permissions group** tab.

Managing Custom Database Objects within a Database

Database object information such as custom database tables, columns, database procedures, functions, triggers or view definitions need to be stored in the database for creating transport packets with the program "Database Transporter" or for creating reports using the system configuration. The DBQueue Processor checks and updates this data.

 **NOTE:** It is not usually necessary to edit the data manually although you might edit the comment for use in reports.

To customize database objects

1. Select the category **Base Data | Advanced | Modified SQL** in the Designer.
2. Select the database object.
3. Modify **Remarks**.

Table 81: Database Object Properties

Property	Description
Processing status	The process state is used for creating custom configuration packages.
Remarks	Additional comments, for example, for using in system configuration reports.
Name	Database object name
Modified	Specifies whether the database object has been changed
Sort order	Order in which the data is presented.
Type	Type of database object, for example, procedure, function, trigger, index, view, custom table, custom column.

Related Topics

- [Transporting One Identity Manager Schema Customizations on page 544](#)

The One Identity Manager Data Model

The One Identity Manager database distinguishes between user data and meta data. Reference data is specified by the application data model and meta data by the system data model. Meta Data includes data for specifying the application data model, for regulating access rights and to influence and control system behavior such as modifying the One Identity Manager interface to meet customer requirements.

The data model is mapped and edited in Designer under the category **One Identity Manager Schema**. This category displays default tables delivered by us and the customer specific tables together with their properties. You also get an overview of the value templates and formatting rules for the database columns and preprocessor dependencies.

- ❶ **IMPORTANT:** Use the program, "Schema Extension" to extend the One Identity Manager data model. Schema extensions are added to the database using "Schema Extension" and the necessary extensions are made in the One Identity Manager data model.
- ❷ **NOTE:** Reports about system configuration and customization of tables and column properties can be found in the category **Documentation**.

Detailed information about this topic

- [Basics of the Data Model on page 162](#)
- [General Advice for Editing Table and Column Definitions on page 165](#)
- [Mapping Table Definitions on page 172](#)
- [Mapping Column Definitions on page 188](#)
- [Mapping Table Relations on page 208](#)

Related Topics

- [Supporting File Groups on page 212](#)
- [Custom Schema Extensions on page 526](#)

Basics of the Data Model

The following types of table are used in the One Identity Manager data model.

Table 82: Table Types

The type of the table.	Description
Simple table	<p>Simple tables are the most common form for storing data.</p> <p>The following columns are defined for simple tables:</p> <ul style="list-style-type: none">• Primary key• Object key (XObjectKey)
Many-to-many table	<p>Many-to-many or M:N tables contain the relationships between two other tables.</p> <p>The following columns are defined for many-to-many tables:</p> <ul style="list-style-type: none">• Twin column primary key as foreign keyBoth columns are defined as foreign key columns on the referenced table.• Object key (XObjectKey) <p>Many-to-many tables are also called assignment tables in this documentation.</p>
Many-to-all table	<p>Many-to-all or M:all tables are a particular type of assignment table, which was developed for One Identity Manager.</p> <p>M:all tables are implemented if part of an assignment (all) can reference different tables, meaning dynamically determined. Valid tables can be limited in this way. For example, the owner of a group can be a user account or a group.</p> <p>Furthermore M:all tables are used if additional information about an assignment is mapped, for example, an assignment's validity period.</p> <p>The following columns are defined for M:all tables:</p> <ul style="list-style-type: none">• Primary key• Foreign key defined as NOT NULL that references the primary key of another table.• Dynamic foreign key defined as NOT NULL that reference the object key (XObjectKey) of the valid tables.• Object key (XObjectKey) <p>You can define more foreign keys and dynamic foreign keys. These columns must be defined as NULL.</p>
Work tables	Worktables are used to store data for which objects cannot be created. No primary key is required for work tables. However, you can define up to two primary keys.

Table 83: Required Columns

Column	Description
Primary key	<ul style="list-style-type: none">If objects are generated from the table through the object layer, the table requires a primary key.If a table represents a many-to-many mapping, a two column primary key is defined. Both primary key columns are defined as foreign key columns in the referenced tables.No primary key is required for work tables.Primary key columns must be defined in Globally Unique Identifier (GUID) format. Default GUID's are created in the format [0-9,a-f](8-4-4-4-12).
XObjectKey	<p>If objects are generated from the table through the object layer, the table must have an object key column. The object key (XObjectKey) is a unique key, which is capable of referencing every object in the database.</p> <p>XObjectKey syntax:</p> <pre><Key><T>TableName</T><P>PrimaryKeyOfRow</P></Key></pre> <p>with:</p> <ul style="list-style-type: none">TableName: table namePrimaryKeyOfRow: primary key column's GUIDAn additional <P>SecondPrimaryKeyOfRow</P> is used for two column primary keys.
Foreign key	<ul style="list-style-type: none">The name of the foreign key column corresponds, as far as possible, to the name of the references table's primary key.Foreign key columns are defined in GUID format.A table is reference through the referenced table's primary key.If the foreign key column is part of a many-to-all table, the column in the One Identity Manager schema is labeled with the option Part of key of many-to-all table (DialogColumn.IsMAllKeyMember).
Dynamic foreign key	<ul style="list-style-type: none">You can use dynamic foreign keys if a reference is able to point to different tables. For example, a user account's manager (table ADSAccount.ObjectKeyManager) can be another user account (table ADSAccount) or a group (ADSGroup).Dynamic foreign keys reference object key (XObjectKey of the permitted tables).Permitted tables can be limited. All tables are permitted, if there are no restrictions.

Column	Description
	<ul style="list-style-type: none"> A dynamic foreign key is labeled in the One Identity Manager schema with the option Dynamic foreign key column (DialogColumn.IsDynamicFK). If the dynamic foreign key is part of a many-to-all table, the column in the One Identity Manager schema is labeled with the option Part of key of many-to-all table (DialogColumn.IsMAllKeyMember).
XDateInserted	The columns contain information about which users made changes at what times. The columns must always exist together.
XDateUpdated	
XUserInserted	
XUserUpdated	
XTouched	This column contains an element's processing status. The process state is used for creating custom configuration packages.
XMarkedForDeletion	<p>This column defines whether the object is marked for deletion. The column exists when:</p> <ul style="list-style-type: none"> The deferred deletion function can be applied to the table. The table is synchronized again a target system and pending objects can be handled.
XOrigin	<p>In order to determine the origin of an assignment, a column XOrigin is defined in a many-to-many or a many-to-all table. The individual bit positions provide the origin of a membership.</p> <p>For more detailed information about calculation, see the One Identity Manager Identity Management Base Module Administration Guide.</p>
XIsInEffect	<ul style="list-style-type: none"> To discover whether an assignment is in effect, a column XIsInEffect is defined on an assignment table. The column only exists if the number of assignments differs from the number of effective assignments. <p>For example, if an employee is disabled, marked for deletion or classified as a security risk, inheritance of company resources can be prohibited for this employee. The group assignment is maintained, this assignment, however, will not be put in effect.</p> <ul style="list-style-type: none"> If column XIsInEffect is used, a column XOrigin must exist. <p>For more detailed information about calculation, see the One Identity Manager Identity Management Base Module Administration Guide.</p>

Column	Description
XDateSubItem	<p>This column contains the change date for dependencies and is required in order to take membership changes in a target system into account during synchronization and provisioning.</p> <p>For more detailed information about synchronizing and provisioning memberships, see the One Identity Manager Target System Synchronization Reference Guide.</p>

General Advice for Editing Table and Column Definitions

- The application and system data model table definitions are stored in the table DialogTable. These predefined One Identity Manager data model table properties are maintained by the schema installation and cannot be edited apart from a few properties.
- Column definitions for application and system data model tables are kept in the table DialogColumn. Predefined column properties of the One Identity Manager data model are maintained through schema installation and cannot be edited apart from a few exceptions.
- Use the program One Identity Manager to customize Schema Extension database model extensions. Schema extensions are added to the database using "Schema Extension" and the necessary extensions are made in the One Identity Manager data model.
- Custom tables are published in the table DialogTable by the program Schema Extension when the schema is extended. You must customize the properties to suit your requirements in the Designer.
- Custom columns are published in the table DialogColumn by the program Schema Extension when the schema is extended. You must customize the properties to suit your requirements in the Designer.
- Run the consistency checks after the custom schema extensions and after changes to table and column definitions and apply the repair methods.
- Some table and column modifications require recompiling the database.
- Reports about system configuration and customization of tables and column properties can be found in the category **Documentation**.

Working with the Schema Editor

You get a complete overview of the One Identity Manager data model in the Schema Editor. The editor is started from the program "Designer" and opens in the document view. Only

additional Schema Editor functions are described in the following.

Menu Items

The following items are added to the menu bar when the editor starts.

Table 84: Menu Items Added by the Editor

Menu	Menu Item	Meaning
Schema	Find table...	Opens the search dialog.
	Export SPML schema information...	Opens the dialog for exporting SPML schema data.
	Check for circular references	Table dependencies are tested for circular references. This menu item is enabled if dependencies are displayed in the schema overview (menu Options Dependencies).
	Reload schema	Updates the schema overview.
Options	Data model	Displays the entire One Identity Manager schema in the schema overview.
	Dependencies	Only displays dependencies in the schema overview.
	Animate movements	Navigates to the selected database table by scrolling the display.
	Hide table relations	Hides the relationships between database tables that are displayed in the schema overview.
	Show disabled columns	Shows/hides disabled columns. This setting applies to the schema overview and the simple schema view.
	Show all columns	Shows all the columns in the schema overview.
	Hide small tables	Hides the columns of small database tables in the schema overview (up to 20 columns).
	Hide all columns	Hides all the columns in the schema overview.
	Save table layout	Saves changes made to the table layout.

Menu	Menu Item	Meaning
View	properties	Shows/hides the object edit view.
	Select columns...	Opens a dialog window for selecting columns to display in the simple schema view.
Help	Help for the One Identity Manager schema.	Opens the help on this topic.
	Schema Editor help	Opens the editor help.

Table 85: Meaning of Toolbar Icons

Icon	Meaning
	Shows entire data model.
	Shows dependencies.
	Exports SPML schema data.
	Tests circular references. This menu item is enabled if dependencies are displayed in the schema overview (menu Options Dependencies).
	Searches for text.
	Zooms in on the view.
	Zooms out on the view.
	Shows all the columns in the schema view.
	Hides the columns of small database tables in the schema view (up to 20 columns).
	Hides all the columns in the schema view.
	Show disabled columns.
	Update the view.

Views in the Schema Editor

The Schema Editor has different views for displaying and editing the One Identity Manager schema:

Table 86: Schema Editor Views

View	Description
Simple schema view	Tables and column definitions are displayed in tabular form in the simple schema view. You can use this view to quickly edit properties. This view is opened from the Designer's task view when a table or a column is selected.
Properties	In the edit view you can edit a database table, a database column or a table relation. The properties of a table, a column or a table relation are displayed depending on what you select. There is a default context menu available for the input fields.
Schema overview	You can get a overview of the entire One Identity Manager data model in the schema overview.

Related Topics

- [Working with the Schema View on page 168](#)

Editing Multiple Properties

It is possible to edit several column definitions or table relations at the same time in the simple schema view.

- Use **SHIFT + SELECT** or **CTRL + SELECT** to select entries in the simplified schema view.

Entries in the list that have different input are labeled with the  icon in the edit view. When one field is edited and saved, all the other entries are given the same value.

Working with the Schema View

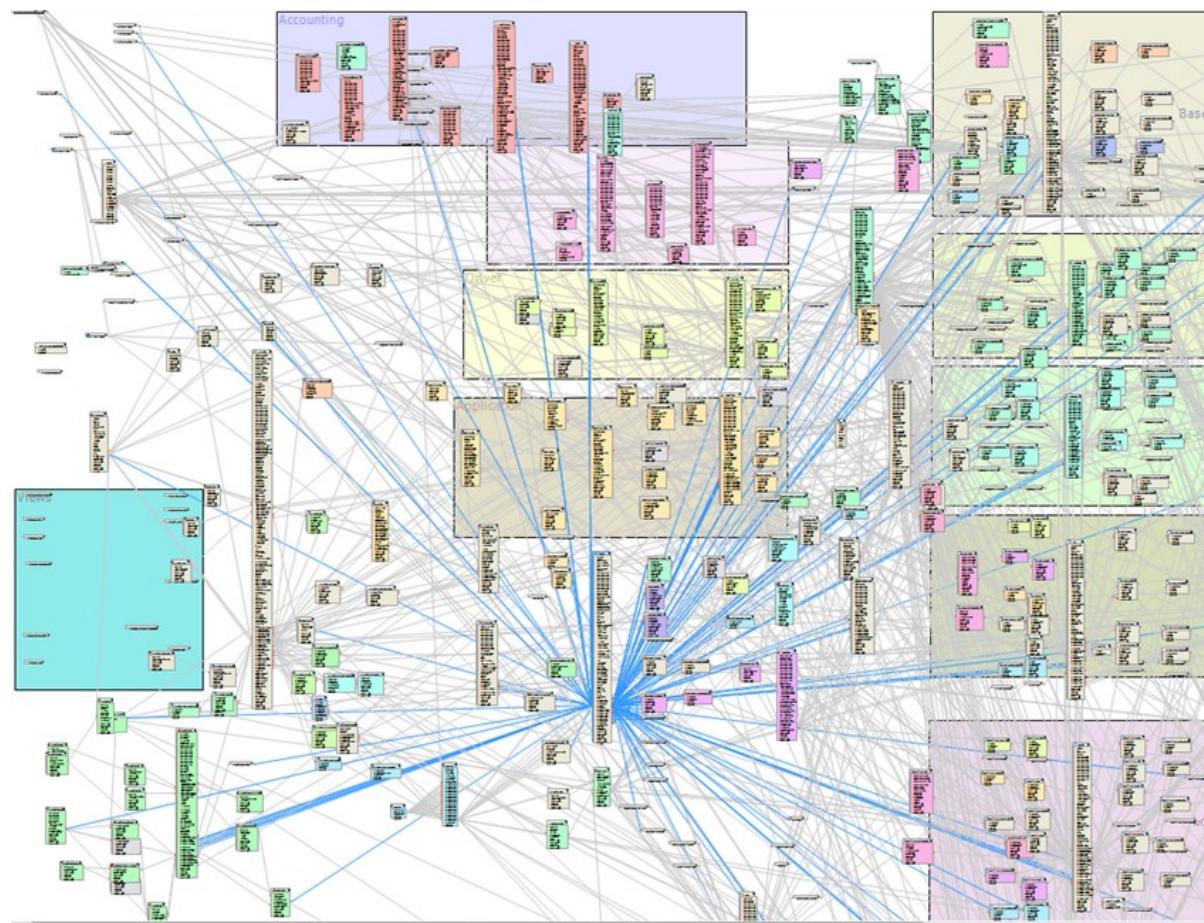
You can get a overview of the entire One Identity Manager data model in the schema overview.

NOTE: Tables and columns that are deactivated through preprocessor conditions are not shown in the view.

To display the schema overview

1. Select the category **One Identity Manager Schema** in the Designer.
 2. Open the schema overview with **One Identity Manager Schema** from the task view.
- NOTE:** When you select a table or column in the Designer, open the schema overview by using the tasks **Show table 'XY' in schema** and **Show column 'XY' in schema**.

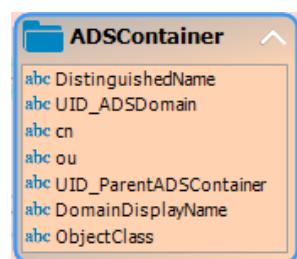
Figure 13: Schema overview



Standard Functions in the Schema View

Tables and their columns are displayed using a special control element. The name of the database table is shown in the header of the control element. All other entries represent columns in the table. Each control element entry has a tooltip. The contents of the tooltip depends on which display mode (data model or dependencies) has been selected. The column entries are labeled with icons that mark particular properties of the columns depending on the display mode.

Figure 14: Control Elements for Displaying Database Tables and their Columns

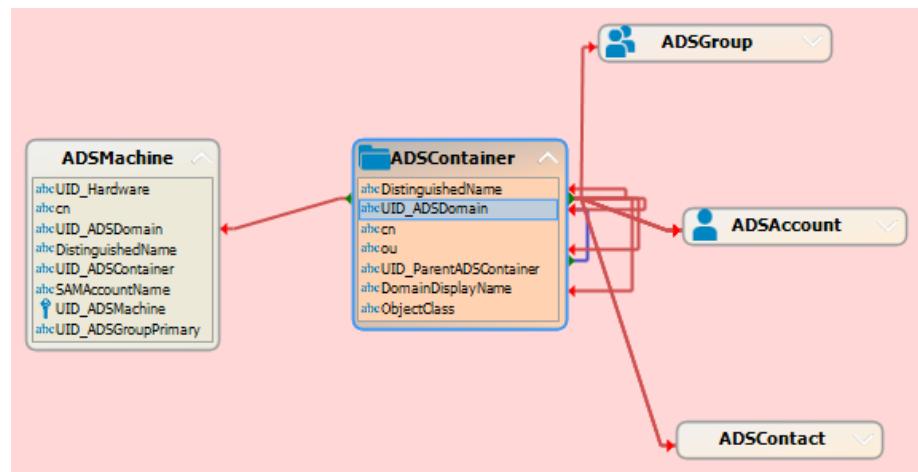


Use the menu items **Options | Show all columns** or **Options | Hide all columns** or the icons in the control element header to control how columns are displayed. Use **Options | Hide small tables** to only show the name of the table for tables with less than 20 columns.

You can change the layout of the control elements in the schema overview with the mouse. Using the menu item **Options | Save table layout** the changes made to the schema layout are saved in the internal database and in the Designer's change log.

Relations between tables or columns are represented by connecting lines. You can control how these are displayed using the menu item **Options | Hide table relations**. If the menu item is disabled all the connectors are shown. If the menu item is enabled then none of the connectors are shown. If a control element is selected the connectors are highlighted anyway without regard to the menu setting.

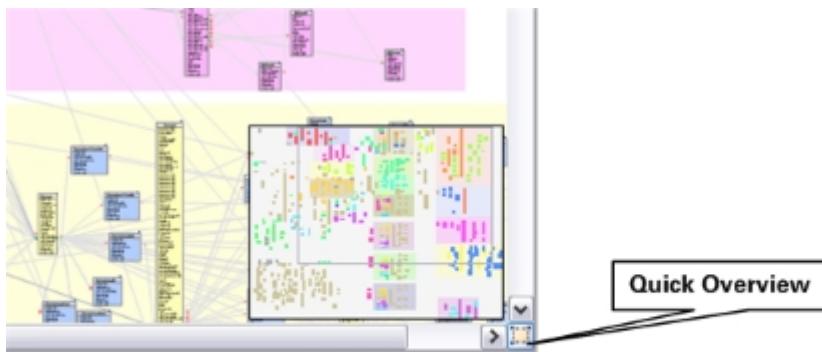
Figure 15: Using Connectors to Illustrate Relations



A connector points to column entries that are related to it. You can navigate between the connection points using the connector. When you select a connector the cursor changes to an arrow icon. Double click on the connector to move the view to other end of the connector. The direction is indicated by the arrow icon. Movement is controlled through the menu item **Options | Animate movements**. When the cursor passes over a connector a tool tip, whose contents depends on the display mode, is shown.

You can use the quick overview to navigate faster around the schema view. On the lower right edge of the schema overview there is a button which you use to open the quick overview. The area of the schema overview that is currently shown in the window is marked with a frame in the quick overview. Using the mouse you can move this frame around in the view. The corresponding area of the schema overview is then shown in the window.

Figure 16: Open Quick Overview



Special Functions for Displaying the Data Model

To display the data model

- Select **Options | Data model**.

In this mode, you obtain an overview of all the database tables with their columns and table relations. A single mouse click on the connector opens the table relation properties in the edit view.

A table entry's tooltip contains the name of the table and the table's preprocessor conditions. A column entry's tooltip contains the name of the column, description, data type and the minimum and maximum length of the column.

A connector's tooltip shows the table relations. This tooltip contains the name of the tables that are related to it and the table relation properties.

Column entries are marked in the control with icons representing special properties.

Table 87: Meaning of the Icons

Icon	Meaning
🔗	The column is a foreign key column (FK).
🔑	The column is a primary key column (PK).
abc	The column is of data type "string" or "text".
☒	The column is of data type "binary".
✓	The column is of data type "Bool".
123	The column is of data type "int", "short" or "byte".
1.3	The column is of data type "double" or "decimal".
📅	The column is of data type "date".

Special Functions for Displaying Dependencies

To display dependencies

- Select Options | Dependencies.

Only tables with columns that have dependencies due to value templates are displayed. Tables and columns without dependencies are not shown.

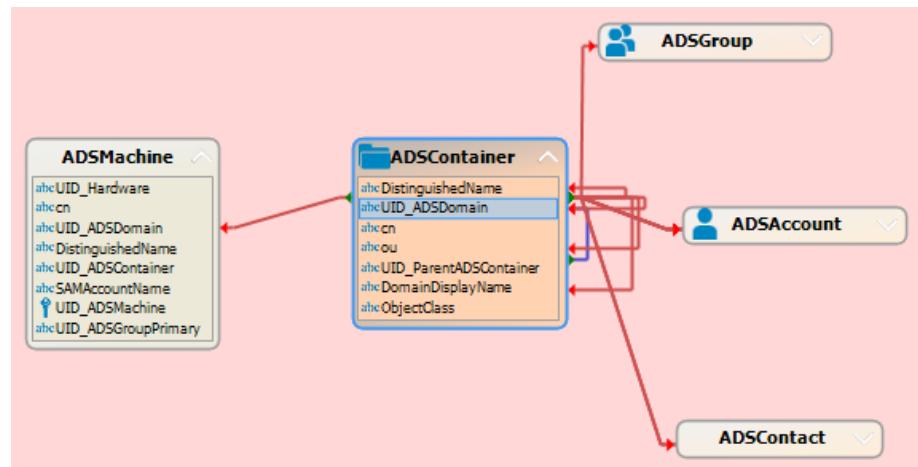
The tooltip for a table entry contains the name of the table. The tooltip for the column entries contains the name of the column. If a column has a value template it is shown in the tooltip. If the column does not have a value template itself but is referenced by value templates belong to other columns then those columns are named in the tooltip.

When you select a column, the connections to other columns are highlighted in color. A tooltip shows the sender and subscriber relationship of the column dependencies. The tooltip contains the names of tables that refer to each other. The sender, subscriber and the part of the value template that gives the reason for the dependency are also shown.

Table 88: Meaning of Colors for Sender Subscriber Relations

Color	Meaning
Blue	Column is sender.
Red	Column is subscriber.

Figure 17: Labeling Dependencies



Mapping Table Definitions

- NOTE:** The default configuration is moved to a configuration buffer during handling. You can retrieve changes from the configuration buffer and restore the default configuration in this way.

To edit table properties

1. Select the category **One Identity Manager Schema** in the Designer.
2. Select the table and start the Schema Editor with the task **Show table definition**.
3. Edit the table properties.

Detailed information about this topic

- [Basics of the Data Model on page 162](#)
- [General Advice for Editing Table and Column Definitions on page 165](#)
- [Table Properties on page 173](#)
- [Table scripts on page 180](#)
- [Database Views of Type "View" on page 181](#)
- [Database Views of Type "Union" on page 183](#)
- [Database Views of Type "Proxy" on page 185](#)
- [Database Views of Type "Read-only" on page 187](#)

Related Topics

- [Mapping Column Definitions on page 188](#)
- [Mapping Table Relations on page 208](#)
- [Custom Schema Extensions on page 526](#)
- [Supporting File Groups on page 212](#)

Table Properties

Table 89: Table Properties

Property	Meaning
Table	Name of the table in the data model.
Usage type	The table's usage type provides the basis for reports and the selection of tasks for daily maintenance. Permitted values are: <ul style="list-style-type: none">Work tablesHistorical transaction The table is a work table and contains reference data. The table contains reference data to create histories.

Property	Meaning						
	<p>data</p> <p>Configurations The table contains data for the system configuration.</p> <p>Materialized data The table contains materialized data. This is recreated through DBQueue Processor calculations.</p> <p>Read-only data The table contains read-only data.</p> <p>User data The table contains user data.</p>						
Display name	The display name is used, for example, to identify the table in a database search or for error output. Translate the given text using the  button.						
Display Pattern	The display template is used to specify the form in which objects will be represented in the administration tool result list. Translate the given text using the  button.						
Display pattern (custom)	Additional display pattern for individual tables containing the object's full name.						
Hierarchy path	<p>Enter the foreign key column that should be used as a basis for displaying tables hierarchically, for example, on assignment forms.</p> <p>Example:</p> <p>An Active Directory user account (table ADSAccount) is typically displayed on an assignment form below its Active Directory container (UID_ADSContainer). The Active Directory container (table ADSContainer) is, on the other hand, displayed underneath its Active Directory domain (column UID_ASDDomain). The path for the hierarchy structure is entered as follows:</p> <table> <thead> <tr> <th>Table</th> <th>Hierarchy path</th> </tr> </thead> <tbody> <tr> <td>ADSAccount</td> <td>UID_ADSContainer,UID_ASDDomain</td> </tr> <tr> <td>ADSContainer</td> <td>UID_ASDDomain</td> </tr> </tbody> </table> <p>An alternative list for objects that do not have values in all foreign keys, can be given by a pipe ().</p> <p>Example:</p> <p>(UID_ADSContainer,UID_ASDDomain UID_ASDDomain)</p>	Table	Hierarchy path	ADSAccount	UID_ADSContainer,UID_ASDDomain	ADSContainer	UID_ASDDomain
Table	Hierarchy path						
ADSAccount	UID_ADSContainer,UID_ASDDomain						
ADSContainer	UID_ASDDomain						
Remarks	Spare text box for additional explanation.						
Cache information	Caching behavior for tables in the Designer. This data is only required for system tables. Cache information for a table is composed of the sort order and loading behavior.						

Property	Meaning
	Permitted values are:
Do not load	The table is not loaded in the Designer.
Base table	The table is loaded before the user interface.
User table	The table is only filled for the current user.
Data table	The table is loaded in background after loading the user interface.
Load BLOBS	Columns with larger data sets (BLOB columns) are loaded.
Preprocessor condition	Tables can have preprocessor conditions added. The table is therefore only available when the preprocessor condition is fulfilled.
Disabled by preprocessor	If a table is disabled by a preprocessor condition, the option is set by the Database Compiler.
Deferred deletion [days]	Delete operations are deferred (0 = delete immediately, other: delete after given number of days).
Icon	Icon representing the table in the administration tool interface.
Background color	Color, with which the control for this table is displayed in the schema overview.
Proxy view	<p>The various target system tables are joined in database layers of type "Proxy" in the Unified Namespace. The table which serves as the proxy view is entered here. Columns of the underlying proxy view are then entered in the column definition.</p> <p>Example:</p> <p>The proxy view UNSRoot is used for mapping the tables ADSDomain and LDAPDomain.</p>
Extensions to proxy view	<p>List of columns as SQL text. These are used in the view's SELECT statement. To use if columns are mapped twice, for example, or if additional columns of the proxy view need to be filled.</p> <p>Example:</p> <p>The view UNSRoot expects the target system type as input in the column UID_DPRNameSpace. This column is not in the tables ADSDomain and LDAPDomain.</p> <p>The proxy view extension is defined as follows:</p>

Property	Meaning	
Table	Table	Extension to proxy view
	ADSDomain	'ADS-DPRNameSpace-ADS' as <code>UID_DPRNameSpace</code>
	LDPDomain	'LDP-DPRNameSpace-LDAP' as <code>UID_DPRNameSpace</code>
Scope hierarchy	Comma delimited list of all FK columns that are required for displaying objects in the scope hierarchy in the Synchronization Editor. List of all columns that lead to tables made available by the parent object.	
Logical disk store	The table's logical disk store. Associated tables are grouped together in logical disk stores. In the default installation, logical disk stores are predefined for the table in each module of the One Identity Manager and the system tables. You cannot change the assignments. You can create your own logical disk storage for grouping custom tables.	
Export for SPML schema	This option determines whether the table should be exported for the SPML schema.	
Many-to-many table	Label for assignment tables (many-to-many tables). Assignment tables are tables used to create relations between two other tables.	
Many-to-ajj table	Marks assignment tables, which have a dynamic foreign key as partner.	
No DB Transport	Tables labeled with this option cannot be excluded from a custom configuration package. These tables are excluded from data transport.	
Assign by event	<p>Specifies how assignments and deletions are handled in tables. This option only applies to assignment tables (many-to-many tables) in the application data model.</p> <p>If the option is not set, assignments and deletions are dealt with directly by the DBQueue Processor.</p> <p>If the option is set, tasks for the process component "HandleObjectComponent" are placed in the Job queue, which then deal with the appropriate operations. This makes it possible to link specific processes directly to the "Assign" and "Remove" events. This behavior has to be implemented on a custom basis.</p>	
Retain in memory	Specifies whether the table contents for the data connection can be buffered. The threshold is defined in the configuration parameter "Common\ResidentTableLimit".	
Type	Table type.	

Property	Meaning
Table 90: Permitted Values	
Table Description types	
Table	The table type is used for simple tables, many-to-many tables, many-to-all tables and work tables.
Base table	This table type is used for simple tables, many-to-many tables and many-to-all tables, to derive database views of type "View".
View	This table type is used for database views of tables with type "base table". Database views with the type "View" represent partial sections of the underlying tables.
Proxy	This table type is used for database view of tables with type "table" or for database view of type "View". Database views with the table type "proxy" are unions of different table views. The column definition is used to map the columns between a "Proxy" type view and the underlying table.
Union	This table type is used for database view of tables with type "table" or for database view of type "View" or "proxy". Database views with the table type "Union" are views of the union of different tables and supply a grouping of different object types with the same context.
Read Only	This table type is used for database view of tables with type "table" or for database view of type "View" or "proxy" or "Union". Views with the table type "read only" can be parts but also unions of the underlying tables.
Module GUID permitted	For more information, see Working with Module Globally Unique Modifiers on page 179.
Module GUID required	For more information, see Working with Module Globally Unique Modifiers on page 179.
Base table	Base table that the view is based on.
Condition for view definition	WHERE clause as database query for setting the database view.
Insert values	Specify default settings for a column that is assigned when a new data set is added. The values are entered in VB.Net syntax.
Selection script	Selection script as a VB.Net term, to determine during runtime whether the object passed belongs to the view.
Additional	Name of an extension to the view definition.

Property	Meaning
view definition	
Generated	Specifies whether the view definition extension is generated by the DBQueue Processor.
Query	<p>Database query as a SELECT statement for setting up the database view.</p> <p>Several extensions for the view definition can be defined. The extensions are linked with each other with the Union operator.</p>
Table scripts	Define actions that are executed before or after saving, loading or discarding an object. The values are entered in VB.Net syntax.
Statistic information	This is information about table sizes, row counts and basic record lengths that are determined once a day by the maintenance tasks. The data material can help to plan capacities and maintenance work on the database.
Customizer	<p>The Customizer contains special methods and has side effects on the table columns. Several Customizers can be defined for one table. Customizers execute processing, logic which is normally implemented in the object code, such as mutual exclusion of properties.</p> <p>There are various Customizers contained in the One Identity Manager default installation, which provide specific behavior. Do not modify Customizers as error-free behavior of One Identity Manager would no longer be guaranteed.</p>
Multicolumn uniqueness	Specifies columns that must be collectively unique. The columns are collected into a unique groups.
Unique group	Name of the group of columns with a unique total value.
Ignore empty values	<p>Specifies whether empty values are permitted. If all the columns in one group are empty, group uniqueness cannot be tested. If this option is not set, empty values are permitted but only once for each column.</p> <p>To prevent empty values, define a minimum column length in the column definition.</p>
Parent tables	Comma delimited list of all foreign key columns required for displaying objects in the scope hierarchy in the Synchronization Editor. List of all columns that lead to tables made available by the parent object.
Condition for transport	Condition for selecting transportable objects. An empty condition means that all object are transferred.

Related Topics

- [Basics of the Data Model on page 162](#)
- [General Advice for Editing Table and Column Definitions on page 165](#)

- [Table scripts on page 180](#)
- [Database Views of Type "View" on page 181](#)
- [Database Views of Type "Union" on page 183](#)
- [Database Views of Type "Proxy" on page 185](#)
- [Database Views of Type "Read-only" on page 187](#)
- [Supporting File Groups on page 212](#)
- [Display Template for Displaying a List on page 268](#)
- [Conditional Compilation using Preprocessor Conditions on page 448](#)
- [Preparing the One Identity Manager Schema for Exporting to the SPML Schema on page 607](#)

Working with Module Globally Unique Modifiers

To transport, for example, predefined reports, processes, workflows or mail definitions with a complete system configuration transport, the objects require a primary key with a module GUID. These objects are identified as part of the system configuration through the module GUID.

Syntax:

The table primary key has the format CCC-[0-9,a-f](32).

 **NOTE:** Entries with a module GUID are transferred automatically to the transport package when a transport of the entire system configuration is created.

You can use the following table definition settings for generating a module GUID:

- If the options **Module GUID permitted** and **Module GUID required** are enabled, the objects have to get a module GUID. The objects in this type of labeled tables are given the module prefix CCC.
- If only the option **Module GUID permitted** is enabled, the objects can get a module GUID in the required format. By default, the objects obtain a default GUID in the format [0-9,a-f](8-4-4-4-12). Create the objects with the prefix CCC if they should obtain a module GUID. You can do this using the Object Browser.

Example

- The DialogGroup table has the options **Module GUID required** and **Module GUID permitted** enabled. When creating a new permissions group, the primary key is automatically generated in the format of a module GUID.
- For the AERole table only the option **Module GUID permitted** is set. To ensure that your own application roles are added to the transport package, create the application roles in the Object Browser with a module GUID.

NOTE: In the default case, the table's primary key is created with a default GUID. To subsequently change a default GUID to a module GUID, you use the Object Browser.

IMPORTANT: Do not execute the following steps for production databases. Only perform these steps within the maintenance window. Otherwise, this could lead to inconsistent data.

To change a default GUID to a module GUID

1. In Object Browser select the object for which you want to change the default GUID.
2. Display the **Properties** context menu.
3. On the **Methods** tab select the `SwitchToModuleGuid()` method and click **Execute**.

To change a module GUID to a default GUID

1. In Object Browser select the object for which you want to change the module GUID.
2. Display the **Properties** context menu.
3. On the **Methods** tab select the `SwitchToNormalGuid()` method and click **Execute**.

Related Topics

- [Table Properties on page 173](#)
- [Configuring a One Identity Manager Database for Testing, Development or Production on page 116](#)
- [Transporting System Configuration on page 554](#)

Table scripts

Table scripts help you to define actions that are executed before or after saving, loading or discarding an object. In this way, substantial changes or value checks that cannot be easily done with formatting rules or templates, can be made to an object by running a table script before it is saved. After the object is saved, changes to other objects can be made or task and processes can be generated with table scripts, for example. The side effect and tasks defined in the Customizer are applied following the table scripts.

You can customize predefined default table scripts and create your own additional table scripts. The default configuration is moved to a configuration buffer during handling. You can retrieve changes from the configuration buffer and restore the default configuration in this way.

Table scripts are stored in VB.Net syntax which allows use of all VB.Net script functions.

IMPORTANT: Compile the database to bring the table scripts into effect.

Table 91: Table scripts

Script	Meaning
Script (OnDiscarded)	The script is run after the object is discarded.
Script (OnDiscarding)	The script is run before the object is discarded.
Script (OnLoaded)	The script is run after the object is loaded.
Script (OnSaved)	The script is run after the object is saved.
Script (OnSaving)	The script is run before the object is saved.

Related Topics

- [Table Properties on page 173](#)
- [Using Scripts on page 453](#)
- [Templates on page 195](#)
- [Formatting Scripts on page 201](#)
- [Compiling a One Identity Manager Database on page 64](#)

Database Views of Type "View"

Database views with the type "View" represent partial sections of the underlying tables. Database views with type "View" are predefined. Templates and formatting rules can be defined for columns in these views.

Database views of type "View" are generally used for displaying roles. For example, the views Department, Locality and Profitcenter are parts of the base table Basetree.

The following information is displayed for a database view of type "View".

Table 92: Properties for Defining a Database View of Type "View"

Property	Meaning
Table	Name of the table in the data model.
Type	Table type "View".
Base table	Base table that the view is based on.
Condition for view definition	Limiting condition as WHERE clause for setting up the database view. The condition relates to the underlying base table.
Columns	A reference is required for each column of the database view to a column in the underlying base column. Make the assignment in the column definition.

Example

The table Department is defined as a database view of type "View".

Table 93: Example of Defining a Database View of Type "View"

Property	Value
Table	Department
Type	View
Base table	BaseTree
Condition for view definition	UID_OrgRoot = 'QER-V-Department'
Column-->Base column (excerpt).	Department.DepartmentName-->BaseTree.Ident_Org Department.Description-->BaseTree.Description
Resulting view definition	create view dbo.Department as select Ident_Org as DepartmentName, Description as Description, ... from BaseTree where UID_OrgRoot = 'QER-V-Department'

You can user insert values to preset certain values in the table when adding a new data set.

Table 94: Defining Insert Values

Property	Value
Insert values	Default settings for individual columns that are assigned when a new data set is added. The values are entered in VB.Net syntax.
Selection script	Selection script as a VB.Net term, to determine during runtime whether the object passed belongs to the view.

 **NOTE:** You must recompile the database if you changed insert values and selection scripts!

Example

The UID_OrgRoot is given the value QER-V-Department when data is added to the table Department. The insert values are defined as follows.

Table 95: Example for defining the insert values.

Insert values	base.putvalue("UID_OrgRoot", "QER-V-Department")
Selection script	Value = (String.Equals(\$UID_OrgRoot\$, "QER-V-Department", StringComparison.OrdinalIgnoreCase))

IMPORTANT: If the base table that the view is based on has a table type "B" then the selection script has to correspond to the view condition. If one of the properties is given then the other one also has to be given.

Related Topics

- [Table Properties on page 173](#)
- [Column Properties on page 189](#)
- [Defining Insert Values on page 269](#)
- [Database Views of Type "Union" on page 183](#)
- [Database Views of Type "Proxy" on page 185](#)
- [Database Views of Type "Read-only" on page 187](#)

Database Views of Type "Union"

Database views with the table type "Union" are views of the union of different tables and supply a grouping of different object types with the same context. Thus the union view QERAccProductUsage determined, which service items are used in which products, for example.

Views of type "Union" are predefined. Templates and formatting rules cannot be defined for columns in these views. The object key column (xObjectKey) must be referenced in the view definition. This makes it possible to create single object with its valid permissions.

"Union" views are mainly used for editing the user interface and for creating reports.

The following information is displayed for a database view of type "Union".

Table 96: Properties for Defining a Database View of Type "Union"

Property	Meaning
Table	Name of the table in the data model.
Type	Table type "Union".
Additional view defin-	Database query as a SELECT statement for setting up the database view.

Property	Meaning
Union	<p>Several extensions for the view definition can be defined. The extensions are linked with each other with the Union operator.</p> <p>When you add a column, an entry is created in the table DialogColumn. When you delete a column, the entry is removed from the table DialogColumn.</p>
Condition for view definition	Limiting condition as WHERE clause for setting up the database view.
Columns	Database view columns.

Example

The table QERAccProductUsage is defined as a database view of type "Union". In the union view, you establish which service item is used in which products. The following example shows an excerpt from the definition based on system entitlements (table ESet) and report subscriptions (table RPSReport).

Table 97: Example of Defining a Database View of Type "Union"

Property	Value
Table	QERAccProductUsage
Type	Union
Columns	TableName, UID_AccProduct, XObjectKey
Extension 1: Additional view definition	ESet
Extension 1: Query	<pre>select 'ESet' as TableName, g.XObjectKey, g.UID_AccProduct from ESet g</pre>
Extension 2: Additional view definition	RPSReport
Extension 2: Query	<pre>select 'RPSReport' as TableName, g.XObjectKey, g.UID_AccProduct from RPSReport g</pre>
Resulting view definition	<pre>create view dbo.QERAccProductUsage as select * from (select convert(varchar(11), null) as TableName, convert (varchar(38), null) as UID_AccProduct, convert(varchar(138),</pre>

Property	Value
	<pre> null) as XObjectKey where 1=0 union all select xxTab.TableName, xxTab.UID_AccProduct, xxTab.XObjectKey from (select 'ESet' as TableName, g.XObjectKey, g.UID_ AccProduct from ESet g) as xxTab union all select xxTab.TableName, xxTab.UID_AccProduct, xxTab.XObjectKey from (select 'RPSReport' as TableName, g.XObjectKey, g.UID_ AccProduct from RPSReport g) as xxTab) as x </pre>

Related Topics

- [Table Properties on page 173](#)
- [Column Properties on page 189](#)
- [Creating a Union View on page 536](#)
- [Database Views of Type "View" on page 181](#)
- [Database Views of Type "Proxy" on page 185](#)
- [Database Views of Type "Read-only" on page 187](#)

Database Views of Type "Proxy"

Database views with the table type "proxy" are unions of different table views. The column definition is used to map the columns between a "Proxy" type view and the underlying table. The DBQueue Processor calculates the actual view definition from the column mapping. This only takes into account tables that are not disabled by a preprocessor condition. Templates and formatting rules cannot be defined for columns in these views.

Views of type "Proxy" are primarily used for displaying the Unified Namespace. For example, the proxy view UNSRoot is used in the Unified Namespace to map the table ADSDomain or LDAPDomain.

The following information is displayed for a database view of type "Proxy".

Table 98: Properties for Defining a Database View of Type "Proxy"

Property	Meaning
Table	Name of the table in the data model.
Type	Table type "Proxy".
Additional view definition	<p>Database query generated as a SELECT statement for setting up the database view. View definition extensions are generated by the DBQueue Processor.</p> <p>The following are taken into account when generating:</p> <ul style="list-style-type: none"> • Tables in which the database view is entered as the proxy view. • Columns that have a reference to a proxy view column. • Columns defined as extensions to the proxy view. <p>The extensions are linked with each other with the Union operator.</p>
Condition for view definition	Limiting condition as WHERE clause for setting up the database view.
Columns	Database view columns.

Example

The following mappings are required to map ADSDomain in Unified Namespace to the database view USRoot.

- The database view UNSRoot is entered as a proxy view in the table ADSDomain.
- The columns of the table ADSDomain to be mapped in the Unified Namespace are given a reference to the corresponding columns in the proxy view.

For example, the column Ident_Domain in the table ADSDomain is mapped to the column Ident_root of the proxy view UNSRoot.

- Columns expected in the database view UNSRoot but do not exist in the table ADSDomain are entered as proxy view extensions in the table ADSDomain.

For example, the view UNSRoot expects a target system type in the column UID_DPRNameSpace. This column is not in the tables ADSDomain. Thus 'ADS-DPRNameSpace-ADS' as UID_DPRNameSpace is entered as an extension to the proxy view in the table ADSDomain.

The DBQueue Processor generates the extended view definition from the data. The following statement is a excerpt from the generated extension.

```
select ... Ident_Domain as Ident_UNSRoot..., 'ADS-DPRNameSpace-ADS' as UID_DPRNameSpace from ADSDomain
```

Related Topics

- [Table Properties on page 173](#)
- [Column Properties on page 189](#)
- [Database Views of Type "View" on page 181](#)
- [Database Views of Type "Union" on page 183](#)
- [Database Views of Type "Read-only" on page 187](#)

Database Views of Type "Read-only"

Views with the table type "read only" can be parts but also unions of the underlying tables. Views with type "read only" are predefined. Templates and formatting rules cannot be defined for columns in these views.

Database views of type "read only" are mainly used to edit the user interface and for creating reports.

The following information is displayed for a database view of type "Read-only".

Table 99: Properties for Defining a Database View of Type "Read-only"

Property	Meaning
Table	Name of the table in the data model.
Type	Table type "Read-only".
Additional view definition	Database query as a SELECT statement for setting up the database view. Several extensions for the view definition can be defined. The extensions are linked with each other with the Union operator. When you add a column, an entry is created in the table DialogColumn. When you delete a column, the entry is removed from the table DialogColumn.
Condition for view definition	Limiting condition as WHERE clause for setting up the database view. The condition is attached to the view definition generated from the extension.
Columns	Database view columns.

Related Topics

- [Table Properties on page 173](#)
- [Column Properties on page 189](#)
- [Creating a Read-Only Database View on page 534](#)
- [Database Views of Type "View" on page 181](#)

- Database Views of Type “Union” on page 183
- Database Views of Type “Proxy” on page 185

Mapping Column Definitions

NOTE: The default configuration is moved to a configuration buffer during handling. You can retrieve changes from the configuration buffer and restore the default configuration in this way.

To edit column properties

1. Select the category **One Identity Manager Schema** in the Designer.
2. Select the table and start the Schema Editor with the task **Show table definition**.

NOTE: Select a table column in the Designer, then you can start the Schema Editor from the task **Show column definition**.
3. Select the column in the Schema Editor and edit the column properties.

Detailed information about this topic

- Basics of the Data Model on page 162
- General Advice for Editing Table and Column Definitions on page 165
- Column Properties on page 189
- Templates on page 195
- Formats for Checking Values on page 200
- Permitted Column Values on page 202
- Column Dependencies for Setting Values on page 205
- Configuring Columns for Full-Text Search on page 206
- Labeling Columns for Translation on page 330

Related Topics

- Mapping Table Definitions on page 172
- Custom Schema Extensions on page 526

Column Properties

Table 100: Column Properties

Property	Meaning						
Table	Name of the table to which the column belongs.						
Column	Name of the table in the data model.						
Display name	Language dependent column name for displaying in the administration tool's user interface. Translate the given text using the  button.						
Comment	Additional information about the column. The remarks are shown in the help function for each administration tool. Translate the given text using the  button.						
Disabled by preprocessor	If a column is disabled by a preprocessor condition, the option is set by the Database Compiler.						
Preprocessor condition	You can add preprocessor conditions to columns. The column is therefore only available when the preprocessor condition is fulfilled. <p> NOTE: You can find an overview of existing preprocessor dependencies in the Designer in the category Database One Identity Manager Schema Preprocessor dependencies.</p>						
Sort order	The sort order specifies the position for displaying the column on the generic form and the custom tabs of the default form. Columns with a value less than one are not displayed.						
Group	Group is used to display the column on general master data forms. A new tab is created for each group on the generic form.						
Base column	If the table is of type "view" then the link to the base table is given here. <p>Example: The database view Department is a sunset of the base table Basetree. The columns of the table Basetree are entered as base columns.</p> <table><thead><tr><th>Column</th><th>Base column</th></tr></thead><tbody><tr><td>Department.DepartmentName</td><td>BaseTree.Ident_Org</td></tr><tr><td>Department.Description</td><td>BaseTree.Description</td></tr></tbody></table>	Column	Base column	Department.DepartmentName	BaseTree.Ident_Org	Department.Description	BaseTree.Description
Column	Base column						
Department.DepartmentName	BaseTree.Ident_Org						
Department.Description	BaseTree.Description						
Adjustment of permitted values list is not allowed	Specifies whether permitted values can be customized for this column.						

Property	Meaning																									
Defined list of values	Marks whether the value in this column must correspond to the values in the List of permitted values , or are empty.																									
List of permitted values	If a column is principally enabled for permitted values editing, that means the option Customizing permitted list not allowed is not set. If the option Defined list of value is set, then you can add to or extend a value list.																									
Defined bitmask	Meaning of each bit position if the column contains a bitmask. The first bit in the definition start with the index 0.																									
Multilingual	Specifies whether this column can be given in multiple languages. Permitted values are: Translation target The column content is displayed in translation. Translation source The column supplies the translation. #LD content The column has contents in #LD notation. The combination of values determines the resulting translation.																									
Syntax	Syntax of Data in this column. The syntax type is used to give the One Identity Manager tools the appropriate syntax highlighting or input assistance. Permitted syntax types are: <table> <tbody> <tr> <td>HTML</td> <td>Input in HTML format</td> </tr> <tr> <td>Picture</td> <td>Images</td> </tr> <tr> <td>SQL.Query</td> <td>Full database queries</td> </tr> <tr> <td>SQL.Special</td> <td>Special syntax for database queries</td> </tr> <tr> <td>SQL.WhereClause</td> <td>WHERE clause for database queries</td> </tr> <tr> <td>Text.Dollar</td> <td>Input in \$ notation</td> </tr> <tr> <td>UNC</td> <td>UNC paths</td> </tr> <tr> <td>URL</td> <td>URL input</td> </tr> <tr> <td>VB.Class</td> <td>Full class definitions</td> </tr> <tr> <td>VB.Instruction</td> <td>"Value = " instructions</td> </tr> <tr> <td>VB.Method</td> <td>Single methods or functions</td> </tr> <tr> <td>XML</td> <td>Input in XML format</td> </tr> </tbody> </table>		HTML	Input in HTML format	Picture	Images	SQL.Query	Full database queries	SQL.Special	Special syntax for database queries	SQL.WhereClause	WHERE clause for database queries	Text.Dollar	Input in \$ notation	UNC	UNC paths	URL	URL input	VB.Class	Full class definitions	VB.Instruction	"Value = " instructions	VB.Method	Single methods or functions	XML	Input in XML format
HTML	Input in HTML format																									
Picture	Images																									
SQL.Query	Full database queries																									
SQL.Special	Special syntax for database queries																									
SQL.WhereClause	WHERE clause for database queries																									
Text.Dollar	Input in \$ notation																									
UNC	UNC paths																									
URL	URL input																									
VB.Class	Full class definitions																									
VB.Instruction	"Value = " instructions																									
VB.Method	Single methods or functions																									
XML	Input in XML format																									

Property	Meaning	
Number of decimal places	This contains the number of decimal places used for displaying real and integer values. A value can be given to three decimal places, for example. Prices are given to two decimal places by default.	
Date add-on	Additional information about displaying dates and times in the user interface.	
Index weighting	<p>Column weighting in indexing. This is used for the full text search index in the Web Portal. Increasing weighting results in a higher position in the search results.</p> <p>If the value is less than or equal to "0", no indexing takes place. If the value is greater than "0", the data value is indexed. Columns to be index are given the weighting "1" in the default installation.</p>	
Table Lookup Support	<p>Each value in these columns are prepared for fast table lookup support. The search is also supported by single values in MVP columns. The internal mapping of prepared data is done in the table QBMSplittedLookup.</p> <p>Permitted values are:</p> <ul style="list-style-type: none"> • Central user account (CentralAccount) • E-Mail Address (EMail) <p>You can extend the list of permitted values and customize the results.</p> <p>The functionality can be used for finding a unique central user account, for example, or a unique default email address for an employee. Columns in the default installation, which are taken into account when mapping the central user account or an email address, are labeled with this property. The results are displayed in the views, QERCentralAccount and QERMailAddress.</p>	
Data type in database	Shows the .Net data type for the column. This is used internally and cannot be edited. The Net data types are mapped internally to SQL data types. If no value is given, the data type is taken from the database schema. Permitted syntax types are:	
.Net Data Types	Data type (SQL Server)	Data type (Oracle)
Binary	image, binary, varbinary (max)	raw, blob
Bool	bit	number (1, 0)

Property	Meaning		
	.Net Data Types	Data type (SQL Server)	Data type (Oracle)
Byte	Tinyint		
Date	datetime		date
Decimal	decimal, numeric		number
Double	float, real		number (38, 16)
Int	Int		number (14, 0)
Long	bignint		number (38, 0)
Short	smallint		
String	nvarchar/varchar/nchar		varchar2
Text	text		clob, nclob
Size in database	Length of the column in the database		
Primary key	The primary key is given when the database is created.		
UID column	Specifies whether this is UID column. This option is only permitted for columns with the .Net data type "String" and length 38.		
Default value	Specifies whether a default value is defined for this column in the database schema.		
BLOB value	This option is used to label text columns whose data contents is so large that they cannot be kept internally in one line in the SQL sever and are therefore saved as a reference. This allows speedier access to the data.		
Log changes	Specifies whether changes to this column are logged.		
Log changes when deleting	Specifies whether the column is logged when an object is deleted.		
Export for SPML schema	This option determines whether the table should be exported for the SPML schema.		
Not for export (XML export)	This column is not exported in data transports. The property is taken into account when data is transported between database.		
Not for import (XML import)	This column is not imported in data transports. The property is taken into account when data is transported between database.		
MVP column	This is a MultiValuedProperty (MVP) whose individual values are separated by char(7) or chr(7).		
Multiline	Specifies whether the column contents can consist of more than		

Property	Meaning
	one line. Columns that are labeled with this option are displayed on a common form with multiline input fields.
Dynamic foreign key	Dynamic foreign keys reference object keys in another tables. The object key is made up of the table name and the value of the primary key of the actual object. Permitted tables can be limited. All tables are permitted, if there are no restrictions.
Column contains description	One column with a description can be labeled with this option per table. The description is only displayed on user interface assignment controls.
Column contains hierarchy information	One column, which maps hierarchy information in readable form can be labeled with this option per table. The column is used for mapping the hierarchy to the mapping control elements in the user interface.
Part of primary key	This column is part of the primary key
Part of alternative primary key	Alternative primary keys are already specified in the default version, but the definition can be customized. Alternative primary keys are used for data transport amongst other things.
Part of the key of a many-to-many table (dynamic)	Labels the foreign key of an many-to-many table (dynamic). A (dynamic) many-to-many table's foreign key and dynamic key are identified with this option.
Display in Filter Designer	The column is displayed in the Filter Designer or the Rule Editor for creating requests.
Recursive key	This option specifies whether this column has a link to a parent object. This input is needed for displaying hierarchical tables. For example, the table ADSContainer contains a column UID_ParentADSContainer with a link to the parent Active Directory container. The column UID_ParentADSContainer is labeled with this option in order to display this hierarchical link on forms.
Encrypted	This option is used to specify whether the value in this column should be encrypted or not. When the database is encrypted the value in this column is encrypted.
	<p>NOTE: If you set this option on database columns, you must encrypt the database again. For more information, see the One Identity Manager Installation Guide.</p>
Permissions not issued automatically	Permissions for predefined permissions groups are not issued automatically for custom schema extension on a predefined table, even if the configuration parameter "Common\AutoExtendPermissions" is set.

Property	Meaning
Proxy view column	If the column is used in a "Proxy" type view, the corresponding column in the view is entered here. For example, the column ADSDomain.DisplayName is mapped in the UNSRoot view to column RootObjectDisplay.
Remarks (custom)	Spare text box for additional explanation.
Max. length	Maximum length of the column. If the value is "0" the length is taken from the database schema.
Foreign key	The column references an object in another table.
Min. length	Minimum length of the column. The minimum length must be at least 1 or more for mandatory columns in the administration tools.
Column format	Specify the format permitted for value in this column. You can control the permitted format for the column with formatting types and formatting scripts.
Overwrites	Specifies whether the template can overwrite or not.
Template	Defines a value template for this column using other columns or a default value for the column. Write the script in VB.Net syntax which allows all VB.Net script functions to be used.
Threshold (abort)	Limit for the number of objects changed by this template. Once this limit has been reached, processing is aborted with an error message.
	<p> NOTE: If a 'abort' threshold value is given, it must be larger than the threshold for asynchronous processing.</p>
Threshold (asynchronous)	Limit for the number of objects changed by this template. Once this limit has been reached, processing takes place synchronously with the One Identity Manager Service.
No automatic truncation by template	If the maximum length exceeded when applying a template, the value is not automatically truncated to the maximum column length if the option is set.
Formatting script	Formatting script for the column. Write the script in VB.Net syntax which allows all VB.Net script functions to be used.
Custom template/formatting not permitted	Specifies whether the default configuration can be changed by the user, for example, display name, templates and formatting rules.
Average column length	Information is determined once a day through the maintenance tasks. The data material can help to plan capacities and maintenance work on the database.

Property	Meaning
Template changed	(Only for internal use) indicates that the template was changed.
No DB Transport	Columns labeled with this option cannot be excluded from a custom configuration package. These columns are excluded from data transport.
No log	Specifies whether the column content is recorded in logs, for example, in the One Identity Manager Service log.

Related Topics

- [Database Views of Type "View" on page 181](#)
- [Editing Value Templates on page 197](#)
- [Formats for Checking Values on page 200](#)
- [Permitted Column Values on page 202](#)
- [Mapping Dynamic Foreign Keys on page 204](#)
- [Configuring Columns for Full-Text Search on page 206](#)
- [Forms for Custom Extensions on page 287](#)
- [Language Dependent Data Representation on page 328](#)
- [Labeling Columns for Translation on page 330](#)
- [Logging Data Changes on page 427](#)
- [Conditional Compilation using Preprocessor Conditions on page 448](#)
- [Using Scripts on page 453](#)
- [Preparing the One Identity Manager Schema for Exporting to the SPML Schema on page 607](#)

Templates

Value templates are implemented in the One Identity Manager for generating user data or for transforming values. You can use these templates to fill object properties with default values or to form property values from other properties. Value templates can take effect within an object as well as between objects. Value templates without dependencies take effect when the value is queried in the column and the column does not have a value assigned. Value templates that refer to other columns are affected when these columns change.

Value templates take effect without regard to the current rights situation. No explicit rights need to be assigned to the dependent columns. When value templates are in use, all the columns that are effected are therefore also loaded if they are not on the current form in the "Manager" program.

 **NOTE:** You can get an overview of existing columns with value template in the category **One Identity Manager Schema | Templates** in Designer.

Column dependencies due to value templates are mapped in the table DialogNotification. The connected properties are shown in the table as sender-subscriber pairs. The column that caused the change is the sender and the column that is changed because of it, is the subscriber. The object links are consolidated by the column relations. The entries are created when the value templates are compiled and updated.

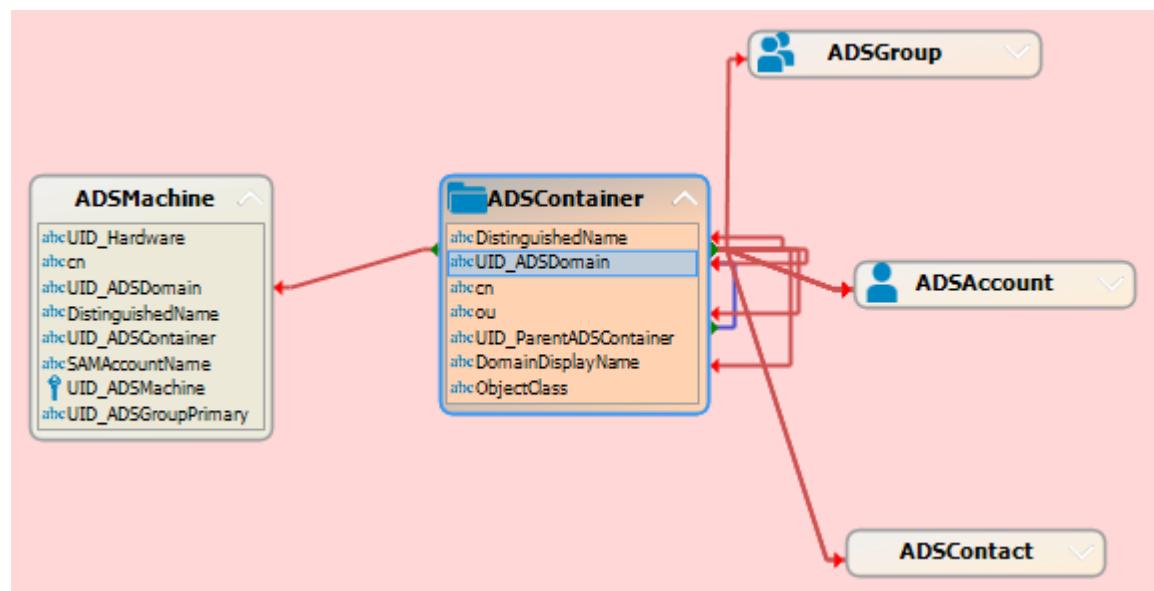
You can see dependencies in the Schema Editor overview. To display dependencies you need to select the menu item **Options | Dependencies**. If a column contains a value template it is displayed in the tooltip. If the column does not have a value template itself but is referenced by value templates belong to other columns then those columns are named in the tooltip.

When you select a column, the connections to other columns are highlighted in color. A tooltip shows the sender and subscriber relationship of the column dependencies. The tooltip contains the names of tables that it refers to. The sender, subscriber and the part of the value template that gives the reason for the dependency are also shown.

Table 101: Meaning of Colors for Sender Subscriber Relations

Color	Meaning
Blue	Column is sender.
Red	Column is subscriber.

Figure 18: Sender Subscriber Relations Schema Editor



Detailed information about this topic

- [Editing Value Templates on page 197](#)
- [Example of Local Value Templates within an Object on page 199](#)
- [Example of Value Templates across Objects on page 199](#)

Related Topics

- [Working with the Schema View on page 168](#)
- [Formats for Checking Values on page 200](#)
- [Permitted Column Values on page 202](#)
- [Column Dependencies for Setting Values on page 205](#)

Editing Value Templates

You can customize predefined default value templates and create your own additional value templates. The default configuration is moved to a configuration buffer during handling. You can retrieve changes from the configuration buffer and restore the default configuration in this way.

NOTE: Remember to take performance into consideration when defining value templates. In certain circumstances, changes to a property could cause large numbers of dependent objects to be changed, saved and processes to be generated through a value template in overwrite mode.

To create a value template

1. Select the category **One Identity Manager Schema** in the Designer.
2. Select the table and start the Schema Editor with the task **Show table definition**.
3. Select the column and edit the following properties on the **Value calculation** tab:

Overwrites	Specifies whether the template can overwrite or not. If this option is set, the value template is always applied. If the option is not set, the value template is only applied when the column is empty.
------------	--

Template	Template script. Write the script in VB.Net syntax which allows all VB.Net script functions to be used.
----------	---

TIP: To display the column that trigger a template, click the link [Triggers for this template](#).

No automatic truncation by template	Specifies whether the value is automatically truncated to the maximum column length if the maximum length is exceeded when applying a template. If this option is set, the value is not truncated to the maximum column length.
-------------------------------------	---

IMPORTANT: Compile the database to bring the value template into effect.

Prevent change to column

You can use value templates to prevent users from changing columns that are filled by a value template. To do this, add the name of this column in the value template in \$-notation. The value template now references itself. Any change to the column is immediately overwritten by the value template. Self-correcting value templates are only effective if the templates are labeled as "overwriteable".

Example:

The user should not be able to change an employee's central user account. This should be prevented by the value template.

- Define a custom value template for the column Person.CentralAccount.
- Set the option **Overwrites**.
- Extend the default value template with the following entry: '\$CentralAccount\$.

```
'$CentralAccount$  
If Not CBool(Session.Variables.Get("FULLSYNC")) Then  
    Value=VI_AE_BuildCentralAccount(GetValue("UID_Person").String,$Lastname$,  
    $Firstname$)  
End If
```

Limit template execution

To limit the number of objects changed by a value template you can define thresholds.

Table 102: Thresholds

Property	Meaning
Threshold (asynchronous)	Maximum number of objects that can be changed by the value template. Once this limit has been reached, processing takes place synchronously with the One Identity Manager Service.
Threshold (abort)	Once this limit has been reached, processing is aborted with an error message. NOTE: If a 'abort' threshold value is given, it must be larger than the threshold for asynchronous processing.

Related Topics

- [Example of Local Value Templates within an Object on page 199](#)
- [Example of Value Templates across Objects on page 199](#)
- [Preprocessor Conditions in VB.Net Expressions on page 451](#)
- [Using Scripts on page 453](#)
- [Compiling a One Identity Manager Database on page 64](#)

Example of Local Value Templates within an Object

The an employee's full name (Person.Internalname) will be derived from its surname (Person.Lastname) and first name (Person.Firstname). The value template for the column Person.Internalname looks like:

```
Value = $Lastname$ & ", " & $Firstname$
```

If the value template is labeled as "Overwrites" then each time Lastname changes a test is done to check for dependent columns that reference this value in a template. If this is the case, the value template is processed and the value is entered into the column Internalname. If the value template cannot overwrite, it only applies if there is no value in the column Internalname.

The columns Person.Lastname and Person.Firstname are the sender and the column Person.Internalname is the subscriber. The mapping for adding a database object in the table DialogNotification is:

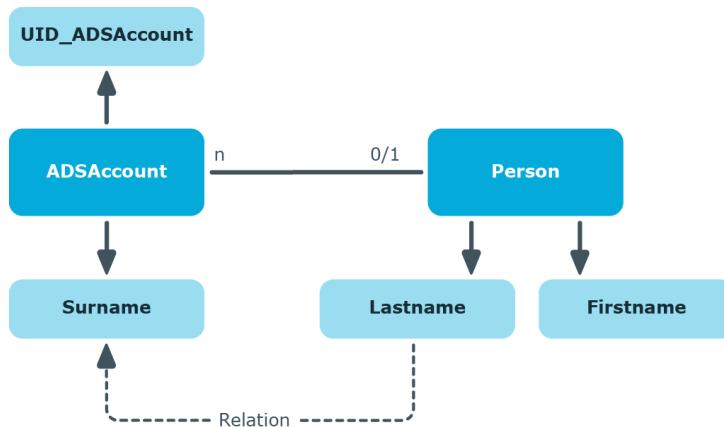
```
person.lastname --> person.internalname
```

```
person.firstname --> person.internalname
```

Example of Value Templates across Objects

If a value template references a value from another object, it can be accessed using the FK relation.

Figure 19: Effect of Cross Object Value Templates



If, for example, the surname of a Active Directory user account (ADSAccount.Surname) is derived from the surname of an employee (Person.Lastname), enter the template for the column ADSAccount.Surname as follows:

```
Value = $FK(UID_Person),Person.Lastname$
```

If the employee's surname changes, the last name of the Active Directory user changes, too. The column Person.Lastname is therefore the sender and the column ADSAccount.Surname is the receiver. The relation is mapped in the table DialogNotification as follows:

Person.Lastname --> ADSAccount.Surname

Formats for Checking Values

You can customize predefined default column formats and create your own additional formats. The default configuration is moved to a configuration buffer during handling. You can retrieve changes from the configuration buffer and restore the default configuration in this way.

You can also use predefined formatting scripts along side formatting types to check values in a column.

NOTE: You can get an overview of existing columns in the system with formatting type or formatting scripts in the category **One Identity Manager Schema | Templates** in Designer.

Detailed information about this topic

- [Formatting Types on page 200](#)
- [Formatting Scripts on page 201](#)

Related Topics

- [Templates on page 195](#)
- [Permitted Column Values on page 202](#)
- [Column Dependencies for Setting Values on page 205](#)

Formatting Types

You can specify column formats based on predefined formatting types. By combining formatting types with each other, you can obtain the formatting you required.

NOTE: Enter the column data, which must be unique when together, into the table definition using multi-column uniqueness. The columns are collected into a unique groups.

To specify formatting types

1. Select the category **One Identity Manager Schema** in the Designer.
2. Select the table and start the Schema Editor with the task **Show table definition**.

- Select the column and specify the formatting types in **Column format** on the **Value calculation** tab.

Table 103: Permitted Formatting Types

Value	Format Type	Permitted values
0	None	No special formatting = default
1	IP address	IP address [0-9] ³ .[0-9] ³ . [0-9] ³ .[0-9] ³
2	MAC-ID	MACID [0-9,A-F]12
4	Drive letter	Drive letter [A-Z]1:
8	Digit	[0-9]+
16	Capital letters	Capital letters
32	Server dependent capital letters	(Only included for compatibility)
64	NT name	All characters permitted except "!@/\:,"[]; - =+*?<>"
128	Telephone number	Telephone number [0123456789#/--*]n
256	Exchange name	All characters permitted except "ÄÖÜäöüß"!§\$%&\ /<>#*{}[] 23~^,"
512	ASCII letters and numbers	All characters of the character set 7-bit
2048	URI	Uniform resource identifier
4096	Email address	Valid email address

IMPORTANT: Compile the database to bring the formatting type into effect.

Related Topics

- [Formatting Scripts on page 201](#)
- [Compiling a One Identity Manager Database on page 64](#)
- [Table Properties on page 173](#)

Formatting Scripts

You can use a formatting script to verify column values. Formatting scripts, as opposed to value templates, are only executed when a value is assigned to the column.

To create a formatting script

1. Select the category **One Identity Manager Schema** in the Designer.
2. Select the table and start the Schema Editor with the task **Show table definition**.
3. Select the column and enter the formatting script for the column in **Formatting script** on the **Value calculation** tab.

Write the script in VB.Net syntax which allows all VB.Net script functions to be used.

IMPORTANT: Compile the database to bring the formatting script into effect.

Example formatting script

The value in the column Mail in the ADSAccount table should correspond to SMPT format. If this is not the case, an error message is sent. The formatting script for the column ADSAccount.Mail can be formulated as follows:

```
Dim str as String = Convert.ToString(Value)
If str.Length > 0 Then
    If Not VID_IsSMTPAddress(str) Then
        Throw New Exception(""" & str & """ is not a valid SMTP address.")
    End If
End If
```

Related Topics

- [Formatting Types on page 200](#)
- [Using Scripts on page 453](#)
- [Compiling a One Identity Manager Database on page 64](#)

Permitted Column Values

You can define a list of permitted values in order to limit a column to specific values. Once the column display name has been created, the list of permitted values is no longer valid. Permitted values are supplied for certain data model columns through schema installation.

NOTE: If a column is principally enabled for permitted values editing, that means the option **Customizing permitted list not allow** is not set, then you can add to or extend a value list.

To specify a list of permitted values

1. Select the category **One Identity Manager Schema** in the Designer.
2. Select the table and start the Schema Editor with the task **Show table definition**.
3. Select the column and set the option **Defined list of values** on the **Column** tab.

4. Enter the **List of permitted values**.

- a. Open the dialog box with the [...] button next to the list.

Table 104: Used Icons

Icon	Meaning
	Insert value.
	Delete value.
	Change value.
	Change sort order.

- b. Enter a value and display text.
- c. Specify the display order.
- d. (Optional) Specify a translation.
- e. Save the changes.

IMPORTANT: Compile the database to bring the list of permitted values into effect.

Example

Only the values "internal" and "external" should be valid in the input field "Spare field no. 1". The list of permitted values is defined as followed:

1=internal 2=external

The string "intern" is therefore displayed on forms in the program "Manager" for an employee with the value "1".

Displaying Columns with Permitted Values in the Manager

There is a special control element used in the program "Manager" for displaying columns is defined a list of permitted values. The control element is displayed as a simple input field if no list is defined. If a list is defined the control element is shown as a menu.

Figure 20: Input Field for List of Defined Values (with and without defined entries)



The control element is only available for columns on default predefined forms as well as custom columns (usually CustomProperty01-CustomProperty10).

Related Topics

- [Templates on page 195](#)
- [Formats for Checking Values on page 200](#)
- [Compiling a One Identity Manager Database on page 64](#)

Mapping Dynamic Foreign Keys

You can use dynamic foreign keys if a reference is able to point to different tables. For example, a user account's manager (table ADSAccount.ObjectKeyManager) can be another user account (table ADSAccount) or a group (ADSGroup).

Dynamic foreign keys reference object key (XObjectKey of the permitted tables. Permitted tables can be limited. All tables are permitted, if there are no restrictions. Restrictions are stored in the table, DialogValidDynamicRef.

To edit dynamic foreign keys

1. Select the category **One Identity Manager Schema** in the Designer.
2. Select the table and start the Schema Editor with the task **Show table definition**.
3. Select the column and set the option **Dynamic foreign key** on the **More** tab.
4. If the dynamic key is part of a many-to-all table, set the option **Part of the key of a many-to-all table**.
5. Enter the following information on the **Valid reference tables** by clicking next to **Dynamic referenced tables** and enter the following information:

Table 105: Properties for Dynamic Foreign Keys

Property	Description
Table	Tables to use for determining the object key.
Only transport as group	The contents of the column are always transferred together with the contents of the referenced column.
Parent relation constraint	Relation constraint, for example, IR - Insert Restrict, DC - Delete Cascade.
Parent relation test instance	This referential integrity should be checked by D - DLL, T - Trigger or N - Nothing. Triggers and constraints are implemented to monitor the database. The triggers and constraints are created automatically and modified as necessary taking the preset restrictions of the DBQueue Processor into account.

Property	Description
Child relation constraint	Relation constraint, for example, IR - Insert Restrict, DC - Delete Cascade.
Parent relation test instance	This referential integrity should be checked by D - DLL, T - Trigger or N - Nothing. Triggers and constraints are implemented to monitor the database. The triggers and constraints are created automatically and modified as necessary taking the preset restrictions of the DBQueue Processor into account.

Table 106: Permitted Restrictions for Testing Referential Integrity

Restriction	Meaning
DeleteNotRestricted (D)	Dependencies are not taken into account on deletion.
DeleteRestrict (DR)	The object can only be deleted when no more references to other objects exist.
DeleteCascade (DC)	All dependent objects are deleted when this object is deleted.
DeleteSetNULL (DS)	All links to other objects are deleted when the object is deleted (SetNULL).
InsertNotRestricted (I)	Dependencies are not taken into account on insertion.
InsertRestrict (IR)	Checks for the referenced object when the object is added.

Related Topics

- [Basics of the Data Model on page 162](#)

Column Dependencies for Setting Values

There may be dependencies between individual values, for example, by using value templates or customizers that require values to be set in a specific order. In the case of administration tools the correct order is enforced through blocking or releasing input fields. In the case of data import and when using SPML and web service interfaces, the correct order for setting values also has to be safeguarded.

The following data sources assume the following sequence for specifying the order for setting values:

1. Customizer

The dependencies between columns and an object are stored in customizers.

2. Custom dependencies

In order to create a customer specific definition of dependence between columns, you select a table column in the Schema Editor and specify a predecessor for the column under **Dependencies**.

3. Column dependencies due to value templates

In this case, values used by a template (for example Person.Firstname, Person.Lastname) are set before values that are created by a template (for example Person.CentralAccount).

If circular dependencies occur whilst determining the order for setting the values, they are aborted at the point of lowest priority.

Related Topics

- [Table Properties on page 173](#)
- [Templates on page 195](#)

Configuring Columns for Full-Text Search

Full-text searching uses an external search index, which returns an object key as result. The object key is used to run a search query in the database. This database search query takes the permissions of the logged in user into account during the search. A maximum of 1000 objects can be returned through the search index.

The One Identity Manager full text search can be used in the Web Portal and in the Manager.

- Prerequisites for using full text search is an application server installed with the search service.
- If you run the Web Portal directly over an application server installed with the search service, you can use the full text search immediately.
- If you are working with the Web Portal and an application server without a search service installed or with a direct database connection, you will need to enter an application server with a search service in the Web Portal configuration file. Full text search is available in the Web Portal once this has been done.
- To use full text search in the Manager, you must run the Manager over an application server with an installed search service.

For more detailed information about installing on an application server and configuring the Web Portal for full text search, see the One Identity Manager Installation Guide.

The following applies to configuring columns for full text search:

- Columns for full text searching must be weighted. The higher the weighting, the higher the position in the search results. Columns for full text search are predefined with "1" in the default installation.

Example:

The column Person.CentralAccount has a weighting value of "1". The column "ADSAccount.SAMAccountName" is weighted with "0.5". This results in the employee being listed before the user account in the full text search.

- Only columns with the data types "string" or "text" can be added to the full text search.
- Table columns with the usage type "work tables" or "historic transaction data" cannot be added in the full text search.

The search service indexes the:

- Column content
- Foreign key column display value
- Display values for lists of permitted values
- Translation for every active language
- Object display value, if the table's primary key column is configured for full text search

The object's display value comes from the display pattern defined for the table. The display value's weighting comes from the table's primary key column weighting

Example:

The column Person.UID_Person is configured for full text search. The display pattern of the table Person is defined as %InternalName% (%CentralAccount%). This indexes the display value "Clara Harris (CLARAH)" for employee "Clara Harris".

The searched index is updated when changes are made to a table with indexed columns, to referenced tables or translations.

Certain important columns are already indexed for full text search in the default installation. You configure more columns for full text searching if you require.

To configure a column for full text search

1. Select the category **One Identity Manager Schema** in the Designer.
2. Select the table and start the Schema Editor with the task **Show table definition**.
3. Select the column and edit the property **Index weighting**.
 - If the value is less than or equal to "0", no indexing takes place.
 - If the value is greater than "0", the data value is indexed.

For more detailed information about using the full text search, see the One Identity Manager Web Portal User Guide and the One Identity Manager User Guide for One Identity Manager Tools User Interface and Default Functions.

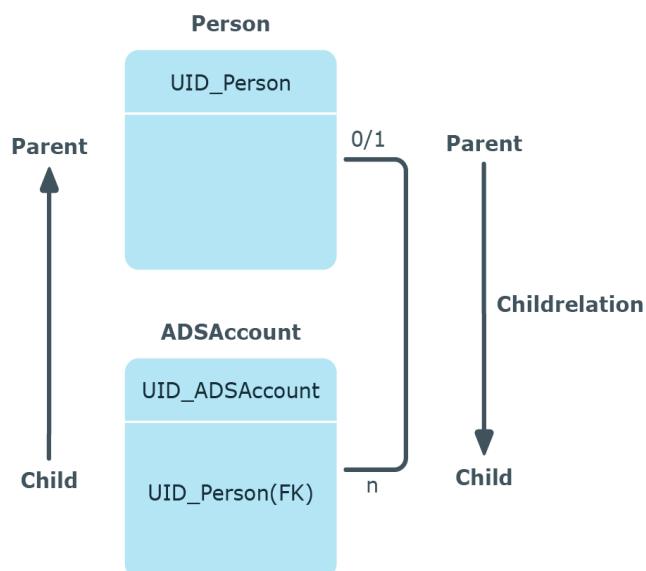
Related Topics

- [Column Properties on page 189](#)

Mapping Table Relations

As you can see from the One Identity Manager data model, parent/child relations exist between objects. When an object is processed by a One Identity Manager DLL, all ForeignKey (FK) objects that are related to this object can be accessed. Use VB.Net notation to access objects access using relations.

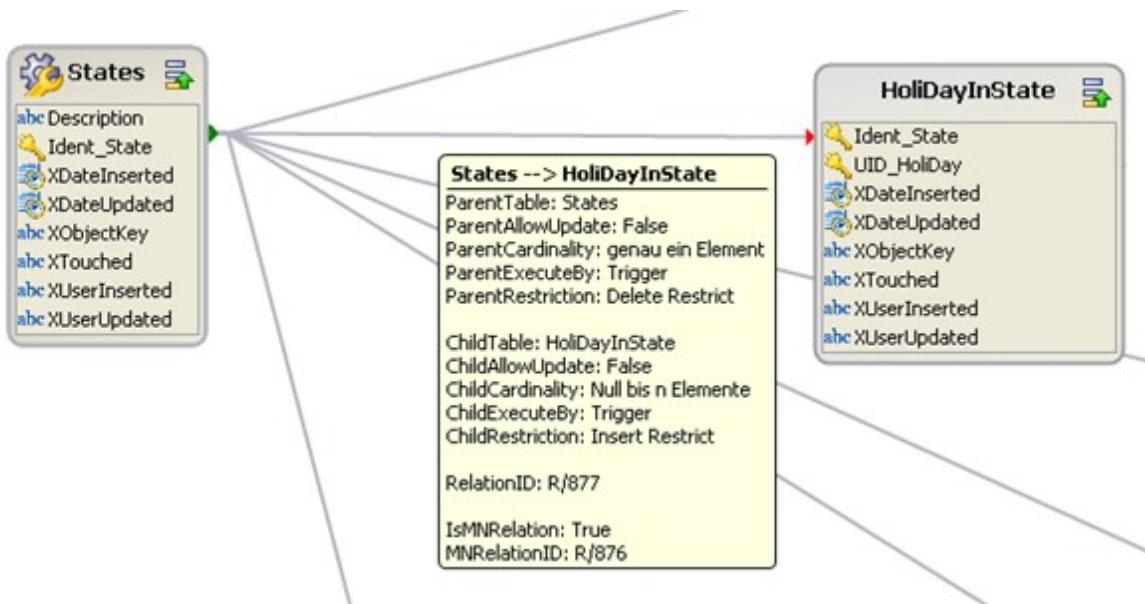
Figure 21: Parent/Child Relation using the Example of an Employee ADSAccount



The tables and column are stored in the table QBMRelation. Predefined relations of the One Identity Manager data model are maintained through schema installation and cannot be edited apart from a few exceptions.

A connector's tooltip shows the table relations in the schema overview (menu item **Options | Data model**). This tooltip contains the name of the tables that are related to it and the table relation properties. A single mouse click on the connector opens the table relation properties in the edit view.

Figure 22: Graphical Representation of Table Relations in the



- ❶ **NOTE:** Custom table relations are always editable. Table relation supplied with the default tables can only be edited if the referential integrity has been tested using the DLL.
- ❷ **IMPORTANT:** Use the program, "Schema Extension" to extend the One Identity Manager data model. Schema extensions are added to the database using "Schema Extension" and the necessary extensions are made in the One Identity Manager data model.

To edit table relations

1. Select the category **One Identity Manager Schema** in the Designer.
2. Select the table and start the Schema Editor with the task **Show table definition**.
3. Select the table relation and edit the properties.

Table 107: Table Relation Properties

Property	Meaning
Display name	Language dependent relation for displaying in the administration tool's user interface.
Only transport as group	In the case of data transport, the contents of tables are always transferred together with the contents of the tables that are referenced. For example, the tables JobChain, Job and JobRunParameter.
Update dependencies modification	When many-to-many entries are added, changed or deleted the value in the XDateSubItem column in one of the parent entries is updated. Required for provisioning memberships in the target system.

Property	Meaning
date	
Export for SPML schema	This option determines whether the table relation should be exported for the SPML schema.
Parent column	Unique parent column identifier.
Configurable parent relation	Specifies whether referential integrity can be configured.
Parent relation test instance	Specifies how referential integrity is tested. Through DLL, Trigger or Nothing. Triggers and constraints are implemented to monitor the database. The triggers and constraints are created automatically and modified as necessary taking the preset restrictions of the DBQueue Processor into account. In the case of customized tables, specify the test instance and the limitations of the One Identity Manager schema extension.
Parent relation constraint	Constraint on the relation, for example, IR - Insert Restrict.
Generated restriction test for parent relation	Abbreviation for triggers and constraints generated automatically by the DBQueue Processor.
Connected column	Unique connected column identifier.
Configurable child relation	Specifies whether referential integrity can be configured.
Child relation test instance	Specifies how referential integrity is tested. Through DLL, Trigger or Nothing. Triggers and constraints are implemented to monitor the database. The triggers and constraints are created automatically and modified as necessary taking the preset restrictions of the DBQueue Processor into account. In the case of customized tables, specify the test instance and the limitations of the One Identity Manager schema extension.
Child relation constraint	Relation restriction, for example, IR - Insert Restrict.
Generated restriction	Abbreviation for triggers and constraints generated automatically by the DBQueue Processor.

Property	Meaning
test for child relation	
Relation ID	Relation identifier. This is used for both directions.
M:N relation	Can relation be reached using an many-to-many relation?
table relation	Unique identifier for table relation.
Relation (base)	Link to underlying base relation assuming a view is part of the relation.
Relation (M:N)	Unique identifier for the M:N relation.

Table 108: Permitted Restrictions for Testing Referential Integrity

Restriction	Meaning
DeleteNotRestricted (D)	Dependencies are not taken into account on deletion.
DeleteRestrict (DR)	The object can only be deleted when no more references to other objects exist.
DeleteCascade (DC)	All dependent objects are deleted when this object is deleted.
DeleteSetNULL (DS)	All links to other objects are deleted when the object is deleted (SetNULL).
InsertNotRestricted (I)	Dependencies are not taken into account on insertion.
InsertRestrict (IR)	Checks for the referenced object when the object is added.

Related Topics

- [Mapping Table Definitions on page 172](#)
- [Mapping Column Definitions on page 188](#)
- [Custom Schema Extensions on page 526](#)
- [Preparing the One Identity Manager Schema for Exporting to the SPML Schema on page 607](#)
- [Language Dependent Data Representation on page 328](#)

Supporting File Groups

One Identity Manager supports file groups to group tables together to help with administration, data assigning and data distribution. One Identity Manager differentiates between logical storage and physical storage.

In the default installation, logical disk stores are predefined for the table in each module of the One Identity Manager and the system tables. You cannot change the assignments. You can create your own logical disk storage for grouping custom tables.

To define logical storage for custom tables

1. Select the category **One Identity Manager Schema | Logical disk stores** in the Designer.
2. Select **Object | New** in the menu.
3. Enter a name and description for the logical storage.
4. Assign custom tables to the logical disk store.
5. Select the menu item **View | Select table relations...** and enable the table **DialogTable**. This shows the tab **Tables** for assigning tables.

You can link logical storage with physical storage - the file groups - in the One Identity Manager schema. If file groups are created on different data medium, you can use parallel accessing to enhance the performance of tables with high change rates. An example of this is tables for processing DBQueue Processor tasks or table for process handling.

 **NOTE:** You cannot move the following groups into other file groups, otherwise proper functioning of the One Identity Manager database cannot be guaranteed.

- DialogColumn
- DialogTable
- DialogValidDynamicRef
- QBMDBQueueTask
- QBMDBQueueTaskDepend
- QBMMModuleDef
- QBMMModuleDepend
- QBMRelation
- QBMViewAddOn
- QBMDiskStoreLogical
- QBMDiskStorePhysical

One Identity Manager supports distribution of table to file groups using a set of database procedures, which you can run in the database using a suitable query tool.



WARNING: Only carry out the following steps for implementing file groups, together with an experienced database administrator.

Ensure that the database cannot be accessed while file groups are being set up, for example, by the Job server, application server, web server, user interfaces, Web Portal. After reactivating the DBQueue Processor, wait until all DBQueue tasks have been processed before you allow new database connections.

To distribute tables to file groups under SQL Server

1. Create your file groups. For detailed information about this, see the documents for your currently installed version of SQL Server.
2. Synchronize the file groups in the One Identity Manager database. Run the following query in the database using a suitable query tool.
`exec QBM_PDiskStorePhysicalSync`
3. Assign physical storage to logical storage in the Designer.
 - a. Select the category **One Identity Manager Schema | Logical disk stores** in the Designer.
 - b. Select the logical disk store and select the file groups under **Physical disk store**.
 - c. Save the changes to the database using **Database | Commit to database....**
4. Disable processing of DBQueue Processor tasks and process handling. Run the following query in the database using a suitable query tool.
`exec QBM_PWatchDogPrepare 1`
`exec QBM_PDBQueuePrepare 1`
5. Move the tables into the configured file groups. Run the following query in the database using a suitable query tool.
`exec QBM_PTableMove`
6. Reactivate the DBQueue Processor. Run the following query in the database using a suitable query tool.
`exec QBM_PDBQueuePrepare 0,1`
`exec QBM_PWatchDogPrepare`

To distribute tables to tablespaces under Oracle Database

1. Create your tablespaces and make them available for use by One Identity Manager database users. For detailed information about this, see the documents for your currently installed version of Oracle Database.
2. Synchronize the tablespaces in the One Identity Manager database. Run the following query in the database using a suitable query tool.
`begin QBM_GCommon2.PDiskStorePhysicalSync(); end;`

3. Assign physical storage to logical storage in the Designer.
 - a. Select the category **One Identity Manager Schema | Logical disk stores** in the Designer.
 - b. Select the logical disk store and select the tablespace under **Physical disk store**.
 - c. Save the changes to the database using **Database | Commit to database....**

4. Disable processing of DBQueue Processor tasks and process handling. Run the following query in the database using a suitable query tool.

```
begin QBM_GWatchDog.PPrepare(1); end;  
begin QBM_GDBQueue.PDBQueuePrepare(1); end;
```

5. Move the tables (including LOBs ad indexes) to the configured tablespaces. Run the following query in the database using a suitable query tool.

```
begin QBM_GCommon2.PTableMove(); end;
```

6. Reactivate the DBQueue Processor. Run the following query in the database using a suitable query tool.

```
begin QBM_GDBQueue.PDBQueuePrepare(0); end;  
begin QBM_GWatchDog.PPrepare(0); end;
```

Granting One Identity Manager Schema Permissions

Permissions for accessing tables and columns of the One Identity Manager schema are themselves mapped in the schema through permissions groups. Permissions groups can be assigned to system users and application roles.

The user's effective permissions depend on the authentication module used for logging into One Identity Manager tools.

- The permissions assigned to the system user are found from the permissions groups for logging into One Identity Manager tools with an authentication module that expects a defined system user.
- Dynamic system users are used for logging into One Identity Manager tools with role-based authentication modules. First, the employee memberships in the One Identity Manager application roles are determined during login. Assignments of permissions group to One Identity Manager application roles are used to determine which permissions groups apply to the employee. A dynamic system user is determined from these permissions groups that will be used for the employee's login.

The system user's effective permissions that are found are not saved in the One Identity Manager schema, but are determined when logging into One Identity Manager tools and then they are loaded.

Permissions groups are also used to control access to parts of the user interface, such as, menu items, forms, tasks and program functions. When a user logs into the One Identity Manager tools, menus, forms and methods are loaded depending on the system user's permissions groups, displaying a user interface customized for this system user.

The One Identity Manager provides permissions groups and system users with a predefined user interface and edit permissions to the One Identity Manager schema's tables and columns. These predefined configurations are maintained by the schema installation and cannot be edited apart from a few properties. Use predefined permissions groups and system users as templates for your own permissions groups and system users.

Detailed information about this topic

- [Predefined Permissions Groups and System Users on page 216](#)
- [Editing Permissions Groups and System Users on page 219](#)
- [Editing Permissions for One Identity Manager Schema Tables and Columns on page 231](#)
- [Rules for Finding Valid Permissions for Tables and Columns on page 233](#)
- [Availability of Certain Functionality on page 241](#)
- [Displaying Permissions for an Object on page 243](#)
- [Displaying the Current User's Permissions on page 244](#)

Related Topics

- [Working with the User Interface on page 246](#)
- [One Identity Manager Authentication Module on page 74](#)
- For more detailed information about implementing and editing application roles, see the One Identity Manager Application Roles Administration Guide.

Predefined Permissions Groups and System Users

One Identity Manager supplies system users and permissions groups with predefined user interfaces (menu items, forms, tasks, program functions) and special access permissions for One Identity Manager schema tables and columns. These predefined configurations are maintained by the schema installation and cannot be edited apart from a few properties. Use predefined permissions groups and system users as templates for your own permissions groups and system users.

NOTE: It is recommended that you set up your own system users and permissions groups whose user interface and access permissions are specially designed to meet the requirements of the administrative tasks.

Table 109: Predefined Permissions Groups

Permissions Group	Description
Permissions group "QBM_BaseRights"	The permissions group "QBM_BaseRights" defines the basic permissions that are sufficient for logging a system user in to the administration tools. This permissions group is always assigned implicitly.
Permissions group "VI_View"	The permissions group "VI_View" owns viewing permissions to all table and columns of the One Identity Manager application data model.

Permissions Group	Description
View"	<p>NOTE: Assign viewing permissions of custom schema extensions to the permissions group. Assign viewing permissions of the module's own tables and columns to the permissions group.</p>
Permissions group "VI_Everyone"	<p>The permissions group "VI_everyone" is assigned user interface form elements, which uses links to the corresponding menu items. These permissions groups also provide functions for Web Portal users.</p> <p>NOTE: Assign the permissions group to your custom system users such that the overview form is fully displayed to the users.</p>
Permissions groups for the One Identity Manager application data model	<p>Permissions groups have edit permissions for One Identity Manager application data model tables and columns. These permissions groups are equipped with menu items, forms, tasks and program functions which allows the application data to be edited with the Manager.</p>
Permissions groups for the One Identity Manager system data model	<p>These permissions groups have permissions for the One Identity Manager system data model tables and columns. These permissions groups are equipped with menu items, forms, tasks and program functionality which allows the application data to be edited, for example, with Designer editors.</p> <p>The permissions group "vid" has all the edit permissions for configuring the system with the Designer.</p>
Role-based permissions group "VI_4_ALLUSER"	<p>The role-based permissions group "VI_4_ALLUSER" provides basic permissions such as menu items, forms, methods and program functions in order to edit application data with the Manager and the Web Portal. This permissions group is always assigned implicitly.</p>
Role-based permissions group "vi_4_ADMIN_LOOKUP"	<p>The permissions group "VI_4_ADMIN_LOOKUP" has viewing permissions for all tables and columns of the One Identity Manager application data model.</p> <p>NOTE: Assign viewing permissions of custom schema extensions to the permissions group. Assign viewing permissions of the module's own tables and columns to the permissions group.</p>
Role-based permissions groups	<p>Role-based permissions groups have edit permissions for One Identity Manager application data model tables and columns. These permissions groups are equipped with menu items, forms, tasks and program functionality which allow the application data to be edited with the Manager and Web Portal. These permissions groups are linked to the One Identity Manager application roles and simplify administration of access permissions in the One Identity Manager role model.</p>

Table 110: Predefined system users

System user	Description
Dynamic system user	Dynamic system users are used for logging into One Identity Manager tools with role-based authentication modules. First, the employee memberships in the One Identity Manager application roles are determined during login. Assignments of permissions group to One Identity Manager application roles are used to determine which permissions groups apply to the employee. A dynamic system user is determined from these permissions groups that will be used for the employee's login.
System User "sa"	The system user "sa" is exclusively user by One Identity Manager Service. This system user is not allocated a permissions groups but has all access permissions, tasks and program functionality.
System User "viadmin"	<p>The system user "viadmin" is the default system user for the One Identity Manager. This system user can be used to compile and initialize the One Identity Manager database and for the first user login to the administration tools.</p> <p>IMPORTANT: The system user "viadmin" is not for use in a live environment! Set up your own system users with the appropriate permissions.</p> <p>The system user "viadmin" has all the permissions predefined by and the entire user interface. The system user "viadmin" also implicitly has all the permissions and user interface components from custom permissions groups.</p> <p>The system user "viadmin" has permissions to set up an employee as One Identity Manager administrator for role-based login. The system user is not a member of the application role themselves.</p>
System user "Synchronization"	The system user "Synchronization" has predefined permissions for setting up and running target system synchronization through an
System user "viHelpdesk"	The system user "viHelpdesk" has predefined permissions and the user interface required to access the One Manager help desk resources with the One Identity Manager.
System Account User "viITShop"	The system user "viITShop" has predefined permissions and the user interface required to access the Manager with the IT Shop.

Related Topics

- [Editing Permissions Groups and System Users on page 219](#)

Editing Permissions Groups and System Users

In the One Identity Manager default installation certain permissions groups and system users already exist with predefined access permissions. Predefined configurations are maintained by the schema installation and cannot be edited apart from a few properties. Set up your own permissions groups and system users such that the access permissions relate to the different administrative tasks required. You can enable permissions to be passed on from one permissions group to other permissions groups by structuring permissions groups hierarchically.

Permissions groups are managed in the Designer in the category **Permissions | Permissions groups**. Here you will find an overview of edit permissions and user interface components that are assigned to individual permissions groups. In addition, the system users are displayed, which the permissions groups are assigned.

System users are displayed in the category **Permissions | System users** in the Designer. You will see an overview of the permissions groups that are assigned to each individual system user.

Related Topics

- [Predefined Permissions Groups and System Users on page 216](#)
- [Permissions Group Dependencies on page 223](#)
- [Copying Permissions Groups on page 224](#)
- [Manually Creating Permissions Groups on page 226](#)
- [Editing System Users on page 227](#)
- [Adding a System User to Permissions Groups on page 229](#)
- [Dynamic system user on page 230](#)
- [Which Employees Use System Users? on page 230](#)

Working with the User & Permissions Group Editor

Use the User & Permissions Group Editor to create and edit system users and permissions groups. The editor is started from the program "Designer" and opens in the document view. Only additional User & Permissions Group Editor functions are described in the following.

Menu Items

The following items are added to the menu bar when the editor starts.

Table 111: Menu Items Added by the Editor

Menu	Menu Item	Meaning
Permissions group	New	Adds a new permissions group.
	Delete	Deletes the selected permissions group after requesting confirmation.
	Assign users...	Opens a dialog window in which system users can be selected for assignment to a permissions group
	Copy permissions groups...	Starts a wizard for copying permissions groups.
	Refresh	Updates the hierarchical view of the permissions groups.
Users	New	Adds a new system user.
	Delete	Deletes the selected system user after requesting confirmation.
	Assign permissions groups...	Opens a dialog window in which permissions groups can be selected for assignment to a system user.
	Create administrator	Creates a new system user and adds it to the default permissions group.
Options	Display permissions group inheritance	Shows/hides a system user's permissions group membership inheritance in the hierarchical view.
View	Properties	Shows/hides the edit view.
	One Identity Manager employees	Shows employees that use this system user.
Help	Users and permissions group management help	Opens the help on this topic.
	User & Permissions Group Editor help	Opens the editor help.

Table 112: Meaning of Toolbar Icons

Icon	Meaning
	Inserts a system user.
	Deletes a system user.
	Inserts a permissions group.
	Deletes a permissions group.

Icon	Meaning
	Create administrator.
	Zooms in on the view.
	Zooms out on the view.
	Hides/shows system user's inherited permissions group memberships in the hierarchical view.
	Updates the permissions group display in the hierarchical view.

Views in the User & Permissions Group Editor

The User & Permissions Group Editor has several views for displaying and editing system users and permissions groups.

Table 113: User & Permissions Group Editor Views

View	Description
System user and permissions groups edit view	The properties of the selected system user or permissions group are displayed in the edit view and may be changed there. A default context menu is available for input fields.
System user and permissions groups edit view	This view displays the permissions groups in their hierarchical form. The permissions group memberships are displayed for one system user. Each permissions group is represented by a permissions group element. Each permissions group element has a tooltip. The contents of the tooltip is made up of the name and description of the permissions group.

TIP: The layout position of the permissions group elements can be changed by using the mouse.

Figure 23: User & Permissions Group Editor with Hierarchical View (Above) and Edit View (Below)

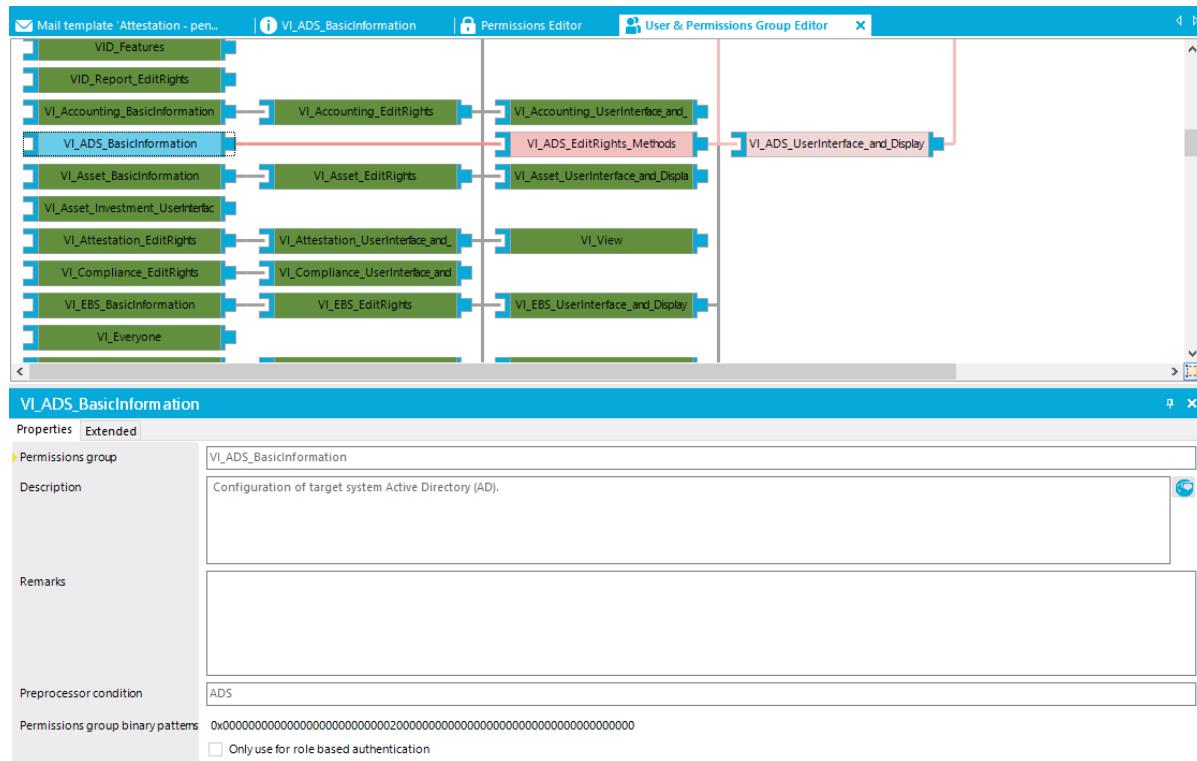


Table 114: Context Menu Items for the Hierarchical View

Context Menu Item	Meaning
Assign users...	Assigns users to the selected permissions group.
Assign permissions group...	Assigns permissions groups to the selected system user.
Inherit permissions from...	The selected permissions group is added to other permissions groups and inherits permissions from those permissions groups.
Permissions inherited by...	More permissions group are added to the selected permissions group. Permissions subgroups inherit permissions from the selected permissions group.
Navigation	Shows all other editors that can be used with the selected object.

Permissions Group Dependencies

You can enable permissions and user interface components to be passed on from one permissions group to other permissions groups by structuring permissions groups hierarchically. This means that inheritance is top down within the hierarchy.

Example

Two permissions groups are defined with the following permissions and user interface components.

Permissions group	Permissions	User interface
A	Viewing permissions	Menu structures and forms
B	Edit permissions	Methods

Permissions group A is above permissions group B in the hierarchy, so that B inherits the permissions group from A. Therefore the viewing and edit permissions and the menu structure, form and methods are available to users of the permissions group B.

You can use the hierarchical view in the User & Permissions Group Editor to set up dependencies between permissions groups. Permissions groups, which are higher up in the hierarchy are displayed further to the right in the User & Permissions Group Editor. When a permissions group is selected in the hierarchical view, the dependencies to other permissions groups are marked in color which also is also used to show the direction of inheritance.

Figure 24: Diagram of Permissions Group Hierarchy (Direction of Inheritance from Right to Left)



Table 115: Meaning of Colors in the Hierarchical Representation

Color	Meaning
Blue	The selected permissions group.
Purple	This permissions group is a child of the selected permissions group and directly inherits from the selected permissions group.
Light purple	This permissions group inherits indirectly from the selected permissions group over the hierarchy.
Red	This permissions group is a parent of the selected permissions group and passes inheritance to the selected permissions group.
Light	This permissions group passes inheritance indirectly to the selected permissions

Color	Meaning
red	group over the hierarchy.
Gray	This permissions group does not inherit or pass inheritance to the selected permissions group.

When a new permissions group is inserted into the hierarchical view the element is initially labeled as "New permissions group". The permissions group name is copied to the permissions group element when the hierarchical view is refreshed.

To display the permissions group elements hierarchically they are attached to each other with a connector. You can control the connection points with your mouse.

- To link permissions group elements, hold down the left mouse button and drag the connector from one permissions groups element to the other. The mouse cursor changes from a sideways pointing arrow to a downwards pointing arrow.
- To release the connection, hold the left mouse button down and drag the connector back from one permissions group element to the other. The mouse cursor changes from a sideways pointing arrow to an upwards pointing arrow in the process.

Related Topics

- [Adding a System User to Permissions Groups on page 229](#)

Copying Permissions Groups

The User & Permissions Group Editor provides a wizard for copying edit permissions and the user interface of an existing permissions group to a new permissions group.

To copy a permissions group

1. Select the category **Permissions | Permissions groups** in the Designer.
2. Select the permissions group you want to copy and start the User & Permissions Group Editor with the task **Edit permissions group**.
3. Select **Permissions groups | Copy permissions group....**

This starts a wizard for copying permissions groups.

4. Click **Next**.
5. The permissions group you want to copy is preselected. Enter a name for the new permissions group in **Copy name**.

A name suggestion is already entered in the field which is made up from the customer prefix and the original permissions group name. You can alter this name but the customer prefix has to remain.

6. Click **Next**.

7. Select the copy options.

Table 116: Copy Options for Permissions Groups

Option	Description
Permissions	Set this option to copy table and column permissions of the selected permissions group to the new permissions group.
Navigation	Select this option to copy menu items, forms and tasks of the selected permissions group to the new permissions group.
System user	Select this option if the system user should be copied to the new permissions group. ⓘ NOTE: Ensure that predefined system users are not copied to the new permissions group.

8. To start compiling, click **Next**.

The copying process may take some time. Each copy step and any error messages are displayed in the dialog window.

9. Click **Next**.

10. Click **Finish** to complete the wizard.

You can then run the following tasks:

- Add the permissions group to permissions group collections
- Adding the system user to the permissions group
- Assign additional edit permissions
- Assign user interface components, such as forms, menu items, tasks or program functions.

Related Topics

- [Manually Creating Permissions Groups on page 226](#)
- [Permissions Groups Properties on page 226](#)
- [Permissions Group Dependencies on page 223](#)
- [Adding a System User to Permissions Groups on page 229](#)
- [Editing Permissions for One Identity Manager Schema Tables and Columns on page 231](#)
- [Working with the User Interface on page 246](#)

Manually Creating Permissions Groups

To create a permissions group

1. Select the category **Permissions** in the Designer.
2. Start the User & Permissions Group Editor with the task **Show / edit permissions group**.
3. Add a new permissions group using the menu item **Permissions groups | New**.
4. Edit the master data for the permissions group.

You can then run the following tasks:

- Add the permissions group to permissions group collections
- Adding the system user to the permissions group
- Assigning Permissions
- Assign user interface components, such as forms, menu items, tasks or program functions.

Related Topics

- [Copying Permissions Groups on page 224](#)
- [Permissions Groups Properties on page 226](#)
- [Permissions Group Dependencies on page 223](#)
- [Adding a System User to Permissions Groups on page 229](#)
- [Editing Permissions for One Identity Manager Schema Tables and Columns on page 231](#)
- [Working with the User Interface on page 246](#)

Permissions Groups Properties

Table 117: Permissions Group Properties

Property	Description
Permissions group	Name of the permissions group. Label your own permissions groups with the prefix 'CCC'
Description	Detailed description of the permissions group's purpose.
Remarks	Spare text box for additional explanation.
Preprocessor condition	You can add a preprocessor condition to permissions groups. This means that the permissions group is only effective when the condition is met.

Property	Description
Permissions group binary pattern	The permissions group binary pattern is used to calculate effective system user permissions. It is provided by the DBQueue Processor.
Only use for role-based authentication	This group includes permissions, form assignments, menu items and program functions for role-based authentication. The permissions group can be assigned to One Identity Manager application roles and is assigned to dynamically determined system users. A direct assignment to non-dynamic system user is not permitted.

NOTE: This function is available if the Identity Management Base Module is installed.

Related Topics

- [Copying Permissions Groups on page 224](#)
- [Manually Creating Permissions Groups on page 226](#)
- [Dynamic system user on page 230](#)
- [Conditional Compilation using Preprocessor Conditions on page 448](#)

Editing System Users

To create a new system user

1. Select the category **Permissions** in the Designer.
2. Start the User & Permissions Group Editor with the task **Show / edit permissions group**.
3. Add a new system user using the menu item **Users | New**.

4. Edit the system user's master data.

Table 118: System User Properties

Property	Description
System user	Name of the system user for logging into the administration tools.
Password and password confirmation	Password for logging into the administration tools as system user.
Remarks	Spare text box for additional explanation.
Read-only	Set this option if the system is member in all permissions group but can only have read access. This results in overwriting all other edit permissions that the system user obtains through permissions group memberships.
Logins	Logins with which the system user can log in to One Identity Manager tools. Enter the login in the form: Domain\User. This information is required if the authentication module "Account-based system user" is used for logging into the One Identity Manager tools.
Administrative user	Specifies whether the user is an administrator. Administrative system users are automatically added to all non role-based permissions groups.
Service account	Specifies whether this is a system user used by a service account. This system user is not allocated a permissions groups but has all access permissions, tasks and program functionality.
External password management	Specifies whether the system user's password is determined by an external password manager. The password cannot be changed in One Identity Manager. Determining the system user's password must be custom implemented.

5. Add the system user to permissions groups.

Related Topics

- [Predefined Permissions Groups and System Users on page 216](#)
- [Adding a System User to Permissions Groups on page 229](#)
- [Dynamic system user on page 230](#)
- [Which Employees Use System Users? on page 230](#)
- [One Identity Manager Authentication Module on page 74](#)

Adding a System User to Permissions Groups

NOTE: You cannot manually add system users to permissions groups for role-based login. Dynamic system users are calculated for role-based login.

When you add a system user to the permissions groups, you allocate permissions for the One Identity Manager database model at the same time and make a user interface available to the user. The system user's effective permissions that are found are not saved in the One Identity Manager schema, but are determined when logging into One Identity Manager tools and then they are loaded.

The permissions group memberships for a system user are shown in the hierarchical view of the User & Permissions Group Editor. Direct and the inherited permissions group memberships are shown for the system user depending on the program settings (menu item **Options | Display permissions group inheritance**).

Figure 25: System User Permissions Group Memberships

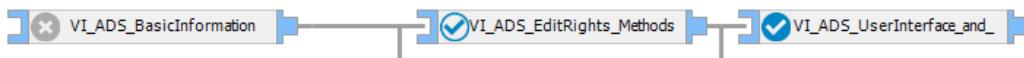


Table 119: Meaning of Icons in the Hierarchical Display

Icon	Meaning
☒	The selected system user is not assigned to this permissions group.
✓	The selected system user is assigned to this permissions group.
✓	The selected system user is indirectly assigned to this permissions group.
✓	The selected system user is directly and indirectly assigned to this permissions group.

To assign a system user to permissions groups

1. Select **Permissions | System users** in the Designer.
2. Select a system user and start the User & Permissions Group Editor with the **Edit system user** task.
3. Select the required permissions group in the hierarchical view. By clicking on the icon you add or delete the selected system user to or from a permissions group.

NOTE: The permissions group "QBM_BaseRights" defines basic permissions that are sufficient for logging a system user in to the administration tools. This permissions group is always assigned implicitly.

NOTE: The system user "viadmin" has all the permissions predefined by and the entire user interface. The system user "viadmin" also implicitly has all the permissions and user interface components from custom permissions groups.

NOTE: Administrative system users are automatically added to all non role-based permissions groups.

Related Topics

- [Dynamic system user on page 230](#)

Dynamic system user

Different role-based authentication modules are available for role-based login on One Identity Manager tools. First, the employee memberships in application roles are determined during log in with role-based authentication. Assignments of permissions group to application roles are used to determine which permissions groups apply to the employee. A dynamic system user is determined from these permissions groups that will be used for the employee's login.

NOTE: If nobody logs in for a long time using role-based authentication using dynamic system users, you should delete these system users on performance grounds.

To delete system users

- Set the configuration parameter "Common\DynamicUserLifetime" in the Designer and enter a maximum retention time (in days) for dynamic system users.
System users, whose retention time has expired, are deleted from the database in the process of daily maintenance work.

NOTE: A new dynamic system user is created if an employee logs in using role-based authentication at a later date.

Related Topics

- [Scheduled Maintenance Tasks on page 653](#)

Which Employees Use System Users?

You cannot change assignments in this view. Employees obtain a system user direct through their master data or dynamically through their One Identity Manager applications roles.

To display which employees are assigned to a system user

1. Select **Permissions | System users** in the Designer.
2. Select a system user and start the User & Permissions Group Editor with the **Edit system user** task.
3. Select the menu **View | One Identity Manager employees**.

You cannot change assignments in this view.

Editing Permissions for One Identity Manager Schema Tables and Columns

You grant One Identity Manager schema tables and columns permissions for permissions groups. You are not permitted to edit predefined permissions groups. Use the User & Permissions Group Editor to set up your own permissions groups.

To edit tables or columns

1. Select the category **Permissions** in the Designer.
2. Start the Permissions Editor using the task **Edit permissions**.
3. Select the permissions group you want to edit in the **Permissions group** menu in the Permissions Editor toolbar.
4. Edit the table or column permissions.
 - To add new permissions, select **New** in the context menu.
When new permissions are added, the tick boxes for the permissions are not set by default. That means, the corresponding permissions are withdrawn. You need to set the tick box to enable the permissions.
 - To delete all table or column permissions, select **Delete** in the context menu .
 - Deselect the tick box to delete single permissions.
5. To specify more conditions for table permissions
 - Open the single permissions view from the **View | Properties** menu, go to the **Permissions filter** tag and enter the condition.
 - Use the **SQL check** button to test the condition. This checks the syntax. The number of objects that comply is returned.

IMPORTANT: Permissions are always edited for permissions group you selected in the **Permissions group** menu. If you want to issue more permissions for permissions groups, select them in the menu first and then edit the permissions.

To copy permissions

1. Select the category **Permissions** in the Designer.
2. Start the Permissions Editor using the task **Edit permissions**.

3. Select the permissions group you want to edit in the **Permissions group** menu in the Permissions Editor toolbar.
4. Select the table or column from which you want to take the permissions in the Permissions Editor.
5. Use **Copy** in the context menu to copy the permissions owned by the selected table for this table or column.
6. Select the table or column for which you want to create the permissions.
7. Use **Paste** in the context menu to insert the copied permissions in the the table for the selected permissions group.

NOTE: To copy all permissions groups for a table or column, use the context menu item **Copy all permissions** and **Paste all permissions**. It does not matter which permissions group you select, in this case.

To display permissions for a table or column

1. Select the category **Permissions** in the Designer.
2. Start the Permissions Editor using the task **Edit permissions for table <table name>**.
3. Select the table or column for which you want to show the permissions in the Permissions Editor.
4. Select the **Object permissions** view.

The view "Summary of all permissions for" displays all permissions groups that own permissions on the table or column by is an extension of the edit permissions view. The permissions in this view cannot be edited.

TIP: A condition is displayed in full by clicking on it.

TIP: Use the context menu **Select in Permissions Editor** to display all permissions in the selected group in the Permissions Editor.

Detailed information about this topic

- [Rules for Finding Valid Permissions for Tables and Columns on page 233](#)
- [Granting Permissions for Tables on page 239](#)
- [Granting Column Permissions on page 240](#)

Related Topics

- [Editing Permissions Groups and System Users on page 219](#)

Rules for Finding Valid Permissions for Tables and Columns

When a system user is used to log into the system, the currently effective permissions for the objects are determined based on the permissions groups. The following rules are used to determine the resulting permissions:

- Permissions from hierarchical permissions groups are inherited from top to bottom. That means that a permissions group contains all the permissions belonging parent permissions groups.
- The number of objects is determined first for hierarchical permissions groups. Column permissions are decided afterwards. In some cases, this results in more permissions than are defined on individual permissions groups.
- A system user receives a permission when at least one of its permissions groups has the permission (directly or inherited).
- The limiting permissions conditions for all the system user's permissions groups are grouped together and used to determine a valid condition for each permission for viewing, editing, inserting and deleting an object.
- Fixed viewing permissions for the database system files are granted by the system, which are sufficient for logging a system user into One Identity Manager tools.
- A system user with read-only permissions only obtains viewing permissions to objects irrespective of any other permissions.
- If permissions are granted on a table for inserting, editing or deleting, viewing permissions are implicit.
- If permissions are granted on a column for inserting, editing or deleting, viewing permissions are implicit.
- If permissions are granted for a table, then viewing permissions are implicitly granted on the primary key column of the table.
- If viewing permissions are granted on a primary key column as a minimum, then viewing permissions are implicitly granted for references table, the primary key column and the columns that are necessary on the referenced table for viewing according to the defined display pattern.
- Permissions for database views of type "proxy" are also valid for underlying tables.
- Database view of type "ReadOnly" only have viewing permissions irrespective of any other permissions.
- If a table or column is disabled due to preprocessor conditions, permissions are not determined for those tables and columns. The table or column is considered not to exist.
- If a permissions group is disabled due to preprocessor conditions, permissions are not taken into account for this permissions group. The permissions group is considered not to exist.

Example of Permissions Grouping using Permissions Groups

The following example shows how to group permissions if the user is directly assigned in permissions groups and the permissions groups are not connected hierarchically.

A system user obtains permissions to the table `ADSAccount` through different permissions groups.

Permissions group	Viewable	Editable	Insertable	Deletable
A	1	1	1	1
B	0	0	0	0

In addition, it is granted permissions to the table `LDAPAccount` through these permissions groups.

Permissions group	Viewable	Editable	Insertable	Deletable
A	1	0	0	0
B	1	1	1	0

Therefore, the system user has effectively the following permissions:

Table	Viewable	Editable	Insertable	Deletable
<code>ADSAccount</code>	1	1	1	1
<code>LDAPAccount</code>	1	1	1	0

Example of Limiting Conditions

A system user obtains viewing permissions to the table `Person` through different permissions groups:

Permissions group	Viewing Condition	Column Viewing Permissions
A		Lastname
B	Lastname like 'B%'	Lastname, Firstname, Entrydate
C	Lastname like 'Be%'	Lastname, Firstname, Gender
D	Lastname like 'D%'	Lastname

This results in the following permissions for the individual employee objects.

Person.Lastname	Visible Columns
Smith	Lastname
Bishop	Lastname, Firstname, Entrydate
Bennett	Lastname, Firstname, Gender
Dummy	Lastname

Working with the Permissions Editor

Use the Permissions Editor to grant permissions groups the permissions for accessing the tables and columns in the One Identity Manager's schema. The editor is started from the program "Designer" and opens in the document view. Only additional Permissions Editor functions are described in the following.

Menu Items

The following items are added to the menu bar when the editor starts.

Table 120: Meaning of Items in the Menu Bar

Menu	Menu Item	Meaning	Key Combination
Permissions	New	Creates table or column permissions for the selected permissions group or system user.	Ins
	Delete	Deletes table or column permissions for the selected permissions group or system user.	Del
	Copy	Copies table (column) permissions from the selected permissions group or system user.	Ctrl + C
	Paste	Inserts copied table (column) permissions into the selected permissions group.	Ctrl + V
	Copy all permissions	Copies all table (column) permissions from the selected permissions group.	Ctrl + Shift + C
	Paste all permissions	Inserts all copied table (column) permissions into the selected permissions group.	Ctrl + Shift + V
	Refresh view	Refreshes permissions display.	
Options	Sort permissions	Sorts the view. Tables and columns with permissions are shown first.	

Menu	Menu Item	Meaning	Key Combination
Tables	Use display values	The display names of the columns and tables are shown. If this option is not set, the table and column names from the One Identity Manager schema are shown.	
	Show all tables	Shows all database model tables.	
	Show non-system tables	Only shows tables from the application data model.	
	Show system tables	Only shows tables from the system data model.	
	Show disabled tables	Shows/hides disabled tables.	
	Define filter...	Opens a dialog window for creating an ad hoc filter.	
	Manage filters...	Opens a dialog window for creating permanent filters.	
	Properties	Shows/hides the edit view.	
View	Object permissions	Shows/hides the objects permissions.	
	Permissions management help	Opens the help on this topic.	
Help	Permissions Editor help	Opens the editor help.	
	Editor help	Opens the help on this topic.	

Table 121: Meaning of Toolbar Icons

Icon	Meaning
	Creates table or column permissions for the selected permissions group or system user.
	Deletes table or column permissions for the selected permissions group or system user.
	Refreshes permissions display.
	Copies table (column) permissions from the selected permissions group or system user.

Icon	Meaning
	Inserts copied table (column) permissions into the selected permissions group.
	Copies all table (column) permissions from the selected permissions group.
	Inserts all copied table (column) permissions into the selected permissions group.
	Starts the wizard for defining custom filters. On completion the tables are shown according to the filter. For more information, see Using User Defined Filters for Searching on page 38.
	Sorts the view. Tables and columns with permissions are shown first.
	The display names of the columns and tables are shown. If the option is not enabled, the technical names according to the One Identity Manager schema are shown.
	Shows disabled tables.

Views in the Permissions Editor

The Permissions Editor has several views for displaying and editing access permissions for permissions groups.

Table 122: Permissions Editor Views

View	Description
Edit and simulated permissions view	This view contains two permissions views. In one, the permissions of a permissions group are edit in the tables and columns of the data model and in the other, the current permissions situation is established by the permissions simulation. For more information, see Functions in the Permissions Edit View on page 237.
Resulting permissions view	All the permissions groups that have permissions for a selected table or column are displayed with their own permissions. The permissions in this view cannot be edited. This shows which of the permissions groups selected in the simulation have which permissions. Effective permissions are also displayed.
Edit individual permissions view	This view is in addition to the edit permissions view. You can make further changes to table or column permissions of permissions groups and user account in this view.

Functions in the Permissions Edit View

This view contains two permissions views. In one, the permissions of a permissions group are edit in the tables and columns of the data model and in the other, the current

permissions situation is established by the permissions simulation.

The permissions displayed in the edit permissions view always relate to the permissions group which is selected in the toolbar. You can add, edit and delete permissions for the permissions group for table or columns.

Each of the permissions in the view is represented by a check box. If the check box is not active the corresponding permission is withdrawn. If the check box is set, the corresponding permission is allowed.

 **NOTE:** Use **SHIFT** + left mouse button or **CTRL** + left mouse button to select multiple tables or columns for editing.

Table 123: Meaning of Icon Used

Icon	Meaning
	Table
	Column
	Foreign key column (FK)
	Primary key column (PK)

Table 124: Context Menu Items for Editing Permissions

Context Menu Item	Meaning
New	Adds a new table or column permission for a permissions group or account user.
Delete	Deletes a table or column permission for a permissions group or account user.
Copy	Copies table (column) permissions from the selected permissions group or system user.
Paste	Inserts copied table (column) permissions into the selected permissions group.
Copy all permissions	Copies all table (column) permissions from the selected permissions group.
Paste all permissions	Inserts all copied table (column) permissions into the selected permissions group.
Navigation	Shows all other editors that can be used with the selected object.

Filtering Entries in the Permissions Editor

Use the **Options** item in the Permissions Editor menu with its predefined filters and filter conditions to limit the number of tables displayed in the editing and simulating

permissions view.

Table 125: Predefined Filters in the Editor

Filter	Meaning
Show all tables	Shows all database model tables.
Show non-system tables	Only shows tables from the application data model.
Show system tables	Only shows tables from the system data model.
Show disabled tables	Disabled tables are displayed as well.

You also have the option to set up ad hoc filters and permanent filters. Ad hoc filters are used for one-off searches. These filters are not saved and are immediately applied to the display. A permanent filter is recommended if you want to reuse it more frequently. Permanent filters are saved and are therefore always available for use.

Related Topics

- [Using User Defined Filters for Searching on page 38](#)

Granting Permissions for Tables

You can give tables the following permissions.

Table 126: Table Permissions

Permissions	Meaning
Viewable	The table data is displayed.
Insertable	New data can be added to the table.
Editable	Table data can be edited.
Deletable	Table data can be deleted.

NOTE: If permissions "Insertable", "Editable" or "Deletable" are granted, "Viewable" permissions are implicit. In this case, the option **Viewable** is grayed out in the Permissions Editor.

Permissions filter

You can limit table elements by setting conditions for viewing, editing, inserting and deleting. This makes it possible to link editability of employee data sets to their names, for example. In this way, a user can allow read access only to employee data where the last names begin with A-F, whereas employee data from G-Z can be given write access.

NOTE: Limiting conditions can only be defined for application database tables.

Table 127: Permissions filter

Condition	Meaning
for viewing	Limiting condition for displaying data sets.
for editing	Limiting condition for editing data sets.
for inserting	Limiting condition for inserting data sets.
for deleting	Limiting condition for deleting data sets.

Format your condition like a where-clause in a database query.

Example

For example, if the user should be able to view all employee, but only edit those with a last name beginning with "B", you can format the condition for editing as follows:

Lastname like 'B%'

TIP: Use the **SQL check** button to test the condition. This checks the syntax. The number of objects that comply is returned.

Related Topics

- [Editing Permissions for One Identity Manager Schema Tables and Columns on page 231](#)
- [Rules for Finding Valid Permissions for Tables and Columns on page 233](#)
- [Granting Column Permissions on page 240](#)

Granting Column Permissions

IMPORTANT: When you grant column permissions, you must apply the same permissions to the tables. This means, a column is only visible when the table is too.

You can grant the following permissions for columns:

Table 128: Column Permissions

Permissions	Meaning
Viewable	The column is displayed.
Editable	The values in the columns can be changed

Permissions Meaning

Insertable The value in the column can be edited when a new data record. Once the data record has been saved it can no longer be edited.

For example, when an Active Directory User is created, an Active Directory Container is defined. Because this is a key field the Active Directory Container cannot be changed after the object has been saved.

NOTE: If permissions "Insertable", "Editable" or "Deletable" are granted, "Viewable" permissions are implicit.

Related Topics

- [Editing Permissions for One Identity Manager Schema Tables and Columns on page 231](#)
- [Rules for Finding Valid Permissions for Tables and Columns on page 233](#)
- [Granting Permissions for Tables on page 239](#)

Availability of Certain Functionality

There is certain functionality in One Identity Manager administration tools that is only available to specific users. This includes exporting data from the Manager, calling the SQL Editor in the Designer or running consistency checks, for example. Furthermore, you can control execution of methods, scripts or processes through program functionality.

Program functions are not assigned to single users but to permissions groups. All users that are assigned to these groups can use the program function.

To make a program function available to users

1. Select the category **Permissions | Program functions** in the Designer.
2. Select the program function and assign it to a permissions group.
 - Select **View | Select table relations...** and enable the table DialogGroupHasFeature. This shows the tab **Permissions groups** for assigning permissions groups.

Assignment to Method Definitions

If a method definition is assigned a program function (table QBMethodHasFeature), the user can only run this method if the necessary program function is granted to him. An error occurs if the user does not own this program function and tries to run it.

To make a method definition available to users using a program function

1. Connect the task definition with the program function.
 - a. Select the category **Permissions | Program functions** in the Designer.
 - b. Select the program function and assign the method definition to it.
 - Select the menu item **View | Select table relations...** and enable the table DialogMethodHasFeature. Use the **Methods** tab displayed in the edit view to assigned the method definition.
2. Assign a permissions group to the program function.

Assignment to Scripts

If a script is assigned a program function (table QBMScriptHasFeature), the user can only run this script if the necessary program function is granted to him. An error occurs if the user does not own this program function and tries to run it.

To make a script available to users using a program function

1. Connect the script with the program function.
 - a. Select the category **Permissions | Program functions** in the Designer.
 - b. Select the program function and assign the script.
 - Select the **View | Select table relations...** in the menu and enable the table DialogScriptHasFeature. You can assign the script on the **Script** tab displayed in the edit view.
2. Assign a permissions group to the program function.

Assignment to Events

In One Identity Manager, how events for stored processes are triggered is linked with the permissions concept. Users can only trigger events on objects like this if they own edit permissions for them. This can lead to table users, who only have viewing permissions, not being able to trigger additional events for processes.

In this case, it is possible, to connect called object events to a program function. An object event (table QBMEvent) can be associated to an event of a process (column JobEventGen._UID_QBMEvent).

If the object event is assigned a program function (table QBMEventHasFeature), users who are granted the program function, can trigger the associated object events and therefore the processes, regardless of their permissions.

To allow triggering a process through a program function

1. Link an object event with the program function.
 - a. Select the category **Permissions | Program functions** in the Designer.
 - b. Select the program function and assign the object event to it.
 - a. Select **View | Select table relations...** in the menu and enable the table QBMEventHasFeature. The **Object events** tab is displayed in the edit view.
2. Assign a permissions group to the program function.

Related Topics

- [Task Definitions for the User Interface on page 322](#)
- [Creating and Editing Events on page 397](#)

Displaying Permissions for an Object

You can display object properties and permissions in the One Identity Manager tools.

To show extended object properties

- Select the object and open the **Properties...** from the context menu.

On the **General** tab you can see the object's general properties, for example, ID, status or primary key.

All the object columns are displayed in a grid on the **Properties** tab with the values. You can choose between a simple column view and the advanced view with additional data for column definitions.

Table 129: Icon used for Column Properties

Icon	Meaning
	Compulsory field.
	No viewing permissions.
	No edit permissions.

On the **Access permissions** tab, you can see which permissions are valid for an object based on permissions groups. The first entry shows the basic permissions for the table. The permissions for this particular object are displayed beneath that. The other entries show the column permissions. By double-clicking on a table, an object or a column entry all the permissions groups are displayed that are used to determine the permissions.

Figure 26: Displaying Object Permissions

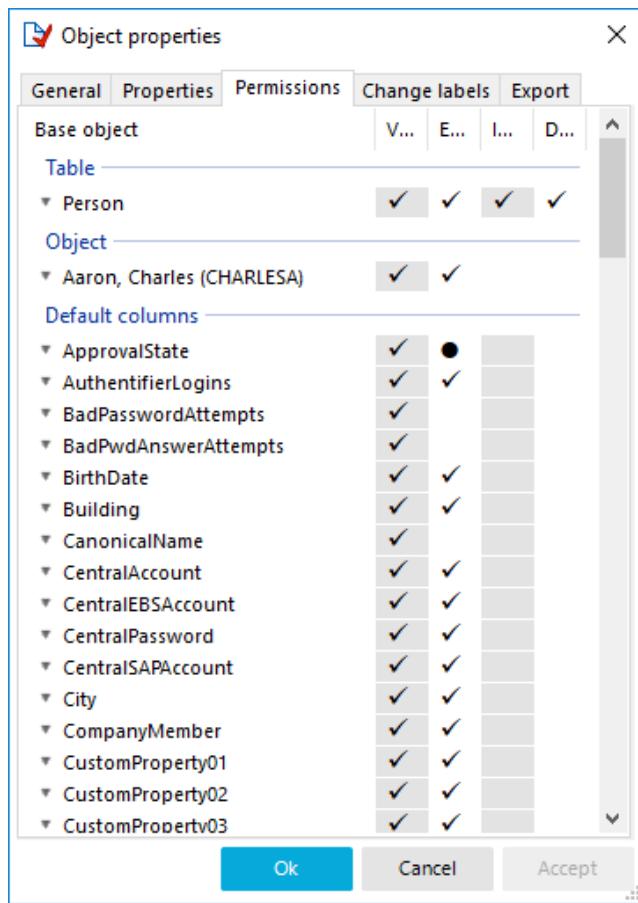


Table 130: Icon used for Permissions

Icon	Meaning
✓	Permissions exist.
●	Permissions have been removed by the object layer
☒	Permissions limited by conditions.

Displaying the Current User's Permissions

To get more information about the current user

- Double-click the icon in the status bar to display more user information.

Table 131: Extra Information about the Current User

Property	Meaning
System user	Name of system user
Authenticated by	Name of the authentication module used for logging in.
Employee UID (UserUID)	Unique ID for the current user's employee if an employee related authentication module is used to log in.
Read-only	The system user has only has read permissions. Modification to data are not possible.
Dynamic user	The current user uses a dynamic system user. Dynamic system users are applied when a role-based authentication module is used.
Remarks	More details about the system user in use.
Permissions group	Permissions groups that are assigned to the system user. Which user interface and editing permissions apply depend on the permissions groups.
Program functions	Program functions assigned to the system user. The menu items and functions available depend on the program functions.

Working with the User Interface

Certain components of the One Identity Manager's graphical user interface are stored in the One Identity Manager schema and can be tailored to suit customer requirements. Menu items in the navigation structure, interface forms, and task definitions can be configured in this way.

Menu items, interface forms and task definitions are assigned to permissions groups. The user's effective components of the user interface depend on the authentication module used for logging in to the One Identity Manager tools. If a user logs in to a One Identity Manager tool, a system user is found and the available menu items, interface forms, task definitions, and individual program functions are identified depending on the permission groups to which this system user belongs and the adapted user interface is loaded.

Data is displayed as objects in the user interface. User interface objects are meta-objects. You provide a selection of configurable elements, which describes how the data stored in the database is perceived. These objects enable data to be distinguished by specific properties. They provide an additional control function for configuring the user interface. Hence, interface forms and tasks are linked to object definitions which means that different forms and tasks are displayed in the user interface depending on which object is selected.

You can only modify the supplied user interface components to a certain extent and they are overwritten by schema installation. You can integrate components of the default user interface into your own user defined user interface. If necessary you can disable individual components of the default user interface to stop them from being displayed. The system users provided are not effected by this limitation. Components labeled as disabled remain so after by schema installation.

Captions are used in the user interface to create user friendly names for different components of the user interface such as menu items, tasks and column names. You can maintain multi-language display text in the One Identity Manager which enables you to display captions in different languages.

The default One Identity Manager installation is supplied in the languages "English - United States [en-US]" and "German - Germany [de-DE]". You can add other languages to the user interface and display text if required. In this case, you should translate the text before One Identity Manager goes live. There is a Language Editor in the Designer to help you do this. A special control element is provided in the One Identity Manager tools which aids multi-language input.

A user interface is always set up for one application. The One Identity Manager's default version supplies applications and predefined navigation menus for the "Manager", "Designer" and "Launchpad".

Detailed information about this topic

- [Object definitions for the User Interface on page 247](#)
- [User Interface Navigation on page 251](#)
- [Forms for the User Interface on page 274](#)
- [Statistics in the One Identity Manager on page 306](#)
- [Extending the Launchpad on page 319](#)
- [Task Definitions for the User Interface on page 322](#)
- [Applications for Configuring the User Interface on page 325](#)
- [Icons and Images for Configuring the User Interface on page 327](#)
- [Language Dependent Data Representation on page 328](#)

Object definitions for the User Interface

The data in the user interface is represented by objects. Objects in the user interface map the data stored in the database. These objects can be configured and enable data to be distinguished by specific properties.

User interface forms and task definitions are linked to object definitions and displayed depending on the selected object definition. Object definitions provide an additional control function for configuring the user interface.

You can assign several objects to each table in the One Identity Manager schema. Basically, each database table should have at least one object definition that is generally valid, that means, without limited selection criterion. Other object definitions then relate to the respective special case limited by the general case.

TIP: To create object definitions for new tables, run the consistency check "Missing DialogObject" in Designer and use the repair mode. You must edit object definitions created like this afterward.

Table 132: Example Relationship between Tables and User Interface Object Definitions

Table	Object definition	Limitation according to Object Definition
Employee	Employee general	None
Employee	Employee dummy	Employees labeled as "Dummy employee".

Detailed information about this topic

- Selection Criteria for Object Definitions on page 248
- Using Display Text for Object Definitions on page 248
- Working with Object Definitions on page 249
- Object Definition Properties on page 250
- Effects of Object Definitions when Displaying Interface Forms on page 286

Selection Criteria for Object Definitions

The table entries to be displayed are found through a selection script and an object definition condition.

- Write a selection script (VB.Net term) that either returns "true" or "false" depending on whether the parent data set belongs to this meta object type or not.
- Write a condition (WHERE clause as database query) such that a meta object can also be used in display list.

IMPORTANT: You must compile the database for the selection criteria to come into effect.

Example of displaying employees labeled as "Dummy employee"

Selection script to determine at run-time whether the data set being dealt with is an employee with the property "dummy employee":

Value = \$IsDummyPerson:Bool\$

Condition (WHERE clause) to select all employees with the property "Dummy employee".

IsDummyPerson=1

Related Topics

- Object Definition Properties on page 250

Using Display Text for Object Definitions

You can define the following captions to represent each object definition in the administration tool user interface.

- List caption

The list caption is used in One Identity Manager tools as the title for result lists. The display text of the object definition that you specified through the selected menu item, is used as the list title.

- Form caption

The form caption is used to display the current object definition, for example, in the Manager status bar.

The current object definition is determined when an item in the administration tool result list is selected. Valid object definitions and thereby the possible captions are determined by selection scripts. Of these, the caption of the object definition with the lowest sort order is shown.

Example

Table 133: Captions depending on the Sort Order of the Object Definitions

Object definition	Selection script	Sort order	Caption
Employee general	None	10	Employees
Employee dummy	Employees labeled as dummy.	1	Dummy employee

When an employee is selected in the result list, the corresponding display text is "Employees". If an employee is labeled as a dummy employee then it is assigned to another object definition using the VB.Net statement and the display text "Dummy employee" is displayed.

Related Topics

- [Object Definition Properties on page 250](#)
- [Editing Lists on page 267](#)

Working with Object Definitions

Predefined configurations are maintained by the schema installation and cannot be edited apart from a few properties. The default configuration is moved to a configuration buffer during handling. You can retrieve changes from the configuration buffer and restore the default configuration in this way.

To define objects definitions

1. Select the category **User Interface | Object definitions** in the Designer.
 2. Select one of the object definitions in the list.
 - OR -
- Add a new object definition using **Object | New** in the menu bar.
3. Enter the object definition's master data.

Related Topics

- [Object Definition Properties on page 250](#)

Object Definition Properties

Table 134: Object Definition Properties

Property	Meaning
Exclusive	Objects labeled with this option are considered exclusive. That means, all other possible matching object definitions are not accepted as valid. If several object definitions of one table are labeled as exclusive, the object definition with the lowest sort order applies.
Display template	The display template specifies the form in which the data sets in the administration tool result lists are displayed.
Display name	The object's display name is used, for example, to identify the table in a database search or for error output. Display names can be given in more than one language.
List caption	Caption used to display the list title in the user interface.
Form caption	Caption used to display the form title in the user interface.
Selection script	Selection script as a VB.Net term, to determine during runtime whether the database object passed down belongs to this object definition. ① NOTE: The database needs to be complied after changing modifying the selection script.
Processing status	Object processing status. The process state is used for creating customer configuration packages.
Condition	Condition required for the object definition to be used for displaying in lists. The condition is defined as a database query WHERE clause. ① NOTE: The selection script and the condition must match. If one of the properties is given then the other one also has to be given!
Remarks	Spare text box for additional explanation.
Disabled by preprocessor	If an object definition is excluded through a preprocessor condition, this option is set by the Database Compiler.
Insert values	Default settings for fields that are assigned when a new data set is added. The input is in VB.Net syntax. ① NOTE: The database needs to be complied after changing modifying the values.

Property	Meaning
Background color	Color, with which the control for this object is displayed in the schema overview.
Object name	Name of the object.
Preprocessor condition	Object definitions can have preprocessor conditions added. This means, an object definition is only available when the preprocessor condition is fulfilled. ⓘ NOTE: You can find an overview of existing preprocessor dependencies in the Designer in the category Database One Identity Manager Schema Preprocessor dependencies .
Sort order	The sort order is used for displaying the form title when an object is selected. The smaller the sort order magnitude, the stronger the restrictions defined for the object.
Icon	Icon for displaying the object definition.
Table	Table for which the object definition is created.

Related Topics

- [Selection Criteria for Object Definitions on page 248](#)
- [Using Display Text for Object Definitions on page 248](#)
- [Display Template for Displaying a List on page 268](#)
- [Language Dependent Data Representation on page 328](#)
- [Defining Insert Values on page 269](#)
- [Conditional Compilation using Preprocessor Conditions on page 448](#)
- [Icons and Images for Configuring the User Interface on page 327](#)

User Interface Navigation

One Identity Manager administration tools with their own user interface are given their own navigation view. The navigation defines specific entry points into the One Identity Manager tool's user interface and controls the user oriented navigation down to the selection of an object in the result list. You can set up the structure of the user interface navigation through a menu. There are different types of menu items with specific uses. You can design a multifaceted navigation by combining different types of menu items.

Detailed information about this topic

- [Navigation View Elements on page 252](#)
- [Recommendations for Editing Menu Navigation on page 253](#)

- [Editing Menu Items on page 258](#)
- [General Menu Item Properties on page 262](#)
- [Creating Database Queries for Data Dependent Menu Items on page 264](#)
- [Editing Lists on page 267](#)
- [Using Links on page 270](#)
- [Using Variables on page 271](#)

Navigation View Elements

Table 135: Types of Menu Items

Type	Description
Menu category	Menu categories are displayed at the navigation top level and provide a method of grouping the data to be managed from different viewpoints. Menu categories constitute entry points into the interface navigation view. Menu categories are displayed as categories in the user interface.
Fixed Menu Item	Fixed menu items are used to organize data more clearly within menu categories. These menu items are always shown in the navigation view. List properties can only be defined for fixed menu items. These specify how the table entries are displayed in the user interface result list.
Data Dependent Menu Item	Data dependent menu items are generated by a database query that returns several data sets as output. These menu items are therefore not individual menu items, but a set of menu items depending on the output of the database query. List properties can be defined for data dependent menu items. These specify how the table entries are displayed in the user interface result list.
Detached Menu Item	Detached menu items are used to group other menu items or to define a main menu item for an application. For example, you can specify a web interface home page with a detached menu item. Detached menu items should always be created at the navigation top level. However, they do not appear in the administration tools navigation view.
Link	Links support the navigation configuration. They are used to reference frequently accessed menu items. Parts of the navigation interface that require an application several times, only need to be set up once. The referenced menu items are always shown in navigation interface as opposed to the links.
Main Form Element	Main form elements are not menu items in the navigation view, but are used as the main elements in object overview forms. All child menu items are assigned to the main element.
Task category	Task categories are represented at navigation top level and are used to organize task oriented workflows. Task categories are not mapped in the

Type	Description
	navigation view but on a special form in the administration tools.
Task	Tasks are used to map single tasks within a task category. They are used, for example, as starting points for administration tool wizards. Tasks are always listed under a task category menu item. Task categories and their tasks are not displayed in the navigation view but on a special form.
Statistics	This menu item is used to display statistics. Statistics are typically displayed in the info system. All statistics that are defined in one menu level can be displayed on one form or as individual menu items. Statistics can also be linked in as form elements.

Related Topics

- [Recommendations for Editing Menu Navigation on page 253](#)
- [Editing Menu Items on page 258](#)
- [General Menu Item Properties on page 262](#)
- [Creating Database Queries for Data Dependent Menu Items on page 264](#)
- [Editing Lists on page 267](#)
- [Using Links on page 270](#)
- [Working with Overview Forms on page 297](#)
- [Linking Statistics into the User Interface on page 310](#)

Recommendations for Editing Menu Navigation

- For fixed and database dependent menu items you can specify list properties like display templates or object definition to be used. These properties determine how the table entries are displayed in the user interface result list.

 **TIP:** You can define display templates for menu items, object definitions and tables.

The display template is determined by the following in order:

1. List display template for the menu item
2. Object definition display template
3. Table display template

- Create menu items you can use as references (links). Thus, the parts of the navigation interface an application uses several times, only need to be created once. The referenced menu items are always shown in navigation interface as opposed to the links.

- Utilize variables in designing menu item names and display templates as well as in insert values and database queries.

TIP: It is best to define the required variable already in the menu item for the menu category. Variables are inherited within a hierarchical navigation. This means that variables can be reused or overwritten on lower hierarchy levels. At run-time the actual value is passed to the variables.

- To display menu items in the user interface, assign the menu items to the program and permissions groups for non role-based and role-based login.

Related Topics

- [Navigation View Elements on page 252](#)
- [Editing Menu Items on page 258](#)
- [Editing Lists on page 267](#)
- [Using Links on page 270](#)
- [Using Variables](#)

Working with the User Interface Editor

Edit the user interface navigation with the User Interface Editor. The editor is started from the program "Designer" and opens in the document view. Only additional User Interface Editor functions are described in the following.

Menu Items

The following items are added to the menu bar when the editor starts.

Table 136: Meaning of Items in the Menu Bar

Menu	Menu Item	Meaning	Key Combination
Menu Item	New	Creates a new menu item	
	Delete	Deletes the selected menu item and all child items after requesting confirmation.	
	Copy	Copies the selected menu.	
	Paste	Inserts copied menu item. If a menu item is copied with child items, then the menu item is inserted with all children.	
	Copy with child nodes	Copies the selected menu item with all its children.	
	Create root node	Creates a new menu category.	
	Reload nodes	Updates the menu items in the list.	F5
Simulation	Start simulation	Starts user interface navigation simulation.	F9
	Simulate overview form	Starts simulation of an overview form.	
	Define simulations data...	Starts a wizard for defining the simulation data.	
Options	Create menu markers...	Opens a dialog window for creating menu markers.	
	Delete menu markers	Deletes menu markers.	
	Show captions	Shows the menu items display text. If this option is not active, the identifiers from the data model are shown.	
	Select columns...	Opens a dialog window for selecting columns that can be additionally displayed in the user interface navigation.	

Menu	Menu Item	Meaning	Key Combination
View	Properties	Shows/hides the edit view.	
	Variable definition	Shows/hides the variable definitions edit view.	
	Permissions group	Shows/hides the permissions group edit view.	
	Related Applications	Shows/hides the application edit view.	
	Forms	Shows/hides the forms edit view.	
Help	Navigation help	Opens the help on this topic.	
	User Interface Editor help	Opens the editor help.	

Table 137: Meaning of Toolbar Icons

Icon	Meaning
	Creates a new menu item.
	Deletes a menu item.
	Copies a menu items.
	Copies menu items with all its children.
	Inserts a copied menu item.
	Toggles display text with data model captions.
	Starts the WHERE clause wizard to define menu markers.
	Deletes menu markers.
	Refreshes the display.
	Starts the simulation.
	Starts the wizard for simulating overview forms.
	Starts the simulation data wizard.

Views in the User Interface Editor

The User Interface Editor makes different views available for editing the user interface:

Table 138: User Interface Editor Views

View	Description
Navigation Overview	Menu items and links are presented below one another. This is where you can create and delete menu items. ① TIP: Use "drag and drop" to move menu items around within the hierarchy.
Properties	Use the view edit the properties of the selected menu item. A default context menu is available for input fields.
Variable definition	In this view, all the variable definitions that belong to the selected menu item are displayed in tabular form with type, name and assigned value. When you click on the variable type, a pop list with all available types is shown. The variable name and the value can be edited by clicking on the appropriate column in the table. Alternatively, you can enter the variable definition in a dialog window. To open the dialog window, select the edit icon in the toolbar.
Permissions group	Displays all available permissions groups. You can assign menu items to the permissions groups or delete existing assignments. These menu items are available to system users depending on their permissions group memberships.
Related Applications	All available applications are displayed. You can assign menu items to the applications or remove existing assignments.
Forms	All available interface forms are shown. You can assign menu items or remove existing assignments. When the associated menu item is selected in the navigation or the item is selected in the result list, the interface forms are shown for all system users without taking their permissions group memberships into account.

Table 139: Item in the Navigation Overview Context Menu

Context Menu Item	Meaning
New	Creates a new menu item.
Delete	Deletes the selected menu item.
Properties	Displays the object properties of the selected entry.
Move up	Moves the selected menu item up one hierarchy level.
Move down	Moves the selected menu item down one hierarchy level
Follow link node	Tracks a link if the menu item is of type "link".
Referenced by	Shows all menu items that are linked to the currently selected menu item.

Context Menu Item	Meaning
Copy	Copies the selected menu item into the clipboard.
Copy with child nodes	Copies the selected menu item with all its child menu items into the clipboard.
Paste	Inserts the copied menu items at the selected position in the navigation.
Navigation	Shows all other editors that can be used with the selected object.

Editing Menu Items

The navigation menu is mapped in the category **User interface | User interface navigation** in the Designer. When you select a menu item in the navigation view, the properties of the menu item are shown in the edit view. The type of menu item determines the availability and editability of the properties.

 **TIP:** You can use the User Interface Editor's wizard to create a preselection of menu items to be edited.

To create a new category

1. Select the category **User Interface | User interface navigation** in the Designer.
2. Start the User Interface Editor with **Edit navigation** in the task view.
3. To create a menu category, select **Menu item | New navigation category**.

 **NOTE:** Change the item type to "Task category" if it should represent a task category.

To create a new menu item

1. Select the category **User Interface | User interface navigation** in the Designer.
2. Start the User Interface Editor with **Edit navigation** in the task view.
3. Select the menu item under which you want to create the menu item in the navigation overview.
4. Select **New** from the context menu.

To copy and paste a menu item

1. Select the category **User Interface | User interface navigation** in the Designer.
2. Start the User Interface Editor with **Edit navigation** in the task view.
3. Select the menu item you want to copy in the navigation overview.

4. Select one of the copy options in the context menu.
 - Select **Copy** to copy the selected menu item.
 - Select **Copy with child items** to copy the selected menu item and the submenu items.
5. Select the menu item under which you want to create the menu item in the navigation overview.
6. Select **New** from the context menu.

TIP: Use "drag and drop" to move menu items around within the hierarchy.

You must make the following assignments to show the menu items in the user interface:

- All menu items that are to be displayed in an application user interface have to be assigned to a permissions group and an application. Use the views **Program** and **Permissions groups** views.
- You can assign user interface forms for individual menu items. When the associated menu item is selected in the navigation or the item is selected in the result list, the interface forms are shown for all system users without taking their permissions group memberships into account. Use the **Form assignment** view to do this. Set the option **Show in navigation** in the view to control how this is displayed.

NOTE: Recompile the database after adding or deleting a menu item.

Related Topics

- [Selecting the Type of Navigation View to Edit on page 259](#)
- [Copy an existing user interface navigation on page 260](#)
- [Compiling a One Identity Manager Database on page 64](#)

Selecting the Type of Navigation View to Edit

You can use the User Interface Editor's wizard to create a preselection of menu items to be edited. The wizard offer several possible methods for selecting the best suited user interface navigation:

- Creating a new user interface navigation
- Selecting menu items in an existing user interface navigation
- Copy an existing user interface navigation

To select a user interface navigation to edit

1. Select the category **User Interface | User interface navigation** in the Designer.
2. To start the wizard, select the task **Load wizard to edit user interface navigation**.
3. Click **Next**.

- Select the loading method by clicking on the corresponding button.

Table 140: Loading Method for Selecting Menus

Loading Method	Description
Create a new user interface navigation	Creates a new user interface navigation with initial menu categories.
Direct selection of defined menu items	To select defined menu items in the wizard, navigation menus are displayed for all applications in the database. Select the menu item you want to edit.
Load menu items through permissions groups or users	To group menu item by permissions groups, select all permissions groups of one system user or single permissions groups.
Copy a user interface navigation	You can use the wizard to copy menu items from an existing permissions group to another permissions group.
Load menu items using WHERE clause	You can load menu items directly using a WHERE clause. Enter the WHERE clause manually or use the WHERE clause wizard to create one.

- Click **Finish** to complete the wizard.

After completing the wizard, the menu items corresponding to the selection are loaded and presented for editing in the User Interface Editor.

Related Topics

- [Copy an existing user interface navigation on page 260](#)
- [Editing Menu Items on page 258](#)

Copy an existing user interface navigation

You can use the wizard to copy menu items from an existing permissions group to another permissions group. The wizard is not just limited to copying each menu item but can also copy the necessary permissions structure.

To copy an existing user interface navigation

- Select the category **User Interface | User interface navigation** in the Designer.
- To start the wizard, select the task **Load wizard to edit user interface navigation**.
- Click **Next**.
- Select the loading method to copy the navigation by clicking on the corresponding button.

5. Select the permissions group whose menu you want to copy. You can limit the permissions groups by selecting a particular system user account or you can directly select the permissions group.
6. Click **Next**.
7. Use **Copy to (new) permissions group** to specify the permissions group to which to copy the parts of the navigation.
 - Enter the name of the new permissions group. Ensure that your own permissions groups begin with the customer prefix.
 - OR -
 - Select an existing permissions group.
8. Enter the **Name prefix** and an optional **Name suffix**.
 This data is used to label the menu items. A name prefix is required for making the new menu item names because names of menu items in the navigation must be unique. Use the customer prefix as name prefix.
9. (Optional) Select copy options.

Table 141: Copy Options

Option	Meaning
Copy column permissions	Copies the column permissions of the permissions group.
Copy table permissions	Copies table permissions of the permissions group.
Copy user interface form assignments	Copies the user interface forms of the permissions group.
Copy method assignments	Copies method definitions of the permissions group.

10. To start copying, click **Next**.

The copy process can take some time depending on the number of selected parts. The components to be copied are displayed.

11. Once processing is complete, click **Next**.
12. Click **Finish** to complete the wizard.

After completing the wizard, the menu items corresponding to the selection are loaded and presented for editing in the User Interface Editor.

The copy process can take some time depending on the number of selected parts. After the wizard has finished, the menu items are displayed in the User Interface Editor for further editing.

Related Topics

- [Selecting the Type of Navigation View to Edit on page 259](#)
- [Editing Menu Items on page 258](#)

General Menu Item Properties

The properties described in the following are valid for all menu items: Other properties may be required for different menu item types.

Table 142: General Menu Item Properties

Property	Meaning
Menu Item	Unique menu item relation. You should try to use a meaningful name that can be passed on in the child structures. This makes it easier to trace the position of child items. The parent menu item and the hierarchy is determined by the insert position in the user interface navigation. The menu item name can contain variables in order to represent the menu items.
Entry type	Menu item entry type.
Caption	Language dependent caption to display the menu item in the user interface. The caption for data dependent menu items can contain fixed strings and variables. Display text for recursive data dependent menu items is inherited from the parent menu item. Translate the given text using the  button.
Sort order	If several menu items have the same parent menu item, the sort order specifies their position in the display order. If the configuration parameter "Re-sort submenu items by caption" is set for the parent menu item, the given sort order does not take effect.
Icon	Icon for displaying the menu item in the navigation. Recursively data dependent menu items inherit the parent icon if none is given.
Overlay icon definition	VB.Net expression for defining overlays for the icon. Is used to display the status in the Launchpad.
Condition	Specify the conditions under which the menu item is displayed in the navigation. The input must satisfy the "WHERE clause" database query syntax. Use may use variables to formulate your condition.
Configuration flag	Special functions are set for menu items with the configuration flag. For more information, see Table 143 on page 263.
Preprocessor condition	You can add preprocessor conditions to menu items. This means that a menu item is only available when the preprocessor condition is fulfilled.

Property	Meaning
	<p>NOTE: You can find an overview of existing preprocessor dependencies in the Designer in the category Database One Identity Manager Schema Preprocessor dependencies.</p>
Disabled by preprocessor	If a menu item is excluded through a preprocessor condition, this option is set by the Database Compiler.
Description	Spare text box for additional explanation.
Disabled	Specifies if the menu item is displayed in the user interface or not. Disabled menu items are never displayed in the user interface.
	<p>NOTE: This change is also permitted for menu items in the default user interface and is not overwritten on schema installation.</p>
Show under "My One Identity Manager"	This option marks the menu items to be displayed in the category My One Identity Manager in the Manager.

Table 143: Configuration Flags for Special Functions

Configuration flag	Description
Auto-reload on insert	If this configuration flag is set, the menu item is reloaded after new data is added.
Hide on empty result	If no submenu items are generated for a menu item labeled the same way during runtime, the menu item is similarly hidden in the user interface.
Not expandable by user	Menu items that are labeled with this option cannot be opened even when there are submenu items. The configuration flag is mainly used in the info system for displaying statistics.
Ignore user interface forms in result list	No forms are provided in the result list for menu items with this option. This can be useful to prevent navigating to objects in the list on an overview form. This is useful if, for example, forms are not defined for some objects in the result list. Otherwise, an empty form is displayed.
Ignore user interface forms	This configuration flag can be used for data dependent menu items. If the configuration flag is set, no object dependent interface forms are displayed when the menu item is selected in the user interface. This configuration flag is mainly used for structuring the user interface for Web front-ends.
Force open menu item	If this configuration flag is set, the menu item is always open. There is no test to see if the menu item is assigned to something, for example, the interface form.

Configuration flag	Description
Re-sort data dependent menu item by caption	This configuration flag can be used for data dependent menu items. The configuration flag should be set if language dependent data is displayed. If the configuration flag is set, the data for menu navigation to be shown is sorted by language after it is loaded.
Re-sort data result by caption	This configuration flag can be used for lists. The configuration flag should be set if language dependent data is displayed. If the configuration flag is set, the data to be shown is sorted by language in the result list after it is loaded.
Re-sort data submenu items by caption	The configuration flag should be set if language dependent data is displayed. If the configuration flag is set, the data for all submenu items to be shown are sorted by language. This enables all user accounts, groups and containers in a container structure, for example, to be sorted alphabetically. The sort order not only affects data dependent menu items but also all submenu items.

Related Topics

- [Navigation View Elements on page 252](#)
- [Editing Menu Items on page 258](#)
- [Creating Database Queries for Data Dependent Menu Items on page 264](#)
- [Editing Lists on page 267](#)
- [Using Links on page 270](#)
- [Working with Overview Forms on page 297](#)
- [Linking Statistics into the User Interface on page 310](#)
- [Extending the Launchpad on page 319](#)
- [Using Variables on page 271](#)
- [Icons and Images for Configuring the User Interface on page 327](#)
- [Conditional Compilation using Preprocessor Conditions on page 448](#)

Creating Database Queries for Data Dependent Menu Items

Data dependent menu items are generated by a database query that returns several data sets as output. These menu items are therefore not individual menu items, but a set of menu items depending on the output of the database query.

For more information about general properties of menu items, see [General Menu Item Properties on page 262](#). The following properties are necessary to put together a database query:

Table 144: Database Query Properties

Property	Meaning
Table	Table that the values are read from.
Sort by	Display elements are sorted by these table columns. The input must satisfy the "Order by" database query syntax. Sorting is given by the columns of the display template if no value is entered. You should use a sort order if the data has a date or represents language dependent data. i NOTE: Use the configuration flag "Re-sort submenu items by caption" to sort by language.
Condition	Condition for limiting the number of results displayed. The input must satisfy the "WHERE clause" database query syntax. You can use variables for formulating a condition. If the menu items are recursively data dependent then variables have to be used. i NOTE: The condition may not contain a JOIN, if necessary the query has to be formulated as a sub query.
Unique	The query result cannot contain doubled items. By setting option, any doubt is eliminated. i NOTE: No interface forms are shown for objects that result from a database query. i NOTE: This option is ineffective if the configuration parameter flag "Force open menu item" is set.
Recursive invocation	This menu item is the recursive successor of the previous menu item. If the option is not set, the results are represented by a flat structure. Set the option if the menu item is required to represent a hierarchical structure. You have to define recursive data dependent menu items below a data dependent item without recursion.

Related Topics

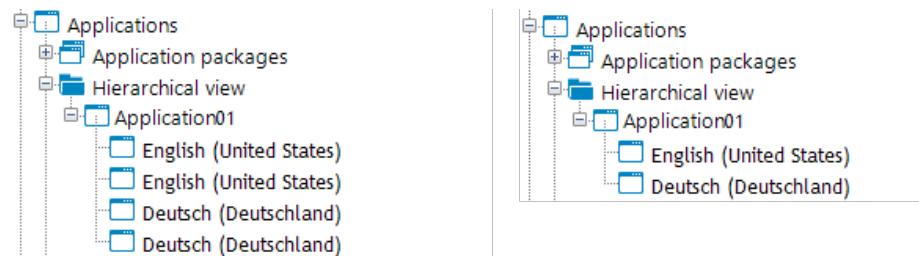
- [General Menu Item Properties on page 262](#)
- [Data Dependent Menu Item Uniqueness on page 266](#)
- [Recursively Data Dependent Menu Items on page 266](#)
- [Using Variables on page 271](#)

Data Dependent Menu Item Uniqueness

Menu items that are labeled with the option **unique** have to contain variables in their names to achieve uniqueness.

IF, for example, all applications (table Application) are grouped by language, the name of the corresponding menu item must contain a variable, which references the column Ident_Language in the table Application.

Figure 27: Example of Data Dependent Menu Item without Uniqueness (left) and with Uniqueness (right)



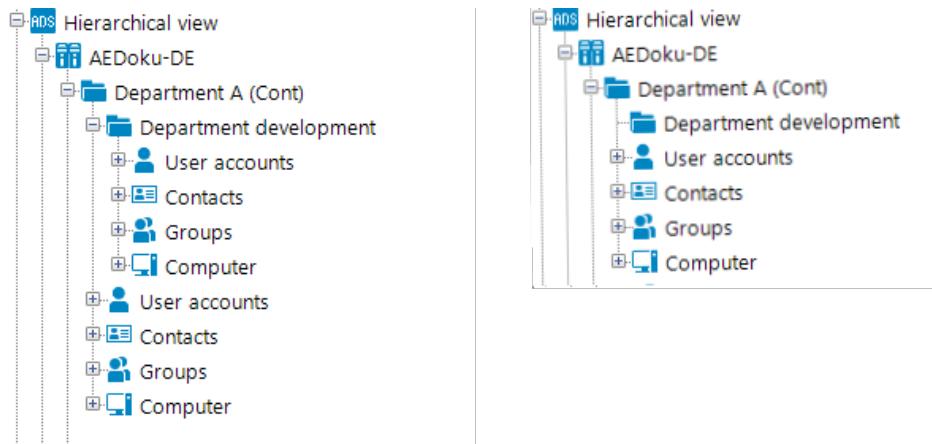
Related Topics

- [Creating Database Queries for Data Dependent Menu Items on page 264](#)
- [Using Variables on page 271](#)

Recursively Data Dependent Menu Items

The heart of the hierarchy is variable replacement. Variables are passed down through the hierarchical navigation view and can therefore be used at lower levels or can be overwritten. In the case of recursive data dependent menu items, the variable contained in a database query is initially replaced by the existing variable value from the parent level and then the query is started. The resulting value immediately determines a new value for the variable that is processed again in the parent node's next step. The original value of the old variable is no longer available after the database query has been executed. If the database query delivers an empty result, the recursion is stopped.

Figure 28: Example of Data Dependent Menu Items with Recursive Calling (left) and without Recursive Calling (right)



Related Topics

- [Creating Database Queries for Data Dependent Menu Items on page 264](#)
- [Using Variables on page 271](#)

Editing Lists

You can apply list properties to fixed and data dependent menu items. These properties determine how the table entries are displayed in the user interface result list.

For more information about general properties of menu items, see [General Menu Item Properties on page 262](#). To define a list, you need to use the following properties:

Table 145: List properties

Property	Meaning
Display template	The display template for displaying table entries in the administration tool result lists are displayed. If a customer specific display template exists it is used instead of the default display template. Syntax: %column name%
Object	Definition of the object which determines the list items.
Condition	Condition for limiting the number of results. The input must satisfy the "WHERE clause" database query syntax. The condition relates to the given object definition. The condition is consolidated with the condition which is already stored for the object definition. The variables can be used that are available in the navigation interface.
Icon	Icon for displaying the items in the list.

Property	Meaning
Order by	Columns to use for the list order. The input must satisfy the "Order by" database query syntax. You should use a sort order if the data has a date or represents language dependent data. Use the configuration flag "Re-sort result by caption" for a language dependent sort order.
Insert values	Insert values initialize individual values when a new data set is added over the result list. Enter insert values in VB.Net syntax. When defining insert values, you can apply the variables currently available in the navigation.
Insert in list permitted	Specifies whether inserting with the context menu is permitted in the corresponding result list. This option is set by taking 'can insert' permissions into account.
Permit deletion in list	Specifies whether deleting with the context menu is permitted in the corresponding result list. This option is set by taking 'can delete' permissions into account.

Related Topics

- [General Menu Item Properties on page 262](#)
- [Display Template for Displaying a List on page 268](#)
- [Defining Insert Values on page 269](#)
- [Using Variables on page 271](#)
- [Language Dependent Data Representation on page 328](#)
- [Object definitions for the User Interface on page 247](#)
- [Editing Permissions for One Identity Manager Schema Tables and Columns on page 231](#)

Display Template for Displaying a List

You use a list display template to specify the form in which the table entries will be represented in the administration tool result list. You can define display templates for menu items, object definitions and table lists.

The display template is determined by the following in order:

1. List display template for the menu item
2. Object definition display template
3. Table display template

The display template for displaying a list can be described in the following syntax:

%columnname%

All the columns that belong to the table that will be displayed can be used in the display template. Variables may not be used in display templates for lists.

Replacing the display templates supports the ?? operator. Thus you can formulate conditional display templates with the following syntax.

```
%columnname1??columnname2??columnname3%  
%columnname1 ?? columnname2%
```

The first column that returns a value from the list of column names is used. Spaces are permitted before and after the ?? operator but not at the begining and end of the display template on performance grounds.

Example of a Display Template

The Active Directory user account (table ADSAccount) should be shown as follows:

Common Name (fully qualified domain name)

The display template for the table ADSAccount to be specified for this purpose is:

```
%cn% (%CanonicalName%)
```

Related Topics

- [Editing Lists on page 267](#)
- [Defining Parameter Values on page 513](#)

Defining Insert Values

You can use insert values to initialize individual values when a new data set is added over the result list. You can apply insert values to interface forms, object definitions, menu item lists and tables.

Enter insert values in VB.Net syntax. The Base. syntax always addresses the object that is currently loaded. Insert values are described with the following syntax:

- Simple value assignment
`Base.PutValue("<column>", <value>")`
- Value assignment with variable replacement (value must be a character string)
`Base.PutValue("<column>", context.Replace(<value>))`

All the columns of the table to be displayed may be applied. You can use variable for defining insert values.

Example

```
Base.PutValue("IsApplicationGroup", 1)  
Base.PutValue("UID_ADSContainer", context.Replace("%cont%"))
```

 **NOTE:** If you changed insert values, you must recompile the database.

Related Topics

- [Using Variables on page 271](#)
- [Compiling a One Identity Manager Database on page 64](#)

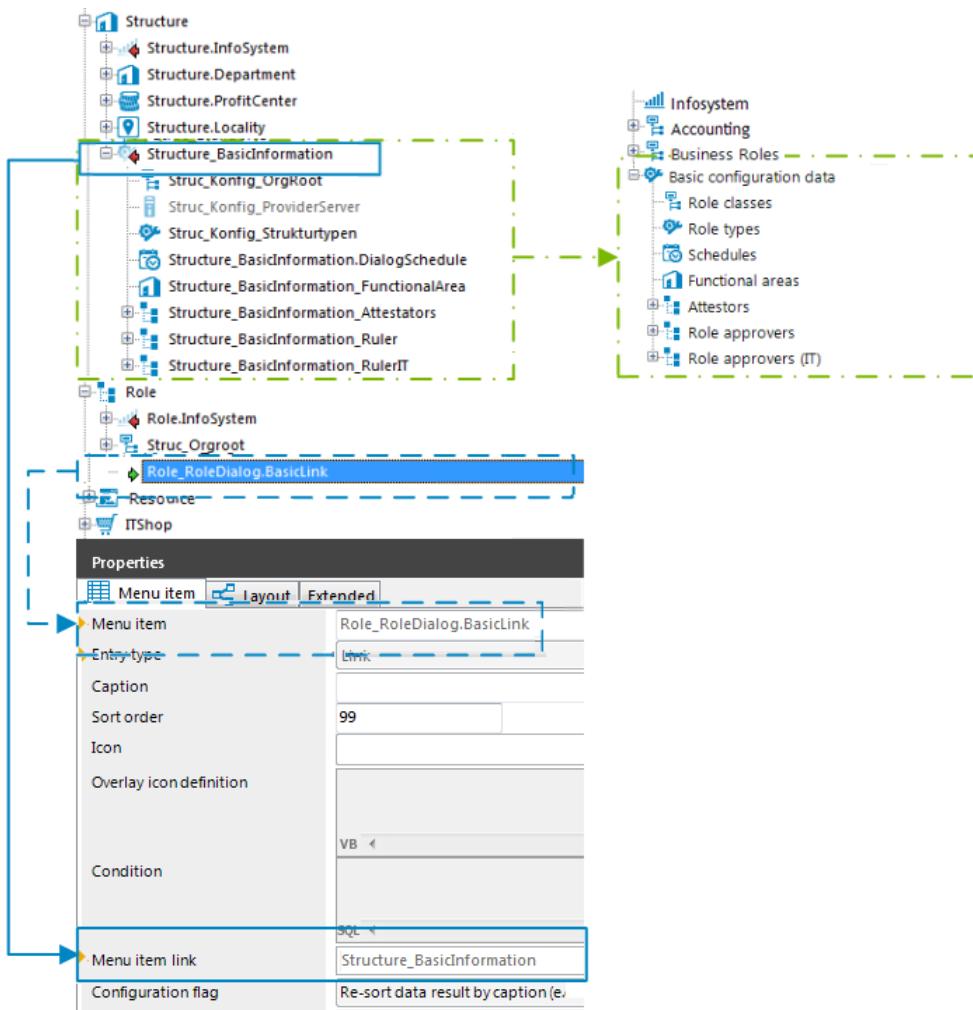
Using Links

Links support the navigation configuration. Links are implemented to reference frequently used menu items. Parts of the navigation interface that require an application several times, only need to be set up once. The referenced menu items are always shown in navigation interface as opposed to the links.

For more information about general properties of menu items, see [General Menu Item Properties on page 262](#).

You specify which menu item is shown at run-time with the property **Menu item link**. Certain link properties are passed on to the menu item. The display text and the icon for the referenced menu item are overwritten by the appropriate values. Even so, the variable definitions are passed on from the link to the referenced menu item.

Figure 29: Structure of the Navigation Interface using Links in the User Interface Editor (left) and How it Appears in the Manager (right).

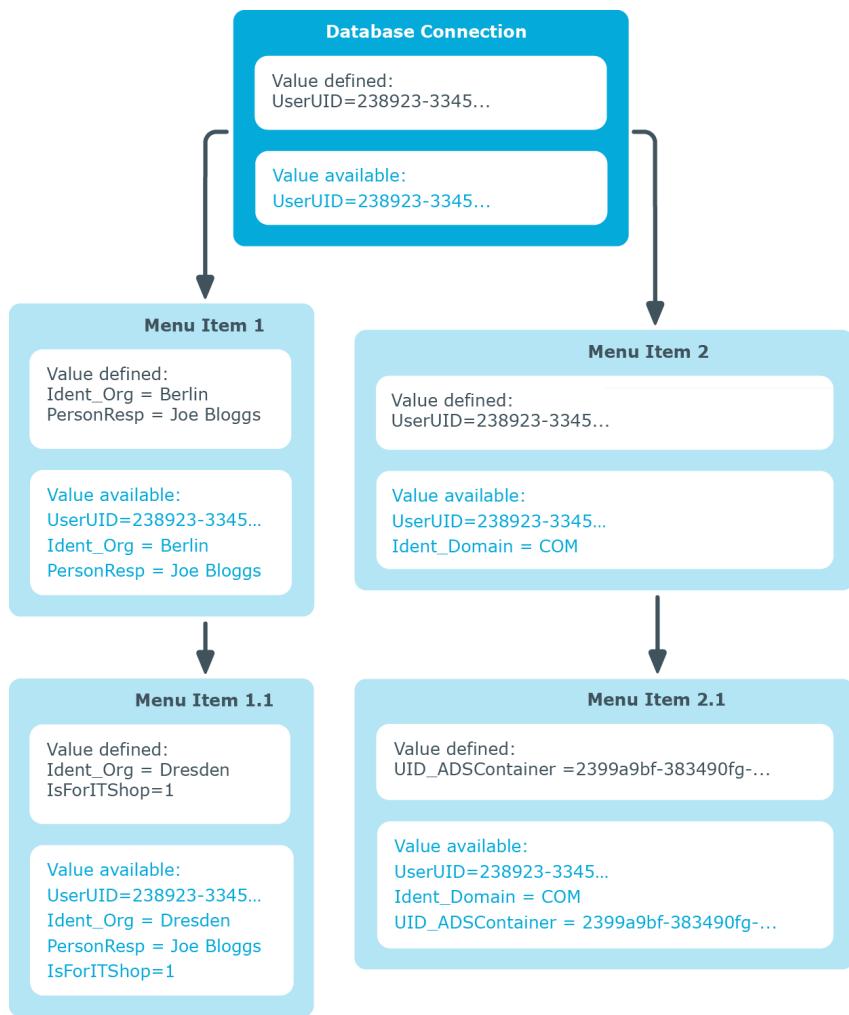


Using Variables

You may use variables to configure identifiers and menu item display templates for menu items in insert values and database queries. In some parts of the navigation interface you have to implement variables as, for example, in the case of formulating database queries for recursive data dependent menu items.

Variables are inherited within a hierarchical navigation. This means that variables in deeper levels of a hierarchy can be reused or overwritten. The actual run-time value is passed to the variable.

Figure 30: Inheriting Variables in a Hierarchical Navigation Interface



The session object variables described in the following, are always available when menu items are being set up.

Table 146: Global Session Object Variables

Variable	Meaning
EnvUserName	Name of user to be authenticated in the environment, for example, "Domain\User" in Active Directory.
LogonUser	DialogUser.Username of the currently logged in user.
DialogUserID	DialogUser.UID_DialogUser of the logged in user.
UserName	Name to be displayed in XUserInserted or XUserUpdated.
UserUID	Logged in user's UID_Person, if user related authentication is being used.
ShowCommonData	Specifies whether system data is shown (1) or not shown (0). The

Variable	Meaning
	variable is evaluated in the Designer by the program settings.
SessionType	Specifies whether a direct database connection or a connection over an application server is supported. Only direct database connection: '%SessionType%' = 'Direct' Only connection through application server: '%SessionType%' = 'AppServer'
	You can also define variables. The variable definition is made up of variable type, variable name and the value. Basically, any string is permitted in the variable definition. However, events have proved that it is a good idea to use a pattern that is unlikely to occur in the data but is accepted as a string by the database server in use.

Table 147: Variable Definitions

Variable Type	Variable name	Value	Usage
Column	Any string	Current object's column name	Only used in data dependent menu items.
Display value	Any string	Current object's column name	Only used in data dependent menu items. The column properties multilingual and List of permitted values are resolved when creating the display value for a column.
Text	Any string	Freely defined value	Can be used in all menu items.

Use the following syntax to access the variables:

%<variable>%

Related Topics

- [Editing and Displaying Variables on page 274](#)
- [General Menu Item Properties on page 262](#)
- [Creating Database Queries for Data Dependent Menu Items on page 264](#)
- [Editing Lists on page 267](#)
- [Using Links on page 270](#)

Editing and Displaying Variables

To edit variables

1. Select the category **User Interface | User interface navigation** in the Designer.
2. Start the User Interface Editor with **Edit navigation** in the task view.
3. Select the menu item in the user interface navigation view.
4. Select the view **Variable definition**.

In this view, all the variable definitions that belong to the selected menu item are displayed in tabular form with type, name and assigned value.

 **TIP:** To show variables inherited from parent menu items, click .

5. Edit the variables.
 - a. Create a new variable using the  button.
 - b. Delete a variable with .
 - c. Edit a variable with .
-  **TIP:** A list of available types is shown by clicking on the variable type in the table grid. The variable name and the value can be edited by clicking on the appropriate column in the table.

The actual value stored in the variable can be shown in the administration tools as additional navigation information.

To display variable values of a menu item in the Manager

- Set the option "Show additional navigation information" in the program settings in the Manager.
- Select menu item in the navigation in the Manager and select **Definition | Defined variables** from the context menu.

Related Topics

- [Using Variables on page 271](#)

Forms for the User Interface

User interface forms are used to display and edit data in the user interface. The basic information for representing data on the user interface forms is described in form definitions and form templates. The form definition referenced by the interface form needs to be found. The form template given in the form definition is checked for existence in the form archive and to see if it labeled for the correct display purposes.

Detailed information about this topic

- [Recommendations for Editing Forms on page 275](#)
- [Editing Interface Forms on page 279](#)
- [Forms for Custom Extensions on page 287](#)
- [Replacing Default Forms with Custom Forms on page 296](#)
- [Working with Overview Forms on page 297](#)

Recommendations for Editing Forms

- You can disable single predefined overview forms or single form elements of an overview form if necessary. This prevents them being displayed in the user interface. They remain disabled even after schema installation.
- The default installation of One Identity Manager already provides a series of form templates and definitions, for example for editing master data as well as many-to-many relations and object relations (Parent/ChildRelation). These can be used for easily creating your own forms.
- If necessary, you can provide your own form templates in a form archive (<MMM>.Forms.vif).
- To display information about a base object, you create an overview form.
 - You can do this using the Overview Form Editor in Designer.
 - Create menu items for object relations you need to display frequently, and use these menu items as reference in the form elements of the overview form.

TIP: You can have the Overview Form Editor create the menu items for object relations.

- Select the object relation you want to display and drag and drop it on an element in the element area of the Overview Form Editor.
- Use the context menu items "Create list element reference" or "Create reference to data element".

The menu items will be entered below the menu item InfoSheets.QIM.Links with the labels InfoSheet.List.<table> and InfoSheet.Node.<table>, respectively.

The condition for the menu items is defined as variable %<table>WhereClause%. In the form element you assign a condition as WHERE clause to the variable.

- To edit the base object master data, you create at least one interface form of form type "Edit".
- To define mappings, you create additional interface forms of form type "MemberRelation".
- Assign the forms and menu items to the Manager.

- Assign the forms and menu items to the permissions groups for non role-based and role-based login.

Related Topics

- [Editing Interface Forms on page 279](#)
- [Forms for Custom Extensions on page 287](#)
- [Replacing Default Forms with Custom Forms on page 296](#)
- [Working with Overview Forms on page 297](#)

Working with the Form Editor

Edit the user interface forms with the Form Editor. The editor is started from the program "Designer" and opens in the document view. Only additional Form Editor functions are described in the following.

Menu Items

The following items are added to the menu bar when the editor starts.

Table 148: Meaning of Items in the Menu Bar

Menu	Menu Item	Meaning	Key Combination
Form	Paste	Creates a new form.	
	Delete	The selected form is deleted after confirming the security alert.	
	Replace by...	Starts the wizard for replacing a form.	
	Refresh view	Updates the form list.	F5
Options	Show captions	Show the interface form display text. If the selection is not enabled, the form names are shown.	
	Tree/list view	Toggles between list and hierarchical form view	
	Select columns...	Opens a dialog window for selecting columns that can be also be displayed in the form list.	
Filter	Define	Opens a dialog window for creating an ad hoc	

Menu	Menu Item	Meaning	Key Combination
	filter...	filter.	
	Delete filter	Deletes the filter.	
	Manage filters...	Opens a dialog window for creating permanent filters.	
View	Properties	Shows/hides the edit view.	
	Objects	Show/hides the object definition view.	
	Permissions group	Shows/hides the permissions group edit view.	
	Menu Items	Shows/hides the menu assignment view	
	Related Applications	Shows/hides the application edit view.	
	Form preview	Shows/hides the form preview.	
Help	Form edit help	Opens the help on this topic.	
	Form Editor help	Opens the editor help.	

Table 149: Meaning of Toolbar Icons

Icon	Meaning
	Adds a new form.
	Deletes a form.
	Toggles between captions and form names.
	Toggles between list and hierarchical form view.
	Refreshes the display.
	Starts the WHERE clause wizard for defining custom filters. After entering the data, the form is displayed with respect to the filter condition.
	Resets custom filter.

Views in the Form Editor

There are several views available for editing forms in the Form Editor:

Table 150: Form Editor Views

View	Description
Form overview	This view displays the interface forms. You can create or delete interface forms in this view. Forms can be displayed hierarchically or in a list. The interface forms are grouped by form template and form definition in the hierarchical representation. ① NOTE: Forms that are disabled by preprocessor conditions are grayed out in the form overview.
Objects	All available object definition are displayed. You can assign interface forms to object definitions or delete existing assignments.
Menu Items	All available menu items are displayed. You can assign menu items or remove existing assignments. When the associated menu item is selected in the navigation or the item is selected in the result list, the interface forms are shown for all system users without taking their permissions group memberships into account.
Permissions group	Displays all available permissions groups. You can assign the interface forms to the permissions groups or delete existing assignments. These interface forms are available to system users depending on their permissions group memberships.
Related Applications	All available applications are displayed. You can assign interface forms to the applications or remove existing assignments.
Properties	Edit the properties for the selected interface form in the edit view and also the form templates and definitions. A default context menu is available for input fields.
Form preview	The form preview shows the contents of the interface form. You can see which base tables will be used to display the data. The permissions of the logged in Designer user are taken into account when loading and displaying an interface form. If a form cannot be loaded, an appropriate error message is displayed.

Table 151: Form Overview Context Menu Items

Context Menu Item	Meaning
Paste	Inserts a new interface form.
Delete	Deletes the selected interface form.
Replace by...	Starts the wizard for replacing a form.
Properties	Displays the object properties of the selected entry.
Select columns...	Opens a dialog window for selecting columns that can be additionally displayed in the user interface navigation.

Editing Interface Forms

User interface forms are connected to object definitions, so that different forms are offered in the user interface depending on which object is selected. These interface forms are made available to system users, taking into account their permissions group memberships, by the additional assignment of interface form to permissions groups. Further more, interface forms can be defined for separate menu items. When the associated menu item is selected in the navigation or the item is selected in the result list, the interface forms are shown for all system users without taking their permissions group memberships into account.

Predefined configurations are maintained by the schema installation and cannot be edited apart from a few properties.

- NOTE:** You can disable individual predefined tasks to prevent them being shown in the user interface. They remain disabled even after schema installation.

To disable a user interface form

1. Select **User Interface | Forms | User interface forms** in the Designer.
2. Select **Edit form** in the task view.
3. Select the user interface form in the Form Editor.
4. Select the tab **User interface form** in the **Properties** view and set the option **Disabled**.

To copy a user interface form

- NOTE:** Use the copy, if you only want to make minimal changes to it, for example, another caption or sort order.

1. Select **User Interface | Forms | User interface forms** in the Designer.
2. Select **Edit form** in the task view.
3. Select the user interface form you want to copy in the Form Editor.
4. Select **Form | Insert** from the menu.
This creates a copy of the selected user interface form.
5. Edit the other user interface form master data.
6. Assign programs, permissions groups, objects and menu items to the user interface form.

- NOTE:** Disable the original user interface form. Otherwise both forms are displayed in the user interface.

To create a new overview form

- NOTE:** Create a new user interface form, for example, if you want to display custom schema extensions in the user interface. One Identity Manager provides a set of form templates and definitions in the default installation. These can be used for easily creating your own forms.

1. Next, create a copy of a existing user interface form.
2. Select the tab **User interface form** in the **Properties** view.
3. Create a new form definition using  next to the field **Form definition**.
4. Enter the form definition name on the **Form definition** tab and select a form template.
5. Edit the other user interface form master data.
6. Assign programs, permissions groups, objects and menu items to the user interface form.

To display a user interface form in the user interface

1. Select **User Interface | Forms | User interface forms** in the Designer.
2. Select **Edit form** in the task view.
3. Select the user interface form in the Form Editor.
4. In order for an interface form in the user interface to show an application you need to assign an application to the form. Use the **Program** view to do this.
5. Then you must assign the form to a permissions group. Use the **Permissions groups** view to do this. Read Effects of Object Definitions when Displaying Interface Forms about controlling the display of forms through object definitions.
6. The form also needs to be linked to a permissions group. Use the **Object relations** view to do this.
7. Then the form is available to Quests that can see this menu item in their navigation view. Use the **Menu assignments** view to do this. Use the to assign a menu item to the interface form.

Related Topics

- [User Interface Form Properties on page 281](#)
- [Form Definitions and Form Templates on page 282](#)
- [Forms for Custom Extensions on page 287](#)
- [Adding in Custom Extensions to the User Interface on page 542](#)
- [Features of the Assignment Form on page 287](#)
- [Effects of Object Definitions when Displaying Interface Forms on page 286](#)
- [Editing Menu Items on page 258](#)
- [Editing Permissions Groups and System Users on page 219](#)
- [Applications for Configuring the User Interface on page 325](#)
- [Object definitions for the User Interface on page 247](#)

User Interface Form Properties

Table 152: User Interface Form Properties

Property	Meaning
Form name	The form name is used to quickly select interface forms, for example, in the Designer. ⓘ TIP: The form name is displayed in the administration tool as extra navigation information.
Form definition	Form definition linked to the user interface form. ⓘ NOTE: Use  next to the input field to link in a new form definition for the interface form.
Caption	Caption shown on the user interface form. The caption is used to represent the interface form in the task view and in the form's context menu in the user interface. Translate the given text using the  button.
Online help link	The form's help key for navigating to the relevant chapter in the online help.
Description	Detailed description of the user interface form. ⓘ TIP: The description is shown as a tooltip in the task view.
Icon	Icon marks the user interface form in the user interface.
Sort order	The sort order determines the position of the interface form in the task view and in the form's context menu in the administration tools. ⓘ NOTE: The interface form of type "Edit" with the lowest sort order is always shown when objects are being added in the Manager.
Preprocessor condition	User interface forms can be given a preprocessor condition. This means that an interface form is only available when the preprocessor condition is fulfilled. ⓘ NOTE: You can find an overview of existing preprocessor dependencies in the Designer in the category Database One Identity Manager Schema Preprocessor dependencies .
Control deactivation	Specifies which button to gray out in the toolbar.

Value	Meaning
New object	The  is grayed out.

Property	Meaning		
	<th>Value</th> <th>Meaning</th>	Value	Meaning
	<td>Reload object</td> <td>The is grayed out.</td>	Reload object	The is grayed out.
	<td>Delete object</td> <td>The is grayed out.</td>	Delete object	The is grayed out.
	<td>Save changes</td> <td>The is grayed out.</td>	Save changes	The is grayed out.
Disabled	Use this option to label interface forms that should not be shown in the user interface. i NOTE: This change is also permitted for user interface forms in the default user interface and is not overwritten on schema installation.		
Disabled by preprocessor	If an object definition is excluded through a preprocessor condition, this option is set by the Database Compiler.		
Show modal	Specifies whether the form is displayed in a separate dialog box. Used by wizards for entering data.		
Open on new tab	The form is opened on a new tab.		
Configuration	<p>The configuration is used to limit the tables and columns on display. Templates for the configuration data definition are found in the pop-up list XML templates.</p> <p>Special properties that have been used in the design of the form, can be passed to the interface form in "SpecialSheetData". For example, the report name and special report parameters can be passed to the report interface form by using this section.</p> <p>Special form properties that have been used in the design of the form are passed on in the section "Properties".</p>		
Insert values	Insert values are only relevant for interface forms of type "Edit". With them you can specify the default values for the columns that are assigned when a new object is added. The input is in VB.Net syntax.		

Related Topics

- [Form Definitions on page 285](#)
- [Defining Insert Values on page 269](#)
- [Conditional Compilation using Preprocessor Conditions on page 448](#)

Form Definitions and Form Templates

Form definitions and form templates make up the basis of interface form design. Form definitions contain information about the data that will appear in the forms, for example,

tables and columns as well as titles for form tabs and root nodes in hierarchically ordered elements (ChildRelationControl, membership tree) for the form templates defined in the form archives (Forms.*.vif).

Detailed information about this topic

- [Form Templates on page 283](#)
- [Form Definitions on page 285](#)
- [Forms for Custom Extensions on page 287](#)

Form Templates

All form templates are found in the category **User Interface | Forms | Form templates** in the Designer. It is not usually necessary to define your own form templates.

To display a form template for a user interface form

1. Select **User Interface | Forms | User interface forms** in the Designer.
2. Select the user interface form in the Form Editor.
3. Select the tab **Form template** in the **Properties** view.

Table 153: Form Template Properties

Property	Meaning
Form source type	Source of the form template. Permitted values are: <ul style="list-style-type: none">• Form For displaying a form from a form archive.• Assembly For displaying controls. It is not necessary to build a form, because the control is displayed directly as form.
Assembly name	Name of the assembly file.
Class	Full type name of the control.
Form template name	The form template name is necessary for loading the form template from the form archive. TIP: The form template name is shown in the administration tools as additional navigation information.
Form archive	Name of the form archive (Forms.*.vif), containing the form template.
Description	Detailed description of the form template.

Property	Meaning
Alternative form template	<p>It might be necessary to use different form templates to display the interface form, for example, to show an the One Identity Manager web interface or in an administration tool.</p> <p>The form templates can be linked in order to avoid adding a form definition and an interface form for each form template. For this, you need to assign an alternative form template to the form template. This alternative form template is used when the conditions for displaying the original template are not fulfilled. The form template referenced is determined in order to display the interface form. The form template given in the form definition is checked for existence in the form archive and to see if it labeled for the correct display purposes. If these conditions are not fulfilled then the alternative form template is tested for suitability. The form template that fulfills the conditions is used for the user interface display.</p>
Form type	Type of form.
Enabled for	<p>This property specifies the intended use of the form template. Permitted values are:</p> <ul style="list-style-type: none"> • Visible in graphical interface • Visible in web application • TimeTrace supported • Multiobject editing possible • Deferred operation possible • Application server not supported

Table 154: Form Types and their Usage

Form type	Usage
Info (I)	Forms of type "Info" are only used to display information. Changes to data on these forms cannot be saved. These forms automatically omitted by the automatic form selection in quick edit mode.
Edit (E)	Forms of type "Edit" are used for editing data. This is the first form to be loaded by the automatic form selection in quick edit mode.
Grid (G)	Forms of type "Grid" are used to show data in table form.
MemberRelation (M)	Forms of type "MemberRelation" are used to represent data in an assignment list (many-to-many relation).
Report (R)	Forms of type "Report" are used to display data in the form of a report.
Virtual (V)	Forms of type "Virtual" are not shown in the form's menu. This form type is used to show editors in the Designer.
Wizard (W)	Forms of type "Wizard" are used to enter data using a wizard. The forms are displayed in a modal dialog window.

Related Topics

- [Editing Interface Forms on page 279](#)
- [Form Definitions on page 285](#)

Form Definitions

Form definitions are found in the category **User Interface | Forms | Form definitions** in the Designer. It is not normally necessary to define your own form definitions.

To display a form template for a user interface form

1. Select **User Interface | Forms | User interface forms** in the Designer.
2. Select the user interface form in the Form Editor.
3. Select the tab **Form definition** in the **Properties** view.

Table 155: Form Definition Properties

Property	Meaning
Form definition name	Name of the form definition. This name is used for displaying the form definition in the Designer.
Form template	Name of the form template to load from the form archive. A form template can be used by several form definitions, such as the form templates for displaying membership trees or the form template for displaying reports. Use the  button next to the input field to integrate a new form template in the form definition.
Base form for form sequence	By entering a form definition as a base for a sequence of forms, you can create a group of form definitions for one object definition. All form sequence form definitions contain the same base form. The definition of the interface form can only be made for this base form. When the interface form is loaded in the display, the referenced form definitions for all other form definitions in the form sequence are also loaded. You can navigate arbitrarily within the form sequence without leaving the scope of the interface form.
Description	Detailed description of the form.
Configuration	The configuration data is used to describe the form properties. The form property's definition is written in XML notation.
Required tables	A form definition can be assigned additional tables that are used to display data.  NOTE: If one of the given tables is disabled by a preprocessor condition then the form definition is also considered to be disabled and the corresponding interface form is not shown in the user interface.

Related Topics

- [Editing Interface Forms on page 279](#)
- [Form Templates on page 283](#)
- [Configuration Data for Displaying Many-to-Many and Object Relations on Forms on page 290](#)

Effects of Object Definitions when Displaying Interface Forms

Interface forms that need to be valid for all entries in a database table are allocated a general object definition. Other limited object definitions can have more interface forms. If an entry is selected in the user interface, the currently valid object definitions are used to gather all the interface forms and display them in the user interface in their sort order in the task view and in the context menu.

Example

The following object definitions with interface forms are set up for the table Person.

Table 156: Example: Interface Forms for Object Definitions

Object definition	Assigned Interface Form
Employee	Business roles
Person_with_ADUserAccount	Active Directory User Accounts
Person_with_LDAPAccount	LDAP User Accounts

The following interface forms are displayed for a "Person" object that fulfills the "Person_with_ADUserAccount" definition:

Business roles

Active Directory user accounts

A "Person" object that satisfies the object definition "Person_with_LDAPUserAccount" is given the interface forms:

Business roles

LDAP User Accounts

Related Topics

- [Object definitions for the User Interface on page 247](#)

Features of the Assignment Form

Forms of type "MemberRelation" are implemented to display data in an assignment list (many-to-many relations). Enter the hierarchy path in the table definition to display the table hierarchically. Enter the foreign key column that the hierarchy should be based on.

Example

An Active Directory user account (table ADSAccount) is typically displayed on an assignment form below its Active Directory container (UID_ADSContainer). The Active Directory container (table ADSContainer) is, on the other hand, displayed underneath its Active Directory domain (column UID_ADSDomain). The path for the hierarchy structure is entered as follows:

Table 157: Example of a Hierarchy Path

Table	Hierarchy path
ADSAccount	UID_ADSContainer, UID_ADSDomain
ADSContainer	UID_ADSDomain

Related Topics

- [Table Properties on page 173](#)

Forms for Custom Extensions

One Identity Manager provides a set of form templates and definitions in the default installation. These can be used for easily creating your own forms.

An other way to create custom forms is to make custom form archives available. Normally, default forms in One Identity Manager are replaced with self developed forms.

Table 158: Form Templates and Definitions for Custom Extensions

Form Template	Form Definition	Usage
FrmCommonChildRelationGrid	VI_Common_ChildRelation_Grid	For editing many-to-many relations with extended properties in the form of a table.
FrmCommonOneChild	A custom form	Assigns many-to-many relations

Form Template	Form Definition	Usage
AndMemberRelation FrmCommonOneMember AndChildRelation	must be created on which the data to be configured is displayed.	and object relations (parent/child relations) on one form. Two tabs for displaying the data are shown on the form.
FrmCommonOneChildRelation	A custom form must be created on which the data to be configured is displayed.	Mapping object relations (Parent/ChildRelation). If several objects relations are represented on a form, the form templates "FrmCommonTwoChildRelation", "FrmCommonThreeChildRelation" can be used instead. One tab is shown per object relation.
FrmCommonOneDynamicRelation	A custom form must be created on which the data to be configured is displayed.	Displays dynamic many-to-many relations whose assigned object is referenced through a dynamic Permitted dynamic objects are found in the table DialogValidDynamicRef. A menu is provided for choosing the object type.
FrmCommonOneGenericRelation	A custom form must be created on which the data to be configured is displayed.	Displaying dynamic many-to-many relations. <ul style="list-style-type: none"> • Base object can be referenced through a dynamic key - OR - • Assigned object is referenced through a dynamic key. In this case, the property "MembersTableName" is defined in the form configuration.
FrmCommonOneMemberRelation	A custom form must be created on which the data to be configured is displayed.	Assigning many-to-many relations. If several many-to-many relations are represented on a form, the form templates "FrmCommonTwoMemberRelation", "FrmCommonFourMemberRelation",

Form Template	Form Definition	Usage
		"FrmCommonFiveMemberRelation" can be used instead. On tab is shown per many-to-many table.
FrmElementNavigation	VI_ElementNavigation	For displaying the overview form.
frmGeneric	VI_Generic_MasterData	For editing object master data.
ReportForm	VI_Report	For displaying reports.
WizardForm	VI_Wizard	For including wizards. The forms are displayed in a modal dialog window.

Detailed information about this topic

- [Editing Interface Forms on page 279](#)
- [Form Definitions and Form Templates on page 282](#)
- [Custom Master Data Forms on page 289](#)
- [Configuration Data for Displaying Many-to-Many and Object Relations on Forms on page 290](#)
- [Replacing Default Forms with Custom Forms on page 296](#)
- [Features of the Assignment Form on page 287](#)
- [Creating Overview Forms on page 300](#)
- [Linking Reports into the User Interfaces on page 524](#)

Custom Master Data Forms

Displaying Columns in Custom Tables

To display custom database table in the administration tool user interfaces and edit the master data:

- Create a user interface form with the form definition "VI_Generic_MasterData". This form definition allocates the control element for editing master data in the user interface.
- Change the order of the input fields on the form using the sort order of the database columns. Columns with a sort order of less than one are not displayed.
- Achieve a better overview of the input fields by grouping database columns. Each group has its own tab. The name of the tag corresponds to the group.

- Columns whose data contents can be multiline are displayed in a multiline field on the generic form. Label these columns as **multiline**.

Displaying Custom Columns in Predefined Tables

Separate tabs can be shown for custom column extensions to default tables on the predefined master data forms. The points listed above apply to predefined master data forms using the form definition "VI_Generic_MasterData".

Otherwise the following prerequisites are required for using this functionality:

- Master data form already has tabs. Simple master data forms without tabs are not extended.
- Change the order of the input fields on the form using the sort order of the database columns. Columns with a sort order of less than one are not displayed.
- Database columns are grouped. Each group has its own tab. The name of the tag corresponds to the group. If no group is given, a tab is shown with the name "Custom".

Related Topics

- [Forms for Custom Extensions on page 287](#)
- [Editing Interface Forms on page 279](#)
- [Column Properties on page 189](#)

Configuration Data for Displaying Many-to-Many and Object Relations on Forms

Form properties are specified by the form definition configuration data. The definition of form properties is written in XML notation.

Example of the Configuration Data Structure

```
<DialogFormDefinition FormatVersion="1.0">
  <ComponentDefinitions>
    <ComponentDefinition Name="TabPage1" Type="VI.Components.TabPage">
      <Properties>
        <Property Name="Caption" Value="Department"/>
        <Property Name="CaptionTranslationSource" Value="DatabaseSchema" />
      </Properties>
    </ComponentDefinition>
    ...
  </ComponentDefinitions>
</DialogFormDefinition>
```

```

<ComponentDefinition Name="MemberRelation1"
Type="VI.Components.MemberRelation">
    <Properties>
        <Property Name="DisplayPattern" Value="" />
        <Property Name="MNBaseColumnName" Value="UID_ADSGroup"
IsMandatory="True" />
        <Property Name="MNTTableName" Value="DepartmentHasADSGroup"
IsMandatory="True" />
        ...
    </Properties>
</ComponentDefinition>
...
</ComponentDefinitions>
</DialogFormDefinition>

```

Displaying Relations

Table 159: Properties of Relation Definitions

Component	Property	Meaning
All		Valid for all maps.
	WhereClause	<p>Limited condition for applying to the displayed objects (member, child).</p> <p>The expression %column% can be used in the WhereClause to reference values of the base object.</p> <p>\$ expressions are permitted to reach other values from the base object, for example \$FK(UID_ADSContainer).UID_ADSDomain\$.</p> <p>Example</p> <pre><Property Name="WhereClause" Value="IsITShopOnly=0" /></pre>
	DisplayPattern	<p>Display pattern for finding the display value of the element. Default is the table display template.</p> <p>Example</p> <pre><Property Name="DisplayPattern" Value="%cn% %info%" /></pre>

Component	Property	Meaning
	DisplayFlatPattern	<p>Special display pattern used for displaying a form's lists with a flat design. Flat displaying is used if there is no hierarchy or the list limit is used. Default is DisplayPattern.</p> <p>Example</p> <pre><Property Name="DisplayFlatPattern" Value="%cn% %info%" /></pre>
	RootNodeCaption	<p>Root node caption. The default is taken from the schema.</p>
	EditWhereClause	<p>Edit condition. The elements that match the condition can be edited. All other elements are also displayed but cannot be edited.</p> <p>Example</p> <pre><Property Name="EditWhereClause" Value="XMarkedForDeletion=0" /></pre>
Member- Relation1- MemberRelationN		<p>Displaying M:N relations</p> <p>Example</p> <pre><ComponentDefinition Name="MemberRelation1" Type="VI.Components.MemberRelatio n"></pre>
	MNTableName	<p>M:N table.</p> <p>Example</p> <pre><Property Name="MNTableName" Value="OrgHasADSGroup" /></pre>
	MNBaseColumnName	<p>Column of the M:N table that points to the base object.</p> <p>Example</p> <pre><Property Name="MNBaseColumnName" Value="UID_ADSGroup" /></pre>
	RootFilterTableName	<p>Table for filtering assignable elements from users. If defined, the control element shows a menu with objects from this table. If, for example,</p> <p>Example</p>

Component	Property	Meaning
		<pre><Property Name="RootFilterTableName" Type="String" Value="OrgRoot" /></pre>
	RootFilterWhereClause	<p>Condition for filtering elements of the RootFilterTableName in the menu.</p> <p>Example</p> <pre><Property Name="RootFilterWhereClause" Type="String" Value="UID_OrgRoot in (select UID_OrgRoot from Org) and exists (select 1 from OrgRootAssign where IsDirectAssignmentAllowed = 1 and UID_OrgRoot=OrgRoot.UID_OrgRoot and UID_BaseTreeAssign='ADS-AsgnBT- ADSGroup')" /></pre>
ChildRelation1- ChildRelationN	RootFil- terMemberWhereClause	<p>Condition formatted after selecting a base object and attached to the WhereClause. The condition must always contains a column relation to the base object.</p> <p>Example</p> <pre><Property Name="RootFilterMemberWhereClause" Type="String" Value="UID_ OrgRoot=N'%UID_OrgRoot%'" /></pre>
	CRTTableName	<p>Displaying parent-child relations.</p> <p>Example</p> <pre><ComponentDefinition Name="ChildRelation1" Type="VI.Components.MemberRelatio n"></pre>
	CRColumnName	<p>Table in which child objects are mapped.</p> <p>Example</p> <pre><Property Name="CRTTableName" Value="ADSAccount" /></pre>
		<p>Child table foreign key that points to the base object.</p> <p>Example</p>

Component	Property	Meaning
	<Property Name="CRCColumnName" Value="UID_Person" />	
	ShowForeign	Specifies whether foreign assignments (object assigned to another object) can be displayed.
		Example <Property Name="ShowForeign" Value="True" />
GenericRelation1- GenericN		Displaying dynamic many-to-many relations.
		Example <ComponentDefinition Name="GenericRelation1" Type="VI.Components.MemberRelatio n">
	MNTableName	M:N table.
		Example <Property Name="MNTableName" Value="ADSPolicyAppliesTo"/>
	MNBaseColumnName	Column of the M:N table that points to the base object.
		Example <Property Name="MNBaseColumnName" Value="ObjectKeyAppliesTo" />
	MNMembersColumnName	Column of the M:N table that points to the members.
		Example <Property Name="MNMembersColumnName" Value="UID_ADSPolicy" />
	MembersTableName	Tables whose objects must be assigned.
		Example <Property Name="MembersTableName" Value="ADSPolicy"/>

Using Tabs

Use the components TabPage to display tabs for the mapped relations. Usually, tabs are used for forms with several relations to display, for example,

"FrmCommonTwoMemberRelation" or "FrmCommonTwoChildRelation".. TabPage1 maps the tab for Relation1, TabPage2 maps the tab for Relation2.

Table 160: Properties of Tab Definitions

Component	Property	Meaning
TabPage1-TabPageN		<p>Displays 1-n tab for each relation to be shown.</p> <p>Example</p> <pre><ComponentDefinition Name="TabPage1" Type="VI.Components.TabPage"></pre>
Caption		<p>Tab captions. Table names or any string can be used as captions.</p> <p>Example</p> <pre><Property Name="Caption" Value="Department"/></pre>
CaptionTranslationSource		<p>Source for translating the tab names.</p> <p>Value="DatabaseSchema" finds the table captions translation from the One Identity Manager schema table given under Caption.</p> <p>Value="TranslationAddOnSource" finds the translation from the text store.</p> <p>Example</p> <pre><Properties> <Property Name="Caption" Value="Department"/> <Property Name="CaptionTranslationSource" Value="DatabaseSchema" /> </Properties> <Properties> <Property Name="Caption" Value="is member of"/> <Property Name="CaptionTranslationSource" Value="TranslationAddOnSource" /> </Properties></pre>

Related Topics

- [Forms for Custom Extensions on page 287](#)
- [Editing Interface Forms on page 279](#)

Replacing Default Forms with Custom Forms

Self developed form templates can be provided for custom forms in a form archive (*.CustomForms.*.vif). You need to add the form template, form definition and interface form with help of the Form Editor if you want to display your custom forms in the user interface.

A wizard is available to swap a default form with all its dependencies for a custom form. The wizard creates the interface form with the form definition and the form template. The properties of the new form are taken from the form it is replacing. The necessary assignments (object definition, menu item, permissions group and application) are created for the new form and the replaced form is disabled.

To replace custom forms with all dependencies

1. Select **User Interface | Forms | User interface forms** in the Designer.
2. Select hierarchical representation of the form overview. Set the menu option **Options | Tree/list view** to do this.
3. Select the form template of the form to replace in the top layer of the hierarchy in the form overview and start the wizard from the context menu **Replace by....**
4. Click **Next**.
5. Select the form archive (*.CustomForms.*.vif) and the form template for the new interface form.
6. Click **Next**.
7. Check the names of the form definition and the user interface form.
The names that are suggested are made up from the customer prefix and the name of the form being replaced. Use **F2** to change the name and the enter button to accept the changes.
8. Click **Next**.
9. Select the permissions group to which to assign the new interface form. Use the **+** button to create a new permissions group.
10. Click **Next**.
11. The settings for form replacement are summarized. To replace the form, click **Finish**.

The wizard is closed after replacement is complete. The new form is displayed in the Form Editor form overview after the wizard is complete and you can continue editing it. The replaced form is disabled and can therefore no longer available in the user interface.

Related Topics

- [Editing Interface Forms on page 279](#)
- [Forms for Custom Extensions on page 287](#)

Working with Overview Forms

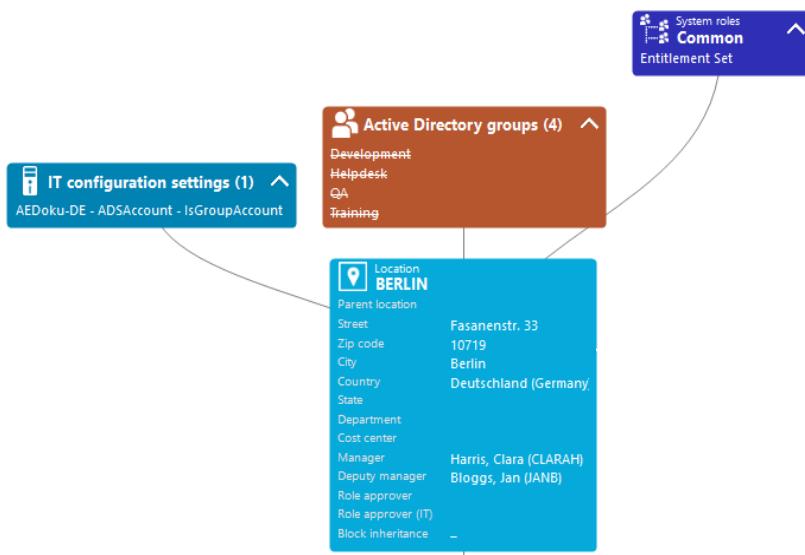
There is a special control element for displaying the overview form in the user interface. The information to be displayed on the overview form is configured with menu items. The menu items are represented as form elements that are linked to each other on the overview form. A hierarchical structure of menu items is also included in the interface configuration.

The base element is a menu item with the input type "main form element". This menu item specifies the main element on the overview form. An interface form that links to this menu item has to be configured in order for it to be displayed in the application. The main form element is always displayed in the middle of the overview form.

The other menu item such as fixed, data dependent, link or statistic menu items are configured under the menu item for the main form element. These menu items are grouped around the main form element on the overview form as additional form elements.

The color and positioning of the form elements on the overview as well as the properties that are shown, are specified by layout information for the menu items.

Figure 31: Example of Elements in an Overview Form



The display text of the menu item, the display text for the objects to be shown and the menu item icon are displayed in the header of a form element. Other data represents the object properties and values. There is a tooltip for each property showing a description for use. Some form element entries are highlighted in color when you click on them with the mouse. You can jump to the referenced object by clicking on the entry with the mouse.

If the form element is used for mapping lists, the items are displayed with their names. The number of items is shown in the form element header. There is an icon in the header for showing and hiding the items. There is an icon in the header for showing and hiding the items. There is no tooltip for list items.

Table 161: Form Element Icon

Icon	Meaning
▼	Show list items.
▲	Hide list items.
●	NOTE: Objects marker for deletion are struck through on the overview form.

Detailed information about this topic

- [Creating Overview Forms on page 300](#)
- [Designing an Overview Form on page 303](#)
- [Disabling Overview Forms on page 305](#)

Working with the Overview Form Editor

Edit overview forms with the Overview Form Editor. The editor is started from the program "Designer" and opens in the document view. Only additional Overview Form Editor functions are described in the following.

Menu Items

The following items are added to the menu bar when the editor starts.

Table 162: Meaning of Items in the Menu Bar

Menu	Menu Item	Meaning	Key Combination
Elements	Create new overview form	Creates a new overview form.	
	Create element	Creates a new form element.	
	Delete element	Deletes the form element,	
	Refresh view	Updates the design view.	F5
	Refresh view automatically	Updates the design view automatically.	

Menu	Menu Item	Meaning	Key Combination
View	Properties	Shows/hides the edit view.	
	Form definition	Displays properties of the main form element's interface form.	
	Variables	Show/hides the variable definition view.	
	Object assignments	Shows/hides the object definition view.	
	Group assignments	Shows/hides the permissions group edit view.	
	Menu assignments	Shows/hides the menu assignment view	
	Product assignments	Shows/hides the application edit view.	
	Object relations	Shows/hides objects permissions view.	
Help	Form Editor help	Opens the editor help.	

Table 163: Meaning of Toolbar Icons

Icon	Meaning
	Creates a new overview form.
	Adds a new form element.
	Deletes a form element.
	Refresh the display.

Views in the Overview Form Editor

There are the several views available for editing overview forms in the Overview Form Editor:

Table 164: Form Editor Views

View	Description
Design view	This view is the design view of the overview form. Element regions are displayed for linking the form elements. The main form element is centered in the overview.
Properties	Use the view edit the properties of the selected menu item. A default context menu is available for input fields.

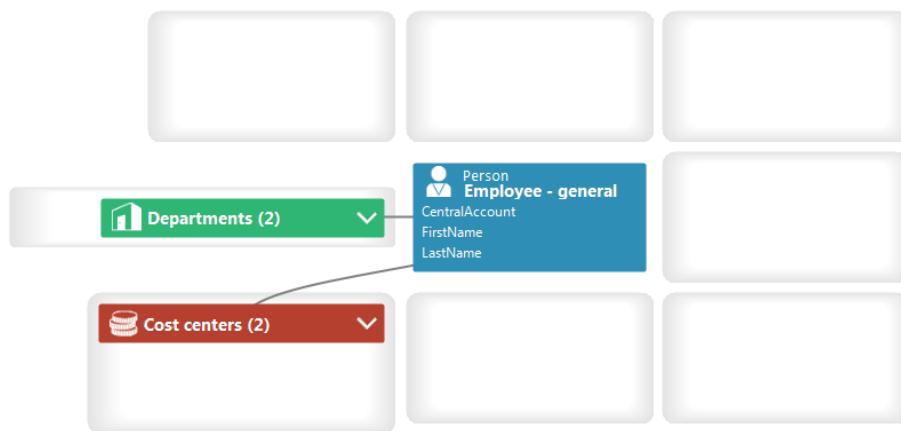
View	Description
Form definition	Displays properties of the main form element's interface form and can be edited. For more information, see Editing Interface Forms on page 279.
Variables	In this view, all the variable definitions that belong to the selected menu item are displayed in tabular form with type, name and assigned value. When you click on the variable type, a pop list with all available types is shown. The variable name and the value can be edited by clicking on the appropriate column in the table. Alternatively, you can enter the variable definition in a dialog window. To open the dialog window, select the edit icon in the toolbar.
Object assignments	All available object definition are displayed. You can assign interface forms to object definitions or delete existing assignments.
Group assignments	Displays all available permissions groups. You can assign the overview forms to the permissions groups or delete existing assignments. These forms are available to system users depending on their permissions group memberships.
Menu assignments	All available menu items are displayed. You can assign menu items or remove existing assignments. When the associated menu item is selected in the navigation or the item is selected in the result list, the interface forms are shown for all system users without taking their permissions group memberships into account.
Product assignments	All available applications are displayed. You can assign interface forms to the applications or remove existing assignments.
Object relations	All the object foreign key relations (FK), object child relations (CR) and object member relations (M:N) are displayed. You can drag and drop an object relation onto an element in the element part of the design view.

Creating Overview Forms

The Overview Form Editor helps you to create overview forms. The Overview Form Editor performs the following steps to create the overview form.

- Creates a menu item with type "Main form element".
- Creates other menu items under the main form element.
- Creates a user interface form for the main form element.

Figure 32: Design View in the Overview Form Editor



To create a new overview form

1. Select the category **User Interface | Forms | Overview forms** in the Designer.
2. Select the task **Create new overview form**.
3. Enter the basic properties for the overview form.

Table 165: Basic Overview Form Data

Property	Meaning
Menu item	Menu item name. You should try to use a meaningful name that can be passed on in the child structures.
Caption	Caption shown on the user interface form. The caption is used to represent the interface form in the task view and in the form's context menu in the user interface.
Object	Definition of the object for which to display the form.
Parent menu item	Parent menu item for group overview forms, normally a menu category.
Product assignment	Program in which the form will be shown.
Group assignment	Permissions group for which the form will be shown.
Display columns	Columns to be displayed on the main form element. TIP: Use Show column captions to switch between column captions and technical names.

4. To create the overview form, click **OK**.

This displays the overview form design in the Overview Form Editor. You can continue editing the overview form.

To add more form elements to the overview form

1. Select the view **Object relations**.

All the object foreign key relations (FK), object child relations (CR) and object member relations (MN) are displayed.

2. Select the object relation you want to display and drag it to one of the elements in the design view.
3. Select the type of menu item you want to create. You have the following options:
 - Create list element
Creates a fixed menu item with predefined list properties.
 - Create data element
Creates a data dependent menu item.
 - Create list element reference
Creates a link to menu item to be displayed as a list.
 - Create reference to data element
Creates a link to a data dependent menu item.

TIP: If you use **Create list element reference** or **Create reference to data element**, the submenu items `InfoSheet.List.<table>` or `InfoSheet.Node.<table>` respectively, are added under the menu item `InfoSheets.QIM.Links`.

The condition for the menu items is defined as variable `%<table>WhereClause%`. In the form element you assign a condition as WHERE clause to the variable.

The menu item master data is created automatically by the Overview Form Editor. The form element is display in the Overview Form Editor's design view. You can continue editing the overview form.

NOTE: You can create more menu items, links or statistics as form elements in the Overview Form Editor design view with **Create element** in the context menu. In this case, enter the master data for the menu item, link or statistics manually.

To create a preview of the overview form

1. Select the category **User Interface | Forms | Overview forms** in the Designer.
2. Select the overview form and open it in the Overview Form Editor.
3. Select the central form element's table in the **Table** menu in the Overview Form Editor's toolbar and select a fixed object to use for the preview from the **Object** menu.

NOTE: Select the entry "No object" in the **Object** menu to end the preview.

Features of the User Interface Form for the Central Form Element of an Overview form

- The user interface form is created with the form definition "VI_ElementNavigation". You have to use the form definition "VI_ElementNavigation" to set up an interface form.
- The name of the central form element is entered in the section "SpecialSheetData" in the user interface form's configuration data.

Example:

```
<DialogSheetDefinition FormatVersion="1.0">
    <SpecialSheetData>VI_Person_Person_Overview</SpecialSheetData>
</DialogSheetDefinition>
```

Features for Mapping Lists to an Overview Form

If the form element is used for mapping lists, the items are displayed with their names. The number of items is shown in the form element header. You can jump to the referenced object by clicking on the entry with the mouse.

Set the configuration option to the value "Ignore user interface forms in result list" to prevent the referenced object from being found in the navigation. This is useful if, for example, forms are not defined for some objects in the result list. Otherwise, an empty form is displayed.

Related Topics

- [Designing an Overview Form on page 303](#)
- [Disabling Overview Forms on page 305](#)
- [User Interface Navigation on page 251](#)
- [Linking Statistics into the User Interface on page 310](#)

Designing an Overview Form

The color and positioning of the form elements on the overview as well as the properties that are shown, are specified by layout information for the menu items. You can modify these properties for predefined overview forms as well.

To customize the form element's layout information

1. Select the category **User Interface | Forms | Overview forms** in the Designer.
2. Select the overview form and open it in the Overview Form Editor.

3. Select the form element in the design view.
4. Select the tab **Layout** in the **Properties** view and change the properties.

Table 166: Layout Information for Form Elements

Property	Meaning
Position	Position of the form elements on the overview form. You cannot align the main form element. The main form element is always displayed in the middle of the overview form. All child menu items are positioned relevant to the main form element.
Background color	Color of the form element for displaying on the overview form. The background color of the main form element cannot be configured. When a link is set up, it is given the background color of the referenced menu item.
Max. similar elements count	<p>If a menu item defines a list of items, each item in the menu item's result list is displayed in a separate form element.</p> <p>Define up to how many items should be displayed in separate form elements. If the number is exceeded the items are grouped into a list and displayed in one form element. Any columns that are given are not displayed in this case.</p> <p>The items are shown with their display mode. There is an icon in the header for showing and hiding the items. There is an icon in the header for showing and hiding the items.</p>
Columns to display	<p>Specify which column should be displayed for the valid object definition. The columns for the main form element refer to the object definitions of the associated overview form. All other form elements get their object definitions from the menu items. When a link is configured, the selected columns of the referenced menu item are initially copied to the link. The order of displaying the properties in a form element corresponds to the column sort order defined in the menu item.</p> <p>TIP: If you apply a list display mode with at most two columns names, you can set up a table in the display mode and achieve double column alignment.</p> <p>You can use scripts in the column definitions to affect the value displayed in the column. The column definition is activated when you click and hold on the column name. Extend the column definition as follows:</p> <pre>Column[S(script name)]</pre>

Designing the Form Element Header

The the menu item display text, display text for the objects to be shown and the menu item icon are displayed in the header of a form element.

- TIP:** An interface form can be opened using the display text in the header of a form element.
- To do this, assign a fixed menu item to the interface form that is allocated below the main form element. The interface form, however, must refer to the main form element, for example, a form for assigning this object.
 - Use the option **Navigation view** in the form assignment view to access forms in the user interface.

Disabling Overview Forms

You can disable single predefined overview forms or single form elements of an overview form if necessary. This prevents them being displayed in the user interface. They remain disabled even after schema installation.

To disable an overview form

1. Select the category **User Interface | Forms | Overview forms** in the Designer.
2. Select the overview form and start the Form Editor with the task **Edit interface form**.
3. Set the option **Disabled**.

To disable a form element on an overview form

1. Select the category **User Interface | Forms | Overview forms** in the Designer.
2. Select the overview form and start the Overview Form Editor with the task **Edit overview form <form name>**.
3. Select the form element in the design view.
4. Set the option **Disabled**.

You can also disable overview forms or single form elements using preprocessor conditions.

- NOTE:** You can find an overview of existing preprocessor dependencies in the Designer in the category Database **One Identity Manager Schema | Preprocessor dependencies**.

Statistics in the One Identity Manager

The One Identity Manager info system provides you with a quick overview of the system situation. These statistics are recalculated at regular intervals and visualized in the user interface using various display elements. Statistic definitions are already supplied with the One Identity Manager. You can create more statistic data in the Designer if required.

The following steps are necessary to make statistics available:

- Create statistic definitions
- Link statistics into the user interface

Detailed information about this topic

- [Working with Statistic Definitions on page 306](#)
- [Linking Statistics into the User Interface on page 310](#)
- [Diagram Types for Visualizing Statistics on page 311](#)
- [Examples of Statistic Definitions on page 315](#)

Working with Statistic Definitions

The basis for the info system is the definition of statistics. Predefined configurations are maintained by the schema installation and cannot be edited apart from a few properties. The default configuration is moved to a configuration buffer during handling. You can retrieve changes from the configuration buffer and restore the default configuration in this way.

To edit an statistic definition

1. Select the category **User Interface | Statistic definitions** in the Designer.
2. Select the statistic definition.
- OR -
To create a new statistic definition, select **Object | New**.
3. Edit the master data.
 - Enter general properties on the **Properties** tab.
 - Enter measurement queries on the **Queries** tab.

Detailed information about this topic

- [General Properties of a Statistic Definition on page 307](#)
- [Querying Statistic Measurements on page 308](#)
- [Examples of Statistic Definitions on page 315](#)

General Properties of a Statistic Definition

Table 167: Properties of a Statistic Definition

Property	Meaning
Statistics	Name of the statistic
Display name	This display name is used to show the statistic definition in the settings for the info system in the administration tools. The display name forms the title of a statistic. Translate the given text using the  button.  NOTE: If a caption is entered in the menu item, it overwrites the statistic definition display name.
Description	Description of the statistic definition. The statistic definition description is shown in the info system settings in the administration tools. Translate the given text using the  button.
Calculation schedule	Select the schedule for calculating the statistic information. Schedules supplied are: "Calculate statistics", "Calculate weekly statistics" and "Calculate monthly statistics on the 1st".  NOTE: Enable the schedules for calculating statistics in the category Basic data General Schedules in the Designer.
Aggregate function	Use the aggregate function if the measurements query returns several values but there should only be one value displayed in the statistics. Example: Determines the number of employees for which a department head is responsible. Use the aggregate function "SUM" to display a statistic for all employees in a department. Do not use an aggregate function to display statistics by department.
Base aggregate function	Use the base aggregate function if a unique base value cannot be attained from the measurements query.  NOTE: Aggregate and base aggregate functions are only evaluated if the formulated measurement value query is limited by a condition on the logged in user.  NOTE: Aggregate and base aggregate functions are only taken into account for statistics that are displayed in the Web Portal.
Threshold green	Threshold in range [0-1]. The base measurement percentage representing a "correct state" is found with the help of this threshold value factor.
Threshold red	The base measurement percentage representing an "accepted state" is found with the help of this threshold value factor.
Unit of	Unit for measured values. This is the unit of measure displayed in the info

Property	Meaning
measure:	system statistics. Translate the given text using the  button.
Time scale	Enter the display accuracy of the data on the time axis for statistic definitions that contain a time query (for example, the number of new employees in the last week). Permitted values are "hour", "day", "week", "month", "quarter" and "year".
Measurement runs to archive	The number of measurement run (apart from the current measurement) to be archived for displaying in the history. Only if the most recent values should remain intact, enter the value "0".
Disabled	Specifies whether the statistic definition is disabled. Statistic definition which are disabled are not calculated.
Preprocessor condition	You can add preprocessor conditions to statistics. This means that a statistic definition is only available when the preprocessor condition is fulfilled.
Disabled by preprocessor	If a statistic definition is excluded through a preprocessor condition, this option is set by the Database Compiler.
Instant calculation	Set this for statistic definitions, which are calculated at the moment they are displayed in the Web Portal. If this option is not set, the statistics are calculated during maintenance tasks.
Imported statistics data	Specifies whether these statistics are calculated at the moment they are displayed (for use in the Web Portal). If this option is not set, the statistics are calculated asynchronously by the DBQueue Processor.

Related Topics

- [Querying Statistic Measurements on page 308](#)
- [Examples of Statistic Definitions on page 315](#)
- [Setting Up and Configuring Schedules on page 134](#)
- [Conditional Compilation using Preprocessor Conditions on page 448](#)

Querying Statistic Measurements

Table 168: Measurement Query Properties

Property	Meaning
Measurements query	Enter the complete database query in SQL syntax to determine the statistic measurements. The query must return the columns ElementName and ElementValue as results. You can also return the columns ElementObjectKey, ElementObjectKey2 and ElementValue2 for displaying statistic information in the Web Portal.

Property	Meaning
	You can, optionally, control the display order of statistic measurements with the column ElementOrder. If the column ElementOrder does not exist, they are sorted by ElementName.
Base measurements query	<p>Enter the complete database query in SQL syntax to determine the statistic measurements. The query must return the columns ElementName and ElementValue as results.</p> <p>You can also return the columns ElementObjectKey, ElementObjectKey2 and ElementValue2 for displaying statistic information in the Web Portal.</p> <p>You can, optionally, control the display order of statistic measurements with the column ElementOrder. If the column ElementOrder does not exist, they are sorted by ElementName.</p> <p>The threshold factors entered in the fields Threshold green and Threshold red refer to the result in the ElementValue column. To determine the base measurement percentage, the result from column ElementValue is applied with 100%.</p> <p>NOTE: The column name ElementName in the base measurements query must match the name of the column ElementName in the measurements query.</p>
Condition	<p>Formulate a condition with which the statistic measurements can be limited to the current user. The condition has to be formulated as a valid where clause for database queries and limits the result of the query further based on the column ElementObjectKey using the variable %UserUID%.</p> <p>NOTE: The condition is only taken into account for statistics that are shown in the Web Portal.</p>

Example of Calculating the Threshold

Threshold factors help to find the base measurement percentage that represents an "Correct state" or a "unacceptable state".

Table 169: Example of Finding the State

Base Measure-ments	Threshold green	Threshold red	Percentage	State
100	0,25	0,75	< = 25	correct
			>25 to >75	acceptable
			>= 75	unacceptable
0,75	0,25	> = 75	> = 75	correct
			<75 to <25	acceptable
			<= 25	unacceptable

Related Topics

- [General Properties of a Statistic Definition on page 307](#)
- [Examples of Statistic Definitions on page 315](#)

Linking Statistics into the User Interface

In order to visualize statistics in the One Identity Manager administration tools, such as the Manager, you have to link the statistics into the user interface as a custom menu item.

You will typically find statistics under the category **Info system** in the administration tools navigation menu. You should set up custom menu items for statistics under an info system like this. All statistics that are defined at one menu level are displayed on one form.

Statistics can also be linked as form elements into overview forms. To do this, use the Overview Form Editor.

NOTE: If you set up a custom info system, ensure that the menu item under which you define the statistics, is labeled with the configuration flags **Not expandable by user** and **Force open menu item**.

For more information about general properties of menu items, see [General Menu Item Properties on page 262](#). Take note of the following properties for menu items.

Table 170: Statistics Properties

Property	Meaning
Item type	Select the item type "Statistics".
Caption	The caption given here, overwrites the statistic definition caption. Leave this field empty if you want to use the statistic definition display name.
Statistics	Enter the statistic definition to be displayed.

Property	Meaning
Diagram type	Select the diagram type that is going to represent the statistic.
Alignment	Positioning of statistics on the overview form. This layout information is used if the statistic is used as a form element on an overview form.
Background	Background color of the form elements on the overview form. This layout information is used if the statistic is used as a form element on an overview form.

All menu items that are to be displayed in an application user interface have to be assigned to a permissions group and an application. Use the views **Applications** and **Permissions groups** for this when you are editing menu items.

Related Topics

- [Diagram Types for Visualizing Statistics on page 311](#)
- [Examples of Statistic Definitions on page 315](#)
- [Editing Menu Items on page 258](#)
- [General Menu Item Properties on page 262](#)
- [Creating Overview Forms on page 300](#)
- [Designing an Overview Form on page 303](#)

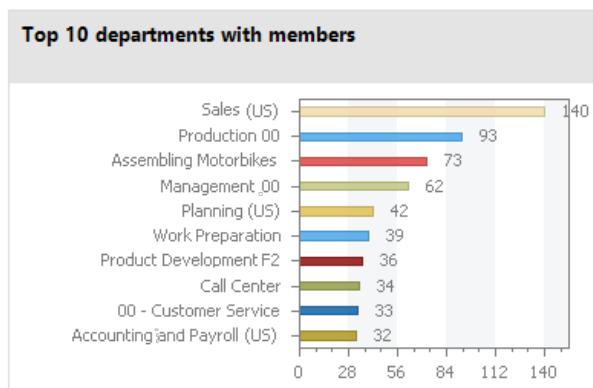
Diagram Types for Visualizing Statistics

There are several diagram types available for visualizing statistics.

Bar Chart

A bar chart can be used to visualize comparisons between measurements. The actual measurement of the column `ElementValue` and the identifier for column `ElementName` are used to label the diagram.

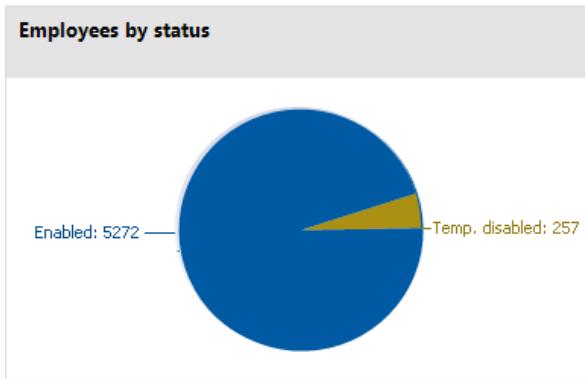
Figure 33: Bar Chart Example



Pie Chart

A pie chart can be used to visualize the measurements as a percentage of the base measurement. The actual measurement of the column ElementValue and the identifier for column ElementName are used to label the diagram.

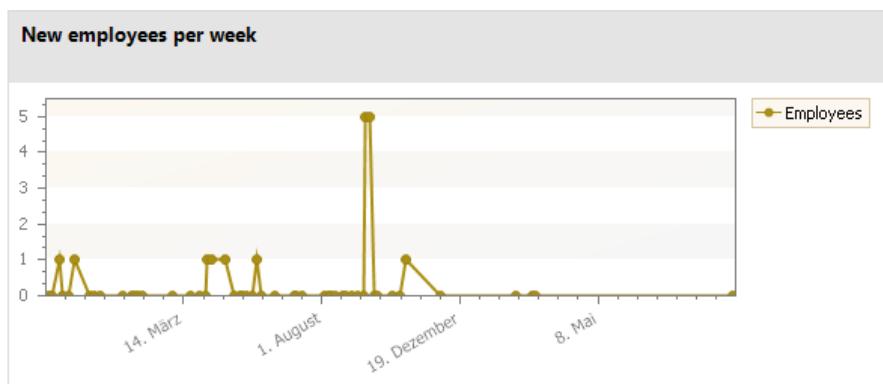
Figure 34: Pie Chart Example



Line Diagram

A line diagram can be used to visualize a data sequence over a specified time period. The time axis is scaled in proportion to the time scale given in the statistic definition. The number of measurements in the line diagram results from measurement runs to archive given in the statistic definition. Click with the mouse on a point of measurement and a tooltip showing the measurement is displayed.

Figure 35: Line Diagram Example



Traffic light

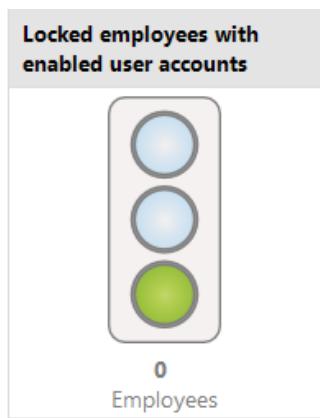
A traffic light diagram can be used to visualize the state of the system. The state is indicated by the color. The threshold factors given in the statistic definition determine when which status is reached.

Table 171: Meaning of the Colors

Color	State
Green	correct
Yellow	acceptable
Red	unacceptable

The actual measurement of the column ElementValue and the identifier for column ElementName are used to label the diagram.

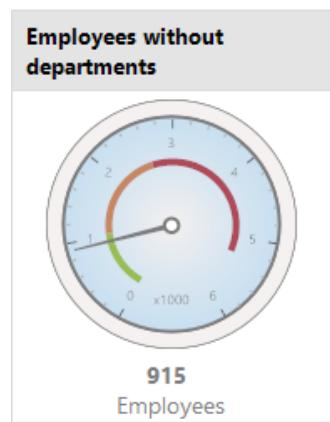
Figure 36: Traffic Light Example



Tachometer

A tachometer diagram can be used to visualize the state of the system in more detail than in a traffic light diagram. The base measurement is also displayed. The state is indicated by the color. The threshold factors given in the statistic definition determine when which status is reached. The actual measurement of the column ElementValue and the identifier for column ElementName are used to label the diagram.

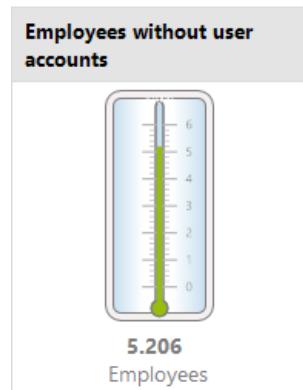
Figure 37: Tachometer Diagram Example



Thermometer

A thermometer diagram can be used to visualize the state of the system in more detail than in a traffic light diagram. The state is indicated by a color scale on the side of the diagram. The threshold factors given in the statistic definition determine when which status is reached. The actual measurement of the column ElementValue and the identifier for column ElementName are used to label the diagram.

Figure 38: Thermometer Diagram Example



Table

This diagram type can be used to visualize the measurements in table form. Enter a number of archived measurements runs in the statistic definition, to present the data over

a specified time period.

Figure 39: Table Example

Number of employees		
	21.09.2017	
Employees	5.274	

Examples of Statistic Definitions

Example 1:

The number of people in the company should be displayed in the statistics. This statistic should be calculated daily. The statistics definition could look like:

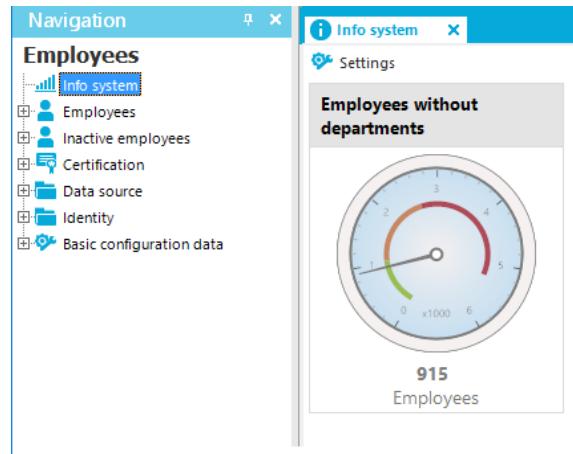
Statistic:	CountEmployees
Display name:	Number of employees
Description:	Finds the number of employees in the company on a daily basis.
Calculation schedule:	Calculate statistics
Measurements query:	<pre>select 'Employees' as ElementName, count (*) as ElementValue from Person</pre>

To display the statistics in the Manager in the category **Employees | Info system**, the following menu item is created:

Menu item:	Person.InfoSystem.CountEmployees
Item type:	Statistics
Sort order	1
Statistic:	Number of employees
Diagram type:	thermometer

The menu item is assigned to the application "Manager" and to an application role and then it can be displayed in the Manager.

Figure 40: Displaying the statistics in the Manager



Example 2:

The number of external employees in the company should be displayed in the statistics. This statistic should be calculated weekly. If more than 20% of employees in the company are externals, the info system should display the state as acceptable instead of a correct. If more than 80% are externals the state should be unacceptable.

Statistic:	CountExternalEmployees
Display name:	Number of external employees.
Description:	Find the number of external employees in the company on a weekly basis.
Calculation schedule:	Calculate weekly statistics
Measurements query:	Select 'Employees' as ElementName, Count (*) as ElementValue from Person where IsExternal = 1
Base measurements query:	Select 'Employees' as ElementName, Count (*) as ElementValue from Person
Threshold green:	0,2
Threshold red:	0,8

To display the statistics in the Manager in the category **Employees | Info system**, the following menu item is created:

Menu item: Person.InfoSystem.CountExternalEmployees

Item type:	Statistics
Sort order	2
Statistic:	Number of external employees.
Diagram type:	Traffic light

The menu item is assigned to the application "Manager" and to an application role and then it can be displayed in the Manager.

Example 3:

The number of employees, for which the current user is entered directly as manager, should be represented in a statistic. Restrictions to the values for the current user are made through a condition.

Statistic:	CountEmployeesPersonHead
Display name:	Supervised employees
Description:	Finds the number of employees for which the manager is responsible on a daily basis.
Calculation schedule:	Calculate statistics
Measurements query:	<pre>select XObjectKey as ElementObjectKey, 'Employees' as ElementName, Count (*) as ElementValue from Person where IsExternal = 1 Group by XObjectKey</pre>
Condition:	<pre>ElementObjectKey in (select XObjectKey from Person where uid_PersonHead = '%useruid%')</pre>

Configure the web project in the Web Designer, to display statistics in the Web Portal info system.

Example 4:

Internal and external employees, which the current user supervises as department manager, should be represented in a statistic. Departments are added here separately to determine clear results for displaying the measurement because a department manager might be responsible for more than one department.

Statistic:	PersonCountInternalExternal_By_Department
Display name:	Number of internal and external employees

Description:	Finds the number of internal and external employees per department on a daily basis.
Calculation schedule:	Calculate statistics
Measurements query:	<pre>select d.XObjectKey as ElementObjectKey, 'Internal' as ElementName, count(p.uid_person) as ElementValue from Department d Left Outer Join Person p on p.UID_Department = d.UID_Department and p.IsExternal = 0 Group By d.XObjectKey UNION ALL select d.XObjectKey as ElementObjectKey, 'External' as ElementName, count(p.uid_person) as ElementValue from Department d Left Outer Join Person p on p.UID_Department = d.UID_Department and p.IsExternal = 1 Group By d.XObjectKey</pre>
Condition:	<pre>ElementObjectKey in (select d.XObjectKey from Department d join helperheadorg hpo on d.UID_Department = hpo.UID_Org where hpo.UID_PersonHead = '%useruid%')</pre>
Aggregate function	SUM

Configure the web project in the Web Designer, to display statistics in the Web Portal info system.

Example 5:

Ten employees with the highest risk index should be found and displayed in a statistic. They should be sorted by measurement unit.

Statistic:	Top10ActivePersonByRiskIndex
Display name:	Top 10 active employees by risk index
Description:	Find ten active employees with the highest risk indexes on a daily basis.
Calculation schedule:	Calculate statistics
Measurements query:	<pre>select top 10 p.InternalName as ElementName, Round(100 * IsNull(p.RiskIndexCalculated, 0), 0) as ElementValue, p.XObjectKey as ElementObjectKey, ROW_NUMBER() over (order by IsNull(p.RiskIndexCalculated, 0) desc,</pre>

```

p.InternalName) as ElementOrder
from Person p
where p.IsInactive = 0
order by ElementOrder

```

Configure the web project in the Web Designer, to display statistics in the Web Portal info system.

Extending the Launchpad

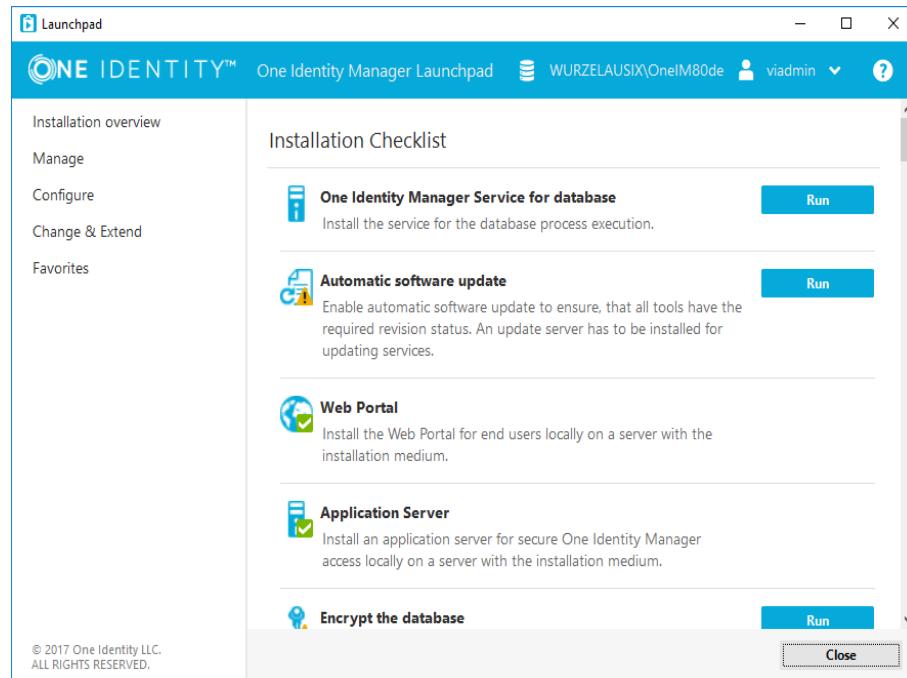
The Launchpad is the central tool for starting administration tools and One Identity Manager configuration tools. Use the Launchpad to check the existing One Identity Manager installation and start One Identity Manager tools for executing individual tasks.

The Launchpad can be customized. You can define your own menu items and action for the Launchpad in the Designer.

You can control how and where menu items are displayed in the Launchpad. You use the menu hierarchy and the different types of menu items to do this. For more detailed information about the structure of a menu hierarchy and the individual menu items and their properties, see [User Interface Navigation on page 251](#).

One Identity Manager supplies a list of Launchpad actions that you can use to start applications. You can also start your own applications over the Launchpad.

Figure 41: The Launchpad User Interface



Recommendations on extending the Launchpad

- To create a new category in the Launchpad navigation on the left-hand side, use the menu item of type "Menu category". The items are shown with their display text.
- To group tasks in the Launchpad main view, use the menu item of type "task category". The items are shown with their display text.
- Apply the entry types "Tasks", "Fixed menu item" or "Data dependent menu item" to each Launchpad task. The items are shown with their display text and description.
- Specify the order for displaying the menu items.
- To display the task status, enter an overlay icon definition on the menu item in VB.net syntax. Use the NavigationNodeState class.

Syntax:

```
public NavigationNodeState(string state, string imageUidOrName, string description)
```

```
public NavigationNodeState(string state, string imageUidOrName, string description, bool enabled, bool visible, int count)
```

Table 172: Parameter of the script NavigationNodeState

Parameter	Description
State	Status, returned, for example "Info", "Ok", "Error", "Warning".
ImageUidOrName	UID or name of the icon to display.
Description	Text, which is displayed as a tooltip.
enabled	Specifies whether the start button for the action is set or not.
visible	Specifies whether the task is shown or not.
count	Number of affected objects.

Calling example:

```
Value = New NavigationNodeState("Ok", "QBM-33228392E9863141A9306B38ADF3D502", #LD("Project is completed.")#)  
Value = New NavigationNodeState("Error", "QBM-a486f0eabf674392bbbdf8572453258c", #LD("Project is not completed.")#)
```

- You can use the condition to specify whether the task is only available for a direct database connection or a connection over an application server. To do this, use the variable SessionType.

Examples of conditions

Only direct database connection: '%SessionType%' = 'Direct'

Only connection through application server: '%SessionType%' = 'AppServer'

If no condition is given, the task is always available.

- If an action is going to be run from a task, link an Launchpad action to the menu item.

This displays the **Start** button for the task. The Launchpad action's description is displayed in the button's tooltip.

To extend the Launchpad

1. Create a new menu item for the Launchpad.
 - a. Select the category **User Interface | User interface navigation | Launchpad** in the Designer.
 - b. Start the User Interface Editor using **Edit navigation for application 'Launchpad'** in the task view.
 - c. Create the menu item.
 - d. Assign the menu item to the application "Launchpad".
 - e. Assign the permissions group menu items to the "QBM LaunchPad".
2. Assign the Launchpad actions to the menu items.
 - a. Select the category **User interface | Launchpad actions** in the Designer.
 - b. Select the menu item **View | Select table relations...** and enable the table DialogTree.
 - c. Select the Launchpad action and assign the action to the menu item on the **Menu items** tab.

Related Topics

- [Launchpad Actions on page 321](#)
- [Editing Menu Items on page 258](#)

Launchpad Actions

One Identity Manager supplies a list of Launchpad actions that you can use to start applications. You can also start your own applications over the Launchpad.

At the start an application, you can pass calling parameters, tasks and task parameters which the application can identify. Variable are permitted in this case. Supported are:

- Environment variables with the syntax %variable%
- Navigation variables with the syntax %variable%
- Columns of the object passed in \$ notation.

To display Launchpad actions

1. Select the category **User Interface | Launchpad actions** in the Designer.
2. Select the Launchpad action The following master data is required for a Launchpad action.

Table 173: Properties of an Action

Properties	Description
Description	Detailed description of the action. The description is displayed in the tooltip.
Executable file	Full name of the executable file.
Execution parameter	Additional calling parameter for starting the application.
Action	Name of the action.
Administrative context	Specifies whether the application can only be started by an administrator. The application expects authentication as an administrator.
Task	Task that must be passed additional as a start parameter.
Task parameter	Additional parameter for the task.

Task Definitions for the User Interface

You can use tasks to perform specific actions on objects with the One Identity Manager tools, for example, unlocking a user account or starting processes to copy applications. The tasks are shown in alphabetical order in the task view of the One Identity Manager tool.

Task definitions are created for object definitions so that different tasks can be shown in the user interface depending on the selected objects. These tasks are also available to Quests, taking their permissions group memberships into account, when tasks are also assigned to permissions groups. Apart from these object dependent task definitions, form tasks are provided through the user interface form and cannot be edited.

Predefined configurations are maintained by the schema installation and cannot be edited apart from a few properties. The default configuration is moved to a configuration buffer during handling. You can retrieve changes from the configuration buffer and restore the default configuration in this way.

To edit a task

1. Select the category **User Interface | User interface navigation | Task definitions** in the Designer.
 2. Select the task.
 - OR -
- To create a new task, select the menu item **Object | New**.
3. Edit the method's master data.

4. Assign a permissions group to the program function.
 - Select the menu item **View | Select table relations...** and enable the table **DialogGroupHasMethod**.
This shows the tab **Permissions groups** for assigning permissions groups.

You can disable individual predefined tasks to prevent them being shown in the user interface. They remain disabled even after schema installation.

To disable a task

1. Select the category **User Interface | User interface navigation | Task definitions** in the Designer.
2. Select the task.
3. Set the option **Disabled** to "False".

If a task definition is assigned a program function, the user can only run this task if the necessary program function is assigned to him. An error occurs if the user does not own this program function and tries to run it. Program functions are not assigned to single users but to permissions groups. All users that are assigned to these groups can user the program function.

To make a method definition available to users using a program function

1. Connect the task definition with the program function.
 - a. Select the category **User Interface | User interface navigation | Task definitions** in the Designer.
 - b. Select the task and assign the program function to it.
 - Select the menu item **View | Select table relations...** and enable the table **DialogMethodHasFeature**. The **Program function** tab is displayed in the edit view you used to assign the program function.
2. Assign a permissions group to the program function.
 - a. Select the category **Permissions | Program functions** in the Designer.
 - b. Select the program function and assign it to a permissions group.
 - Select **View | Select table relations...** and enable the table **DialogGroupHasFeature**. This shows the tab **Permissions groups** for assigning permissions groups.

Detailed information about this topic

- [Task Definition Properties on page 324](#)

Task Definition Properties

Table 174: Task Properties

Property	Meaning
Task name	Name of the task.
Caption	The display name is used to display the task in the administration tool task view. Display names can be given in more than one language.
Description	Description of the method. The description is shown as a tooltip in the user interface task view.
Object	Definition of the object for which the task is made available.
Enabled for	This property specifies the intended use of the task. The task can be displayed in the graphical user interface (FatClient) and in the web application (Web client) and is labeled accordingly. These conditions determine whether a task can be shown in the appropriate administration tool.
Task behavior	Sets the behavior of the task. The following options are permitted: <ul style="list-style-type: none">No data Default. The task is available for single object and multiple object editing. Changes are executed separately for each object, even if multiple edit is used.Save required The task saves data. A corresponding alert message is displayed.Single objects only This method is only permitted for single objects.Save required + single objects only The task saves data. A corresponding alert message is displayed. This method is only permitted for single objects.Execute on multiple objects This method is available for multiple editing of objects. Changes are executed for all objects together through a multi-object.Save required + execute on multiple objects The task saves data. A corresponding alert message is displayed. This method is available for multiple editing of objects. Changes are executed for all objects together through a multi-object.
Icon	Icon for displaying the task in the user interface.
Script	Task script. You can use function calls or command input in VB.Net statements for the task script. The currently loaded object is always addressed with base .

Property	Meaning
	<p> NOTE: The database needs to be complied after changing a task script.</p>
Disabled	Specifies if the task is displayed in the user interface or not. Disabled tasks are never displayed in the user interface. Predefined system users are not effected by this limitation. This modification is also permitted for predefined default user interface tasks and is not overwritten when the schema is installed.
Processing status	The process state is used for creating custom configuration packages.
Permissions group	Permissions group, whose users can use these tasks.
	<p>To assign a permissions group</p> <ul style="list-style-type: none"> Select the menu item View Select table relations... and enable the table <code>DialogGroupHasMethod</code>. <p>This shows the tab Permissions groups for assigning permissions groups.</p>
Program function	Program function, which is linked to the task definition.
	<p>To assign a program function</p> <ul style="list-style-type: none"> Select the menu item View Select table relations... and enable the table <code>DialogMethodHasFeature</code>. <p>The Program function tab is displayed in the edit view you used to assign the program function.</p>

Related Topics

- [Task Definitions for the User Interface on page 322](#)
- [Using Scripts on page 453](#)
- [Using #LD Notation on page 464](#)
- [Language Dependent Data Representation on page 328](#)
- [Compiling a One Identity Manager Database on page 64](#)

Applications for Configuring the User Interface

The One Identity Manager's default version supplies applications and predefined navigation menus for the One Identity Manager tools "Manager", "Designer", "Launchpad" and the One

Identity Manager web interface. Predefined configurations are maintained by the schema installation and cannot be edited apart from a few properties. It is not usually necessary to define your own applications. You might possibly need your own applications for a customer specific web interface.

The available programs are shown in the category **Base data | Security settings | Programs** in the Designer.

Table 175: Predefined Programs

Product	Meaning
Default	Default for front-ends without special usage, for example Job Queue Info or Report Editor. Required to determine the authentication module.
Designer	Program for the configuration tool, the Designer.
Manager	Program for the administration tool, the Manager.
Launchpad	Program for the tool Launchpad.
WebDesigner	Program for installing the Web Portal.
WebDesignerEditor	Program for the Web Designer to configure and extend the Web Portal.
Application server	Program for installing the application server.
SOAP Service	Application for installing the SOAP Web Service.
SPML Service	Program for installing the SPML Web service.

Program Properties

Table 176: Program Properties

Property	Meaning
Program	Name of the program.
Remarks	Comments about the program.
Start menu item	If the given start menu item is available to a system user in a program's navigation menu, the program navigates straight to this position in the menu when it starts up. You can specify, for example, a home page for a system user with this feature. This function is mainly used by web interfaces.
Configuration data	Configuration data is used to determine a system user by the dynamic authentication module. You can also adapt the configuration data for the default applications that are supplied.

Property	Meaning
Min. version	Lowest version of the application that can run with the database version in use. This input is used solely as information, the version number is not verified.
Engine based	Specifies whether menu navigation and forms can be assigned to the program.
Processing status	The process state is used for creating custom configuration packages.
Authentication module	Authentication module used by the program.
	To display authentication modules <ul style="list-style-type: none"> • Select the menu item View Select table relations... and enable the table DialogProductHasAuthentifier.
Form	Forms displayed in the program.
	To display a form <ul style="list-style-type: none"> • Select the menu item View Select table relations... and enable the table DialogProductHasSheet.
Menu	Menus displayed in the program.
	To display menu items <ul style="list-style-type: none"> • Select the menu item View Select table relations... and enable the table DialogTreeInDialogProduct.
System user	System users that use the program.
	To display system users <ul style="list-style-type: none"> • Select the menu item View Select table relations... and enable the table DialogUserConfiguration.

Related Topics

- [Applications for Configuring the User Interface on page 325](#)
- [Configuration Data for System User Dynamic Authentication on page 108](#)

Icons and Images for Configuring the User Interface

When you are configuring the One Identity Manager tools' user interfaces you can add icons and images for displaying in different parts of them. The default version of the One Identity

Manager supplies the icons and images that you can use for configuring the user interface and to create reports. Predefined configurations are maintained by the schema installation and cannot be edited apart from a few properties.

Icons are required to be in PNG format with sizes of 16x16 pixels, 24x24 and 32x32 pixels for the graphical interface.

Icons are required for the following use cases.

Table 177: Meaning of the Icons

State	Usage
Normal	Icons showing an enabled state. These icons must exist in the One Identity Manager database.
Inverted	Icons that show an enabled state on a black background. These icons can be converted automatically.
Disabled	Icons showing a disabled state. These icons must exist in the One Identity Manager database.

To add an icon

1. Select the category **Base Data | General | Icons** in the Designer.
2. Select **Object | New** in the menu.
3. Give the icon a name.
4. Upload the icon using .

To add images for reports

1. Select the category **Base Data | General | Large images** in the Designer.
2. Select **Object | New** in the menu.
3. Give the image a name.
4. Upload the image using .
5. Save the image with the  button.

The image is displayed with certain important image properties.

Language Dependent Data Representation

The One Identity Manager supports language dependent representation of data. You can use this feature to edit display text in different languages for the One Identity Manager tool user interfaces. You can also create multi-language text for process information output, script processing as well as processing messages.

The One Identity Manager default installation is supplied in the languages "English - United States [en-US]" and "German - Germany [de-DE]". You can use other languages if required. To do this it is advisable to translate the required text before starting to use the One Identity Manager. There is a Language Editor in the Designer to help you do this. A special control element is provided in the One Identity Manager tools which aids multi-language input.

Detailed information about this topic

- [Basic Rules for Using Language Dependent Data on page 329](#)
- [Labeling Columns for Translation on page 330](#)
- [Editing Translations in the Language Editor on page 333](#)
- [Importing Translations on page 334](#)

Related Topics

- [Enabling More Languages for Displaying and Maintaining Data on page 126](#)

Basic Rules for Using Language Dependent Data

In order to use multi-language data representation in the One Identity Manager, the following prerequisites need to be fulfilled:

- The language is set up in the database and labeled with **Select in front-end**.
- A fallback language for the database is declared. The language "English - United States [en-US]" is used in the One Identity Manager default installation. This language is used if there is no translation available for language dependent drilling in the user's requested language.
- The option **Multilingual** has to be set on the column definitions in order to use multi-language display text.
- Source and target of the translation are known.
- #LD notation is used for outputting language dependent data from within VB.Net expressions. #LD text is automatically extracted for translation. To do this, a column must be labeled as #LD content.

The translations are stored in the table DialogMultiLanguage. A key, the language and the translation are entered into the table.

Example

The text "Resource" should be shown as column name for the column Ident_QERResource if the login language is "English - United States [en-US]" and "Ressource" if the login language is "German - Germany [de-DE]". The value "Car" is entered in the column Ident_

QERResource. If the login language is "English - United States [en-US]", the word "car" should be shown. If the login language is "German - Germany [de-DE]", the word "Auto" should be shown.

Table 178: Example for Language Dependent Entries in the table "DialogMultiLanguage" with Default Language "English - United States [en-US]"

Column Name	Key	Language	Value
DialogColumn.Caption	Resource	English - United States [en-US]	Resource
	Resource	German - Germany [de-DE]	Resource
Ressource.Ident_QERResource	Car	English - United States [en-US]	Car
	Car	German - Germany [de-DE]	Auto

Related Topics

- [Labeling Columns for Translation on page 330](#)
- [Editing Translations in the Language Editor on page 333](#)
- [Enabling More Languages for Displaying and Maintaining Data on page 126](#)
- [Using #LD Notation on page 464](#)

Labeling Columns for Translation

Columns must be marked for translation in order to enter multilingual captions.

To mark a One Identity Manager database column for translation

1. Select the category **One Identity Manager Schema** in the Designer.
 2. Select the table and start the Schema Editor with the task **Show table definition**.
- NOTE:** Select a table column in the Designer, then you can start the Schema Editor from the task **Show column definition**.
3. Select the column in the Schema Editor and edit the property **Multilingual** and specify the following settings. The combination of values determines the resulting translation. The following values are permitted:

Value	Meaning
Translation target	The translated column content is displayed.
Translation source	The column supplies translations.

Value	Meaning
#LD content	The column has content in #LD notation. The contents are extracted for the translation.
A translation target is normally the same as the translation source. If the translation, however, is taken from another translation source, enter this additionally as a language dependency.	
<ul style="list-style-type: none"> Change to the Multilanguage dependencies tab and select the column to act as translation source under Translation source. <p>NOTE: The column that is used as translation source, must be labeled as "translation source".</p>	

Example: a column is translation target and source

You want the contents of the column QERRessource.Ident_QERResource to be translated. The value "Car" is entered in the column Ident_QERResource. If the login language is "English - United States [en-US]", the word "car" should be shown. If the login language is "German - Germany [de-DE]", the word "Auto" should be shown. The translation itself maintained in the column QERRessource.Ident_QERResource.

- Label the column QERRessource.Ident_QERResource with the value "translation target" in the **Multilingual** property.
- Label the column QERRessource.Ident_QERResource with the value "translation source" in the **Multilingual** property.
- Translate the entry for QERRessource.Ident_QERResource in the Language Editor

When the column is loaded, it is determined that QERRessource.Ident_QERResource should be translated. The corresponding key for QERRessource.Ident_QERResource is taken from the table DialogMultiLanguage as translation and displayed in the current user's login language.

Example: a column is translation target and takes its translation from another translation source

The action is displayed in the Manager process view in the current user's login language. The contents of the column DialogProcess.DisplayName are taken from the column JobEventGen.ProcessDisplay. The column JobEventGen.ProcessDisplay may use #LD notation to create the display string.

- Label the column JobEventGen.ProcessDisplay with "translation source" and the value "#LD content" in the **Multilingual** property.
- Label the column DialogProcess.DisplayName with the value "translation target" in the **Multilingual** property and enter the column JobEventGen.ProcessDisplay as **Multilanguage dependencies**.
- Translate the entry for JobEventGen.ProcessDisplay in the Language Editor.

When the column is loaded, it is determined that `DialogProcess.DisplayName` should be translated. The corresponding key for `JobEventGen.ProcessDisplay` is taken from the table `DialogMultiLanguage` as translation and displayed in the current user's login language.

Using the text store

Translations, which occur frequently or cannot be associated with a particular database column, can be stored in text memory (table `QBMTranslationAddOnSource`). The Web Portal, for example, takes its translations from the text store. In the same way, output text from database triggers is found in the text store. The text store is also used as a fallback when a fitting translation cannot be found through other translation sources. You can also reference the text store as a translation source.

To enter an item in the text store

- Select the category **Base Data | Localization | Translatable texts** in the Designer.
- Select **Object | New** and enter the translation key.
- Translate entries for `QBMTranslationAddOnSource.EntryKey` in the Language Editor.

Related Topics

- [Editing Translations in the Language Editor on page 333](#)
- [Column Properties on page 189](#)
- [Using #LD Notation on page 464](#)

Working with the Language Editor

Use the Language Editor to enter caption translations. The editor is started from the program "Designer" and opens in the document view. Only additional Language Editor functions are described in the following.

Menu Items

The following items are added to the menu bar when the editor starts.

Table 179: Menu Items Added by the Editor

Menu	Menu Item	Meaning
Translation	Reload translations	Updates the display with translations.
Help	Multi-language support help	Opens the help on this topic.
	Language Editor help	Opens the editor help.

Table 180: Meaning of Toolbar Icons

Icon	Meaning
	Shows untranslated captions.
	Entries filtered by the term entered in the field.
	Resets the filter.
	Updates the display with translations.
	Imports the translation.

Editing Translations in the Language Editor

With the Language Editor you can carry out translations for:

- The content of column labeled for multi-language input
- #LD expressions from columns containing VB.Net code
- Text stored in the text store (table QBMTranslationAddOnSource)

All translatable entries that are shown with their translation status in the Language Editor translation table.

TIP: Click with the mouse in a column header to sort by the selected column.

The following information is displayed.

Table 181: Information in the Translation Table

Properties	Meaning
Status	Status of the item.
Table	Translation source table.
Column	Translation source column.
Module	Module belonging to the translation.
Usage	Number of time the translation is used.
Key	Key value to be translated.
Checked	Specifies if the translation has been tested.
Language	Translation in the selected language.

To edit a translation in the Language Editor

1. Select the category **Base data** in the Designer.
2. Start the Language Editor using the task **Language Editor**.

3. Select the languages of the translations you want to edit under **Select languages** in the toolbar.
4. Use filters to limit how much data is displayed, if necessary.
 - Click with the mouse in a column header to sort by the selected column.
 - To limit column entries, click on the arrow in the column header. This opens a text box into which you can enter filter text. If a filter is defined, a  icon is displayed in the column header. To remove the filter, click on the arrow in the column header and select **Remove**.
 - You can also enter a filter using the **Filter** option in the toolbar. Apply the filter to the key and its translations by clicking the  button. To reset the filter, click the  icon.
- If you double-click on a translation and select text in the field, the selected text is automatically copied to the **Filter** text box.
5. Run one of the following tasks.
 - Enter the translation. You can enter a translation by double-clicking on the text field.
 - When you have checked the translation, set the **Checked** box.

 **TIP:** Use the context menu **Show usage** to display where this text is displayed. The **Usage** displays the number of times the text is used. By double-clicking on an item, a dialog box opens showing the object's properties.

6. Commit the changes to the main database. To do this, select **Database | Commit to database...** from the menu.

After the changes have been committed to the main database, the system data is recalculated by the DBQueue Processor in order to make the new multi-language data available to all system users.

Related Topics

- [Labeling Columns for Translation on page 330](#)
- [Importing Translations on page 334](#)

Importing Translations

The One Identity Manager default installation is supplied in the languages "English - United States [en-US]" and "German - Germany [de-DE]". Other languages are made available with a One Identity Manager Language Pack CSV files for translating the Web Portal. For more information about how the Web Portal uses languages, see the One Identity Manager Installation Guide.

 **NOTE:** You will find the One Identity Manager Language Pack in the Support Portal under <https://support.oneidentity.com/>

The import:

- Creates the translations in the table `DialogMultiLanguage`.
- Updates currently existing entries based on the key, the table and the column.
- Deletes the entries.

To import the language files

1. Select the category **Base data** in the Designer.
2. Start the Language Editor using the task **Language Editor**.
3. Select the  icon in the Editor's toolbar.
4. Select *.csv files with the required language and click **Open**.
This starts the import. This may take some time.
5. Commit the changes to the main database. To do this, select **Database | Commit to database...** from the menu.

After the changes have been committed to the main database, the system data is recalculated by the DBQueue Processor in order to make the new multi-language data available to all system users.

Process Orchestration in One Identity Manager

One Identity Manager uses so called 'processes' for mapping business processes. A process consists of process steps, which represent processing tasks and are joined by predecessor/successor relations. This functionality allows flexibility when linking up actions and sequences on object events. Processes are modeled using process templates. A process generator (Jobgenerator) is responsible for converting script templates in processes and process steps into a concrete process in the 'Job queue'.

The server service "One Identity Manager Service" ensures distribution in the network of data managed in the One Identity Manager database. The One Identity Manager Service performs data synchronization between the database and any connected target systems and executes actions at the database and file level. The One Identity Manager Service retrieves process steps from the JobQueue. Process steps are executed by process components. One Identity Manager Service also creates an instance of the required process component and passes the parameters to the process step. Decision logic monitors the execution of the process steps and determines how processing should continue depending on the results of the executed process components. The One Identity Manager Service enables parallel processing of process steps because it can create several instances of process components.

The One Identity Manager Service is the only One Identity Manager component authorized to make changes in the target system.

Detailed information about this topic

- [Declaring the Job Server on page 337](#)
- [One Identity Manager Service Configuration on page 353](#)
- [Handling Processes in the One Identity Manager on page 384](#)

Related Topics

- [Monitoring Process Handling with Job Queue Info on page 611](#)

Declaring the Job Server

The One Identity Manager Service handles defined processes. To execute the processes, the One Identity Manager Service has to be installed on the One Identity Manager network server.

Each One Identity Manager Service within the network must have a unique queue identifier. The process steps are requested by the Job queue using exactly this queue name. Enter this queue name in the One Identity Manager Service configuration file. Create a corresponding Job server entry for every queue.

To declare a Job server in the One Identity Manager

- Create an entry for the Job server with the Job Server Editor in the Designer.
- Specify the Job server's server functions.
- Specify the Job server's machine role.
- Install the One Identity Manager Service.

You can use Job Server Editor to install, configure and start the One Identity Manager Service remotely.

- Make any other changes to the One Identity Manager Service configuration settings.

This configuration is already created when the One Identity Manager Service is installed. Use Job Server Editor to modify each configuration setting.

Execute the necessary steps with the Job Server Editor. Start the Job Server Editor in the category **Base data | Installation | Job server** using the task **Edit Job server** in the Designer.

Detailed information about this topic

- [Editing a Job Server on page 340](#)
- [Server Functions of a Job Server on page 343](#)
- [Job Server Machine Roles on page 344](#)
- [Installing the One Identity Manager Service on the Job Server on page 346](#)
- [Customizing the One Identity Manager Service Configuration for a Job Server on page 349](#)
- [One Identity Manager Service Configuration on page 353](#)

Related Topics

- [Working with the Job Server Editor on page 338](#)

Working with the Job Server Editor

You use the Job Server Editor to edit the Job server attributes and specify the server function. Install and configure the One Identity Manager Service also using the Job Server Editor. The editor is started from the program "Designer" and opens in the document view. Only additional Job Server Editor functions are described in the following.

Menu Items

The following items are added to the menu bar when the editor starts.

Table 182: Meaning of Items in the Menu Bar

Menu	Menu Item	Meaning
Job server	New	Adds a new Job sever.
	Delete	Deletes selected job server.
	Start HTTP request.	Displays the HTTP Servers home page.
	Transfer configuration to Job server	A process is created that updates the configuration file on the Job server.
	Reload job servers	Updates the job server view.
View	Install service...	Starts the service installation wizard
	Properties	Shows/hides the edit view.
	Server functions	Show/hides the server function definition view.
	Machine role	Show/hides the machine role definition view.
	Configure One Identity Manager Service	Shows/hides the One Identity Manager Service Configuration window.
Help	Select columns...	Opens a dialog window to select columns to be shown additionally in the Job server view.
	Job Server Editor help	Opens the editor help.

Table 183: Meaning of Toolbar Icons

Icon	Meaning
	Adds Job server.
	Deletes Job server.
	Shows HTTP Servers home page.

Icon	Meaning
	Sends configuration.
	Installs service.
	Refreshes display.

Table 184: Meaning of icons in the One Identity Manager Service Configuration Toolbar

Icon	Meaning
	Loads configuration file.
	Saves configuration file.
	Removes configuration file.
	Views file using the Job Service Configuration.
	Displays file in XML format.
	Runs verification test.

Views in the Job Server Editor

The Job Server Editor has several different views for displaying and editing a job server.

Table 185: Editor Views

View	Description
Job server view	This view shows the Job server and its links in hierarchical form. You may add or delete Job servers.
Job server edit view	Use the edit view to edit the properties of selected job servers. A default context menu is available for input fields.
Server functions	In this view, you can assign server function to a Job server.
Configure One Identity Manager Service	Use the One Identity Manager Service configuration to configure the service for a selected Job server. A default context menu is available for input fields.
Machine role	In this view, you can assign machine roles to a Job server.

Table 186: Job Server Context Menu Items

Context Menu Item	Meaning
New	Adds a new Job sever.
Delete	Deletes the selected Job server.
Install service...	This starts the One Identity Manager Service remote installation and configuration wizard.
Select columns...	Opens a dialog window to select columns to be shown additionally in the Job server view.

Editing a Job Server

Each One Identity Manager Service within the network must have a unique queue identifier. The process steps are requested by the Job queue using exactly this queue name. Enter this queue name in the One Identity Manager Service configuration file. Create a corresponding Job server entry for every queue.

To edit a Job server

1. Select the category **Base Data | Installation | Job server** in the Designer.
2. Start the Job Server Editor using the task **Edit job server....**
3. Enter a new Job server using the menu item **Job servers | New**.
- OR -
Select an existing Job server in the Job server overview.
4. Edit the Job server's master data.
5. Select the menu item **View | Server functions** and specify the server functionality.
6. Select the menu item **View | Machine roles** and assign roles to the server.
The machine roles expected by a server function, are already assigned.

Detailed information about this topic

- [Job Server Properties on page 341](#)
- [Server Functions of a Job Server on page 343](#)
- [Job Server Machine Roles on page 344](#)
- [Job Server Statistic Information on page 345](#)

Job Server Properties

Table 187: Job Server Properties

Property	Meaning
Server	Job server name.
Full server name	Full server name in accordance with DNS syntax. Example: <Name of servers>.<Fully qualified domain name>
Server is cluster	Specifies whether the server maps a cluster.
Server belongs to cluster	Cluster to which the server belongs. i NOTE: The properties Server is cluster and Server belongs to cluster are mutually exclusive.
IP address (IPv6)	Internet protocol version 6 (IPv6) server address.
IP address (IPv4)	Internet protocol version 4 (IPv4) server address.
Coding	Character set coding that is used to write files to the server.
Parent Job server	Name of the parent Job server.
Executing server	Name of the executing server. The name of the server that exists physically and where the processes are handled. This input is evaluated when One Identity Manager Service is automatically updated. If the server is handling several queues the process steps are not supplied until all the queues that are being processed on the same server have completed their automatic update.
Queue	Name of the queue to handle the process steps. Each One Identity Manager Service within the network must have a unique queue identifier. The process steps are requested by the job queue using exactly this queue name. The queue identifier is entered in the One Identity Manager Service configuration file.
Server operating system	Operating system of the server. This input is required to resolve the path name for replicating software profiles. Permitted values are "Win32", "Windows", "Linux" and "Unix". If the input is empty, "Win32" is assumed.
Service	One Identity Manager Service user account information. In order to replicate

Property	Meaning
account data	between non-trusted systems (non-trusted domains, Linux server) the One Identity Manager Service user information has to be declared for the servers in the database. This means that the service account, the service account domain and the service account password have to be entered for the server.
One Identity Manager Service installed	<p>Specifies whether a One Identity Manager Service is installed on this server. This option is enabled by the procedure QBM_PJobQueueLoad the moment the queue is called for the first time.</p> <p>The option is not automatically removed. If necessary, you can reset this option manually for servers whose queue is no longer enabled.</p>
Stop One Identity Manager Service	<p>Specifies whether the One Identity Manager Service has stopped. If this option is set for the Job server, the One Identity Manager Service does not process any more tasks.</p> <p>You can make the service start and stop with the appropriate administrative permissions in program "Job Queue Info".</p>
No automatic software update	<p>Specifies whether to exclude the server from automatic software updating.</p> <p>NOTE: Servers must be manually updated if this option is set.</p>
Software update running	Specifies whether a software update is currently being executed.
Extended properties	Additional information about Job servers. Shows the job server UID and the creation data (user, date). These cannot be edited.
Server Function	Server functionality in One Identity Manager. One Identity Manager processes are handled depending on the server function.
Machine role	Role of the Job server in One Identity Manager. Installation packages to be installed on the Job server are found depending on the selected machine role.

Related Topics

- [Server Functions of a Job Server on page 343](#)
- [Job Server Machine Roles on page 344](#)
- [Job Server Statistic Information on page 345](#)
- [JobServiceDestination on page 365](#)
- [Stopping the System \(Emergency Stop\) on page 626](#)

Server Functions of a Job Server

The server function defines the functionality of a server in One Identity Manager. One Identity Manager processes are handled depending on the server function.

 **NOTE:** More server functions may be available depending on which modules are installed.

Table 188: Permitted Server Functions

Server Function	Remark
Update Server	This server executes automatic software updating of all other servers. The server requires a direct connection to the database server that the One Identity Manager database is installed on. The server can execute SQL tasks. The server with the installed One Identity Manager database, is labeled with this functionality during initial installation of the schema.
SQL processing server	This server can process SQL tasks. Several SQL processing servers can be set up to spread the load of SQL processes. The system distributes the generated SQL processes throughout all the Job servers with this server function.
One Identity Manager Service installed	Server on which a One Identity Manager Service is installed.
SMTP host	Server from which the One Identity Manager Service sends email notifications. Prerequisite for sending mails using the One Identity Manager Service is SMTP host configuration.
Default report server	Server on which reports are generated.

Related Topics

- [Job Server Properties on page 341](#)
- [Job Server Machine Roles on page 344](#)
- [Job Server Statistic Information on page 345](#)
- [Machine Roles and Server Functions on page 150](#)

Job Server Machine Roles

Specify which role the Job sever assumes in the One Identity Manager. Installation packages to be installed on the Job server are found depending on the selected machine role.

Table 189: Machine Role and Installation Package Options

Machine role	Description of the Installation Package
Workstation	Contains all basic components for installing tools on an administrative workstation.
	Administrator Contains One Identity Manager administration tools required by default users for fulfilling their task with One Identity Manager. As well as the tools, which ensure basic functionality for working with One Identity Manager, this includes the Manager as the main administration tools.
	Configuration Contains all One Identity Manager tools of the default user and additional programs for configuring the system. These include, for example, Configuration Wizard, Database Compiler, Database Transporter, Crypto Configuration, Designer, Web Designer and configuration tools for the One Identity Manager Service.
	Development & Testing Contains the One Identity Manager tools for developing and testing custom scripts and forms, for example, the System Debugger.
	Monitoring Contains One Identity Manager programs for monitoring the system status, for example Job Queue Info.
Documentation	Contains One Identity Manager documentation in different languages.
Server	Contains all the basic components for setting up a server.
	Job server Contains the One Identity Manager Service and basic processing components. Additional machine roles contain connectors for synchronizing individual target systems.

NOTE: The machine roles "API" and "Web" are shown additionally in the category **Base Data | Installation | Machine roles**. These are reserved for internal user and cannot be changed or assigned.

Related Topics

- [Job Server Properties on page 341](#)
- [Server Functions of a Job Server on page 343](#)
- [Job Server Statistic Information on page 345](#)
- [Installing the One Identity Manager Service on the Job Server on page 346](#)
- [Machine Roles and Server Functions on page 150](#)

Job Server Statistic Information

Table 190: Configuration Parameter for Calculating Statistic Information

Configuration parameter	Meaning
Common\JobQueueStats	If this configuration parameter is set, One Identity Manager Service statistic data is written to the database (table JobQueueStats).
Common\JobQueueStats\MaxAge	This configuration parameter specifies how many days the statistic data will be kept in the database.

This Job server statistic data is evaluated and creates a basis for configuration recommendations for Job server load intervals. The data for the last 100 days is included in the calculation of the configuration recommendations. You should take these recommendations into account when configuring One Identity Manager Service.

To find statistics

- Set the configuration parameter "Common\JobQueueStats" in the Designer.
- Set the Designer configuration parameter "Common\JobQueueStats\MaxAge" and enter the retention period for the statistics (in days).

New statistics are created for the Job server by each action in the Job queue (such as, adding, changing or deleting processes). The DBQueue Processor task QBMJobQueueStatsShrink compresses the statistics. The compression takes place for every hour prior to the current hour.

To display Job server statistics

1. Select the category **Base Data | Installation | Job server** in the Designer.
2. Start the Job Server Editor using the task **Edit job server**.
3. Select the Job server to edit in the Job server overview.
4. Select the columns with statistic information using **Select columns...** in the context menu.

These columns are highlighted in the color in the view.

Table 191: Columns for Displaying Statistic Information

Column	Name	Meaning
AverageLoad	Average processes/hour	Average number of processes per hour.
MaxLoad	Maximum processes/hour	Maximum number of processes per hour.
LoadDuration	Recommended load interval (seconds)	Configuration recommendation for the parameter "StartInterval" in the One Identity Manager Service configuration.
StatisticsDuration	Recommended statistic interval (seconds)	Configuration recommendation for the parameter "StatisticInterval" in the One Identity Manager Service configuration.

Installing the One Identity Manager Service on the Job Server

You have the option to install certain Job servers remotely in the Job Server Editor. The remote installation wizard executes the following steps:

- Installs the One Identity Manager Service components.
- Configures the One Identity Manager Service.
- Starts the One Identity Manager Service.

Prerequisites for Remote Installation

- The Job server is entered in the database
- There is a user account with sufficient permissions for installing the One Identity Manager Service.
- Remote installation is only supported within a domain or a trusted domain.

To install the One Identity Manager Service remotely

1. Select the category **Base Data | Installation | Job server** in the Designer.
2. Start the Job Server Editor using the task **Edit job server**.
3. Select the Job server to edit in the Job server overview.
4. Select the **Job server | Install service...** from the menu.

This starts the One Identity Manager Service remote installation wizard.

5. Click **Next**.

6. Enter the One Identity Manager Service configuration settings.

Initial configuration of the service is already predefined for the database connection. To use this template, enter the connection data for process collection. In order to extend the configuration, each configuration section of the One Identity Manager Service is listed in the module list.

- SQL Server

Select **Process collection | sqlProvider**, click **Connection string**, click **Edit** and enter the following data.

Table 192: SQL Server Database Connection Data

Data	Description
Server	Database server.
Windows authentication	Specifies whether Windows authentication is used. This type of authentication is not recommended. If you decide to use it anyway, ensure that your environment supports Windows authentication.
User	Database user.
Password	Database user password.
Database	Database.

- Oracle Database

Select **Process collection | OracleJobProvider**, click **Connection string**, click **Edit** and enter the following data.

Table 193: Oracle Database Connection Data

Data	Description
Direct access (without Oracle client)	Set this option for direct access. Deactivate this option for access via Oracle Clients. Which connection data is required, depends on how this option is set.
Server	Database server.
Port	Oracle instance port.
Service name	Service name.
User	Oracle database user.

Data	Description
Password	Database user password.
Data source	TNS alias name from TNSNames.ora.

7. Click **Next**.
8. Specify the installation source. Select the directory with the installation files.
9. If the database is encrypted, select the file with the private key.
10. Enter the installation data.

Table 194: Installation Data

Data	Description
Computer	Server on which to install and start the service from.
Service account.	User account data for the One Identity Manager Service. For more information about user account requirements, see the One Identity Manager Installation Guide.
	<p>To enter a user account for the One Identity Manager Service</p> <ul style="list-style-type: none"> • Set the option Local system account. This starts the One Identity Manager Service under the account "NT AUTHORITY\SYSTEM". - OR - • Enter user account, password and password confirmation.
Installation account	Data for the administrative user account to install the service.
	<p>To enter an administrative user account for installation</p> <ul style="list-style-type: none"> • Enable the option Current user. This uses the user account of the current user. - OR - • Enter user account, password and password confirmation.

11. Click **Next**.

Installation of the service occurs automatically and may take some time.

NOTE: The One Identity Manager Service is entered in the server's service administration with the name "One Identity Manager Service".

12. Click **Close** to end the workflow wizard.

NOTE: Using the menu item **Job server | Start HTTP request...**, you can call up the One Identity Manager Service HTTP server for a Job server. This displays the various One Identity Manager Service services.

NOTE: If you are working with an encrypted One Identity Manager database, see the advice for working with an encrypted database in the One Identity Manager Installation Guide.

Related Topics

- Declaring the Job Server on page 337
- One Identity Manager Service Configuration on page 353

Customizing the One Identity Manager Service Configuration for a Job Server

This configuration is already created when the One Identity Manager Service is installed. Use the Job Server Editor to modify each configuration setting.

NOTE: You can also adjust the configuration settings in the program "Job Service Configuration".

To modify the One Identity Manager Service configuration

1. Load the Job server configuration into the database. Configure and enable the schedule "Get configuration file from the Job server and write in the Job server configuration" in the Designer.
2. Modify the Job Server Editor configuration.
3. Deploy the modified configuration to the Job server.

To modify the One Identity Manager Service configuration on a Job server

1. Select the category **Base Data | Installation | Job server** in the Designer.
2. Start the Job Server Editor using the task **Edit job server**.
3. Enable the **One Identity Manager Service configure** view.
4. Select the Job server to edit in the Job server overview.
5. Edit the configuration settings.
TIP: Use the and buttons to change the configuration data.
6. Save the configuration using .
7. Use the button to test the configuration.
8. Deploy the modified configuration to the Job server using **Job server | Deploy Job server configuration** from menu.

This generates a process, which updates the configuration file on the Job server.

NOTE: Using the menu item **Job server | Start HTTP request...**, you can call up the One Identity Manager Service HTTP server for a Job server. This displays the various One Identity Manager services.

Related Topics

- [One Identity Manager Service Configuration on page 353](#)
- [Template for the Configuration File on page 350](#)
- [Selecting Modules Types and Editing the Parameters on page 350](#)
- [Configuration File Verification Test on page 352](#)

Template for the Configuration File

 **NOTE:** The templates are only available in the program "Job Service Configuration".

Templates for a One Identity Manager Service configuration are supplied.

- SQL Server direct

This template already contains the most important modules with settings for a simple One Identity Manager Service configuration with a direct connection to a SQL Server. You can load the template using the menu item **Templates | SQL server direct**. After loading, the configuration the template needs to be modified as required.

- Oracle Server direct

This template already contains the most important modules with settings for a simple One Identity Manager Service configuration with a direct connection to a Oracle Database. You can load the template using the menu item **Templates | SQL server direct**. After loading, the configuration the template needs to be modified as required.

Related Topics

- [Customizing the One Identity Manager Service Configuration for a Job Server on page 349](#)
- [Selecting Modules Types and Editing the Parameters on page 350](#)
- [Configuration File Verification Test on page 352](#)
- [One Identity Manager Service Configuration on page 353](#)

Selecting Modules Types and Editing the Parameters

Each of the configuration sections are listed in the module list. A selection of module types is available for certain modules.

To select a module type

1. Click on the module in the module list.
2. Use **Insert** to open the module type menu.
3. Select the module type you want from the list and insert it with the **OK** button.

To change the name of a module type

1. Click on the module in the module list.
2. Select the module type and click **Rename**.
3. Change the name of the module type.
4. Press return.

To delete a module type

1. Click on the module in the module list.
2. Select the module type and click **Delete**.

To edit a parameter value

1. Select the parameter in the **Properties** column.
2. Click **Edit**.

When a item is selected in the module list, all possible parameters and their values are displayed. You can change some values by clicking in input field or on the option button in the **Value** column.

 **NOTE:** The parameter description in each module contains the parameter name, in brackets, which is used in the configuration file.

The following icons are used:

Table 195: Meaning of the Icons for the Module Parameters

Icon	Meaning
	The parameter is preset with a standard value. The value is passed as a string.
	Compulsory input. The parameter must be altered as required. The value is passed as a string.
	The parameter is preset with a standard value. The value is passed as an integer.
	The parameter can be activated and deactivated.
	This parameter is added during run-time. The One Identity Manager Service does not need to be restarted.
	The parameter takes effect after the One Identity Manager Service is restarted.

Related Topics

- [Customizing the One Identity Manager Service Configuration for a Job Server on page 349](#)
- [Template for the Configuration File on page 350](#)
- [Configuration File Verification Test on page 352](#)
- [One Identity Manager Service Configuration on page 353](#)

Configuration File Verification Test

The verification test ensures that the minimum requirements for a configuration file are met.

To start the verification test

- Use the button to test the configuration.
Errors and warnings are sent to a message window.

Table 196: Verification Test Error Output

Error	Output
No Job provider found.	error
No Logwriter found.	error
No input in compulsory field.	error
No Job destination found.	warning
No plugins found.	warning

Related Topics

- [Customizing the One Identity Manager Service Configuration for a Job Server on page 349](#)
- [Template for the Configuration File on page 350](#)
- [Selecting Modules Types and Editing the Parameters on page 350](#)
- [One Identity Manager Service Configuration on page 353](#)

One Identity Manager Service Configuration

The server service "One Identity Manager Service" ensures distribution in the network of data managed in the One Identity Manager database. The One Identity Manager Service performs data synchronization between the database and any connected target systems and executes actions at the database and file level. The One Identity Manager Service retrieves process steps from the JobQueue. Process steps are executed by process components. One Identity Manager Service also creates an instance of the required process component and passes the parameters to the process step. Decision logic monitors the execution of the process steps and determines how processing should continue depending on the results of the executed process components. The One Identity Manager Service enables parallel processing of process steps because it can create several instances of process components.

A Job provider function makes a Job destination process step available within the One Identity Manager Service. The Job destination function handles the process steps and returns a result to the Job provider. The Job provider evaluates the result.

The combination of a Job provider on one server and a Job destination on another server is called a "Job gate". The Job provider and Job destination are configured within the Jobgate such that they can communicate with each other.

Figure 42: One Identity Manager Service Operating Mode

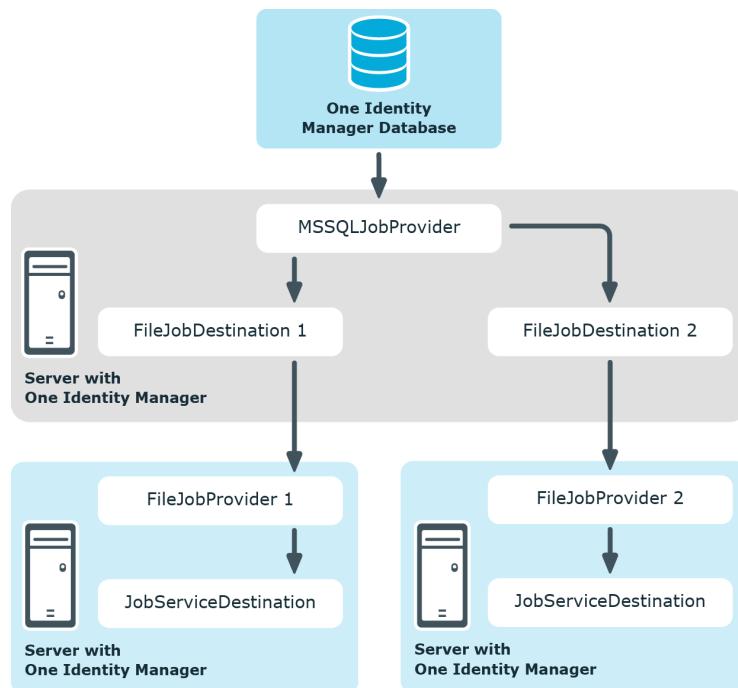


Table 197: One Identity Manager Service Provider

Provider	Description
MSSQLJobProvider	The MSSQLJobProvider collects process steps from the One Identity Manager database under SQL Server and sends them to a Job destination.
OracleJobProvider	The OracleJobProvider collects process steps from the One Identity Manager database under Oracle Database and sends them to a Job destination.
FileJobProvider	The FTPJobProvider reads process queries and results from files and writes them to file. These files can be processed by FileJobGate (File or FTP Job destination). The data is transferred using these files.
FTPJobProvider	The FTPJobProvider is based on the functionality of the FileJobProvider. The FTPJobProvider reads process queries and results from files and writes them to file. After the files have been created in the local directory, the FTPJobProvider connects to the FTP server and transfers all the files. A connection is also made to the FTP Server when it gets a signal and the data is collected.
HTTPJobProvider	The HTTPJobProvider receives process steps from a parent Job server. The data transfer is done by HyperText Transfer Protocol (HTTP).
AppServerJobProvider	The AppServerJobProvider gets process steps from the application server and sends them to a Job destination.

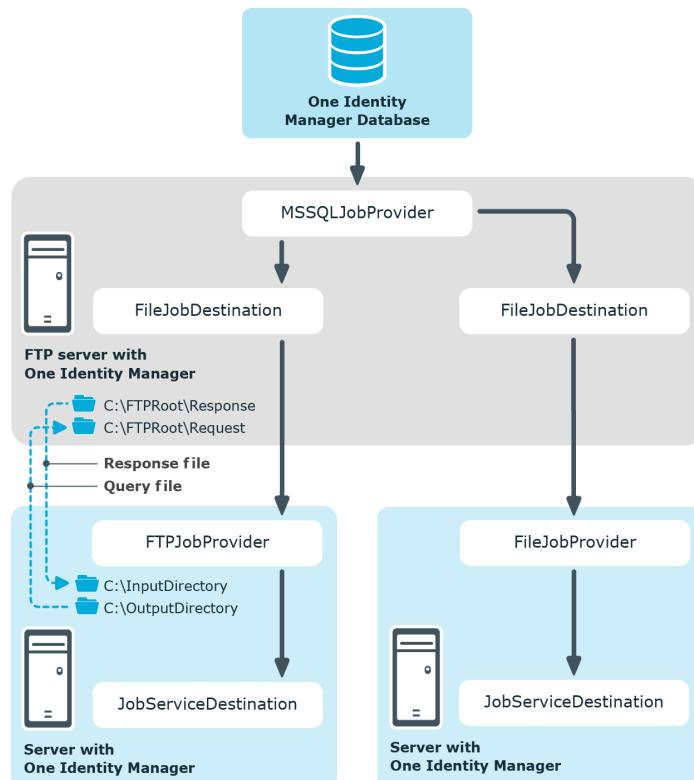
Table 198: One Identity Manager Service Job Destinations

JobDestination	Description
JobServiceDestination	The JobServiceDestination is the One Identity Manager Service component that actually deals with handling process steps. It requests the process steps from the Job provider, processes them with the process component and returns the result.
FileJobDestination	The FileJobDestination processes the process steps that are queued by the FileJobGate (FileJobProvider or FTPJobProvider) and returns the results to the Job provider.
FTPJobDestination	The FTPJobDestination handles the process steps that are queued in the FileJobGate (FileJobProvider or FTPJobProvider) and returns the results to the Job provider.
HTTPJobDestination	An HTTPJobDestination sends process steps to a follow-on Job server. The data transfer is in HyperText Transfer Protocol.

Table 199: One Identity Manager Service Job Gates

Jobgate	Description
HTTPJobGate	Consists of HTTPJobProvider and HTTPJobDestination.
FileJobGate	Consists of FileJobProvider, FileJobDestination, FTPJobProvier and FTPJobDestination. JobProvider and JobDestinations can be combined with each other.

Figure 43: Example Configuration for FileJobGate



Detailed information about this topic

- [One Identity Manager Service Configuration Files on page 356](#)
- [Process Collection Module on page 357](#)
- [The Jobdestination Module on page 365](#)
- [The Configuration Module on page 373](#)
- [The Logwriter Module on page 374](#)
- [The Dispatcher Module on page 377](#)
- [The Connection Module on page 378](#)
- [The HTTP Authentication Module on page 379](#)

- [The Plug-ins Module on page 379](#)
- [Module File with Private Key on page 383](#)

One Identity Manager Service Configuration Files

Configuration One Identity Manager Service and its plug-ins with a configuration file. The file has to reside in the same directory as the file viNetworkService. The configuration file is necessary both for One Identity Manager Service on a windows based operating system and for the Linux daemon.

Two configuration file formats are supported:

- Jobservice.cfg

Jobservice.cfg is an XML configuration file in our own format. The advantage of this file is that run-time loading is supported.

- viNetworkService.exe.config

The viNetworkService.exe.config file is the default configuration file for .NET exes and has the specified format.

The system initially searches for the parameter in the configuration file Jobservice.cfg in order to determine the setups. If the parameter is not found, the file viNetworkService.exe is automatically used. Thus the One Identity Manager Service can only work with the configuration file viNetworkService.exe.config.

Edit the One Identity Manager Service configuration in the category **Base Data | Installation | Job server** in the Designer or with the program "Job Service Configuration".

There is one unique section in the file for each of the different modules in One Identity Manager Service.

Table 200: One Identity Manager Service Modules

Module	Description
Process collection	Specify the Job provider in this module.
JobDestination	Specify Job destinations in this module.
Configuration	Standard configuration settings for One Identity Manager Service are in this module.
LogWriter	This module writes One Identity Manager Service messages to a log file.
Dispatcher	Use this module to configure the One Identity Manager Service as a dispatcher. The process requests from the child Job server are buffered, processed and forwarded.

Module	Description
Connection	With this module you can set special configuration settings for the behavior of the One Identity Manager Service.
HTTP authentication module	Use this module to specify how authentication works on an HTTP server so that extended services can be accessed, for example, displaying the log file or the status display.
Plugins	Specify which plugins should be installed in this module.
File with the private key	In this module, you provide the data for files with a private key. Use this module if you are working with more than one private key.

Detailed information about this topic

- [Customizing the One Identity Manager Service Configuration for a Job Server on page 349](#)
- [One Identity Manager Service Configuration on page 353](#)
- [Process Collection Module on page 357](#)
- [The Jobdestination Module on page 365](#)
- [The Configuration Module on page 373](#)
- [The Logwriter Module on page 374](#)
- [The Dispatcher Module on page 377](#)
- [The Connection Module on page 378](#)
- [The HTTP Authentication Module on page 379](#)
- [The Plug-ins Module on page 379](#)
- [Module File with Private Key on page 383](#)
- [One Identity Manager Service Configuration Files on page 660](#)

Process Collection Module

In this module you define the Job providers. The Job provider supplies a Job destination process step and evaluates the result. The following module types may be selected:

- [MSSQLJobProvider](#)
- [OracleJobProvider](#)
- [FileJobProvider](#)
- [FTPJobProvider](#)
- [HTTPJobProvider](#)
- [AppServerJobProvider on page 364](#)

You can configure any number of Job providers in one configuration file. The associated configuration sections are determined by name. Therefore, you should rename Job providers that are added.

Related Topics

- [Selecting Modules Types and Editing the Parameters on page 350](#)

MSSQLJobProvider

The MSSQLJobProvider handles One Identity Manager database process requests on an SQL Server server.

Following parameters are available:

- Connection Parameter (ConnectionString)

This parameter defines the access data for the database server and the database to be used.

Table 201: SQL Server Database Connection Data

Data	Description
Server	Database server.
Windows authentication	Specifies whether Windows authentication is used. This type of authentication is not recommended. If you decide to use it anyway, ensure that your environment supports Windows authentication.
User	Database user.
Password	Database user password.
Database	Database.

- Max. number of pending requests (RequestQueueLimit)

The MSSQLJobProvider internally caches process requests that are queried by the database. This value defines the maximum number of cache entries. The default value is 1000.

- Max. number of pending results (ResultQueueLimit)

The MSSQLJobProvider internally caches the process results that are written to the database. This value defines the maximum number of cache entries. The default value is 10000.

OracleJobProvider

The OracleJobProvider handles One Identity Manager database process requests on an Oracle Database server.

Following parameters are available:

- Connection Parameter (ConnectionString)

This parameter defines the access data for the database server and the database to be used.

Table 202: Oracle Database Connection Data

Data	Description
Direct access (without Oracle client)	Set this option for direct access. Deactivate this option for access via Oracle Clients. Which connection data is required, depends on how this option is set.
Server	Database server.
Port	Oracle instance port.
Service name	Service name.
User	Oracle database user.
Password	Database user password.
Data source	TNS alias name from TNSNames.ora.

- Max. number of pending requests (RequestQueueLimit)

The OracleJobProvider internally caches process requests that are queried by the database. This value defines the maximum number of cache entries. The default value is 1000.

- Max. number of pending results (ResultQueueLimit)

The OracleJobProvider internally caches the process results that are written to the database. This value defines the maximum number of cache entries. The default value is 10000.

FileJobProvider

In the FileJobProvider transfers data using files. Process requests and results are written to file or read from file. These files can be processed by FileJobDestination.

Following parameters are available:

- Backup transferred files (BackupFiles)

If this option is set, all the file are moved to a directory "Backup" irrespective of errors. In the default case (not set) only files with errors are saved.

- Check file index (CheckInputIndex)

If this option is set, the file name index is checked to see if has increased in size. Files with the same or a lower index are not processed. This option is not set by default.

- Max.number of process trees in one transfer file (MaxListCount)

This setting specifies the maximum number of process steps that can be grouped together in one file. This allows limiting of the file size.

- Use encoding (UseEncryption)

The data is encoded when written to file.

 **NOTE:** The encryption setting must be configured the same in the Job provider and associated Job destination.

- Notification procedures (EventTypes)

The FileJobProvider supports three different methods of acquiring information about new data. The notification methods can be combined when separated by commas.

Table 203: Supported Notification Procedures

Method	Description
Timer	Querying newly added data is done in a specified time interval.
HTTP	The provider queries the parent Jon server over HTTP and processes the added data after it has answered.
FSEvent	Queries newly added data after a file system event.

Example:

TIMER,FSEVENT

- HTTP notification destination computer (HostName)

Enter the name of the target computer here that will receive the queries if the "HTTP" notification methods are used.

- Port for HTTP notification (Port)

Enter the transfer port here if the "HTTP" notification method are used.

- Monitoring interval for input directory (TimerInterval)

Enter the time interval in milliseconds if the "timer" notification method is used.

- Directory for receiving input (InputDirectory)

The module reads and processes the process files (*.fjg) in this directory.

NOTE: It is necessary to ensure that the Job provider and associated Job destination use the same directory. The input and output directories are correspondingly reversed.

- Destination directory (OutputDirectory)
The processed files are written to this directory.

- List of subdirectories (SubDirectories)

A list of directory names separated by a pipe character "|" can be entered here. All the directories are then monitored and processed correspondingly. The following directory structure is expected:

SubDirectories = "ServerA|ServerB"

...

Request

 ServerA

 ServerB

Response

 ServerA

 ServerB

where Request and Response are directories enter in the parameters "InputDirectory" and "OutputDirectory".

NOTE: Only the "Timer" notification method can be used. The notification methods "HTTP" and "FSEvent" are not available!

- Automatic identification of subdirectories (AutoSubDirectories)

If this option is enabled, the module automatically processes all the files in the subdirectories. Processing is not recursive.

Related Topics

- [FTPJobProvider on page 361](#)
- [FileJobDestination on page 368](#)
- [FTPJobDestination on page 370](#)

FTPJobProvider

After the files have been created in the local directory, the FTPJobProvider connects to the FTP server and transfers all the files. After a signal, a connection is set up to the FTP Server and the data is transferred. The directories Request and Response are expected to be found on the FTP Server. The names of these directories are fixed and cannot be changed. The software components (Job provider/Job destination) deposit or collect the files from here. The FTP user requires the necessary access rights to create, rename and delete files.

Following parameters are available:

- Backup transferred files (BackupFiles)

If this option is set, all the file are moved to a directory "Backup" irrespective of errors. In the default case (not set) only files with errors are saved.

- Check file index (CheckInputIndex)

If this option is set, the file name index is checked to see if has increased in size. Files with the same or a lower index are not processed. This option is not set by default.

- Max.number of process trees in one transfer file (MaxListCount)

This setting specifies the maximum number of process steps that can be grouped together in one file. This allows limiting of the file size.

- Use encoding (UseEncryption)

The data is encoded when written to file.

 **NOTE:** The encryption setting must be configured the same in the Job provider and associated Job destination.

- Notification procedures (EventTypes)

The FileJobProvider supports three different methods of acquiring information about new data. The notification methods can be combined when separated by commas.

Table 204: Supported Notification Procedures

Method	Description
Timer	Querying newly added data is done in a specified time interval.
HTTP	The provider queries the parent Jon server over HTTP and processes the added data after it has answered.
FSEvent	Queries newly added data after a file system event.

- HTTP notification destination computer (HostName)

Enter the name of the target computer here that will receive the queries if the "HTTP" notification methods are used.

- Port for HTTP notification (Port)

Enter the transfer port here if the "HTTP" notification method are used.

- Monitoring interval for input directory (TimerInterval)

Enter the time interval in milliseconds if the "timer" notification method is used.

- Directory for receiving input (InputDirectory)

The module reads and processes the process files (*.fjg) in this directory.

NOTE: It is necessary to ensure that the Job provider and associated Job destination use the same directory. The input and output directories are correspondingly reversed.

- Destination directory (OutputDirectory)
The processed files are written to this directory.

- List of subdirectories (SubDirectories)

A list of directory names separated by a pipe character "|" can be entered here. All the directories are then monitored and processed correspondingly. The following directory structure is expected:

SubDirectories = "ServerA|ServerB"

...

Request

 ServerA

 ServerB

Response

 ServerA

 ServerB

where Request and Response are directories enter in the parameters "InputDirectory" and "OutputDirectory".

NOTE: Only the "Timer" notification method can be used. The notification methods "HTTP" and "FSEvent" are not available!

- Automatic identification of subdirectories (AutoSubDirectories)

If this option is enabled, the module automatically processes all the files in the subdirectories. Processing is not recursive.

- FTP server (FTPServer)

Enter the name or the IP address of the FTP Server.

- FTP port (FTPPort)

If the FTP server does not use the default port 21 for FTP transfer, you can enter the appropriate port.

- FTP user account (FTPUser)

Enter the user account with which the FTPJobProvider logs on onto the FTP Server.

- FTP password (FTPPassword)

Enter the user account password for the FTP login.

Related Topics

- [FileJobProvider on page 359](#)
- [FileJobDestination on page 368](#)

- [FTPJobDestination on page 370](#)

HTTPJobProvider

The HTTPJobProvider receives process steps from a parent Job server. The data transfer is carried out by HTTP.

Following parameters are available:

- Receiver port (ParentPort)
Enter the HTTP port of the parent Job server.
- Receiver server (ParentServer)
Enter the DNS name or the IP address of the parent Job server.
- Number of retries (Retries)
This value defines how many time the module retries the data transfer if it fails.
- Time interval between retries (RetryDelay)
This time delay defines how long a module waits after a failed process transfer before retrying.

Timeout format:

day.hour:minutes:seconds

- Target domain (RemoteDomain)
Enter the user account domain on the remote HTTP server.
- Target user account (RemoteUser)
Enter the user account that the HTTPJobProvider uses to log onto the HTTP server.
- Target password (RemotePassword)
Enter the password that the user account uses to log onto the HTTP server.

Related Topics

- [HTTPJobDestination on page 372](#)

AppServerJobProvider

The AppServerJobProvider gets process steps from the application server and sends them to a Job destination.

Following parameters are available:

- Authentication data (AuthenticationString)

Select the authentication module. Depending on the authentication module, other data may be required, for example, user and password.

- Max. number of requests (RequestQueueLimit)

The AppServerJobProvider internally caches the process requests that are queried. This value defines the maximum number of cache entries. The default value is 1000.

- Max. number of pending results (ResultQueueLimit)

The AppServerJobProvider caches process results internally. This value defines the maximum number of cache entries. The default value is 10000.

- Connection Parameter (ConnectionString)

Enter the application server's web address.

Related Topics

- [One Identity Manager Authentication Module on page 74](#)

The Jobdestination Module

In this module you can define Job destinations. This handles the process steps and returns an result to the Job provider. The following module types may be selected:

- [JobServiceDestination](#)
- [FileJobDestination](#)
- [FTPJobDestination](#)
- [HTTPJobDestination](#)

Within a configuration file you can configure as many Job destinations as you wish. The associated configuration sections are determined by name. Therefore the Job destinations that are added can be renamed.

Related Topics

- [Selecting Modules Types and Editing the Parameters on page 350](#)

JobServiceDestination

The JobServiceDestination is the One Identity Manager Service module that deals with the handling process steps. A JobServiceDestination requests the process steps from the Job provider, processes them using process components and returns the result.

Following parameters are available:

- External slot count (ExternalSlots)

This parameter specifies how many external processes (`StudioProcessor.exe`) the One Identity Manager Service opens to handle process components.

- Environment variables for external slots (ExternalSlotEnvironment)

This parameter contains a list of environment variables that you should set for external slot processes. Enter the variables in a pipe ('|') delimited list.

`Variable1=value1|Variable2=value2...`

- External 32-bit slot count (ExternalSlots32)

This parameter specifies how many external processes in 32-bit memory (`StudioProcessor.exe`) the One Identity Manager Service opens to handle process components.

- Environment variables for external 32-bit slots (ExternalSlotEnvironment32)

This parameter contains a list of environment variables that you should set for external 32-bit slot processes. Enter the variables in a pipe ('|') delimited list.

`Variable1=value1|Variable2=value2...`

- Internal slot count (InternalSlots)

This parameter specifies how many internal slots One Identity Manager Service makes available for internal process components processing.

- File with private key (PrivateKey)

Enter the file with the encryption data. The default file is `private.key`. The encryption file has to be in the installation directory of all servers with One Identity Manager Service. If the One Identity Manager Service finds a private key on start up, it places it in the per-user key container and deletes the file from the hard drive.

Use the program "Crypto Configuration" in order to create an encryption file and to encrypt the database information.

NOTE: If you are working with an encrypted One Identity Manager database, see the advice for working with an encrypted database in the One Identity Manager Installation Guide.

- Encryption scheme (EncryptionScheme)

Specify the encryption scheme you want to use.

Table 205: Encryption scheme

Method	Description
RSA	RSA encryption with AES for large data (default).
FIPSCompliantRSA	FIPS certified RSA with TripleAES for large data. This method is used if encryption must match the FIPS 1040-2 standard. The local security policy "Use FIPS compliant algorithms for encryption, hashing, and signing" must be enabled.

- Job provider ID (ProviderID)

Enter the name of the Job provider that will be used if more than one Job provider is being processed by the One Identity Manager Service. If this is empty the first Job provider is used.

- Private key ID (PrivateKeyId)

Identifier of the private key. If no ID is given, the file "private.key" is searched for. Use this parameter if you work with several private keys, for example, if One Identity Manager Service data must be exchanged between two encrypted One Identity Manager databases. Enter the private key in the module, "File with private key". If One Identity Manager only uses one encrypted database, you can enter the key file in the parameter "File with private key (PrivateKey)" as an alternative.

- Queue (Queue)

Each One Identity Manager Service within the network must have a unique queue identifier. The process steps are requested by the Job queue using exactly this queue name. Enter this queue name in the One Identity Manager Service configuration file. Create a corresponding Job server entry for every queue.

- Timeout for process queries (RequestTimeout)

This input specifies a time after which a process request can be said to have failed and is sent again.

Timeout format:

day.hour:minutes:seconds

- Process query interval (StartInterval)

This property defines a time interval in which One Identity Manager Service can request new process steps. Click on this icon to discard the change. The default value is 90 seconds. Suggestions for configuring the time interval are calculated from Job server statistical data.

- Interval of time allowed for statical calculations (StatisticInterval)

This property defines the time interval (in seconds) in which One Identity Manager Service's processing speed statistics are supplied to the database. Click on this icon to discard the change. The default value is set to 4 times the process request interval. Suggestions for configuring the time interval are calculated from Job server statistical data.

- Max. reuse of external processors (MaxExternalSlotReuse)

This value specifies how many times an external processor can be reused before the process is unloaded and restarted. The default value "0" means that the process is not unloaded until it is no longer in use. The default value is 1000.

Related Topics

- [Declaring the Job Server on page 337](#)
- [Job Server Properties on page 341](#)

- [Job Server Statistic Information on page 345](#)
- [Module File with Private Key](#)

FileJobDestination

The FileJobDestination processes the process steps that are queued by the FileJobGate (FileJobProvider or FTPJobProvider) and returns the results to the Job provider.

Following parameters are available:

- Backup transferred files (BackupFiles)

If this option is set, all the file are moved to a directory "Backup" irrespective of errors. In the default case (not set) only files with errors are saved.

- Check file index (CheckInputIndex)

If this option is set, the file name index is checked to see if has increased in size. Files with the same or a lower index are not processed. This option is not set by default.

- Max.number of process trees in one transfer file (MaxListCount)

This setting specifies the maximum number of process steps that can be grouped together in one file. This allows limiting of the file size.

- Use encoding (UseEncryption)

The data is encoded when written to file.

NOTE: The encryption setting must be configured the same in the Job provider and associated Job destination.

- Notification procedures (EventTypes)

The FileJobProvider supports three different methods of acquiring information about new data. The notification methods can be combined when separated by commas.

Table 206: Supported Notification Procedures

Method	Description
Timer	Querying newly added data is done in a specified time interval.
HTTP	The provider queries the parent Jon server over HTTP and processes the added data after it has answered.
FSEvent	Queries newly added data after a file system event.

Example:

TIMER,FSEVENT

- HTTP notification destination computer (HostName)
Enter the name of the target computer here that will receive the queries if the "HTTP" notification methods are used.
- Port for HTTP notification (Port)
Enter the transfer port here if the "HTTP" notification method are used.
- Monitoring interval for input directory (TimerInterval)
Enter the time interval in milliseconds if the "timer" notification method is used.
- Directory for receiving input (InputDirectory)
The module reads and processes the process files (*.fjg) in this directory.

NOTE: It is necessary to ensure that the Job provider and associated Job destination use the same directory. The input and output directories are correspondingly reversed.

- Destination directory (OutputDirectory)
The processed files are written to this directory.
- List of subdirectories (SubDirectories)
A list of directory names separated by a pipe character "|" can be entered here. All the directories are then monitored and processed correspondingly. The following directory structure is expected:

SubDirectories = "ServerA|ServerB"

...

Request

 ServerA

 ServerB

Response

 ServerA

 ServerB

where Request and Response are directories enter in the parameters "InputDirectory" and "OutputDirectory".

NOTE: Only the "Timer" notification method can be used. The notification methods "HTTP" and "FSEvent" are not available!

- Automatic identification of subdirectories (AutoSubDirectories)
If this option is enabled, the module automatically processes all the files in the subdirectories. Processing is not recursive.
- Job provider ID (ProviderID)
If several Job providers are processed by the One Identity Manager Service, enter the name of the Job provider to use. If this is empty the first Job provider is used.

Related Topics

- [FileJobProvider on page 359](#)
- [FTPJobProvider on page 361](#)
- [FTPJobDestination on page 370](#)

FTPJobDestination

FTPJobDestination handles the process steps that are queued in the FileJobGate (FileJobProvider or FTPJobProvider) and returns the results to the Job provider.

Following parameters are available:

- Backup transferred files (BackupFiles)

If this option is set, all the file are moved to a directory "Backup" irrespective of errors. In the default case (not set) only files with errors are saved.

- Check file index (CheckInputIndex)

If this option is set, the file name index is checked to see if has increased in size. Files with the same or a lower index are not processed. This option is not set by default.

- Max.number of process trees in one transfer file (MaxListCount)

This setting specifies the maximum number of process steps that can be grouped together in one file. This allows limiting of the file size.

- Use encoding (UseEncryption)

The data is encoded when written to file.

 **NOTE:** The encryption setting must be configured the same in the Job provider and associated Job destination.

- Notification procedures (EventTypes)

The FileJobProvider supports three different methods of acquiring information about new data. The notification methods can be combined when separated by commas.

Table 207: Supported Notification Procedures

Method	Description
Timer	Querying newly added data is done in a specified time interval.
HTTP	The provider queries the parent Jon server over HTTP and processes the added data after it has answered.
FSEvent	Queries newly added data after a file system event.

Example:

TIMER,FSEVENT

- HTTP notification destination computer (HostName)
Enter the name of the target computer here that will receive the queries if the "HTTP" notification methods are used.
- Port for HTTP notification (Port)
Enter the transfer port here if the "HTTP" notification method are used.
- Monitoring interval for input directory (TimerInterval)
Enter the time interval in milliseconds if the "timer" notification method is used.
- Directory for receiving input (InputDirectory)
The module reads and processes the process files (*.fjg) in this directory.

NOTE: It is necessary to ensure that the Job provider and associated Job destination use the same directory. The input and output directories are correspondingly reversed.

- Destination directory (OutputDirectory)
The processed files are written to this directory.
- List of subdirectories (SubDirectories)
A list of directory names separated by a pipe character "|" can be entered here. All the directories are then monitored and processed correspondingly. The following directory structure is expected:

SubDirectories = "ServerA|ServerB"

...

Request

 ServerA

 ServerB

Response

 ServerA

 ServerB

where Request and Response are directories enter in the parameters "InputDirectory" and "OutputDirectory".

NOTE: Only the "Timer" notification method can be used. The notification methods "HTTP" and "FSEvent" are not available!

- Automatic identification of subdirectories (AutoSubDirectories)
If this option is enabled, the module automatically processes all the files in the subdirectories. Processing is not recursive.
- Job provider ID (ProviderID)
If several Job providers are processed by the One Identity Manager Service, enter the name of the Job provider to use. If this is empty the first Job provider is used.

- **FTP server (FTPServer)**
Enter the name or the IP address of the FTP Server.
- **FTP port (FTPPort)**
If the FTP server does not use the default port 21 for FTP transfer, you can enter the appropriate port.
- **FTP user account (FTPUser)**
Enter the user account with which the FTPJobProvider logs on onto the FTP Server.
- **FTP password (FTPPassword)**
Enter the user account password for the FTP login.

Related Topics

- [FileJobProvider on page 359](#)
- [FTPJobProvider on page 361](#)
- [FileJobDestination on page 368](#)

HTTPJobDestination

An HTTPJobDestination sends process steps to a child Job server. The data transfer is carried out by HTTP.

Following parameters are available:

- **Recipient port (ChildPort)**
Enter the HTTP-Port of the child server.
- **Job provider ID (ProviderID)**
Enter the name of the Job provider that will be used if more than one Job provider is being processed. If this is empty the first Job provider is used.
- **Number of retries (Retries)**
This value defines how many time the module retries the data transfer if it fails.
- **Time interval between retries (RetryDelay)**
This time delay defines how long a module waits after a failed process transfer before retrying.
Timeout format:
day.hour:minutes:seconds
- **Target domain (RemoteDomain)**
Enter the user account domain on the remote HTTP server.
- **Target user account (RemoteUser)**
Enter the user account that the HTTPJobProvider uses to log onto the HTTP server.

- Target password (RemotePassword)

Enter the password that the user account uses to log onto the HTTP server.

Related Topics

- [HTTPJobProvider on page 364](#)

The Configuration Module

The standard One Identity Manager Service configuration settings are specified in this module.

Following parameters are available:

- Verbose logging (VerboseLogging)

Set this parameter to get detailed message about starting and stopping the One Identity Manager Service.

- DebugMode

In DebugMode One Identity Manager Service writes additional information to the log file. For example, all the parameters and results that are passed to a component are written to the log file.

- Component debugging mode (ComponentDebugMode)

When set, individual One Identity Manager Service process components write additional process information to a log file.

NOTE: The parameters "DebugMode" and "ComponentDebugMode" are used to localize errors. Setting these parameters is not recommended during normal work hours because system performance is affected.

- IP address of the HTTP server (HTTPAddress)

If One Identity Manager Service is running on a computer with several network cards, you can use this parameter to define which service should work over which IP address. If no IP address is entered, then all of them are used.

- HTTP server port (HTTPPort)

Every One Identity Manager Service automatically works as an HTTP server. This parameter specifies the port that One Identity Manager Service works with. The port 1880 is the default value. The HTTP Server communicates through:

`http://<server name>:<port number>`

Example:

`http://<server name>:1880`

- Logging of Job provider and current instance (LogDestinationAndProviderId)

The Job providers ID and the current instance are written in the process step log messages.

- Language

This parameter specifies the language for One Identity Manager Service error messages and output. Permitted input is "deutsch" or "english". The default is "English".

- Use SSL (UseSSL)

Set this option if an HTTP Server secure connection is available. The server is accessed over HTTPS in the browser.

- Do not protect encrypted configuration values (DoNotProtectEncryptedValues)

Nomally, encrypted values from the Jobservice.cfg are additionally protected by the data protection API. This prevents use by other accounts or servers. This option switches off additional protection to use it on other cluster nodes, for example.

- Wait time on failed start (WaitTimeOnFailedStart)

Use this parameter to set the amount time to wait after a start has failed, before restarting. The default value is 90 seconds.

- Retries on failed start (RetriesOnFailedStart)

Use this parameter to specify the number of retries for the One Identity Manager Service start up process. Default value is 5 retries.

Related Topics

- [Extended Debugging in the One Identity Manager Service on page 640](#)

The Logwriter Module

This module writes the One Identity Manager Service messages. The following module types may be selected:

- [EventLogLogWriter](#)
- [FileLogWriter](#)

EventLogLogWriter

This module writes One Identity Manager Service log events to a log file. The event log can be displayed using the event window of the Microsoft Management Console, for example.

Following parameters are available:

- Event log (EventLog)

Enter the name for the event log to which the messages should be written. The messages are written to the application log if the default value "Application" is used.

NOTE: If several One Identity Manager Service write event logs on a server, ensure that the first 8 letters of the log name are unique.

- Severity level (LogSeverity)

Specifies the warning level for logging messages. By default, only warnings and serious errors are logged.

Table 208: Warning Levels for Logging

Severity level	Description
Info	All messages are written to the event log. The event log quickly becomes large and confusing.
Warning	Only warnings and exception errors are written to the event log (default).
Serious	Only exception errors are written to the event log (exceptions).
<ul style="list-style-type: none">Event ID Define an ID with which messages are written in the event log.Category Define a category with which messages are written to the event log.Source Define a name for the source with which messages are written to the event log.	

Related Topics

- [FileLogWriter on page 375](#)
- [One Identity Manager Service Logging on page 636](#)

FileLogWriter

The FileLogWriter writes One Identity Manager Service messages into a log file. The log file can be displayed in a browser.

Following parameters are available:

- Log file (OutPutFile)

The parameter contains the name of the log file including its directory. Log information for the One Identity Manager Service is written to this file.

IMPORTANT: Ensure that the given directory exists. If the file cannot be created, no error output is possible. In this case, the error messages appear in the Windows event log or, under Linux, in /var/log/messages.

- Renaming interval for the log file (LogLifeTime)

In order to avoid unnecessarily large log files, the module supports the functionality of exchanging the log file with a history list. The LogLifeTime specifies the maximum life of a log file before it is renamed as backup. If the log file has reached its maximum age, the file is renamed (i.e. as JobService.log_20040819-083554) and a new log file is started.

Timeout format:

day.hour:minutes:seconds

- Process step log duration (JobLogLifeTime)

Use this parameter to specify the length of time process step logs are kept. After this expires, the logs are deleted.

Timeout format:

day.hour:minutes:seconds

For test purposes, you can enable logging of individual process steps in the Job Queue Info. The process step's processing messages with the NLog warning level "Debug" are written to a separate log. The files are stored in the log directory.

Repository structure:

```
<Protokollverzeichnis>\JobLogs\<first 4 digits of the UID_Job>\Job_<UID_Job>_<yyyymmdd>_<Timestamp>.log
```

- Max.number of archived log files (HistorySize)

This attribute limits the number of log files. If several log files exist, the oldest backup file is deleted when a new log file is created so that the limit is not exceeded.

- Max .log file size (MB) (MaxLogSize)

Use this parameter to specify the maximum size for the log file. Once the log file has reaches the limit, it is renamed into a backup file and a new log file is created.

- Max. length of the parameter (ParamMaxLength)

This parameter defines how many character can be in a job so that it is still written to the log file.

- Severity level (LogSeverity)

Specifies the warning level for logging messages. Only warnings and fatal errors are logged by default.

Table 209: Warning Levels for Logging

Severity level	Description
Info	All messages are written to the log file. The log file quickly becomes large and cumbersome.
Warning	Only warnings and exception errors are written to the log file

Severity level	Description
	(default).
Serious	Only exception errors are written to the log file.

Related Topics

- [One Identity Manager Service Logging on page 636](#)
- [Displaying the One Identity Manager Service Log File on page 638](#)
- [Enabling Extended Logging of Process Steps on page 622](#)

The Dispatcher Module

In a hierarchical server structure a server can be used as a proxy server for other servers. The proxy server makes requests at set time intervals for process steps to be processed on a server and sends them to the next server. If the request load needs to be minimized, a proxy server is recommended.

Following parameters are available:

- Acts as proxy for other servers (`IsProxy`)

This parameter specifies if a server is acting as a proxy server. Set this option if the server should be a proxy server.

- Proxy request interval (`ProxyInterval`)

The `ProxyInterval` sets the time interval in seconds, after which the proxy server acting as deputy for another server, should renew a request to the database.

The following guidelines can be used as orientation for the configuration of One Identity Manager Service polling intervals in a cascading environment:

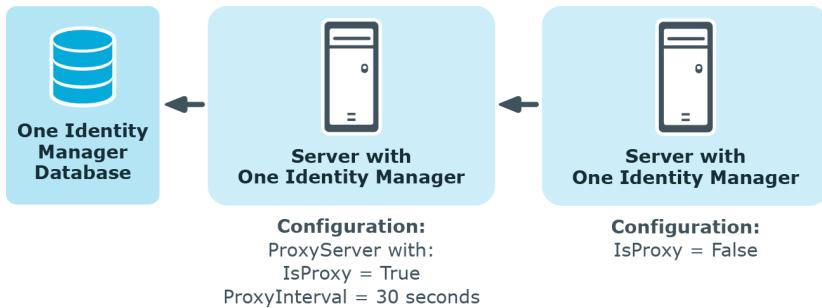
Table 210: Polling Interval Guidelines for One Identity Manager Service

Parameter	Root Server (direct connection to database)	Leaf Server (connected using HTTP/File or other.)
<code>JobServiceDestination.Startinterval</code>	90 seconds	600 seconds
<code>JobServiceDestination.Statisticinterval</code>	360 seconds	600 seconds
<code>Dispatcher.ProxyInterval</code>	180 seconds	
<code>Dispatcher.IsProxy</code>	True	False

The proxy mode of a root server ensures that, acting on behalf of the leaf server, process steps are queried in shorter proxy intervals. When a root server is restarted it can take a

while until all the leaf servers have send their first requests (in this case max.6000 seconds), but then the system takes over.

Figure 44: Dispatcher Configuration Example



The Connection Module

With this module you can set special configuration settings for the behavior of the One Identity Manager Service.

The parameters in this module are:

- Directory for generating logging (JobGenLogDir)

Log files are created in this directory that record process generation instructions from One Identity Manager Service.

- Suppress reload beep (NoReloadBeep)

When this parameter is set the beep is switched off that is made when buffered dialog data is loaded.

- BLOB read operation logging (LogBlobReads)

Use this parameter to specify whether read operation on text and binary LOB field are written to the SQL log.

- Reload interval for cache (CacheReloadInterval)

Use this parameter to enter a time interval for updating the local cache. Click on this icon to discard the change. The parameter overwrites the setting in the configuration parameter "Common\CacheReload\Interval".

- Expression for tracing the stack position (ObjectDumpStackExpression)

This expression specifies when an extra stack trace is written to the object log. If the current row in the object log matches the regular expression, the stack trace is written in the object log.

Example expression: "Lastname"

If the current contain the value "Lastname" the stack trace is also copied to the log.

NOTE: This parameter is used for localizing errors. It is not recommended to set this parameter in normal working conditions on performance grounds.

Related Topics

- [Writing the One Identity Manager Log on page 634](#)
- [One Identity Manager Service Logging on page 636](#)

The HTTP Authentication Module

Every One Identity Manager Service automatically works as an HTTP server. Which services the One Identity Manager Service provides depends on the plug-ins configurations. Use this module to specify how authentication works on an HTTP server so that other services can be accessed, for example, displaying the log file or the status display.

The following module types may be selected:

- BasicHttpAuthentication

To access the HTTP server with this authentication type, enter a specific user account (user) and the associated password (password).

- WindowsHttpAuthentication

Use this authentication type to specify an Active Directory group, whose users can be authenticated on the HTTP server. Either an SID or the Active Directory group name can be entered into the Job server domain. If the Active Directory groups are not in the Job server domain, you must use the SID.

 **NOTE:** If no model is given, no authentication is required. All users can access the service.

The Plug-ins Module

Plug-ins are program classes that One Identity Manager Service loads and that extend the functionality of the service. The following plug-ins are available:

- [HTTPLogPlugin](#)
- [ScheduleCommandPlugin](#)
- [DBSchedulerWatchDogPlugin](#)
- [RequestWatchDogPlugin](#)
- [PerformanceCounterPlugin](#)
- [DebugMailPlugin](#)
- [ShareInfoPlugin](#)
- [RemoteConnectPlugin](#)

HTTPLogPlugin

The HTTPLogPlugin writes a log file that records the One Identity Manager Service HTTP requests.

Enter the following parameter:

- Output file (LogFile)

Enter the name of the file that is to record the messages. The file is written in Apache HTTP Server Combined Log Format.

ScheduleCommandPlugin

This plug-in calls up an external program in regular intervals. This is useful, for example, when process steps need to be routed over their own transfer methods.

The following parameters are necessary:

- Run command (Command)

This parameter defines the command that is to be carried out, including the command line options. This will be executed as a cmd, therefore built-in commands are possible.

- Service start Command (StartCommand)

This command is run when One Identity Manager Service is started.

- Service start Command (StopCommand)

This command is run when One Identity Manager Service is stopped.

- Interval (Interval)

This parameter specifies how often the command should be called. Click on this icon to discard the change. While the command is running, the timer is stopped so that the calls do not overlap.

- Command output to log file (OutputToLog)

If this parameter is set the command output is written to the One Identity Manager Service's log file also on success. Otherwise only errors are written to the log file.

- Severity level (LogSeverity)

This is where the warning level is given with which the warnings appear in the log file. Permitted are "Info", "Warning" and "Serious".

DBSchedulerWatchDogPlugin

This plugin checks, at regular intervals, whether the database schedule for the DBQueue Processor is enabled and starts it if necessary.

 **NOTE:** The plug-in should only be started on one Job server in the network. Starting it on an SQL processing server is recommended.

The following parameters are necessary:

- Monitoring interval (Interval)

Use this parameter to specify how regularly the DBQueue Processor is checked. Click on this icon to discard the change.

- Job provider ID (ProviderID)

This parameter contains the ID of the Job provider to be used. Enter the Job provider name. If this is empty the first Job provider is used.

RequestWatchDogPlugin

This plug-in restarts One Identity Manager Service when less than a defined number of requests are made within a specified interval.

The following parameters are necessary:

- Monitoring interval (Interval)

This parameter sets the monitoring interval. Click on this icon to discard the change.

- Min. number of requests (MinRequests)

This parameter contains the minimum number of requests that need to be made within the interval. When setting this value, take into account that a DBSchedulerWatchDogPlugin is possibly in use that also posts requests.

PerformanceCounterPlugin

This plugin exports the One Identity Manager Service status values as performance counter. This makes monitoring through a system monitor (for example Perfmon) possible. The list of currently available performance counters is displayed under <http://servername:1880/PerfCounter>.

The following parameters are necessary:

- Value types to specify (CounterType)

Use this parameter to specify which value types are made available as performance counters. Int and long values should be entered directly, time values should be entered as long values (number of milliseconds).

- Polling interval (PollingInterval)

This parameter to specify the interval for exporting the performance counter. Click on this icon to discard the change.

- Category

Use this parameter to specify which category the performance counters for this One Identity Manager Service are displayed under. This data is required if several One Identity Manager Services with PerformanceCounterPlugin are enabled.

NOTE: If the error "At least one service could not be started" occurs after restarting the One Identity Manager Service, then make the WMI Performance Adapter service a dependency of One Identity Manager Service.

DebugMailPlugin

If this plug-in is enabled, email notifications generated by the One Identity Manager Service are not send but are kept in a drop folder. The file names contain the time stamp in this case. If a mail contains HTML text, a HTML file with the HTML body is saved that has the same name as the text file to be created. Attachments are also saved in this way.

Enter the following parameter:

- Drop folder (DropFolder)

Directory for storing email notifications.

NOTE: The plugin only works for processes executed internally in the One Identity Manager Service.

NOTE: If this plugin is enabled, no email notification are sent through the One Identity Manager Service. This plug-in is only used for localizing errors. It is not recommended to set this parameter in normal working conditions.

ShareInfoPlugin

This plugin is required for solving Samba shares (smb.conf) under the Linux operating system. The plugin solves UNC paths to local paths. This plugin does not required any parameters.

NOTE: Install the plugin if the One Identity Manager Service executes copy actions between servers with Linux operating system.

RemoteConnectPlugin

To configure synchronization with a target system, One Identity Manager must load the data from the target system. One Identity Manager communicates directly with target system to do this. because of the firewall configuration, for example, you can set up a remote connection. Prerequisite for this is that the RemoteConnectPlugin is installed on the Job server.

The plugin requires the following parameters:

Table 211: RemoteConnectPlugins Parameters

Parameter	Value	Description
Authentication method (AuthenticationMethod)	ADSGroup	Method with which incoming queries can be authenticated. Permitted values: ADGroup
Permitted AD group (ADGroupAuthPermittedGroup)		Distinguished name or object SID of the Active Directory group whose members are permitted to use a remote connection. This parameter is only required for the authentication methods "ADGroup".
Port (Port)	2880	Port for reaching the server.

 **NOTE:** Authentication of a remote connection can only be done through an Active Directory group.

Module File with Private Key

In this module, you provide the data for files with a private key. Use this parameter if you work with several private keys, for example, if One Identity Manager Service data must be exchanged between two encrypted One Identity Manager databases. If no key is entered, the private key file from the JobServiceDestination parameter "File with private key (PrivateKey)" is used.

To enter a file with a private key

1. Click **New**.
2. Enter the private key ID in the **Property** column. The ID is expected in the JobServiceDestination in the parameter "Private key identifier (PrivateKeyId)". The default key has the ID "Default".
3. Enter the path of the key file in the **value** column. You can enter the absolute or relative path to the One Identity Manager Service.

Example of the configuration in the file Jobservice.cfg.

```
configuration>
  <category name="privatekeys">
    <value name="Default">private.key</value>
    <value name="Key2">key2.key</value>
    <value name="OtherKey">C:\Path\To\Other.key</value>
  </category>
```

```
</configuration>
```

Related Topics

- [JobServiceDestination on page 365](#)

Handling Processes in the One Identity Manager

One Identity Manager uses so called 'processes' for mapping business processes. A process consists of process steps, which represent processing tasks and are joined by predecessor/successor relations. This functionality allows flexibility when linking up actions and sequences on object events.

So-called process tasks are used to perform single elementary tasks at system level, for example, adding a directory. A process component consists of one or more process tasks and its parameters. Process components are defined in the tables Jobcomponent, Jobtask and Jobparameter along with their process tasks and parameters. Predefined configurations are maintained by the schema installation and cannot be edited apart from a few properties.

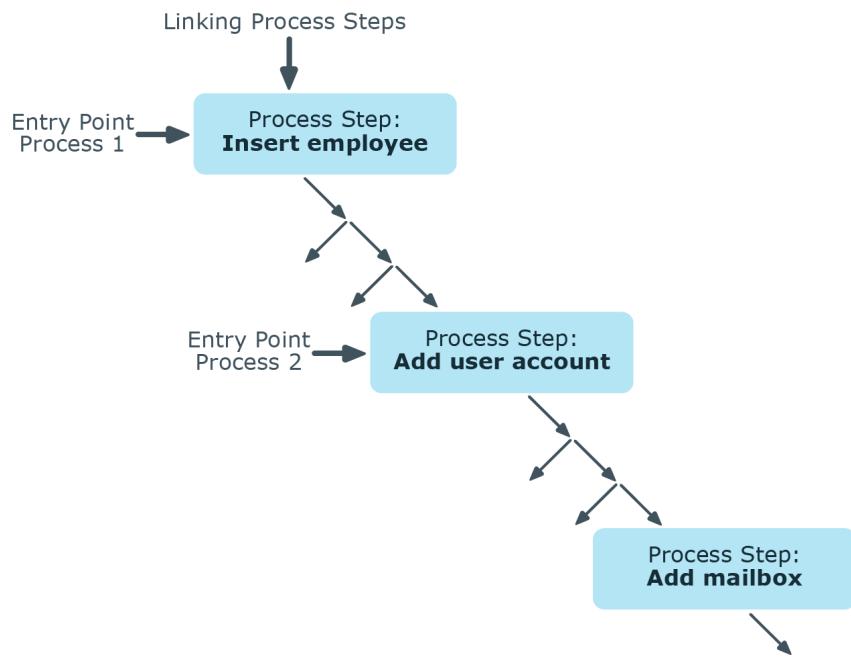
Processes are modeled using process templates. A process generator (Jobgenerator) is responsible for converting script templates in processes and process steps into a concrete process in the 'Job queue'.

One Identity Manager Service, a service running on the target system, collects the process steps from the Job queue. The process steps are executed by process components in the target system. The One Identity Manager Service also creates an instance of the required process component and passes the parameters to the process step. Decision logic monitors the execution of the process steps and determines how processing should continue depending on the results of the executed process components. The One Identity Manager Service enables parallel processing of process steps because it can create several instances of process components. The One Identity Manager Service is the only One Identity Manager component authorized to make changes in the target system.

The following illustration shows a chain of process steps with which you can add an employee, set up an Active Directory user account for him or her and finally add a mailbox.

You can reproduce this sequence in a process. However, you can also define entry points for other processes. The result of entering at point "process 1" is the addition of an employee with an Active Directory account with a mailbox. Joining at entry point "process 2" only results in the addition of an Active Directory user account with a mailbox.

Figure 45: Creating a Single Process by Linking Process Steps



Detailed information about this topic

- [Defining Processes on page 391](#)
- [Executing Processes Automatically on page 416](#)
- [Process Components on page 419](#)

Working with the Process Editor

The Process Editor is the program that you use to define and edit processes. The editor is started from the program "Designer" and opens in the document view. Only additional Process Editor functions are described in the following.

Menu Items

The following items are added to the menu bar when the editor starts.

Table 212: Meaning of Items in the Menu Bar

Menu	Menu Item	Meaning	Key Combination
Process	New	Creates a new process.	Ctrl + N
	Delete	Deletes the selected process after requesting confirmation.	
	Rearrange	Rearranges the process within the process document.	
	Error checking	Carries out a validity check on the process. Messages are sent to the "Process error" view.	
	Compare processes...	Open a dialog window for comparing processes.	
	Compile	Compiles the selected process on a test basis. Messages are sent to the "Compiler error" view.	
	Compile and save to database	Compiles the selected process and assembly is saved in the database. Messages are sent to the "Compiler error" view.	
	Export...	Exports the selected process as XML file.	
	Import...	Imports the selected process into the database from an XML file.	
	Copy...	Starts a copy wizard for a process.	
	View	Swaps between views.	
	View/ Edit view	Default view for editing processes.	
	View/Simulation view	Starts the process simulation wizard.	
	View/ Source code view	Only shown when compiler errors have occurred.	
Process step	New	Adds a new process step into the process document.	
	Delete	Deletes the selected process step from the process document.	

Menu	Menu Item	Meaning	Key Combination
	Import	Imports a process step from anywhere in the database into a process document.	
	Search	Searches for a process step within the selected process.	CTRL + F
	Copy	Copies the selected process step into the clipboard.	
	Cut	Copies the selected process step into the clipboard and deletes it from the process document.	
	Paste	Inserts the selected process step from the clipboard into the process document.	
View	Process errors	Shows/hides view "Process errors".	
	Compiler errors	Shows/hides the "Compiler errors" view.	
	Parameters/Events	Shows/hides view "Parameter" or "Events" respectively.	
	Properties	Shows/hides the edit view.	
Help	Process editing help	Opens help window on this theme.	
	Process Editor help	Opens the editor help.	

Table 213: Meaning of the Icons in the Toolbar

Icon	Meaning
	Creates a new process
	Deletes a process
	Executes an validity check for this process.
	Rearranges the process.
	Compiles the process.
	Compiles the process and save assemblies.
	Exports a process.
	Imports a process.
	Insert a new process step.
	Deletes a process step.

Icon	Meaning
	Searches for a process step and imports it into the process document.
	Copies the process into the clipboard.
	Deletes the process step from the list but retains it in the clipboard.
	Inserts a process step from the clipboard.
	Shows/hides edit view.
	Shows/hides simulation view.
	Shows/hides source code view.
	Shows the process generator log (after simulation).
	Zooms in on the view.
	Zooms out on the view.
	Shows/hides edit view.
	Shows/hides compiler errors.
	Shows/hides process errors.

Table 214: Process Document Context Menu Items

Context Menu Item	Meaning
New	Creates a new process step element for editing a process step.
Delete	Deletes the selected process step.
Copy	Copies the selected process step into the clipboard.
Paste	Insert the process step stored in the clipboard
Arrange	The elements are restore to their default positions.
Properties	Displays the object properties of the selected entry.

Views in the Process Editor

The Process Editor has several views for editing processes and process steps.

Table 215: Process Editor Views

View	Description
Process document	The process document contains special control elements that allow a process to be displayed and edited with its process steps. A separate document is opened for each process.

View	Description
Edit view for processes and process steps	In this edit view you can change the properties of processes and process steps. The process or process step properties that are shown, depends on which elements are selected in the process document.
Edit view for events and parameters	This edit view enables you to alter the event properties for a process or the parameters for a process step. Either parameters or events are shown depending on which elements is selected in the process document.
Process validity check view	The result of the validity check is displayed in this view and is retained until the next validity check.
Compiler errors	Errors that occur when a process is compiled are displayed in this view.
Source code view	The source code is displayed if errors occur during compilation. This view is only for displaying the source code. It cannot be edited here.
Simulation view	When you change to the simulation view, a wizard is started that tests how a process is generated.

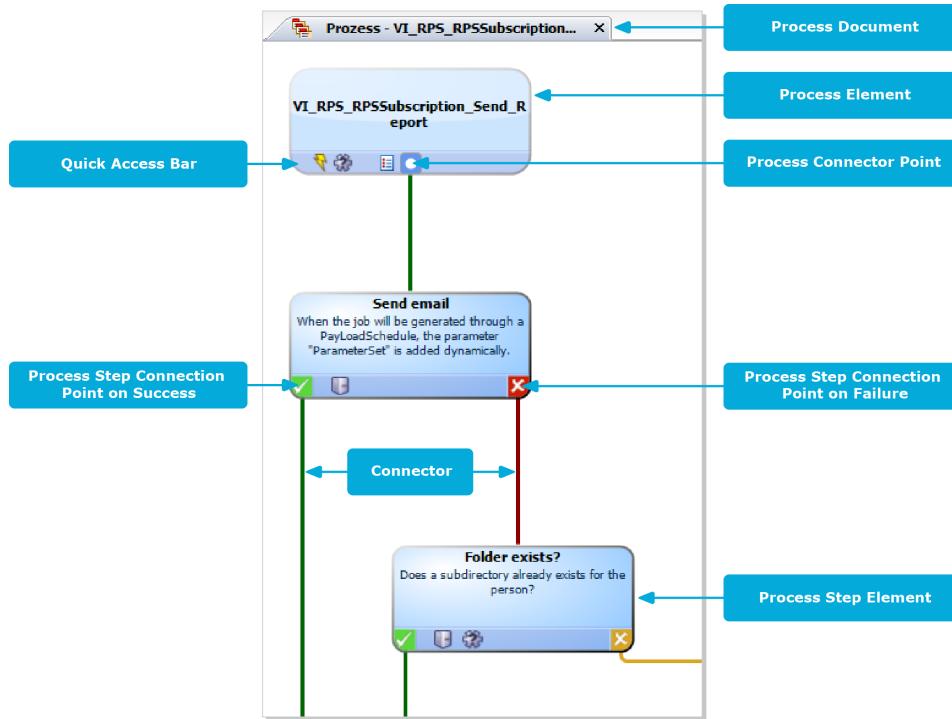
Related Topics

- [Working with the Process Document on page 389](#)
- [Checking the Validity of a Process on page 413](#)
- [Simulating Process Generation on page 411](#)

Working with the Process Document

The process document contains special control elements that allow a process to be displayed and edited with its process steps.

Figure 46: Representing a Process in the Process Editor



When you add a new process, an initial process document with one process element is created. When you add a process step, the associated process step element is created.

Individual elements are linked to each other with a connector. Activate the connection points with the mouse.

- To create a connection, click on a connection point, hold down the left mouse button and pull a connector to the second connection point.
- To delete a connection, select a connection end-point and confirm the deletion request that appears. Confirm the security prompt with **OK**.

Double-click on the process or process step element to open the respective edit view, where you can make your changes.

Each element has a tooltip. A process element's tooltip displays the name and description of the process. A process step element's tooltip displays the name and description of the process step as well as the description of the process function used.

Each element contains a quick access menu bar. The icons represent special properties of processes or process steps. The icon's tooltip shows more detailed information about a property. Double-clicking on a icon respectively opens the process or process step's edit view and jumps to the corresponding property.

Table 216: Quick Access Icons

Icon	Meaning
	Events are defined.
	Process is not generated.
	Process in wait mode on error.
	Processing is split. The connection point on error and the connector to the subsequent process step are colored yellow.
	Runtime errors are ignored. The connection point is colored gray on error. No process step is possible on error.
	If an error occurs, no more process steps are handled for this process.
	A generating condition exists.
	Process information is enabled.
	A script for selecting a server or server mask is entered.
	Messaging on error and on success is enabled.

Some important properties are shown by the color of the element.

Table 217: Colors of Elements

Color	Meaning
Blue	Default.
Yellow	The verification test resulted in a warning or information.
Red	The verification test failed.
Gray	The process is disabled.

You can drag and drop elements in the process document. Use **Arrange** in the context menu to reset the elements to their default positions. The position of each element is transferred to the One Identity Manager database when the entire process is saved. The layout is therefore available to all users when you restart the Designer.

Defining Processes

Processes are edited and displayed in the category **Process Orchestration** in the Designer. Apart from the default processes supplied by us and customer specific processes, you also get an overview of the process components with their process tasks and parameters. You can also set up process plans that are available for triggering processes on a cyclical basis.

So-called process tasks are used to perform single elementary tasks at system level, for example, adding a directory. A process component consists of one or more process tasks and its parameters.

NOTE: Predefined configurations are maintained by the schema installation and cannot be edited apart from a few properties. The default configuration is moved to a configuration buffer during handling. You can retrieve changes from the configuration buffer and restore the default configuration in this way.

The following steps are required to set up a process

1. Create up a process.
2. Specify which events to trigger.
3. Create the process steps.
4. Edit the parameters.
5. Test the process.
6. Compiles the process.

Detailed information about this topic

- [Creating and Editing Processes on page 392](#)
- [Creating and Editing Events on page 397](#)
- [Creating and Editing Process Steps on page 399](#)
- [Process Step Parameters on page 406](#)
- [Simulating Process Generation on page 411](#)
- [Checking the Validity of a Process on page 413](#)
- [Compiling a Process on page 415](#)
- [Process Components on page 419](#)

Creating and Editing Processes

IMPORTANT: The process and process steps are not created until the entire process is saved in the One Identity Manager database. After this, other users can use the Process Editor to make changes to the process. However, it cannot be generated yet. The process has to be compiled before it can be generated.

To create a custom process

- Create a new process.
 - OR -
- Copy an existing process and edit the copy.

To create a new process

1. Select the category **Process Orchestration** in the Designer.
2. Start the Process Editor using the task **Create a new process**.

This makes a new element for the process and opens it in the Process Editor.

To edit an existing process

1. Select the process in the category **Process Orchestration** in the Designer
2. Start the Process Editor with **Edit process 'XY'** in the task view.

The process is opened in the Process Editor.

To copy a process

1. Select the process in the category **Process Orchestration** in the Designer
2. Start the Process Editor with **Edit process 'XY'** in the task view.
3. Select **Process | Copy...** from the menu.
Starts a copy wizard for a process. The wizard start up screen displays the name of the process to be copied.
4. Click **Next**.
5. Specify the name of the new process and set the copy options.

Table 218: Copy Options

Option	Meaning
Rename process steps	If you set this option you have the option to rename each of the process steps individually in the next step of the wizard.
Copy events	Set this option if you want the events that are assigned to this process to be copied as well.
Disable source process	This option specifies whether the source process should be disabled after copying. If you set this option the source process is set to "do not generate".
Disable copied process	Use this option to specify whether the process should be disabled after copying. If you enable this option the process is set to "do not generate".

6. Click **Next**.
7. (Optional) rename each process step and click **Next**.

You can change these by clicking on the new process step name.

 **NOTE:** This step is only available if you have set the copy option **Rename process steps**.

- To start compiling, click **Next**.

During the process the action that is currently being executed is displayed in a status bar. The process is opened in the Process Editor and you can continue editing.

Detailed information about this topic

- [Properties for a Process on page 394](#)
- [Using Process Specific and Global Variables for the Process Definition on page 396](#)
- [Thresholds for Handling Processes on page 397](#)
- [Creating and Editing Events on page 397](#)
- [Creating and Editing Process Steps on page 399](#)
- [Comparing Processes on page 410](#)
- [Exporting and Importing Processes on page 411](#)

Properties for a Process

Table 219: Properties for a Process

Property	Meaning
Name	Name of the process. The name of the process must be unique. Label custom processes with the prefix "CCC_".
Table	The process is generated on the event from this base object (table).
Description	Additional description of the process.
Remarks	Additional remarks about the process.
Process	Process UID. These cannot be edited.
Process information	Specifies whether this step is logged. Logging depends on the setting of the configuration parameter "Common\ProcessState\ProgressView".

Table 220: Permitted Values

Value	Description
None	The process information is not logged.
Full process tracking	The process information is logged and displayed in the Manager.
Web Portal tracking	The process information is logged and displayed in the Manager and in the Web Portal.
Process information term	VB.Net expression for displaying the display name in the process view.

Property	Meaning
Pre-script for generating	The pre-script is executed before other scripts are run. You can find global variables with a pre-script or define process specific variables that can then be used within the process and process steps, for example, in generating conditions, sever selection scripts or parameters. i NOTE: When a process is being handled, the generating pre-script is executed first and then the generating condition is evaluated.
Generating condition	Define a condition in VB.Net syntax for the process step, which is used to decide whether the process is generated. If a generating condition is given, the process is only generated if the condition is fulfilled. You can find an example scripts on the installation medium in directory QBM\d-vd\AddOn\SDK\ScriptSamples.
Do not generate	Use this option to decide whether a process will be generated. If the option is set, the process will not be generated and cannot be compiled. i NOTE: If this option is set for this process, it remains in the 'set' state during update migration and is not reset.
Preprocessor condition	You can specify a preprocessor condition for a process for conditional compiling. A process is only available, therefore, if the preprocessor condition is fulfilled.
Disabled by preprocessor	If a process step is disabled by a preprocessor condition, the option is set by the Database Compiler.
Threshold (warning)	Maximum number of processes for a queue that can be present at the same time. A warning is sent if the number is exceeded. The One Identity Manager Service continues handling processes all the same.
Threshold (disable)	Maximum number of processes for a queue that can be present at the same time. If this number is exceeded, other processes are set to OverLimit and not handled by the One Identity Manager Service.

Related Topics

- [Using Process Specific and Global Variables for the Process Definition on page 396](#)
- [Thresholds for Handling Processes on page 397](#)
- [Logging Process Information during Process Handling on page 429](#)
- [Conditional Compilation using Preprocessor Conditions on page 448](#)
- [Using Scripts](#)

Using Process Specific and Global Variables for the Process Definition

Process specific variables are local data spaces when a process is generated. This enables variables to be defined that can be made further use of within processes and process steps, for example, generating conditions, server selection scripts or in parameters.

- 1 | **NOTE:** It is recommended only to set process specific variables in the pre-script and to have read access to them during further usage.

Pre-script syntax:

```
values("Name") = "value"
```

Usage in the following process and process step code sections:

```
Value = values("Name")
```

Additional global variables can be used to control process generation, which are made available over the session object. These variables are valid as long as the session is active. All custom variables defined for the session object can be used in addition to predefined variables. Custom global variables can be defined through scripts, methods or customizers, for example, and used in the processes.

- 1 | **NOTE:** Global variables should only be used with read access in processes.
- 1 | **NOTE:** When a process is being handled, the generating pre-script is executed first and then the generating condition is evaluated. It is recommended to evaluate global variables that are used in the generating condition in the pre-script as well. This can prevent unnecessary data access.
- 1 | **NOTE:** If a custom session variable is defined, it must be removed again afterward. Otherwise it remains for the rest of the session and, in certain circumstances, the wrong processes can be generated.

Example of use:

The process should only be generated for a full synchronization. The session variable "FullSync" is used for this. This variable can take the value "true" or "false". The variable is available to all processes that are generated within full synchronization.

The variable is queried in the pre-script for generating and the generating condition. This way, loading of unnecessary objects is already prevented by executing the pre-script.

Generating pre-script:

```
If CBool(Session.Variables("FULLSYNC")) Then  
    values("Name1") = "value1"  
    values("Name2") = "value2"  
    ...  
End If
```

Generating condition:

```
Value = CBool(Session.Variables("FULLSYNC"))
```

Related Topics

- [Pre-scripts for using in Processes and Process Steps on page 461](#)
- [Querying Session Object Global Variables on page 463](#)

Thresholds for Handling Processes

Table 221: Configuration Parameter for Notifications

Configuration parameter	Meaning
Common\MailNotification\NotifyAboutWaitingJobs	Specifies whether a message should be sent if the process steps have a particular execution state in the job queue.

In order to prevent bulk modifications, you can specify how long each process can remain in the Job queue. Use the values **Threshold (warning)** and **Threshold (disable)** to do this. You can use the database script `SDK_SetLimitationCount_in_Jobchain` to initially fill the process data. You can find an example of a configuration file on the installation medium in directory `QBM\dvd\AddOn\SDK\SQLSample`.

If the warning threshold is exceeded, a message is sent by email to a specified recipient. Prerequisites for using the notification system is an SMTP host set up for sending mail and activation of the configuration parameter for mail notification.

If the disable threshold is exceeded, the affected processes are given the status "Overlimit" in the Job queue. These processes are no longer collected by One Identity Manager Service for processing and remain in the Job queue. You can re-enable the process steps in the program, "Job Queue Info".

If the configuration parameter "Common\MailNotification\NotifyAboutWaitingJobs" is set, an additional email is sent when processes are labeled with the status "Overlimit2 and a corresponding entry is made in the update server's event log.

Related Topics

- [Properties for a Process on page 394](#)
- [Setting up the Mail Notification System on page 122](#)
- [Reinstating Process Steps and Processes on page 621](#)

Creating and Editing Events

Events are defined to assign processes to objects. Processes cannot be generated until a link has been created between object, event and process. The following predefined events are available. These are described in the following table.

Table 222: Predefined Events

Event	Comment
Insert	Event created when an object is created. Available for all objects.
Update	Event created when an object is changed. Available for all objects.
Delete	Event created when an object is deleted. Available for all objects.
Execute	Event created by DBQueue Processor when the execution time is reached of a delayed operation.

Other events are provided by the Customizer. These events are described in the Customizer documentation. You can define other custom events to trigger processes.

In One Identity Manager, how events for stored processes are triggered is linked with the permissions concept. Users can only trigger events on objects like this if they own edit permissions for them. This can lead to table users, who only have viewing permissions, not being able to trigger additional events for processes.

In this case, it is possible, to define so called object events and connect them to a program function. An event, which is defined for a process, is linked with an object event. If the object event is assigned a program function, users that owns the program function, can trigger the associated object event and therefore the process, depending on their permissions.

To create

1. Open the process in the Process Editor.
2. Click on the element for the process in the process document.
3. Select the **Events** view and click .
4. The table's object events are displayed in the **Object event** menu. Select an valid object event or create a new object event using the  button.

To allow triggering a process through a program function

1. Select the category **Permissions | Program functions** in the Designer.
2. Select the menu item **View | Select table relations...** and enable the tables QBMEventHasFeature and DialogGroupHasFeature.
The tabs **Object events** and **Permissions groups** are displayed in the edit view.
3. Select the program function and assign an object event and permissions groups to it.

Related Topics

- [Tracking Changes with Process Monitoring on page 425](#)

Creating and Editing Process Steps

A process consists of process steps, which represent processing tasks and are joined by predecessor/successor relations.

To add process steps within a process with the you can:

- Create a new process step
- Copy an existing process step and edit the copy
- Import an existing process step into the process

To create a new process step

1. Open the process in the Process Editor.
2. Select **Process step | New** from the menu.

This makes a new element for the process step and displays it in the Process Editor.

3. Edit the process' master data.
4. Link the process step with the process.

To edit an existing process step

1. Open the process in the Process Editor.
2. Click on the element for the process step in the process document.

NOTE: To edit several process steps, hold down the **CTRL** key and click on the process steps.

Input fields with different values are marked with  in the process step edit view. The value in the input field is copied to selected process steps after the changes have been saved.

To copy a process step

1. Select the process step to copy and use **Copy** in the context menu or **CTRL + C** to copy the process step to the clipboard,

NOTE: To edit several process steps, hold down the CTRL key and click on the process steps.

2. Insert the process step using **Paste** in the context menu or **CTRL + V**.

The process step is given a new UID and all the process steps are copied.

3. Edit the process step's master data
4. Link the process step with the process.

To import a process step

1. Open the process in the Process Editor.
2. Select **Process step| Import** from the menu.

Table 223: Meaning of the Icons in the Toolbar

Icon	Meaning
	Searches for a process step.
	Imports the process step.
	Specifies the search options.

3. Enter a search string and use the search options to specify which objects should be searched for.

The given objects are searched for internally by a WHERE clause. If several objects are found they are appended, internally, with a 'Join' condition.

Table 224: Searchable Objects and Properties

Search in Objects	Properties to be Searched
Process	Name
Process step	Name, description, generating condition, server selection script
Parameter	Name, value
Process component	Component class, component assembly
Process task	Name
Parameter template	Name, value template

4. Start the search

The process steps that are found are displayed in the result list.

5. Select the process steps you want from the list and import them into the process document with in the toolbar or by double clicking in the process document.
6. Edit the process step's master data
7. Link the process step with the process.

Detailed information about this topic

- [Process Step Properties on page 401](#)
- [Specifying the Execution Server on page 403](#)

- [Notifications about Process Step Handling on page 405](#)
- [Process Step Parameters on page 406](#)
- [Creating and Editing Processes on page 392](#)

Process Step Properties

Table 225: General Process Step Properties

Property	Meaning
Name	Name of the process step.
Process task	Process task to execute for the process component. When you select a process task you define which action is executed by the process step. The process task parameter templates are copied to the process step as parameters. This means that every process step that uses this process task can pass other parameter values. The original is not altered.
Description	Additional description of a process step.
Priority	The priority sets the precedence in the Job queue for adding and processing the process step. The value ranges from 1 to 15. The higher the value, the sooner the process step will be processed.
Process information	Specifies whether this step is logged. Logging depends on the setting of the configuration parameter "Common\ProcessState\ProgressView".

Table 226: Permitted Values

Value	Description
None	The process information is not logged.
Full process tracking	The process information is logged and displayed in the Manager.
Web Portal tracking	The process information is logged and displayed in the Manager and in the Web Portal.
Process information term	VB.Net expression for displaying the display name in the process view.
Depth of detail	Severity level for mapping process information.
Notification (success)	Specifies whether notification is sent on success.
Notification (error)	Specifies whether notification is sent on error.

Property	Meaning
Pre-script for generating	The pre-script is executed before other scripts are run. You can find global variables with a pre-script or define process specific variables that can then be used within the process, for example, in generating conditions, sever selection scripts or parameters.
Generating condition	Define a condition in VB.Net syntax for the process step, which is used to decide whether the process step is generated. If a generating condition is given, the process step is only generated if the condition is fulfilled.
Preprocessor condition	You can specify a preprocessor condition for a process step for conditional compiling. A process step is, therefore, only available if the preprocessor condition is fulfilled.
Disabled by preprocessor	If a process step is disabled by a preprocessor condition, the option is set by the Database Compiler.
Server function	Specifies the server types for this process step. Specifies the permitted server types for this process step. The selection must lead to a unique result, for example SQL processing Server.
Script for server selection	If it is not possible for the Job Generator to decide which server to use based on the server function, you can use a selection script in VB.net syntax for more a detailed evaluation.
Wait mode on error	If a specific condition is not fulfilled at a particular point in the process step, One Identity Manager Service can repeat the process step. Setting this option results in the process step being re-run depending on latency and retries.
Latency (mins)	Latency period in minutes. Number of minutes a process step, if it has failed, is deferred until the next retry.
Retries	Number of retries.
Split processing	Process steps that are only required for branching the process are labeled with this option. An example could be a process step that checks for the existence of a directory. The next process step to be processed is either the step on success or the step on error (without generating an error message) depending on the return value.
Ignore errors	Specifies whether errors are ignored during execution. In this case the following process step is still carried out despite the previous step not being correctly processed.
Stop on error	If an error occurs when a process step is executed, it remains in the Job queue and is given the status "Frozen". In this case, no more process steps are collected for processing and they remain in the Job queue. You can re-enable the process steps with "Frozen" status in the program "Job Queue Info".

Property	Meaning
	If the configuration parameter "Common\MailNotification\NotifyAboutWaitingJobs" is set, an additional email is sent when processes are labeled with the status "Frozen" and a corresponding entry is made in the update server's event log. Prerequisites for using the notification system is an SMTP host set up for sending mail and activation of the configuration parameter for mail notification.
Log errors to journal	If this option is set, the error message from process handling is logged to the database journal. Error messages from process handling can be recorded in the process history.
Log mode	You can enable an extended logging mode for process step messages in Job Queue Info. Use this logging mode to provide individual processing steps with continuous extended logging. Use the value "always" to log process step messages on success and failure. Use the value "error" to log process step messages only on failure.
Process History	Specifies whether process step notification is written to the process history.

Related Topics

- [Specifying the Execution Server on page 403](#)
- [Notifications about Process Step Handling on page 405](#)
- [Setting up the Mail Notification System on page 122](#)
- [Using Process Specific and Global Variables for the Process Definition on page 396](#)
- [Process Components on page 419](#)
- [Logging Process Information during Process Handling on page 429](#)
- [Reinstating Process Steps and Processes on page 621](#)
- [Enabling Extended Logging of Process Steps on page 622](#)
- [Logging Messages in the Database Journal on page 632](#)
- [Recording Messages in the Process History on page 433](#)
- [Conditional Compilation using Preprocessor Conditions on page 448](#)
- [Using Scripts on page 453](#)

Specifying the Execution Server

You specify which server should handle each process step. You can select the executing server using the server function or a selection script. Server selection should always end with a unique result. The selection script is evaluated first to determine the server. If a

server cannot be determined in this way, the server function is analyzed. The first server that is found is used for executing the process step.

Selecting the Server with a Server Function

The most common server functions are predefined, for example, domain controller or SQL processing server. Enter a server function directly if you can determine the server uniquely.

 **NOTE:** More server functions may be available depending on which modules are installed.

Table 227: Permitted Server Functions

Server Function	Remark
Update Server	This server executes automatic software updating of all other servers. The server requires a direct connection to the database server that the One Identity Manager database is installed on. The server can execute SQL tasks. The server with the installed One Identity Manager database, is labeled with this functionality during initial installation of the schema.
SQL processing server	This server can process SQL tasks. Several SQL processing servers can be set up to spread the load of SQL processes. The system distributes the generated SQL processes throughout all the Job servers with this server function.
One Identity Manager Service installed	Server on which a One Identity Manager Service is installed.
SMTP host	Server from which the One Identity Manager Service sends email notifications. Prerequisite for sending mails using the One Identity Manager Service is SMTP host configuration.
Default report server	Server on which reports are generated.

Selecting the Server with a Selection Script

If it is not possible to decide which server should be used based on the server function (for example, because several SMTP servers exist), you can use a server script for more a detailed evaluation.

To find the server with a selection script, use a VB.Net expression, which:

- Returns a string with the Job server UID
- Returns a string with data for a WHERE clause for database queries. The selection must return a string, which begins with WHERE and contains a logical condition. The WHERE clause is applied to the table QBMServer.

Alternatively, you can enter the queue to be handled by the process step directly into the selection script. Each One Identity Manager Service within the network has a unique queue name. Only process steps that have this exact queue name are requested from the Job queue.

Syntax for direct Queue input:

DIRECT:<queue>

Example:

Value = "DIRECT:\Server01"

Related Topics

- [Using Scripts on page 453](#)

Notifications about Process Step Handling

You have the possibility to send a message when a process step has succeeded or when it has failed. Prerequisite for using the notification system is an SMTP host, set up for sending mail and activation of the configuration parameter for mail notification. Use the various configuration parameters for mail notifications for setting up notifications.

To configure mail notification for a process step

1. Open the process in the Process Editor.
2. Click on the element for the process step in the process document.
3. Set the options **Notification (success)** and **Notification (error)** on the **General** tab.
4. Enter data for sending notifications on the **Notification on success** and **Notification on error** tabs.

Table 228: Notification Properties

Property	Meaning
Sender email address	Email address whoever sent the message.
Recipient email address	Email address of the recipient of the message.
Subject	Subject of the email.
Message	The message to send.

NOTE: All data must be entered in VB.Net syntax. Use #LD notation for language dependent formatting of the information.

1 **NOTE:** Messages are only sent during processing if all the data is entered for a case (failure, success)!

Example for Configuring an Email Message

Sender email address	Value = Connection.GetConfigParm("Common\MailNotification\DefaultSender")
Recipient email address	Value = Connection.GetConfigParm("Common\MailNotification\DefaultAddress")
Subject	Value = #LD("Error updating the Active Directory user account {0}.", \$CanonicalName\$) #
Message	Value = #LD("The user account {0} could not be updated.)#

The process VID_SendMail (table DialogDatabase) is used to send email notifications from the process handling. This process uses the parameters of the database procedure vid_InsertForSendMail. To customize this process, create a copy of the process and edit it.

1 **TIP:**

The database procedure vid_InsertForSendMail provides the parameter pcAdditionalMessage for sending error messages by email which have been logged by the One Identity Manager Service.

To access this functionality, use the variable [AdditionalMessage] when you set up your failure notification message.

Example of a message

```
Value = "Process failed." & vbCrLf _  
    & vbCrLf _  
    & "-----"  
    & vbCrLf _  
    & "[AdditionalMessage]"
```

Related Topics

- [Setting up the Mail Notification System on page 122](#)
- [Using Scripts on page 453](#)
- [Using #LD Notation on page 464](#)

Process Step Parameters

When you select a process task you specify which action will be executed by the process step. The process task parameter templates are copied to the process step as parameters.

This means that every process step that uses this process task can pass other parameter values. The original is not altered.

Compulsory parameters are immediately entered into the process step when the process task is selected. Then, you need to enter any optional parameters individually. When a parameter is added, the value template is copied from the parameter template. Templates for parameter values are mostly predefined, for example, procedures that evaluate object UIDs and note them accordingly.

Detailed information about this topic

- [Editing Process Step Parameters on page 407](#)
- [Allocating Parameter Values on page 408](#)

Editing Process Step Parameters

To edit process step parameters

1. Open the process in the Process Editor.
2. Click on the element for the process in the process document.
3. Select the **Parameter** view.
This displays all the parameters defined for the process.
4. Check whether the required parameters are assigned and edit the parameters.
You can add, delete and edit parameters.

TIP: You can directly edit the parameters directly by simply clicking once on the entry.

Table 229: Meaning of Icon Used

Icon	Meaning
✓	Mandatory process task parameter.
▢	Optional process task parameter which is assigned to the process step.
✗	Optional process task parameter which is not assigned to a process step.

Table 230: Parameter Properties

Property	Meaning
Name	Name of the parameter. NOTE: The name of a parameter should not be changed. The parameter for the process component "HandleObjectComponent" is an exception to this.

Property	Meaning
Hidden	This option specifies whether the parameter is shown in the One Identity Manager Service log file and in the program "Job Queue Info". Values for hidden parameters are shown as <HIDDEN>. Only "viadmin" system users have permission to see these parameters in Job Queue Info.
Encrypted	Specifies whether the parameter is encrypted when passed (assuming the database is encrypted). Encrypted parameters are shown as <HIDDEN> in the One Identity Manager Service log file and in the program "Job Queue Info".
	<p>NOTE: If the option is already set in the parameter template, the parameter should also be encrypted when it is passed.</p>
Contains encrypted components	Specifies whether encrypted sequences are contained in this value. Use this option to pass complex parameter (for example, PowerShell scripts) containing encrypted sequences (for example, passwords). Encrypted parts of a parameter are shown as <HIDDEN> in the One Identity Manager Service log file and in the program "Job Queue Info".
Value template	Define value templates in VB.Net syntax. When a parameter is added, the value template is copied from the parameter template.
	<p>NOTE: You can restore default value templates in the Process Editor. Select the  button in the View Parameter menu and click in the parameter's edit view on the Template.</p>
Type	Type of parameter. The values IN, OUT and INOUT are permitted. Parameters of type OUT and INOUT are parameters which a process component can use to output a value. This value is then available to subsequent process steps in the process and can be used as a value for IN parameters.

Related Topics

- [Allocating Parameter Values on page 408](#)
- [Using Scripts on page 453](#)

Allocating Parameter Values

Define value templates in VB.Net syntax. The following statements can be used for allocating values:

- None
- Object columns or columns of a related object

Syntax:

```
Value = ${<column name>:<data type>$}
Value = ${FK(<foreign key column>).}<column name>:<data type>$
```

Example:

```
Value = $Lastname$
Value = $PasswordNeverExpires:bool$
Value = $FK(Ident_Domain).Description$
```

- Parameter from the optional parameter collection

Syntax:

```
Value = ${PC(<parameter name>)}
```

Example:

```
Value = ${PC(SRCUID_Application)}
```

- Out parameter

Parameters of type "OUT" or "INOUT" are parameters that can be used by process components to output a value. This value is then available to subsequent process steps in the process and can be used as a value for IN parameters.

When you use OUT parameters you need to take care that these contain data at runtime. Alternatively, when the text is processed "&OUT(<parameter name>)&" is entered, which means that the variable will not be replaced.

Syntax:

```
Value = "&OUT(<Parameter name>)&"
```

Example:

```
Value = "&Out(FileSize)&"
```

- Global variables owned by the set up program

Syntax:

```
Value = Variables("<variable name>")
```

Example:

```
Value = Variables("GENPROCID")
```

```
Value = Variables("FULLSYNC")
```

- Process or process step variables created locally by a pre-script

Syntax:

```
Value = values("Name")
```

Example:

```
Value = Values("FirstHomeServer")
```

- Querying Configuration Parameters

The full path for the configuration parameter always has to be entered.

Syntax:

```
Value = Session.Config().GetConfigParm("<full path>")
```

Example:

```
Value = Session.Config().GetConfigParm("TargetSystem\ADS\RestoreMode")
```

- VB.net

Enter any VB.Net statement.

Related Topics

- [Editing Process Step Parameters on page 407](#)
- [Using Scripts on page 453](#)

Searching for an Entry within a Process

To search for an entry within a process

1. Open the search dialog box using **CTRL + F**.
2. Enter the search text under **Text**.
3. Start the search with **Search**.
4. Use **F3** to search next.

This searches for the text in the process and process steps.

Table 231: Objects and Properties to be Searched

Search in Objects	Properties to be Searched
Process	Name
Process step	Name, description, generating condition, server selection script

Comparing Processes

To find differences between two processes

1. Open the process in the Process Editor.
2. Select **Process | Compare processes...** from the menu.
The current process is already selected as **Process A**.
3. Select the process to compare it with in **Process B**.

4. (Optional) use the button to specify which process properties you want to include in the comparison. By default, all the properties of the processes, process steps and events are compared.
5. Start the comparison with .

Differences in the processes are highlighted in the output text.

TIP: Mark the text and click the button to copy the text to the clipboard.

Exporting and Importing Processes

Exporting and importing processes is implemented through XML files.

To export a process to an XML file

1. Open the process in the Process Editor.
2. Select **Process | Export...** from the menu.
3. Enter the file name and click **Save**.

To import a process from an XML file

1. Select the category **Process Orchestration** in the Designer.
2. Start the import by selecting **Import process** in the task view.
3. Select the XML file and click **Next**.

The process is opened in the Process Editor.

Related Topics

- [Creating and Editing Processes on page 392](#)

Simulating Process Generation

You can use simulation to test whether a selected process can be generated successfully or whether the syntax for passing parameters is correct. This makes it easier to alter processes if necessary.

- NOTE:** The option **Do no generate** is not taken into account when you simulate a process. Disable this option for process simulation.
- TIP:** An entry in the Process Editor toolbar is created for every simulation, which you can use to rerun the simulation without having to specify the simulation data again.

To generate a process for testing

1. Open the process in the Process Editor.
2. Select **Process | View | Simulation view** from the menu.
This starts the process simulation wizard.
3. Click **Next**.
4. Select the event for which the process will be generated and specify the database connection for simulating. Select **Designer database** or **Main database**.
5. Click **Next**.
6. Select the object to simulate with this event.
7. Click **Next**.
8. (Optional) Change the object properties.
9. Click **Next**.
10. (Optional) Define parameters for the parameter collection. Enter the parameter name and value.

NOTE: For processes generated with these parameter collection, you must specify the parameter and the value to pass (for example, parameter SourceDir when copying profiles or parameter ConfigName for loading a target system). No parameter collection is used for processes generated for the default events (insert, update, delete).

Table 232: Actions for Defining Parameters

Action	Description
Load process steps	Loads process steps and their parameters.
Insert	Inserts a parameter in the parameter collection.
Delete	Deletes a parameter from the parameter collection.

11. Click **Next**.
12. (Optional) Specify global session object variables to use for simulation, for example "FullSync". Enter the parameter name and the parameter value.

Table 233: Actions for Defining Variables

Action	Description
Insert	Inserts a variable.
Delete	Deletes a variable.

13. Click **Next**.

14. Select preprocessor conditions to be taken into account when the process is generated.
15. Click **Next**.
16. Start the simulation with **Finish**.

The simulation process can take some time. The assemblies generated are saved locally on the workstation on which the simulation is executed. A simulation does not, therefore, have any effect on other users.

After the simulation is complete the generated process is shown in the Process Editor. The process steps are shown in color depending on the generation result.

Table 234: Simulation Color Code

Color	Meaning
Gray	Process step not generated.
Blue	Process step successfully generated.

- NOTE:** Double-click on a successfully generated process step to display properties and parameters with the simulated data in the edit view.
- NOTE:** You can swap between the edit view and the simulation view using the menu **Process | View** in order to make any further changes.

To display a process generator log

- Select **Process | View | process generator log**.

Checking the Validity of a Process

- NOTE:** Before you compile a process, you should carry out a validity check of the process and process steps.

To check a process

1. Open the process in the Process Editor.
2. Select **Process | Error checking**.

The result of the check is shown in the view **Validity Check** and is retained until the next validity check.

Table 235: Icon used in the Validity Check

Icon	Meaning
	No error found.
	Error
	Warning, information.

- ❶ **TIP:** If you double-click on an error message in the **Validity check** view, you jump to the corresponding entry in the process document which you can then edit.
- ❷ **TIP:** Process or process step elements are highlighted in yellow to indicate a warning or information. If a error occurs, the elements are highlighted in red.

Table 236: Possible Reasons for Process Failure

Error Category	Possible Cause
Error	The process does not have a name. No base object given. The given generating condition does not correspond to required notation (value =).
Warning	The process does not have a base process step. The process has no event.
Information	The option Do not generate is set.

Table 237: Possible Reasons for Process Step Failure

Error Category	Possible Cause
Error	The process step does not have a name. No process task assigned. The given generating condition does not correspond to required notation (value =). No execution server specified (server selection script or server mask). Process step name not unique. Process step has no parameters. The given parameter value does not correspond to required notation (value =).
Warning	Process step not linked into the process.

Related Topics

- [Compiling a Process on page 415](#)

Compiling a Process

Once you have created, imported or made changes to a process, you need to compile it. The process cannot be generated until it has been compiled.

NOTE: Before you compile a process, you should carry out a validity check of the process and process steps.

Compiling takes place for each base object, that means that all processes that belong to a base object are translated. The assemblies are created and placed on the workstation where generating will take place. During translation, the source is checked for errors. This process may required some time.

There are two methods for compiling a process in the Process Editor:

Local Compiling

Use this method to compile a process for testing.

To compile a process for testing

1. Open the process in the Process Editor.
2. Select **Process | Compile** from the menu..

Compiling and Saving Assemblies to the Main Database

If the process has been test compiled, use this method to add assemblies that are generated into the main database after compiling the process. Once the changes have be integrated the altered processes are immediately available in the system.

To compile a process and save the assemblies to the main database

1. Open the process in the Process Editor.
2. Select **Process | Compile and save to database** from the menu.

Displaying Errors

Error messages during compiling are displayed in the "Compiler errors" view. The source code is displayed if errors occur during compilation. This view is only for displaying the source code. It cannot be edited here.

NOTE:

If several users edit processes of the same base object, any error messages are also sent to other users. However, these cannot be changed by the current user.

A double-click on the error message in the "Compiler error" view takes you straight to the corresponding line in the process. Here, you can edit it.

When you double-click on a message in the "Compiler errors" view, it jumps to the corresponding row in the source code view.

Related Topics

- [Checking the Validity of a Process on page 413](#)

Executing Processes Automatically

Set up process plans to execute cyclical processes to put into effect, for example, regular synchronization with a target system environment. Process plans are connected to schedules and can therefore be executed at regular intervals.

The following steps are necessary to execute processes automatically:

1. Creating a Process Plan

A process plan contains the basic configuration for automatically running a process.

2. Setting up and configuring a schedule

A schedule include the configuration of execution times for executing processes regularly.

Detailed information about this topic

- [Creating a Process Plan on page 418](#)
- [Setting Up and Configuring Schedules on page 134](#)

Working with the Process Plan Editor

A separate editor is used for creating process tasks. The editor is started from the program "Designer" and opens in the document view. Only additional Process Editor functions are described in the following.

Menu Items

The following items are added to the menu bar when the editor starts.

Table 238: Meaning of Items in the Menu Bar

Menu	Menu Item	Meaning
Process plan	New	Adds a new process plan.
	Delete	Deletes the selected process plan after requesting confirmation.
	Start process plan now	The selected process plan is executed immediately. A process for executing is queued in the One Identity Manager database.
	Show captions	Technical name or captions are displayed in the user's login language in the list view.
	Refresh	Updates the list view. The process plan start times are reloaded from the One Identity Manager database.
View	Properties	Shows/hides the edit view.
Help	Help for process automation	Opens the editor help.

Table 239: Meaning of the Icons in the Toolbar

Icon	Meaning
	Reload and refresh the list view.
	Creates a new process plan.
	Deletes the process plan.
	Toggles between technical identifiers or captions in the user's login language.
	Executes the process plan.

Views in the Editor

The editor has several views for displaying and editing schedules.

Table 240: Editor Views

View	Description
List view	The editor list view displays all the process plans, the time they were last executed and the next planned execution time.
Edit view	You can edit the properties for a process plan in the edit view. There is a default context menu available for the input fields.

Table 241: Meaning of List View Icons

Icon	Meaning
	The process plan schedule is not enabled.
	The process plan was executed according to plan.
	The process plan was not executed. This state can occur if the task could not be executed to plan or if the schedule was reenabled and the time had not been reached for the initial run.

Table 242: List View Context Menu Items

Context Menu Item	Meaning
Add process plan	Adds a new process plan.
Delete process plan	Deletes the selected schedule.
Edit process plan	Edits the new schedule.
Edit process	Open the editor for the process which is executed by the process plan.
Execute	The selected process plan is executed immediately. A process for executing is queued in the One Identity Manager database.
Select columns...	Opens a dialog window for selecting columns to be displayed in the list. Specify which properties should be shown in the list and the order they should be shown in. You can enter the column width and alignment of the column description.
Navigation	Shows all other editors that can be used with the selected object.

Creating a Process Plan

A process plan contains the basic configuration for automatically running a process.

To edit a process plan

1. Select the category **Process Orchestration | Process automation** in the Designer.
2. Start the editor using the task **Edit process plans**.

3. Create a new process plan using **Process plan | New**.

- OR -

Select an existing process plan.

4. Edit the process plans master data.

TIP: You can execute the process plan immediately from the context using **Execute** or **Process plan | Start process plan now**. The process is queued in the One Identity Manager database.

TIP: You can see which process is triggered from the context item **Edit process**.

Table 243: Process Plan Properties

Property	Meaning
Name	Name of the process plan. Translate the given text using the  button.
Table	Base object (table) for which the process plan will run.
Event	Event to be executed. All base object events are listed for new process plans.
Activation schedule	Schedule that contains the execution time for the process plan. NOTE: Create a new schedule using  next to the menu.
Max.execution time (hours)	Enter the number of hours after which the process plan should automatically quit.
Description	Enter a detailed description of the process plan.
Condition	Limiting condition for elements to which the scheduled task will be applied. The input must satisfy the "WHERE clause" database query syntax.
Parameters	List of parameters that are set when the process is generated from this process plan.

Related Topics

- [Setting Up and Configuring Schedules on page 134](#)
- [Creating and Editing Events on page 397](#)

Process Components

Process components and their process tasks form a framework that all process steps can be based on. The tables Jobcomponent, JobTask and Jobparameter define the complete range of One Identity Manager's own process components and process task with the associated parameters.

Process tasks are used to carry out single basic jobs at system level, for example, adding directories. A process component consists of one or more process tasks and its parameters.

When a process is created, the parameter templates for the process task are copied and entered in the process step. This means that every process step that uses this process task can pass other parameter values. The original is not altered.

NOTE: The information available for the process components is added through migration and cannot be edited.

To obtain a complete overview of process components and their process tasks and parameters

- Select the report **Process components** in the category **Documentation | System configuration reports** in the Designer.

To display individual process components and their process tasks and parameters

- Select the category **Process Orchestration | Process automation** in the Designer.

The following table contains short descriptions of the process components.

Table 244: Short Descriptions of Process Components

Component	Description
AutoUpdateComponent	This process component maps the One Identity Manager Service built-in-tasks.
CommandComponent	This process component runs any command.
DelayComponent	This process component controls the start time of the following process steps.
FileComponent	This process component creates, deletes, copies and modifies file and directories and also their access permissions. The program RSync is required as prerequisite for using the process component under Linux (download from: http://www.itefix.no/i2/index.php or http://sourceforge.net/project/showfiles.php?group_id=69227&package_id=68081).
	The program XCacls is required as prerequisite for setting permissions. You can find this in the your server installation resource kit.
FtpComponent	This process component can transfer file by FTP.
HandleObjectComponent	This process component runs default and custom events for database objects. Each assigned default process is

Component	Description
	generated as in the front-ends (for example, the Manager). The component also makes it possible to initiate so called CustomEvents for triggering object related generation of a special process.
LogComponent	This process component is used to log messages, for example, in the result log.
MailComponent	This process component can send emails.
PowerShellComponent	This process is used for calling Windows PowerShell. Version 2.0 of Windows PowerShell must be installed.
PowershellComponentNet4	This process is used for calling a .NET 4 Windows PowerShell. A version later than 2.0 of Windows PowerShell must be installed.
ProjectorComponent	This process component contains tasks for synchronizing and provisioning data with the One Identity Manager database.
ReportComponent	This process component can create reports and export them in various file formats.
ScriptComponent	This process component runs the scripts from the assemblies.
SQLComponent	This process component runs SQL queries and can be used to determine the number of data records and the existence of data records.
SubversionComponent	This process component runs the sub version operations. The process component only runs under 64-bit operating systems. Prerequisites for using Process Components
	<ul style="list-style-type: none"> • The program SharpSVN, version 1.8.9 is installed. The program is not part of One Identity Manager. • The version for Microsoft .NET Framework 4.0 is stored in the One Identity Manager Service installation directory.
ZipComponent	This process component creates or unpacks ZIP files.

Detailed information about this topic

- [Properties of Process Components, Process Tasks and Parameter Templates on page 422](#)

Properties of Process Components, Process Tasks and Parameter Templates

Table 245: Process Component Properties

Property	Meaning
Display name	Name of component for displaying.
Component class	Component class.
Assembly name	Name of the component.
Description	Description of component functionality.
Remarks	Additional remarks about the process component.
Max. instances	This value specifies the maximum number of instances in which this process component is allowed to run in a queue in the Job server. Permitted values: <ul style="list-style-type: none">• -1: All instances of this process component are processed sequentially. It must be ensured that these components are run exclusively on one Job server, which means no other queue can exist to process these components.• 0: All instances of this process component can be process at the same time.• 1 or -1: Exact number of instances of a process component, which can be processed at the same time. <p>NOTE: The value is only used if the maximum number of instances of a process function is set to "0". Otherwise, the value applies that is set for the process task.</p>
Configuration	Definition of possible additional options for the component in XML syntax.

Table 246: Process Task Properties

Property	Meaning
Name	Name of the process task.
Operating system class	Specifies the operating system on which the process task can be run. Permitted values are "Win32", "Linux" and "ALL" where the value "ALL" specifies that this process function can be run on any operating system.

Property	Meaning
Execution type	The execution type specifies whether the process components for the process task should be executed in by One Identity Manager Service (internal) or in its own process (external).
Description	Description of the process task.
Max. instances	This value specifies the maximum number of instances that can be run by One Identity Manager Service in parallel per process task. Permitted values: <ul style="list-style-type: none"> • -1: All instances of this process task are processed sequentially. • 0: The maximum number of instances given for the process component is used. • 1 or -1: Exact number of instances of a process task, which can be processed at the same time.
Last step in the partial process tree	Specifies whether a process task is principally marks the end of a partial process tree.
Component	Process component to which the process function belongs.
Direct database connection required	Specifies whether a process task requires a direct database connection.
Exclusive per object	Specifies whether execution of the process task is done exclusively per object. If this option is set, only one specific object is ever executed for a process step with this process function. There is no parallel processing.

Table 247: Parameter Template Properties

Property	Meaning
Name	Name of the parameter.
Value template	Default template for finding values. When a parameter is added to a process step, the value template is taken from the parameter template. Define value templates in VB.Net syntax.
Value template (example)	Example of the value template.
Description	Description of the parameter.
Type	The values IN, OUT and INOUT are permitted. Parameters of type OUT and INOUT are parameters which a process

Property	Meaning
	component can use to output a value. This value is then available to subsequent process steps in the process and can be used as a value for IN parameters.
Optional	Labels parameter as a mandatory or optional parameter.
Hidden	This option specifies whether the parameter is shown in the One Identity Manager Service log file and in the program "Job Queue Info". Values for hidden parameters are shown as <HIDDEN>. Only "viadmin" system users have access permission to see these parameters in Job Queue Info.
Encrypted	This option specifies whether the parameter is encrypted when it is passed.
Contains encrypted components	Specifies whether encrypted sequences are contained in this value.
Process task	Process task to which the parameter belongs.

Tracking Changes with Process Monitoring

With the One Identity Manager it is possible to create a change history for objects and their properties. This can be used to fulfil reporting duties for internal committees and legal obligations for providing documentary evidence. Different methods can be used to track changes within the One Identity Manager. With this combination of methods, all changes that are made in the One Identity Manager system can be traced.

- Recording changes to data

Data changes can be recorded for add or delete operations on objects and up to and including changes to individual object properties.

- Recording process information

Recording process information allows all processes and process steps to be tracked while being processed by One Identity Manager Service.

- Recording Messages in the Process History

In the process history, success and error messages from handling each process step in the Job queues are recorded by the One Identity Manager Service.

All entries logged in One Identity Manager are initially saved in the One Identity Manager database. The proportion of historical data to total volume of a One Identity Manager database should not exceed 25%. Otherwise performance problems may arise. You must ensure that log entries are regularly removed from the One Identity Manager database and archived. For more information about archiving data, see the One Identity Manager Data Archiving Administration Guide.

Detailed information about this topic

- [Basics for Process Monitoring on page 426](#)
- [Logging Data Changes on page 427](#)
- [Logging Process Information during Process Handling on page 429](#)
- [Recording Messages in the Process History on page 433](#)
- [Archiving and Deleting Recordings on page 439](#)

Basics for Process Monitoring

Table 248: Configuration Parameters for Process Monitoring

Configuration parameter	Effect
Common\ProcessState	Process monitoring can be configured if the configuration parameter is set. The data is displayed in the Manager process view.

To use process monitoring in One Identity Manager.

- Set the configuration parameter "Common\ProcessState".
- You can control the extent of the logging using the configuration settings for each method.

The methods implemented by the One Identity Manager allows all modifications to the system that are triggered by a user action to be monitored. Each action in One Identity Manager is labeled with a unique ID number. This ID number is called a GenProcID. All changes that can be traced back to the same cause are given the same GenProcID and are grouped in this way. If a previously stored action does not pass a GenProcID to the current action, a new ID is automatically created.

If an action is triggered from the One Identity Manager's object layer the GenProcID is written to the context data of the database connection. The logged in user is also noted in the context data and is made available in this way.

A new GenProcID is generated by the trigger if an action takes place directly in the database or through an application that works without the One Identity Manager object layer. This GenProcID is valid for the duration of the database connect, which means that all changes belong to the same action and link to the same GenProcID. The user data is made up of the database user's name, the MAC address and the workstation name as well as the application name.

All actions (process triggers) that cause changes to the system, and their actual status information are logged internally in the status table DialogProcess. Logging takes place independent of the chosen change history method. This log writing therefore provides a starting point for monitoring and allows the changes based on one action to be grouped together.

The following information is recorded for one action:

- ID number (GenprocID)
- Display name for the action
- Base object that the action is triggered for
- User that triggered the action
- Time of action
- Object key for selecting the process trigger

- Comment on the action
- Current process status

NOTE: The information is displayed in the Manager's process view. For more information, see the One Identity Manager User Guide for One Identity Manager Tools User Interface and Default Functions.

Detailed information about this topic

- [Logging Data Changes on page 427](#)
- [Logging Process Information during Process Handling on page 429](#)

Logging Data Changes

Table 249: Configuration Parameter for Logging Data Changes

Configuration parameter	Effect
Common\ProcessState	Process monitoring can be configured if the configuration parameter is set. The data is displayed in the Manager process view.
Common\ProcessState\PropertyLog	When this configuration parameter is set, changes to individual values are logged and shown in the process view.
Common\ProcessState\PropertyLog\AllDefaultPropertiesForModel	If this configuration parameter is set, the most important columns of the One Identity Manager schema to monitor are labeled for logging.
Common\ProcessState\PropertyLog\AutoTrackAlternatePK	If the configuration parameter is set, properties are logged even if parts of an alternative key change. This only applies for tables that are transportable and where at least one property is labeled for logging. This configuration parameter only affects system components.
Common\ProcessState\PropertyLog\AutoTrackAlternatePK\Payload	If the configuration parameter is set, properties are still logged even if parts of an alternative key change. This only applies for tables that are transportable and where at least one property is labeled for logging. This configuration parameter only has an effect on user components.

Add, change and delete operations can be recorded for objects. The trigger GenProcID is passed as well, so that the changes to one object can be grouped together.

NOTE: Displaying an object's change history is done in the Manager process view. For more information, see the One Identity Manager User Guide for One Identity Manager Tools User Interface and Default Functions.

The following prerequisites are required to log data changes:

- Set the configuration parameter "Common\ProcessState" in the Designer.
- Set the configuration parameter "Common\ProcessState\PropertyLog" in the Designer.
- Label columns for which changes will be logged.
- Label columns to be logged when an object is deleted.

TIP: If the configuration parameter "Common\ProcessState\PropertyLog\AllDefaultPropertiesForModel" is set, One Identity Manager schema columns labeled for logging changes and deletions. Define which columns are affected in the table QBMVDefaultHistoryColumns.

To log a column

1. Select the category **One Identity Manager Schema** in the Designer.
2. Select the table and start the Schema Editor with the task **Show table definition**.
3. Select the column and select the tab **More**.
 - To log changes to data in the column, set the option **Log changes**.
 - To log the deleting data in the column, set the option **Log changes when deleting**.

The data changes are stored in the tables DialogWatchOperation and DialogWatchProperty. An entry is also created in the status table DialogProcess for the triggering action.

The following information is collected for these operations:

- Adding an object

When a new object is added, the object key, object display name, date of insertion and user are logged.

- Changing an object

When a column is changed the old value, change date and user are logged. Changes to properties that belong to the alternative foreign key are recorded depending on the configuration parameters

"Common\ProcessState\PropertyLog\AutoTrackAlternatePK" and
"Common\ProcessState\PropertyLog\AutoTrackAlternatePK\PayLoad".

- Deleting an object

When an object is deleted, the columns to be logged an all primary key columns are logged. The value, deletion date and user are logged.

Related Topics

- [Mapping Column Definitions on page 188](#)
- [Basics for Process Monitoring on page 426](#)
- [Logging Process Information during Process Handling on page 429](#)

Logging Process Information during Process Handling

Table 250: Configuration Parameters for Logging Process Information

Configuration parameter	Effect
Common\ProcessState	Process monitoring can be configured if the configuration parameter is set. The data is displayed in the Manager process view.
Common\ProcessState\ProgressView	When the parameter is set, changes in the context of process monitoring and display of information takes place in the process view.

The following prerequisites are required to log process information during process execution:

- Set the configuration parameter "Common\ProcessState" in the Designer.
- Set the configuration parameter "Common\ProcessState\ProgressView" in the Designer. Specify the scope of logging through the configuration parameter option.

Table 251: Scope of Process Logging

Option	Description
1	Full process tracking Process information from all processes marked for process tracking is logged.
2	Web Portal tracking Only process information for process marked for process tracking the Web Portal is logged. (default)
<ul style="list-style-type: none">• Label the process and process steps for process tracking and define templates for event, process and process step process information.	

NOTE: The information is displayed in the Manager's process view. For more information, see the One Identity Manager User Guide for One Identity Manager Tools User Interface and Default Functions.

If the configuration parameter "Common\ProcessState\ProgressView" is enabled, the Job Generator creates an entry in the status table while generating process information for processes and process steps.

Right at the start, the Job Generator uses the GenProcID for the generating operation. If there is no GenProcID passed at runtime, a new one is automatically created. This ID is written to the global variable "GENPROCID" for the current database connection object before the process is generated. Therefore, it can be used by all processes. All partial steps that are triggered by a generating operation are grouped together in this way and logged. Bulk operations such as synchronization and CSV import are an exception. In this case, a new GenProcID is created for each individual step in tracking the object changes and not for the process as a whole.

An entry is set up in the status table DialogProcessStep for each process step that is marked for tracking. For each process that has at least one such process step, an entry is made in the status table DialogProcessChain. For each generating operation that has caused an entry in the status table DialogProcessChain, an entry is written to the status table DialogProcess. At the same time, the Job Generator creates the display name for the process view by executing the given VB.Net expression for the process information.

The possible processing states and additional information available for the respective processing statuses are listed in the following tables.

Table 252: Possible Process States

Process State	Description
Initial	<generated> ::= "G"
End of processing	<p><finalstate> ::= <ended> <failed> <not executed></p> <p>where:</p> <ul style="list-style-type: none"> <ended> ::= "E" (processing successful) <failed> ::= "F" (processing unsuccessful) <not executed> ::= "N" (no longer accessible during processing)
In progress	<p><workingstate> ::= <delayed> <processing> [<ProcessStateAddON>]</p> <p>where:</p> <ul style="list-style-type: none"> <delayed> ::= "D" (processing delayed) <Long delayed> ::= "L" (processing was put on hold) <processing> ::= "P" (in progress) <ProcessStateAddON> (optional additional information)

Table 253: Possible Additional Information

Additional Information	Description
Processing deferred	<datetime> ::= <YYYY> - <MM> - <DD> <HH> : <NN> :

Additional Information	Description
until	<SS> where: <YYYY> ::= 1980..9999 <MM> ::= 01..12 <DD> ::= 01..31 <HH> ::= 00..23<NN> ::= 00..59 <SS> ::= 00..59
Retries	<retryinfo> ::= 1..99

Related Topics

- [Setting Up Process Information for Process Handling on page 431](#)
- [Basics for Process Monitoring on page 426](#)
- [Logging Data Changes on page 427](#)

Setting Up Process Information for Process Handling

You can set up templates for creating process information for processes, process steps and events with the Process Editor in the Designer. Use #LD notation for language dependent definition of process information.

To edit process information a process

1. Open the process in the Process Editor in the Designer.
2. Click on the element for the process in the process document.
3. Select how the process information should be logged by selecting a value from the **Process information** menu.

Table 254: Permitted Values

Value	Description
None	The process information is not logged.
Full process tracking	The process information is logged and displayed in the Manager.
Web Portal tracking	The process information is logged and displayed in the Manager and in the Web Portal.

- Enter the template for process information in the **Process information** text box.

To edit process information for a process

- Open the process in the Process Editor in the Designer.
- Click on the element for the process step in the process document.
- Select how the process information should be logged by selecting a value from the **Process information** menu.

Table 255: Permitted Values

Value	Description
none	The process information is not logged.
Full process tracking	The process information is logged and displayed in the Manager.
Web Portal tracking	The process information is logged and displayed in the Manager and in the Web Portal.

- Select **Detail depth** for the process information on the **Process tracking** tab. The following options are available: "Basic information", "Advanced information" and "Full information".

You use depth of detail to control how process information is displayed in the Manager's process view. This provides users with different layers of detail for process information depending on the configuration in the Manager's user interface.

- Enter the template for process information in the **Process information** text box.

To edit process information for events

- Open the process in the Process Editor in the Designer.
- Click on the element for the process in the process document.
- Select the **Events** view.
- Select the event and option the edit dialog box with .
- Enter the template for process information in the **Event process information** text box.
- Click **OK**.

IMPORTANT: At least one event process must have process tracking enabled in order to generate process information for events.

If several processes point to one event, the event with a process information template is found that has the lowest generating order specified in its process configuration. This template is evaluated and shown in the process view.

If there no template is available, the information is evaluated as follows:

<table> - <event> - <object display name>

Related Topics

- [Using #LD Notation on page 464](#)
- [Creating and Editing Processes on page 392](#)
- [Creating and Editing Events on page 397](#)
- [Creating and Editing Process Steps on page 399](#)

Recording Messages in the Process History

Table 256: Configuration Parameter for Logging Messages in the Process History

Configuration parameter	Meaning
Common\ProcessState\JobHistory	Logs entries in the table JobHistory.

Processes being handled are logged in the process history. The process history can be analyzed with the help of the "Job Queue Info" program.

To log messages to the process history

- Set the configuration parameter "Common\ProcessState\JobHistory" in the Designer and specify a range for the log entries.

Table 257: Permitted Values for the Configuration Parameter "Common\ProcessState\JobHistory"

Value	Meaning
NO	No messages are written to the process history.
ALL	All handled process steps are written to the process history.
ERROR	Only failed process steps are written to the process history.
ERROROrSELECTED	Failed process steps and process steps marked with the option Process history are logged in the process history.
SELECTED	Only process steps marked with the option Process history are logged in the process history.

Log entries in the process history are exported from the One Identity Manager database at regular intervals. One Identity Manager provides various methods to do this. [For more information, see Archiving and Deleting Recordings on page 439](#).

Related Topics

- [Monitoring Process Handling with Job Queue Info on page 611](#)

Process Tracking for DBQueue Processor Operations

In order to track inherited calculations as a result of changes to the system, the GenProcID is always passed to the DBQueue Processor operation. There may only be one entry in the DBQueue for each operation and object in case of follow-on operations. To map such processes, a new GenProcID is issued and used in subsequent processes. The conflicting processes and their GenProcID's are saved in the table DialogProcessSubstitute.

When a new GenProcID is created for conflicting processes, the following rules apply:

- Several of the same DBQueue Processor operations on one object are merged into one process (one GenProcID). This uses existing substitute processes if the number is identical to the predecessor (with respect to the root process).
- If further conflicts occur in the sequence, the GenProcIDs that have already been replaced are reset to the original and a new substitute is created.
- A substitute is only valid for one set of original processes.

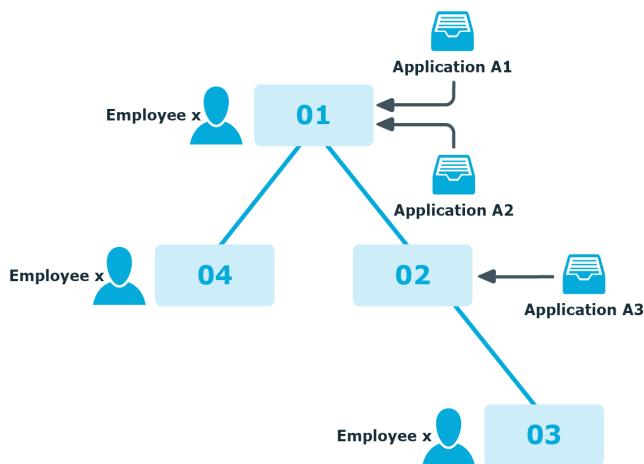
Related Topics

- [Example for Replacing the GenProcID on page 434](#)
- [Processing DBQueue Tasks on page 645](#)

Example for Replacing the GenProcID

A hierarchical role structure exists which consists of 4 roles O1, O2, O3 and O4. Employee X is assigned to roles O1, O4 and O3. The assignment of applications to roles is depicted in the following:

Figure 47: Role Structure as in the Example Above



Three processes run between two DBQueue Processor runs, each with its own GenProcID:

- P1: Application A1 is assigned to the role O1
- P2: Application A2 is assigned to the role O1
- P3: Application A3 is assigned to the role O2

The following operations are in the DBQueue (table DialogDBQueue) and in the process information:

Operation	Object	GenProcID
OrgHasApp	O1	P1
OrgHasApp	O1	P2
OrgHasApp	O2	P3

The operation OrgHasApp cannot be subdivided with respect to O1 because the union of the applications is not yet computed. At this point, there is no more information available as to which GenProcID has been entered for which application by the assignment.

In order to achieve uniqueness for the combination of operation and object, a new GenProcID P4 is introduced and the two O1 operations are compacted into this GenProcID. P1 and P2 are noted in the table DialogProcessSubstitute as possible predecessors of P4 (but not clearly in the individual actions).

Operation	Object	GenProcID
OrgHasApp	O1	P4
OrgHasApp	O2	P3

The following constellations can occur depending on whether the operation OrgHasApp is processed as single step or in bulk:

Case 1) O1 is calculated and then O2.

Case 2) O2 is calculated and then O1.

Case 3) O1 and O2 are calculated together in a bulk operation.

After these operations have been executed and assuming that they all cause changes to the total sets affected the following situation arises:

Case 1) O1 is calculated and then O2.

Operation	Object	GenProcID
OrgHasApp	O2	P3
OrgHasApp	O4	P4
OrgHasApp	O2	P4
OrgHasApp	O3	P4
PersonHasApp	X	P4

Before the next DBQueue Processor run the GenProcID's must be compressed again, because the OrgHasApp operation did not produce a unique result for the object O2. P5 is introduced with possible predecessors P4 and P3.

Operation	Object	GenProcID
OrgHasApp	O2	P5
OrgHasApp	O4	P4
OrgHasApp	O3	P4
PersonHasApp	X	P4

Now the calculation is done for O2:

Operation	Object	GenProcID
OrgHasApp	O3	P5
PersonHasApp	X	P5
OrgHasApp	O4	P4
OrgHasApp	O3	P4
PersonHasApp	X	P4

Because O3 is not unique, P6 is introduced with possible predecessors P4 and P5.

Operation	Object	GenProcID
OrgHasApp	O3	P6
PersonHasApp	X	P5
OrgHasApp	O4	P4
PersonHasApp	X	P4

After O3 and O4 have been calculated we have the following situation:

Operation	Object	GenProcID
PersonHasApp	X	P6
PersonHasApp	X	P5
PersonHasApp	X	P4

There is no uniqueness for object X such that P7 is introduced with possible predecessors P4, P5 and P6.

Case 2) O2 is calculated and then O1.

Operation	Object	GenProcID
OrgHasApp	O1	P4
OrgHasApp	O2	P3

After execution the following entries are in the DBQueue:

Operation	Object	GenProcID
OrgHasApp	O1	P4
OrgHasApp	O3	P3

The following situation is the result after the next step:

Operation	Object	GenProcID
OrgHasApp	O3	P3
OrgHasApp	O4	P4
OrgHasApp	O2	P4
OrgHasApp	O3	P4
PersonHasApp	X	P4

To achieve uniqueness for O3 a process P5 with possible predecessors P3 and P4 is created:

Operation	Object	GenProcID
OrgHasApp	O3	P5
OrgHasApp	O4	P4
OrgHasApp	O2	P4
PersonHasApp	X	P4

After calculating we have the following situation:

Operation	Object	GenProcID
PersonHasApp	X	P5
PersonHasApp	X	P4

There is no uniqueness for object X such that P6 is introduced with possible predecessors P4 and P5.

Case 3) O1 and O2 are calculated together in a bulk operation.

Operation	Object	GenProcID
OrgHasApp	O1	P4
OrgHasApp	O2	P3

After the first step in the calculation the following entries are in the DBQueue:

Operation	Object	GenProcID
OrgHasApp	O4	P4
OrgHasApp	O2	P4
OrgHasApp	O3	P4
OrgHasApp	O3	P3
PersonHasApp	X	P4

Uniqueness is achieved for O3 by introducing P5 with possible predecessors P3 and P4:

Operation	Object	GenProcID
OrgHasApp	O4	P4
OrgHasApp	O2	P4
OrgHasApp	O3	P5
PersonHasApp	X	P4

After the next step in the calculation, the following content is found

Operation	Object	GenProcID
OrgHasApp	O3	P4
PersonHasApp	X	P4
PersonHasApp	X	P5

After O3 has been reached in the next run and has not created a new PersonHasApp entry, only X exists with P4 and P5 because X already exists with P4.

Operation	Object	GenProcID
PersonHasApp	X	P4
PersonHasApp	X	P5

There is no uniqueness for object X such that P6 is introduced with possible predecessors P4 and P5.

Archiving and Deleting Recordings

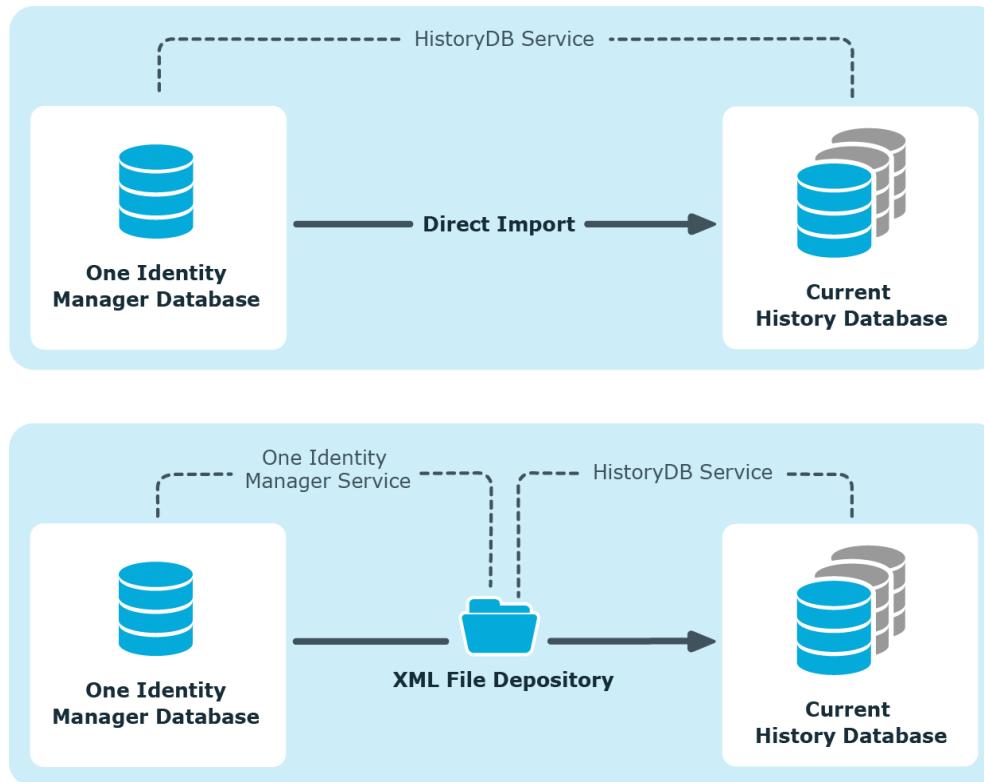
All entries logged in One Identity Manager are initially saved in the One Identity Manager database. The proportion of historical data to total volume of a One Identity Manager database should not exceed 25%. Otherwise performance problems may arise. You must ensure that log entries are regularly removed from the One Identity Manager database and archived.

The following methods are provided for regularly removing data recorded from the One Identity Manager database:

- The data can be transferred directly from the One Identity Manager database into a One Identity Manager History Database. This is the default procedure for data archiving. Select this method if the servers on which the One Identity Manager database and the One Identity Manager History Database are located have network connectivity.

- The data can be exported in XML files. These can be loaded into the One Identity Manager History Database on a scheduled basis. Use this method if the One Identity Manager database and the One Identity Manager History Database are not in the same network segment. Alternatively you can load the XML files into another archiving system provided by the company.
- The data is deleted from the One Identity Manager database after a certain amount of time without being archived.

Figure 48: Transferring Records to the One Identity Manager History Database



All records in the One Identity Manager database that are triggered by an action are grouped together into a process group based on an ID number, the GenProcID for direct transfer to a History Database or for exporting to XML files. The exported process groups along with the associated records are deleted from the One Identity Manager database once the export has been successfully completed.

The following conditions have to be met to facilitate direct transfer to a One Identity Manager History Database or to export XML files:

- The subsection of records is configured for export.
- The retention period for all records that belong to a process group has ended, not taking into account whether the section of record is labeled for export or not.
- There are no processes enabled with the process group GenProcID in the DBQueue, Job queue or as planned operations.

- There is at least one record in the subsection of records for the triggered action that should be exported.

Both databases for archiving records in a One Identity Manager History Database - the One Identity Manager database and the One Identity Manager History Database - have to be configured. For more detailed information about archiving data in the History Database, see the One Identity Manager Data Archiving Administration Guide.

Selecting a Method

Select the basic procedure by setting the configuration parameter "Common\ProcessState\ExportPolicy". If the configuration parameter is disabled, the data remains in the One Identity Manager database. If the configuration parameter is enabled, the selected procedure is applied.

Table 258: Permitted Values for the Configuration Parameter "Common\ProcessState\ExportPolicy"

Value	Meaning
FILE	The data is exported to XML files after a specified time period has expired.
HDB	The files are transferred directly to the One Identity Manager History Database after a specified time period has expired.
NONE	The data is deleted in the One Identity Manager database after the specified time period has expired.

After selecting the basic procedure, you can specify whether data is exported or deleted for each subsection of records individually. You use configuration parameters to make the choice for each subsection.

Table 259: Configuration Parameter for Handling Change Data

Configuration parameter	Meaning
Common\ProcessState\PropertyLog\IsToExport	Exports the data changes. If this configuration parameter is not set the information is deleted once the retention period has expired.
Common\ProcessState\PropertyLog\LifeTime	This configuration parameter specifies the maximum retention period in the database for log entries from change tracking.

Table 260: Configuration Parameter for Handling Process Information

Configuration parameter	Meaning
Common\ProcessState\ProgressView\IsToExport	Exports the data in the process information. If this configuration parameter is not set the information is deleted once the retention period has expired.
Common\ProcessState\ProgressView\LifeTime	This configuration parameter specifies the maximum length of time that log data from process information can be kept in the database.

Table 261: Configuration Parameter for Handling Process History

Configuration parameter	Meaning
Common\ProcessState\JobHistory\IsToExport	Exports the information in the process history. If this configuration parameter is not set the information is deleted once the retention period has expired.
Common\ProcessState\JobHistory\LifeTime	This configuration parameter specifies the maximum retention period in the database for log entries from process history.

Specifying Data Retention Periods

Once the retention period has ended, the recorded data is either exported or deleted from the One Identity Manager database depending on which archiving method has been chosen. A longer retention period should be selected for subsections whose records will be exported than for those that will be deleted.

NOTE: If you do not specify a retention period, the records for this subsection will be deleted daily from the One Identity Manager database within the DBQueue Processor daily maintenance tasks.

The recordings are not exported until the retention period for all subsections has expired and no other active processes for the process group (GenProcID) exist in the DBQueue, process history or as planned operation.

Example 1

Records are transferred directly to the One Identity Manager History Database. The following configurations are selected for each subsection:

Configuration	Process Information	Process History	Data Changes
Export data	No	No	Yes
Retention period	3 days	4 days	5 days

This results in the following sequence:

Time	Process Information	Process History	Data Changes
Day 3	Data is deleted from the One Identity Manager database	No action	No action
Day 4	-	Data is deleted from the One Identity Manager database	No action
Day 5	-	-	Data is transferred to the One Identity Manager History Database and then deleted from the One Identity Manager database

Example 2

Records are transferred directly to the One Identity Manager History Database. The following configurations are selected for each subsection:

Configuration	Process Information	Process History	Data Changes
Export data	Yes	No	Yes
Retention period	3 days	4 days	5 days

This results in the following sequence:

Time	Process Information	Process History	Data Changes
Day 3	No action because the retention period has not ended for all subsections	No action	No action
Day 4	No action because the retention period has	Data is deleted from the One Identity Manager database	No action

Time	Process Information	Process History	Data Changes
	not ended for all subsections	Identity Manager database	
Day 5	Data is exported and then deleted	-	Data is transferred to the One Identity Manager History Database and then deleted from the One Identity Manager database

Configuring the One Identity Manager History Database for Archiving

- Enable the configuration parameter "Common\ProcessState\ExportPolicy" in the Designer and enter the value HDB.
- Configure the subsections for export and define a retention period.
- Check the value of the configuration parameter "Common\ProcessState\PackageSizeHDB". This parameter specifies the maximum number of process groups to be transferred to the History Database. The default value is 10000.
- Configure the One Identity Manager History Database for importing data. For more detailed information, see the One Identity Manager Data Archiving Administration Guide

Related Topics

- [Selecting a Method on page 441](#)

Configuring Archiving with XML Files

- Enable the configuration parameter "Common\ProcessState\ExportPolicy" in the Designer and enter the value FILE.
- Configure the subsections for export and define a retention period.
- Specify a repository for the XML files and the executing server using the configuration parameters "Common\ProcessState\ExportPolicy\ExportPath" and "Common\ProcessState\ExportPolicy\ExportServer".
- Exporting is carried out at regular intervals by the One Identity Manager Service. Configure and enable the schedule "Export process information" in the Designer.

NOTE: A file ProcessInfoExport.log is created in the depository during export. It logs the actions that are executed.

- Configure the One Identity Manager History Database for importing data. For more detailed information, see the .One Identity Manager Data Archiving Administration Guide

Alternatively you can load the XML files into another archiving system provided by the company.

Related Topics

- [Selecting a Method on page 441](#)

Direct Deletion of Records in the One Identity Manager Database

If records from a subsection should be kept in the One Identity Manager database for a certain amount of time but are not archived later however, then you have the following options:

- To exclude subsection from archiving do not configure it for export, but only specify a retention period.
- To delete all subsections with archiving, specify the retention period. Set the configuration parameter "Common\ProcessState\ExportPolicy" in the Designer and enter the value NONE.

The records are deleted from the DBQueue Processor database by the One Identity Manager when the retention period has ended. In addition, all entries for triggered actions that have no corresponding records in the subsections are deleted.

NOTE: If you do not specify a retention period, the records from the subsection are deleted from the One Identity Manager database during DBQueue Processor daily maintenance tasks.

If there is a large amount of data, you can specify the number of objects to delete per DBQueue Processor operation and run in order to improve performance. You use configuration parameters to make the choice for each subsection.

Table 262: Configuration Parameters for Deleting logged Data Changes

Configuration parameter	Meaning
Common\ProcessState\PropertyLog\Delete	This configuration parameter allows configuration of deletion behavior for logged data changes.
Common\ProcessState\PropertyLog\Delete\BulkCount	This configuration parameter contains the number of entries to be deleted in an operation.

Configuration parameter	Meaning
Common\ProcessState\PropertyLog\Delete\TotalCount	This configuration parameter the total number of entries to be deleted in one processing run.

Table 263: Configuration Parameters for Deleting Process Information

Configuration parameter	Meaning
Common\ProcessState\ProgressView\Delete	This configuration parameter allows configuration of deletion behavior for process information.
Common\ProcessState\ProgressView\Delete\BulkCount	This configuration parameter contains the number of entries to be deleted in an operation.
Common\ProcessState\ProgressView\Delete\TotalCount	This configuration parameter the total number of entries to be deleted in one processing run.

Table 264: Configuration Parameters for Deleting Process History

Configuration parameter	Meaning
Common\ProcessState\JobHistory\Delete	This configuration parameter allows configuration of deletion behavior for the process history.
Common\ProcessState\JobHistory\Delete\BulkCount	This configuration parameter contains the number of entries to be deleted in an operation.
Common\ProcessState\JobHistory\Delete\TotalCount	This configuration parameter the total number of entries to be deleted in one processing run.

Table 265: Configuration Parameters for Deleting Process Status Entries

Configuration parameter	Meaning
Common\ProcessState\Delete	This configuration parameter allows configuration of deletion behavior for process status entries.
Common\ProcessState\Delete\BulkCount	This configuration parameter contains the number of entries to be deleted in an operation.

Configuration parameter	Meaning
Common\ProcessState\Delete\TotalCount	This configuration parameter the total number of entries to be deleted in one processing run.

Related Topics

- [Selecting a Method on page 441](#)

Conditional Compilation using Preprocessor Conditions

Conditional compiling of program code is integrated into the One Identity Manager. Conditional compilation allows parts of the program code to be parsed whereas other parts remain untouched.

Conditional compiling has the following advantages:

- Assemblies are reduced in size
- System configuration organization
- Improves clarity for the model and rights
- Speeds up processing
- Hides unnecessary data in all VB.Net expressions
- Hides unnecessary model components

The One Identity Manager conditional compiling is controlled using preprocessor conditions. Preprocessor conditions can be used in:

- Objects with the property **Preprocessor condition**.
- VB.Net expressions

Configuration parameters and their options define the possible preprocessor conditions.

In order to become effective on a system-wide basis, every modification to preprocessor relevant configuration parameters as well as modifications to preprocessor conditions on objects and VB.Net expressions requires the One Identity Manager database to be recompiled.

Detailed information about this topic

- [Preprocessor Relevant Configuration Parameters on page 449](#)
- [Preprocessor Conditions in Objects on page 449](#)
- [Preprocessor Conditions in VB.Net Expressions on page 451](#)
- [Evaluation of Preprocessor Conditions during Compilation on page 452](#)

Preprocessor Relevant Configuration Parameters

The option **Preprocessor relevant parameter** is used to label a configuration parameter as preprocessor relevant. A preprocessor expression is entered in the associated configuration parameter option.

When a preprocessor relevant configuration parameter is set it is valid globally across the system. The preprocessor condition does not come into effect until the database has been compiled.

- ❶ **IMPORTANT:** Each time a preprocessor relevant configuration parameter and the options are changed, the One Identity Manager database needs to be recompiled.
- ❶ **NOTE:** Predefined preprocessor configuration parameters are overwritten during schema installation. Define company specific preprocessor relevant configuration parameters and options in the configuration parameter "Custom".

To display preprocessor relevant configuration parameters

1. Select the category **Base data | General | Configuration parameters** in the Designer.
2. Select **View | Preprocessor definitions** from the menu.

This displays all the processor conditions. Double-click on an entry to display the configuration parameter.

- ❶ **NOTE:** You can find an overview of existing preprocessor dependencies in the Designer in the category Database **One Identity Manager Schema | Preprocessor dependencies**.

Related Topics

- [Editing Configuration Parameters on page 119](#)
- [Preprocessor Conditions in Objects on page 449](#)
- [Preprocessor Conditions in VB.Net Expressions on page 451](#)
- [Evaluation of Preprocessor Conditions during Compilation on page 452](#)

Preprocessor Conditions in Objects

You can enter preprocessor conditions directly for certain objects. To simplify entering conditions, these objects have a property **Preprocessor condition** in which you can write a preprocessor expression.

- IMPORTANT:** Each modification to preprocessor objects requires recompiling the One Identity Manager database.
- TIP:** You can link preprocessor expressions together with AND, OR, NOT, ().
- NOTE:** You can find an overview of existing preprocessor dependencies in the Designer in the category Database **One Identity Manager Schema | Preprocessor dependencies**.

Example

The column Person.RiskIndexCalculated should only be shown in the interface if the risk function is set. The column Person.UID_PersonMasterIdentity is only shown if an employee has several identities to manage. The following preprocessor conditions are entered in the table DialogColumn.

Table 266: Example for Preprocessor Conditions

Table	Column	Preprocessor condition
Employee	RiskIndexCalculated	COMPLIANCE
Employee	UID_PersonMasterIdentity	PERSON_MASTERIDENTITY

If you change the setting of a preprocessor relevant configuration parameter, tasks are created for the DBQueue Processor to calculate all preprocessor and calculation tasks for the affected objects. The option **Disabled by preprocessor** is updated for each object. If the re-interpretation of the preprocessor conditions leads to a change in the option, the preprocessor interpretation tasks that follow are generated for the dependent objects. User rights can also be affected. After DBQueue Processor has processed the tasks, the database needs to be recompiled.

The interpretation of preprocessor conditions has the following effects:

- If a table is disabled by a preprocessor condition then all the columns and object definitions that relate to the table and the user interface forms and the associated navigation are disabled.
- If a primary key column is disabled, all the foreign key columns that refer to this PK are disabled.
- If a primary key member is disabled by a previous rule (by an many-to-many table, for example), then this primary key's table and all further columns belonging to this table are disabled. This method has the advantage that, for example, when a table such as ADSGroup is disabled then all assignments are automatically disabled, such as the table, DepartmentHasADSGroup.

Related Topics

- [Preprocessor Relevant Configuration Parameters on page 449](#)
- [Preprocessor Conditions in VB.Net Expressions on page 451](#)

- Evaluation of Preprocessor Conditions during Compilation on page 452

Preprocessor Conditions in VB.Net Expressions

Preprocessor conditions can be used in VB.Net expressions. The preprocessor conditions are defined in the options for the preprocessor relevant configuration parameters.

IMPORTANT: Every modification to preprocessor conditions in VB.Net expressions requires recompiling the One Identity Manager database.

Script code that is dependent on a preprocessor condition has to be passed in an #if...then...#else statement. The interpretation of the preprocessor conditions is not carried out until the script is generated.

Syntax

```
#If <preprocessor_condition_1>
    ' code, for this preprocessor condition
#ElseIf <preprocessor_condition_2> then
    ' code, for this preprocessor condition
#Else
    ' other code
#Endif
```

Preprocessor conditions can be linked with the following operators:

AND
OR
NOT
()

Example

The preprocessor condition "ITSHOP" is entered into the table DialogColumn for the column ADSGroup.IsForITShop. The template in column ADSGroup.DisplayName should reference the column IsForITShop. In order to remain compatible the following construction has to be used for the template:

```
#If ITSHOP Then
    If $IsForITShop:Bool$ And $UID_AccProduct$ <> "" Then
        Value = $FK(UID_AccProduct).Ident_AccProduct$
    Else
```

```
    value = $cn$  
End If  
  
#Else  
    value = $cn$  
#End If
```

Related Topics

- [Preprocessor Relevant Configuration Parameters on page 449](#)
- [Preprocessor Conditions in Objects on page 449](#)
- [Evaluation of Preprocessor Conditions during Compilation on page 452](#)

Evaluation of Preprocessor Conditions during Compilation

In order to become effective on a system-wide basis, every modification to preprocessor relevant configuration parameters as well as modifications to preprocessor conditions on objects and VB.Net expressions requires the One Identity Manager database to be recompiled.

The following is true for compiling:

- Internal program code in the form of an #if...then...#else statement is created for objects that have a preprocessor condition. Program code in sections whose preprocessor condition does not apply basically do not exist for the compiler and are not parsed by the compiler. Therefore these objects are not considered to exist.
- VB.Net expressions that contain preprocessor conditions are compiled. The program code exists. The interpretation of the preprocessor conditions is not carried out until the script is generated.

These templates are valid for compiling:

- Templates for columns that are disabled by preprocessor conditions are not compiled and the resulting relations are not saved in the table DialogNotification. These columns are therefore considered to be non-existent.
- Templates that relate to disabled columns cause a compiler error message if the corresponding part of code is not linked in a preprocessor statement.

One Identity Manager Scripts

Scripts are used in the One Identity Manager to monitor and maintain data consistency and customer business logic in the database. Scripts can be used to:

- Test column values
- Trigger events
- Create, change and delete objects and therefore manipulate the database

Detailed information about this topic

- [Using Scripts on page 453](#)
- [Message Output on page 454](#)
- [Implementing Dates on page 454](#)
- [Using Dollar \(\\$\) Notation on page 455](#)
- [Using base. Object on page 460](#)
- [Calling Functions on page 460](#)
- [Pre-scripts for using in Processes and Process Steps on page 461](#)
- [Using Session Services on page 461](#)
- [Using #LD Notation on page 464](#)
- [Scripts in the Script Library on page 467](#)
- [Testing Scripts with the System Debugger on page 474](#)

Using Scripts

One Identity Manager scripts are written in VB.Net syntax, which allows all VB.Net functions to be used. The values to be edited are given as preprocessor instructions.

NOTE: You can find detailed examples for syntax and usage of scripts on the installation medium in the directory QBM\dvd\AddOn\SDK\ScriptSamples.

You can use scripts in:

- Templates and formatting scripts (DialogColumn table)
- Table scripts (DialogTable table)
- Script library (DialogScript table)
- Tasks (DialogMethod table)
- Object definition selection scripts (DialogObject table)
- Views selection scripts (DialogTable table)
- Scripts to find the servers to execute process steps (Job table)
- Process step parameters (Jobrunparameter table)
- Process control notification (Job table)
- Generating conditions for process steps and processes (Job and JobChain tables)
- Process step and process pre-scripts (Job and JobChain tables)
- Process information (Job, JobChain and JobEventgen table)
- Mail templates (table DialogRichMailBody)

Message Output

You should never use the VB.Net functions MsgBox and Inputbox on servers. Use the functions VID_Write2Log, RaiseMessage or AppData.Instance.RaiseMessage.

Related Topics

- [Outputting Custom Messages in the One Identity Manager Service Log File on page 641](#)

Implementing Dates

Time stamps, such as insert dates or modification dates, are stored in the database with the respective UTC. The object layer transforms this time data into the currently valid time zone data when an object is loaded. The user, therefore, sees all the values in local time. When an object is saved the current time zone data is transformed into UTC data.

NOTE: The use of DateTime.Now in scripts must be critically tested. It is better to use DateTime.UtcNow than DateTime.Now to display the value to users.

It is not recommended to convert dates, in non-U.S. notation, from type "String" to type "DateTime" in scripts:

```
Value = CDate("31.12.2014")
```

This always causes a problem if the script is running on a U.S. system. In the best case, you are sent an error message like "Cast from string...to type Date is not valid". In the

worst case the wrong date is returned as month and day are swapped (3.12.2014 becomes 12.3.2014).

If possible, you should avoid a string conversion altogether in this case. The type "DateTime" provides several functions for this purpose. For the example above, that would be:

```
Value = new DateTime(2014, 12, 31)
```

If the data type "String" however, should be used then the ISO date notation should be applied as it is always converted correctly:

```
Value = CDate("2014-12-31")
```

```
Value = CDate("2014-12-31 15:22:12")
```

The complicated version is to input the language code format for the date:

```
Value = DateTime.Parse("12.31.2014", new CultureInfo("en-US"))
```

```
Value = DateTime.ParseExact("12.31.2014", "mm.dd.yyyy", CultureInfo.InvariantCulture)
```

Using Dollar (\$) Notation



NOTE: Dollar notation is used to access object properties in One Identity Manager.

If you want to user dollar notation in scripts with it representing access to column names, the character must be doubled to mask.

Example:

In Windows PowerShell scripts, instead of:

```
theScript.AppendLine("foreach ($Domain in $Domains)")
```

use:

```
theScript.AppendLine("foreach ($$Domain in $$Domains)")
```

If you are using dollar (\$) notation you need to ensure that the value is allocated the correct data type. Dollar notation returns a "String" type as default. If another data type is given it is internally converted with "ToString".

Permitted data types are:

Binary

Bool

Byte

Date

Decimal

Double

Int

Long

Short

String (default)

Text

Detailed information about this topic

- [Accessing Local Object Columns on page 456](#)
- [Accessing Object Columns Connected by a Relation on page 457](#)
- [Accessing the Old Column Value on page 457](#)
- [Accessing Column Display Values on page 458](#)
- [Accessing References in Comments on page 459](#)
- [Accessing Local Object Meta Values on page 459](#)

Accessing Local Object Columns

Syntax

```
$<column name>:<data type>$
```

Example for use in templates:

The Active Directory user display name should comprise of the first and last name of the Active Directory user. The template for `ADSAccount.DisplayName` is:

```
If $Givenname$<>"" And $Surname$<>"" Then
    Value = $Surname$ & " " & $Givenname$
ElseIf $Givenname$<>"" Then
    Value = $Givenname$
ElseIf $Surname$<>"" then
    Value = $Surname$
End If
```

If an employee is disabled, the leaving date should be set. The template for `Person.Exitdate` is:

```
If $IsActive:bool$ Then
    Value = Date.Today
End If
```

Related Topics

- [Accessing Object Columns Connected by a Relation on page 457](#)
- [Accessing the Old Column Value on page 457](#)

- Accessing Column Display Values on page 458
- Accessing References in Comments on page 459
- Accessing Local Object Meta Values on page 459

Accessing Object Columns Connected by a Relation

The only relation currently permitted is the foreign key relation.

Syntax

```
$FK(<foreign key column>).<column name>:<data type>$
```

Example for use in templates:

An Active Directory user's first name should be based on the assigned employee. The template for ADSAccount.Givenname is:

```
Value = $FK(UID_Person).Firstname$
```

Related Topics

- Accessing Local Object Columns on page 456
- Accessing the Old Column Value on page 457
- Accessing Column Display Values on page 458
- Accessing References in Comments on page 459
- Accessing Local Object Meta Values on page 459

Accessing the Old Column Value

Syntax

```
$columnname[o]$
```

Example for use in process step parameters:

Optional process step parameters are not generated if the value is set to "Nothing" or the value template is not assigned. This makes it possible to limit the number of parameters for target system components. If such a value should be emptied then an empty string should be given instead of "nothing".

A value template may look like this:

```
If $Lastname[o]$ <> $Lastname$ Then
```

```
Value = $Lastname$  
End If
```

Related Topics

- [Accessing Local Object Columns on page 456](#)
- [Accessing Object Columns Connected by a Relation on page 457](#)
- [Accessing Column Display Values on page 458](#)
- [Accessing References in Comments on page 459](#)
- [Accessing Local Object Meta Values on page 459](#)

Accessing Column Display Values

The column properties "IsMultiLanguage" and "LimitedValues" (list of permitted values) are resolved when creating the display value for a column.

Syntax

```
$columnname[D]$
```

Example of use:

A list of permitted values is defined for the restriction type of the IBM Notes server restrictions.

```
PrivateList=Run Personal Agent RestrictedList=Run Restricted Agent  
UnrestrictedList=Run Unrestricted Agent
```

If a server restriction has the value "Private list" the value "Run personal agent" is displayed on the information form.

Example for use in templates:

The display value for the server restriction should be formatted from the name of the IBM Notes user and the display value of the restriction type.

```
Value = vid_Left($FK(UID_NotesUser).FullName1st$,39) & " [" & vid_Left  
($NotesAgentMgrType[D]$, 22) & "]"
```

Related Topics

- [Accessing Local Object Columns on page 456](#)
- [Accessing Object Columns Connected by a Relation on page 457](#)
- [Accessing the Old Column Value on page 457](#)

- [Accessing References in Comments on page 459](#)
- [Accessing Local Object Meta Values on page 459](#)

Accessing References in Comments

The preprocessor also interprets references that are embedded in comments, for example, `$Lastname$`. Referencing a column in a script comment results in the script being run when the column value is changed.

Example for use in templates:

An employee's starting date is filled with a template. This template should run when the employee's surname changes. The template for Person.Entrydate is therefore:

```
'$Lastname$  
Value = Date
```

Related Topics

- [Accessing Local Object Columns on page 456](#)
- [Accessing Object Columns Connected by a Relation on page 457](#)
- [Accessing the Old Column Value on page 457](#)
- [Accessing Column Display Values on page 458](#)
- [Accessing Local Object Meta Values on page 459](#)

Accessing Local Object Meta Values

Syntax

`[$IsLoaded]:Bool$`

Table 267: Meta Values and their Meaning

Meta value	Meaning
IsLoaded	This value specifies whether the object is loaded from the database.
IsChanged	This value specifies whether the object is altered when it is loaded from the database.
IsDifferent	This value specifies whether the new value is different from the old value. You can access to the column through: <code>Columnname[C]</code> .
IsDeleted	This value specifies whether the object is marked for deletion.

Related Topics

- [Accessing Local Object Columns on page 456](#)
- [Accessing Object Columns Connected by a Relation on page 457](#)
- [Accessing the Old Column Value on page 457](#)
- [Accessing Column Display Values on page 458](#)
- [Accessing References in Comments on page 459](#)

Using `base.` Object

The `Base.` syntax always accesses the object that is currently loaded. The `Base.object` can be used in tasks, selection scripts for object definitions and insert values. `Base.object` cannot be used in templates, formatting scripts and processes.

Syntax

- Simple value assignment
`Base.PutValue("<column>", <value>)`
- Value assignment with variable replacement (value must be a character string)
`Base.PutValue("<column>", context.Replace(<value>))`

Example

```
Base.PutValue("IsForITShop", 1)
Base.PutValue("UID痈ADSContainer", context.Replace("%cont%"))
```

Calling Functions

Functions are stored in the script library (DialogScript table).

Example

```
Public Function BuildInternalName(ByVal Firstname As String, ByVal Lastname As
String) As String
    BuildInternalName = Lastname & Firstname
End Function

Usage in template for Person.Internalname:
Value = BuildInternalName($Firstname$, $Lastname$)
```

Pre-scripts for using in Processes and Process Steps

Pre-script code is code that is executed before other scripts are run. You can define process specific variables. Process specific variables are local data spaces when a process is generated. This enables variables to be defined that can be made further use of within processes and process steps, for example, generating conditions, server selection scripts or in parameters.

- **NOTE:** It is recommended only to set process specific variables in the pre-script and to have read access to them during further usage.

Pre-script syntax

```
values("Name") = "value"
```

Usage in the following process and process step code sections:

```
Value = values("Name")
```

Related Topics

- [Using Process Specific and Global Variables for the Process Definition on page 396](#)
- [Querying Session Object Global Variables on page 463](#)

Using Session Services

The session object is the instance which makes data available to the user session. This includes the current user, their user groups and program functions. Furthermore, the session object make various services available for accessing data. The session object services provided are made available through a generic interface (Resolve (Of Service)()). In the following, examples are provided of frequently used service.

- **NOTE:** You can find a complete description of all parameters in the VI.DB.DLL documentation.

Detailed information about this topic

- [Querying Configuration Parameters on page 462](#)
- [Testing the Existence of Certain Database Entries on page 462](#)
- [Querying Session Object Global Variables on page 463](#)

Querying Configuration Parameters

The full path for the configuration parameter always has to be entered when configuration parameters are queried.

Syntax

```
Session.Config().GetConfigParm("<full path>")
```

When a configuration parameter is tested in a generating condition in VB.NET syntax, the function returns a string. In order to compare this value to a numerical value, the configuration parameter has to be set and contain a numerical value. This depends on the implicit value type conversion from VB.NET. If the configuration parameter is not enabled, the function returns an empty string that cannot be compared to a numerical value. This results in a VB.NET runtime error. Configuration parameter values are therefore always compared to strings.

Do not use:

```
Session.Config().GetConfigParm("QER\Person\User\DeleteOptions\Homedir")=1
```

but:

```
Session.Config().GetConfigParm("QER\Person\User\DeleteOptions\Homedir")=1
```

In order to ensure that a logical value is always returned, the function VID_IsTrue should be used.

Example

```
If VID_IsTrue(Session.Config().GetConfigParm  
("QER\Person\User\DeleteOptions\Homedir")) Then ...
```

Related Topics

- [Testing the Existence of Certain Database Entries on page 462](#)
- [Querying Session Object Global Variables on page 463](#)

Testing the Existence of Certain Database Entries

 **NOTE:** The test should take place without taking access permissions into account.

Syntax

```
Session.Source().Exists("<Tablename>","<WhereClause>")
```

Example

```
Session.Source().Exists("Person", "CentralAccount = '" & accnt & "' and uid_person <> '' & uid_person & "")
```

Related Topics

- [Querying Configuration Parameters on page 462](#)
- [Querying Session Object Global Variables on page 463](#)

Querying Session Object Global Variables

Global variables are allocated by the set up program. All environment variable and custom variables defined for the session object can be used in addition to predefined variables. Custom session variables can be defined, for example, through scripts, methods or customizers.

NOTE: If a custom session variable is defined, it must be removed again afterward. Otherwise it remains for the rest of the session and, in certain circumstances, the wrong processes can be generated.

Syntax:

```
Variables("<Variable name>")
```

Example for use in process step parameters:

```
Value = Variables("GENPROCID")
```

```
Value = CBool(Session.Variables("FULLSYNC"))
```

Table 268: Permitted Predefined Global Variables

Variable	Meaning
EnvUserName	Name of user to be authenticated in the environment, for example, "Domain\User" in Active Directory.
FullSync	Variable is set by all synchronizers ("True", "False").
GenProcID	Unique Process ID number
LogonUser	DialogUser.Username of the currently logged in user.
DialogUserID	DialogUser.ID_DialogUser of the logged in user.
UserName	Name displayed in XUserInserted or XUserUpdated.
UserUID	Logged in user's UID_Person, if user related authentication is being used.

Variable	Meaning
ShowCommonData	Specifies whether system data is shown (1) or not shown (0). The variable is evaluated in Designer by the program setting "Show system information".
Feature_<Featurename>	Queries additional program functions (DialogFeature) that are available for the user. The value is "1" when the program function is available, otherwise the variable is not set.
ManageOutstandingOperation	This variable is used to differentiate between executing operations during post-processing of outstanding objects in target system synchronization. Permitted values are "Delete", "DeleteState" and "Publish".

Related Topics

- [Querying Configuration Parameters on page 462](#)
- [Testing the Existence of Certain Database Entries on page 462](#)

Using #LD Notation

#LD notation is used for displaying language dependent information. #LD notation is mainly used in process tracking and processing notification, but it can also be used in scripts that are stored in the script library.

Table 269: Using #LD Notation

Context	Table.column
Process tracking	Job.ProcessDisplay - based on DialogProcessStep.Displayname
	JobChain.ProcessDisplay - based on DialogProcessChain.Displayname
	JobEventgen.ProcessDisplay - based on DialogProcess.Displayname
Process handling notification	Job.NotifyAddress and Job.NotifyAddressSuccess
	Job.NotifyBody and Job.NotifyBodySuccess
	Job.NotifySender and Job.NotifySenderSuccess
	Job.NotifySubject and Job.NotifySubjectSuccess
	JobRunParameter.ValueTemplate (only process components: MailComponent)
Templates	DialogColumn.Template and DialogColumn.CustomTemplate

Context	Table.column
Formats	DialogColumn.FormatScript and DialogColumn.CustomFormatScript
Method definitions	DialogMethod.MethodScript
Insert values	DialogObject.InsertValues, DialogTable.InsertValues, DialogTree.ListInsertValues and DialogSheet.InsertValues
Selection scripts	DialogTable.SelectScript and DialogObject.SelectScript
Process generating scripts	JobChain.GenCondition and Job.GenCondition JobChain.PreCode and Job.PreCode Job.ServerDetectScript

Syntax

Value=#LD[<language>](<key>,{<parameter>}*)#

where:

<Language> Optional language or culture for the output.

<Key> Basis string with place holder. The place holder syntax corresponds to a format place holder in .Net ({0} to {9})

<Parameter> Parameter for replacing the place holder (comma delimited)

Related Topics

- [Example for Use in Process Tracking on page 465](#)
- [Example for Specifying Language on page 466](#)

Example for Use in Process Tracking

A change is made to an employee (e.g. because they have moved). The process information for the event "Update" on the base object "Person" could be formulated as:

Value = #LD("Change properties of employee {0}.", \$Internalname\$)#+

with:

Internalname = JSmith

results in the following output text in the process view:

Änderung der Daten der Person JSmith.

 **NOTE:** Prerequisite for language dependent representation is the definition of the corresponding caption for displaying the text in the active languages.

The captions for language dependent text are entered in `DialogMultiLanguage` when the script is compiled. The key (column `Entrykey`), the language (column `Ident_Language`) and the language dependent replacement (column `EntryValue`) are entered in this table. The key should be in the corresponding default language. If a language caption has not been entered, the key is used as the display text. Use the Language Editor to add translations for the captions in other languages.

In order to display the language dependent process information for the example above, it could be formulated as follows:

Display text caption in the `DialogMultiLanguage` table:

Language	English
Key	Changed properties of employee {0}.
Value	Changed properties of employee {0}.
Language	German
Key	Changed properties of employee {0}.
Value	Änderung der Daten der Person {0}.

Template for the process information:

```
Value = #LD("Change properties of employee {0}.", $Internalname$)#+
```

with:

```
Internalname = JSmith
```

results in the following display text in the process view:

System's user language:	Output:
German	Änderung der Daten der Person JSmith.
English	Changed properties of employee JSmith.

Related Topics

- [Editing Translations in the Language Editor on page 333](#)

Example for Specifying Language

#LD notation also supports the specification of which language to use. This is particularly useful in cases where users need to receive system messages in their preferred language.

Example

Output always in English

```
Value = #LD["english"]("Test: {0}", <parameter>)#+
```

```
Value = #LD["en-US"]("Test: {0}", <parameter>)#
```

Output in the default language:

```
Value = #LD("Test: {0}", <parameter>)#
```

```
Value = #LD[""]("Test: {0}", <parameter>)#
```

Using a variable:

```
Dim lang As String = "english"
```

```
Value = #LD[lang]("Test: {0}", <parameter>)#
```

You do not need to enter the language in square brackets, it is optional. It is important, however, that the language statement is a string. If the language is not given or the resulting string is empty or "nothing", the language is used that is currently set up for the application.

Scripts in the Script Library

The script library contains source code for all the scripts used in One Identity Manager. The default scripts that we supply cannot be edited. These scripts are overwritten during schema installation even if they are used in custom scripts.

The scripts are stored in the category **Script library** in the Designer and created and edited with the Script Editor.

Detailed information about this topic

- [Editing Scripts on page 470](#)
- [Overriding Scripts on page 472](#)
- [Testing Script Compilation on page 473](#)
- [Testing a Script with the Script Editor on page 473](#)
- [Testing Scripts with the System Debugger on page 474](#)
- [Support for Scripting on page 469](#)

Working with the Script Editor

Create scripts in the script library and edit them with the Script Editor. The editor is started from the program "Designer" and opens in the document view. Only additional Script Editor functions are described in the following.

Menu Items

The following items are added to the menu bar when the editor starts.

Table 270: Menu Items Added by the Editor

Menu	Menu Item	Meaning	Key Combination
Edit	Undo	Restores state before last change.	
	Redo	Restores state after last change.	
	Cut	Removes selected script code and stores it in the clipboard.	
	Copy	Copies selected code into the clipboard.	
	Paste	Inserts code stored in the clipboard.	
	Delete	Deletes the selected script code.	
	Increase indent	Moves the selected script code to the right in the script window.	
	Decrease indent	Moves the selected script code to the left in the script window.	
	Search text...?	Opens dialog box for searching text.	
Script	New	Creates a new script.	
	Delete	Deletes the selected script.	
	Save	Save changes to the script.	
	Compile Script	Tests script compilation. Messages are sent to the <Compiler error> view.	F9
	Insert code snippet	A code snippet is inserted into the script.	F2
Options	Line number	Shows/hides line numbers in the script window.	
	Text block	Expands/collapses text blocks in the script window.	
View	Compiler errors	Shows/hides the "Compiler errors" view.	
	Test script	Shows/hides the "Test script" view.	
Help	Script Editor	Opens the editor help.	

Table 271: Meaning of Toolbar Icons

Icon	Meaning
	Shows/hides line numbers.

Icon	Meaning
	Expands/collapses text blocks.
	Undo last change.
	Redo last change.
	Removes selected script code and stores it in the clipboard.
	Copies selected code into the clipboard.
	Inserts code stored in the clipboard.
	Deletes the selected script code.
	Decreases insert.
	Increases insert.
	Inserts code snippet.
	Search and replace.
	Saves changes to the script.
	Creates a new script.
	Deletes the selected script.
	Compiles the script.

Support for Scripting

Additional input aids are provided for creating script code.

Syntax Highlighting

The input fields support syntax highlighting depending on the syntax type.

Auto-Completion

Auto-completion can be used when creating script code. The amount of scripted code to enter is reduced by displaying the names of properties or functions that can be used. Automatic completion is called with the key combination **CTRL + SPACE** at the appropriate point in the editor. The contents of the list is determined by the key words in the code.

Entering Code Snippets

Input fields that required data in VB.Net syntax support code snippets. Standard code snippets are available in the option "Visual Basic". The option "Object layer" contains special code snippets for the One Identity Manager object layer.

You can insert code snippets using the following options:

1. Using the icon 
 - Select the  in the menu bar.
 - Select the option "Object layer" or "Visual Basic".
 - Select the code snippet.
2. Using a shortcut
 - Press the **F2** key.
 - Select the option "Object layer" or "Visual Basic".
 - Select the code snippet.
3. Using an aliases
 - Enter an alias.
 - Use **Tab** to insert the code snippet.

 **NOTE:** Remember that the connection name is case sensitive.

 **NOTE:** If you select the code snippet using a shortcut or the  icon, a short description and the shortcut name is displayed in a tooltip.

 **TIP:** You can use custom code snippets. To do this, create a directory `CustomSnippets` in the One Identity Manager installation directory to store the code snippets. Use Visual Studio documentation to develop your own code snippets.

Editing Scripts

Scripts are displayed in the category **Script Library** in the Designer. Scripts are organized into the subcategories custom, product dependent and general usage scripts. You can gather all the information about usage, for example, in column definitions, processes or other scripts, in the script overview.

Use the Script Editor to create and edit your scripts.

 **IMPORTANT:** After creating and editing the script, you should test compiling the script. Compile the scripts in the script library for this script to take effect.

To create a new script

1. Select the category **Script Library** in the Designer.
2. Start the Script Editor using the task **Create a new script**.

3. Edit the master data.

Table 272: Script Master Data

Property	Description
Script	Name of script. Label custom scripts with the prefix "CCC_".
Description	Detailed description of the script's functionality.
Script code	One Identity Manager scripts are written in VB.Net syntax, which allows all VB.Net functions to be used. The values to be edited are given as preprocessor instructions. You can find detailed examples for syntax and usage of scripts on the installation medium in the directory QBM\dvd\AddOn\SDK\ScriptSamples.

To edit a script

1. Select the script in the category **Script Library** in the Designer
2. Select the task **Edit script 'name of script'**.
3. Edit the script master data.

To copy an existing script

1. Select the script you want to copy in the category **Script Library** in the Designer.
2. Select the task **Copy script 'name of script'**.
This starts the Script Editor and opens the dialog box "Copy script".
3. Verify the following data and it necessary.

Table 273: Copying a Script

Old script name	Name of the copied script.
Script	The name of the new script is made up of the prefix "CCC_" and the name of the old script. You can change the name. Label custom scripts with the prefix "CCC_".
Script code	The script code from the original is copied over. If necessary, you can modify the script code of the script to copy beforehand.

4. To create the copy, click **OK**.
5. Edit the script master data.

Detailed information about this topic

- [Overriding Scripts on page 472](#)
- [Testing Script Compilation on page 473](#)

- [Testing a Script with the Script Editor on page 473](#)
- [Compiling a One Identity Manager Database on page 64](#)
- [Using Scripts on page 453](#)

Overriding Scripts

You might want to label scripts for overriding if there are limits to how much you can modify default scripts. Scripts, which can be overridden are labeled with the property "overridable".

NOTE: Only the default scripts that are supplied can be overridden. Custom scripts cannot be overridden because these are saved in the script class 'Custom scripts'.

To override a script

1. Select the script to override in the category **Script Library | Overridable scripts** in the Designer.
2. Select the task **Copy script 'name of script'**.
3. This starts the Script Editor and opens the dialog box "Copy script".
4. Modify the following data:
 - Script name
The new script name is made up of the old script name. You can change the name.
5. To create the copy, click **OK**.
6. Replace the property "overridable" with "overrides" in the script header.
7. Modify the other script code accordingly to suit your requirements.

IMPORTANT: After creating and editing the script, you should test compiling the script. Compile the scripts in the script library for this script to take effect.

Syntax example

```
Public overridable Function My_Function() as Boolean
  'script code of the original version
End Function

Public overrides Function My_Function() as Boolean
  'Custom script code
End Function
```

Related Topics

- [Editing Scripts on page 470](#)
- [Testing Script Compilation on page 473](#)
- [Testing a Script with the Script Editor on page 473](#)
- [Compiling a One Identity Manager Database on page 64](#)
- [Using Scripts on page 453](#)

Testing Script Compilation

If you have created a new script, you need to compile it. The script is not executable until it has been compiled. You can test script compilation in the Script Editor.

To test compiling scripts

- Select the script in the category **Script Library** in the Designer
- Select the task **Edit script 'name of script'**.
- Start compilation with the icon , menu item **Script | Compile script** or **F9**.

All scripts are converted during compilation. The assemblies are created and placed on the workstation where generating will take place. During the conversion, the script code is tested for validity. This process may required some time.

Error messages are sent to the "Compiler error" view. A double-click on the error message takes you straight to the corresponding line in the script code view where you can edit it. It can be modified at this point.

IMPORTANT: Once you have tested the script it needs to be added to the One Identity Manager database and compiled with the Database Compiler.

Related Topics

- [Compiling a One Identity Manager Database on page 64](#)

Testing a Script with the Script Editor

You can use the Script Editor to test a script.

To test a script

1. Select the script in the category **Script Library** in the Designer
2. Select the task **Edit script 'name of script'**.
3. Select the **Test script** view.

4. Select the script to test in the menu and modify the parameters as required.

All the parameters to be passed to the script are displayed with their data types. You can edit the values. You can also predefine values for the script base class variables "Base" and "Value" as input parameters in order to work with the script.

5. Select one or more options for running the test from the **Options** list.

Option	Description
Use master connection	This option specifies whether the script test is tested against the main database or an internal SQLite database. Scripts that relate to the application part of the One Identity Manager data model should always be tested with the main database. Scripts for system parts can be tested with the main database or the internal SQLite database.
Use transaction	Use this option to specify whether a script is going to be executed within a transaction including subsequent rollback or whether it is going to be executed immediately against the database.
Write SQL Log	You can record the database actions in an SQL log whilst the script is running. The output is displayed in a separate dialog window. The execution time of the script is logged in addition to the statement being executed.

6. Select the **Start** button to run the script test.

The test results are displayed in the **Result** field after the script has been run.

Related Topics

- [Testing Scripts with the System Debugger on page 474](#)

Testing Scripts with the System Debugger

The System Debugger gives you the opportunity to test scripts, templates, formatting rules, methods and table scripts. Visual Studio debug and edit options are available to you.

The following software must be installed to use the System Debugger:

- Visual Studio 2010 or later
- Microsoft .NET Framework Version 4.5.2 or later

NOTE: Microsoft .NET Framework version 4.6 is not supported.

NOTE: To use the System Debugger with privileges without starting Visual Studio, you must install the One Identity Manager components in a local directory which is not controlled through user accounts.

Detailed information about this topic

- [Loading the System Library on page 475](#)
- [Testing and Editing Scripts on page 476](#)
- [Testing Templates and Formatting Scripts on page 479](#)
- [Testing Methods on page 480](#)
- [Testing Table Scripts on page 480](#)
- [Logging Database Queries and Object Actions on page 481](#)

Loading the System Library

A solution template `SystemLibrary.sln` with the solution "SystemLibrary" is loaded for editing and testing in System Debugger when the Visual Studio is called.

The following projects are defined in the solution.

Table 274: Solution Project Files

Project	Script File	Description
Methods	Methods.vb	This script file contains all methods.
Scripts	VIScripts.vb	This script file contains all predefined scripts from the model components. Do not edit these scripts as they are overwritten by migration.
	VIDScripts.vb	This script file contains all predefined scripts from the system components. Do not edit these scripts as they are overwritten by migration.
	CustomerScripts.vb	This script file contains custom scripts. Add new scripts here.
SystemDebugger	Main.vb	Start up project.
Tables	Tables.vb	This script file contains all the table scripts.
Templates	Templates.vb	This script file contains all templates and formatting scripts.

To load the system library

1. Select **Start | One Identity | One Identity Manager | Configuration | System Debugging.**

- OR -

Run the file `SystemLibrary.sln` from the installation directory.

2. Check whether the project "SystemDebugger" is entered in Visual Studio as the start project.
3. Start the solution with **F5** in Visual Studio.
4. Connect to the database.
5. Check the solution file directory and the options for creating the script library.

Options	Meaning
Export locked scripts	Specifies whether only active scripts or other locked scripts are loaded from the library as well.
Update project references	Specifies whether references used in scripts are loaded as well.
Create backups of existing files	Specifies whether backups of existing files are made.

6. Click **OK**.

The script library files are filled with data from the database.

7. Confirm reloading of each project in Visual Studio.

8. Start the solution with **F5** in Visual Studio.

The source code generated for the solution is compiled.

9. Reconnect to the database.

Starts the System Debugger.

NOTE: The project setting **Start external program** must be set to <InstallFolder>\SystemLibrary\SystemDebugger\bin\Debug\SystemDebugger.Oracle.exe.

Related Topics

- [Testing and Editing Scripts on page 476](#)
- [Testing Templates and Formatting Scripts on page 479](#)
- [Testing Methods on page 480](#)
- [Testing Table Scripts on page 480](#)

Testing and Editing Scripts

The System Debugger gives you the opportunity to test and, if necessary, edit scripts that are stored in the script library. Here you have Visual Studio debug and edit options available, in contrast to the Script Editor.

Detailed information about this topic

- [Testing a Script on page 477](#)
- [Editing a Script in the System Debugger](#)
- [Saving Modified Scripts on page 478](#)

Related Topics

- [Testing Templates and Formatting Scripts on page 479](#)
- [Testing Methods on page 480](#)
- [Testing Table Scripts on page 480](#)
- [Logging Database Queries and Object Actions on page 481](#)

Testing a Script

To test a script

1. Select the desired script in the System Debugger in **Scripts**.
2. Enter value for the script parameters as required.
3. Check the options for executing the script.

Option	Meaning
Run in debug mode	Jumps into the source code. This allows you to use all Visual Studio debugging options.
Define base data	The variables "base" and "value" of the script base class can be pre-allocated as input parameters to be used in the script. Example: Base is initialized with a DB object key in order to use base.GetValue ("column name").String.
Transaction with Rollback	Use this option to specify whether the script is executed within a transaction with subsequent rollback or the script is executed immediately against the database.

4. Select **Start**.

The script starts executing. After the script has executed, the result and the execution time of the script is displayed.

Related Topics

- [Editing a Script in the System Debugger on page 478](#)
- [Saving Modified Scripts on page 478](#)
- [Testing a Script with the Script Editor on page 473](#)

Editing a Script in the System Debugger

You can use the Script Editor to create scripts. Enter the name of the script in the Script Editor and a skeleton script body. This you can export to the script library where you can edit the script with the System Debugger.

To edit a script, select it in the script library, start the script in debug mode. Visual Studio debug and edit options are available to you.

IMPORTANT: You are not permitted to edit VI-Key comments in the source code or to delete them because they label each code block and are needed for backing up scripts in the database.

Related Topics

- [Testing a Script on page 477](#)
- [Saving Modified Scripts on page 478](#)

Saving Modified Scripts

To save a script in the database

1. Select **Scripts | Save script...** from the menu.
This opens a dialog box displaying script name, database object, database connection and script code to be added.
 2. Select a change label under **Change labels** to group your changes.
 3. Click **Save**.
- NOTE:** Ensure you recompile the database after modifying scripts.

Related Topics

- [Editing a Script in the System Debugger on page 478](#)
- [Testing a Script on page 477](#)
- [Compiling a One Identity Manager Database on page 64](#)

Testing Templates and Formatting Scripts

You can use the System Debugger to load objects from the database and debug templates and formatting scripts as they execute.

To test a template

1. Select the table and table script from **Templates** in the System Debugger.
2. Select the column with the template you want to test under **Notifier column**.
3. Select the object in **Database object** to which to apply the template.
4. Check the options for executing the template.

Option	Meaning
Transaction with Rollback	Use this option to specify whether the template is executed within a transaction with subsequent rollback or the template is executed immediately against the database.

5. Select one of the following actions to test the template.

Action	Meaning
Save	Saves the object.
Discard	Discards changes made to the object.
Load	The object is reloaded.
New	Creates a new object.
Execute	Executes the template.

To test a formatting script

1. Select the column with the formatting script in **Formats** in the System Debugger.
2. Select the object in **Database object** to which to apply the formatting script.

Related Topics

- [Testing and Editing Scripts on page 476](#)
- [Testing Methods on page 480](#)
- [Testing Table Scripts on page 480](#)
- [Logging Database Queries and Object Actions on page 481](#)

Testing Methods

You can use the System Debugger to debug methods in runtime.

To test a method

1. Select the method in **Method** in the System Debugger.
2. Select the object to apply the method to under **Base object**.
3. Check the options for executing the method.

Option	Meaning
Transaction with Rollback	Use this option to specify whether the method is executed within a transaction with subsequent rollback or the method is executed immediately against the database.
4. Select Start .	The method starts executing.

Related Topics

- [Testing and Editing Scripts on page 476](#)
- [Testing Templates and Formatting Scripts on page 479](#)
- [Testing Table Scripts on page 480](#)
- [Logging Database Queries and Object Actions on page 481](#)

Testing Table Scripts

You can use the System Debugger to load objects from the database and debug table scripts as they execute.

To test table scripts

1. Select the table and table script from **Tables** in the System Debugger.
2. Select the object to test the table script on under **Database object**.

- Check the options for executing the table script.

Option	Meaning
Transaction with Rollback	Use this option to specify whether the table script is executed within a transaction with subsequent rollback or the table script is executed immediately against the database.

- Select the following actions to test the table script.

Action	Meaning
Save	Saves the object. (OnSaved, OnSaving)
Discard	Discards changes made to the object. (OnDiscarded, OnDiscarding)
Load	The object is reloaded. (OnLoaded)
New	Creates a new object.

Related Topics

- [Testing and Editing Scripts on page 476](#)
- [Testing Templates and Formatting Scripts on page 479](#)
- [Testing Methods on page 480](#)
- [Logging Database Queries and Object Actions on page 481](#)

Logging Database Queries and Object Actions

Use database query and object action logging in the System Debugger to look for errors and optimize scripts during development. The execution time and the command that was run are logged.

- SQL log
Open the log dialog box by selecting **View | SQL protocol** from the menu.
- Object log
Open the log dialog box by selecting **View | Object log** from the menu.

Table 275: Functions for Logging Database Queries and Object Actions

Icon	Meaning
	Starts recording.

Icon	Meaning
	Stops recording.
	Copies logged data to the clipboard.
	Save logged data in a file.
	Deletes the logged data.

Maintaining Mail Templates

The One Identity Manager provides the means to send email notifications. For example, notifications can be sent from process handling, about attestation or the status of IT Shop requests.

A mail template consists of general master data such as target format, important or mail notification confidentiality and one or more mail definitions. Mail text is defined in several languages in the mail template. This ensures that the language of the recipient is taken into account when the email is generated.

There is a One Identity Manager in the Mail Template Editor to simplify writing notifications. You can use the Mail Template Editor to create and edit mail text in WYSIWYG mode.

Email notifications are generated through default processes during process handling. To use email notifications based on mail templates for other business procedures, for example creating user accounts, you have to create custom mail templates and processes. The process task "SendRichMail" is provided for this through the process component "MailComponent".

Detailed information about this topic

- [Creating and Editing Mail Templates on page 485](#)

Working with the Mail Template Editor

Edit mail templates with the Mail Template Editor. The editor is started from the program "Designer" and opens in the document view. Only additional Mail Template Editor functions are described in the following.

Menu Items

The following items are added to the menu bar when the editor starts.

Table 276: Meaning of Items in the Menu Bar

Menu	Menu Item	Meaning
Mail template	Save	Saves the mail template.
	New	Creates a new mail template.
	Delete	Deletes the mail template.
	Copy mail template...	Creates a copy of the current mail template and saves it under a new name.
	Mail preview	Creates a mail preview.
View	Mail definition	Shows/hides the mail definition edit view.
Help	Mail Template Editor help	Opens the editor help.

Table 277: Meaning of Toolbar Icons

Icon	Meaning
	Saves the template.
	Creates a new mail template.
	Deletes a mail template.
	Copies a mail template.
	Creates a preview.

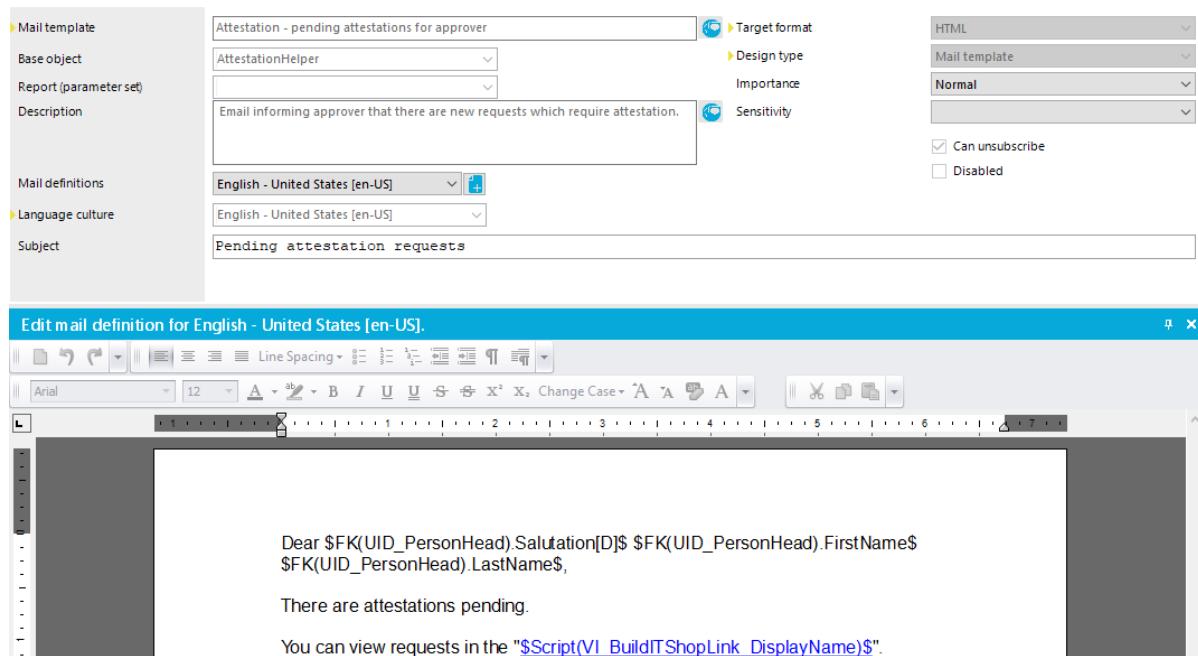
Views in the Mail Template Editor

The Mail Template Editor has different view for editing the mail template.

Table 278: Mail Template Editor Views

View	Description
Mail template properties	Use the view to edit the properties of the selected mail template. A default context menu is available for input fields.
Mail definitions	There is a mail text editor in this view for editing the mail definition with edit and formatting function in the style of Microsoft Word.

Figure 49: Views in the Mail Template Editor



Creating and Editing Mail Templates

A mail template consists of

1. General master data, such as, end format, importance or confidentiality.
2. One or more mail definitions. Mail text is defined in several languages in the mail template.

To edit a mail template

1. Select the category **Mail templates** in the Designer.
2. Select the mail template and start the Mail Template Editor using the task **Edit mail template '<template name>'.**

To create a new mail template

1. Select the category **Mail templates** in the Designer.
2. Select the mail template and start the Mail Template Editor using the task **Create a new mail template.**

To copy a mail template

1. Select the category **Mail templates** in the Designer.
2. Select the mail template you want to copy and start the Mail Template Editor using the task **Edit mail template '<template name>'.**

3. Select **Mail templates | Copy mail template...** in the menu.
4. Enter the name of the new mail template and click **OK**.
The new mail template is displayed in the Mail Template Editor. Now, you can edit the mail template.

To preview the mail template.

1. Select the category **Mail templates** in the Designer.
2. Select the mail template and start the Mail Template Editor using the task **Edit mail template '<template name>'**.
3. Select **Mail templates | Mail preview** in the menu.
4. Select the base object and click **OK**.

Detailed information about this topic

- [General Properties of a Mail Template on page 486](#)
- [Creating and Editing an Email Definition on page 487](#)
- [Customizing Email Signatures on page 493](#)

General Properties of a Mail Template

Table 279: Mail Template Properties

Property	Meaning
Mail template	Name of the mail template. This name will be used to display the mail templates in the administration tools and in the Web Portal. Translate the given text using the  button.
Base object	Mail template base object. A base object only needs to be entered if the mail definition properties of the base object are referenced.
Report (parameter set)	Report, made available through the mail template.
Description	Mail template description. Translate the given text using the  button.
Target format	Format in which to generate email notification. Permitted values are:
Value	Description
HTML	The email notification is formatted in HTML format. HTML format can contain formatting.
TXT	The email notification is formatted in text format. Text format cannot contain any formatting.

Property	Meaning
Design type	Design in which to generate the email notification. Permitted values are:
Value	Description
Mail template	The generated email notification contains mail text corresponding to the mail definition.
Report	The email notification is generated with the report contained under Report (parameter set) as mail body.
Mail template, report as attachment	The generated email notification contains mail text corresponding to the mail definition. The report entered in the Report (parameter set) field is attached to the mail as PDF file.
Importance	Importance for the email notification. Permitted values are "low", "normal" and "high".
Confidentiality	Confidentiality for the email notification. Permitted values are "normal", "personal", "private" and "confidential".
Can unsubscribe	Specifies whether the recipient can unsubscribe email notification. If this option is set, the emails can be unsubscribed through the Web Portal.
Disabled	Specifies whether this mail template is disabled.
Mail definitions	Unique name for the mail definition.
Language culture	Language which applies to the mail template.
Subject	Subject of the email message
Mail body	Content of the email message.

Related Topics

- [Creating and Editing an Email Definition on page 487](#)

Creating and Editing an Email Definition

Mail texts can be defined in these different languages in a mail template. This ensures that the language of the recipient is taken into account when the email is generated.

To create a new mail definition

1. Open the mail template in Mail Template Editor.
2. Click the  button next to the **Mail definition** list.

3. Select the language culture you want the mail definition to apply to from the **Language culture** menu.

All active language cultures are shown in the list. To use other languages, enable the corresponding countries.

4. Enter the subject in the **Subject** field.
5. Edit the mail text in the **Mail definition** view with the help of the Mail Text Editor.
6. Save the changes.

To edit an existing mail definition

1. Open the mail template in Mail Template Editor.
2. Select the language in the **Mail definition** list.
3. Edit the mail subject line and the body text.
4. Save the changes.

Related Topics

- [Displaying Country Information on page 127](#)

Using Base Object Properties

You can use all the properties of the object entered under **Base object** in the subject line and in the mail body. You can also use the object properties that are referenced by foreign key relation.

To access properties use dollar notation. For more information, see the One Identity Manager Configuration Guide.

Example

An IT Shop requester should receive email notification about the status of the request.

Table 280: Email Notification Properties

Property	Value
Base object	PersonWantsOrg
Subject	"\$DisplayOrg[D]\$" status change
Mail body	Dear \$FK(UID_PersonOrdered).Salutation[D]\$ \$FK(UID_PersonOrdered).FirstName\$ \$FK(UID_PersonOrdered).LastName\$, The status was changed on the following request on \$DateHead:Date\$. Product: \$DisplayOrg[D]\$

Property	Value
Requested by:	\$DisplayPersonInserted\$
Reason:	\$OrderReason\$
Current status of your request:	
Approval:	granted
Approver:	\$DisplayPersonHead[D]\$
Reason:	\$ReasonHead[D]\$

The generated email notification could look like the following, for example, once it has been formatted:

Subject: "Service Notebook" status change

Dear Ms Monica Fletcher.

The status was changed on the following request on 03/08/2011 11:14:53.

Product:	Service Notebook
Requested by:	Fletcher, Monica
Reason:	For on-site processing

Current status for your request:

Approval:	granted
Approver:	Rippington, Rudiger
Reason:	approved

Related Topics

- [Using Dollar \(\\$\) Notation on page 455](#)

Use of Hyperlinks in the Web Portal

You can insert hyperlinks to the Web Portal in the mail body. If the recipient clicks on the hyperlink in the email, the Web Portal is opened on that web page and further actions can be carried out. In the default version, this method is implemented for IT Shop requests, in Identity Audit, for policy checking and in attestation.

Prerequisites for using this method

- The configuration parameter "QER\WebPortal\BaseURL" is set and contains the Web Portal URL.

`http://<Server>/<App>`

with:

`<Server> = Server name`

`<App> = Web Portal installation directory path`

To add a hyperlink to the Web Portal into the mail text

1. Click in the mail body at the point where you want to add the hyperlink.
2. Open the context menu and select **Hyper Link....**
3. Enter the hyperlink in **Display text**.
4. Set the option **File or website**.
5. Enter the address of the page to be opened in the Web Portal in **Address**.
6. To accept the input, click **OK**.

Related Topics

- [Default Functions for Creating Hyperlinks on page 490](#)
- [Using Process Parameters in Hyperlinks on page 492](#)

Default Functions for Creating Hyperlinks

Several default functions are available to help you create hyperlinks. You can use these functions to directly insert a hyperlink in a mail body or into processes.

Direct Function Input

A function is referenced in the **Address** field when a hyperlink is inserted:

`$Script(<Function>)$`

Example:

```
$Script(VI_BuildITShopLink_Show_for_Requester)$  
$Script(VI_BuildAttestationLink_Approve)$  
$Script(VI_BuildComplianceLink_Show)$  
$Script(VI_BuildQERPolicyLink_Show)$
```

Default Functions for Requests

The script `VI_BuildAttestationLinks` contains a collection of default functions for composing hyperlinks to directly grant or deny approval of requests from email notifications.

Table 281: Functions of the Script "VI_BuildAttestationLinks"

Function	Usage
<code>VI_BuildAttestationLink_Show</code>	Opens the attestation page in the Web Portal.
<code>VI_BuildAttestationLink_Approve</code>	Approves an attestation and opens the attestation page in the Web Portal.
<code>VI_BuildAttestationLink_Deny</code>	Denies an attestation and opens the attestation page in

Function	Usage
	the Web Portal.
VI_BuildAttestationLink_AnswerQuestion	Opens the page for answering a question in the Web Portal.
VI_BuildAttestationLink_Pending	Opens the page with pending attestations in the Web Portal.

Default Functions for IT Shop Requests

The script VI_BuildITShopLinks contains a collection of default functions for composing hyperlinks to directly grant or deny approval of IT Shop requests from email notifications.

Table 282: Functions of the Script "VI_BuildITShopLinks"

Function	Usage
VI_BuildITShopLink_Show_for_Approver	Opens the overview page for request approval in the Web Portal.
VI_BuildITShopLink_Show_for_Requester	Opens the overview page for requests in the Web Portal.
VI_BuildITShopLink_Approve	Approves a request and opens the approvals page in the Web Portal.
VI_BuildITShopLink_Deny	Denies a request and opens the approvals page in the Web Portal.
VI_BuildITShopLink_Unsubscribe	Opens the email notification configuration page in the Web Portal. This function is used in processes for unsubscribing email notifications.
VI_BuildITShopLink_AnswerQuestion	Opens the page for answering a question in the Web Portal.
VI_BuildITShopLink_Reject	Opens the page with denied requests in the Web Portal.
VI_BuildAttestationLink_Pending	Opens the page with pending requests in the Web Portal.
VI_BuildITShopLink_Unsubscribe	Creates the link for canceling email notification.

Default Functions for Identity Audit

The script VI_BuildComplianceLinks contains a collection of default functions for composing hyperlinks for exception approval of rule violations.

Table 283: Functions of the Script, "VI_BuildComplianceLinks"

Function	Usage
VI_BuildComplianceLink_Show	Opens the exception approval page in the Web Portal.

Default function for policy checking

The script VI_BuildComplianceLinks contains a collection of default functions for composing hyperlinks for exception approval of policy violations.

Table 284: Functions of the Script, "VI_BuildComplianceLinks"

Function	Usage
VI_BuildQERPolicyLink_Show	Opens the exception approval page in the Web Portal.

Related Topics

- [Using Process Parameters in Hyperlinks on page 492](#)

Using Process Parameters in Hyperlinks

Use this method to pass additional parameters to a function. Email notifications are generated during the process handling. The process task "SendRichMail" is provided for this through the process component "MailComponent".

To compose a hyperlink, for example, to unsubscribe email notifications, within a process, use the spare process parameters [ParamName 1-n] and [ParamValue 1-n] from the process component.

Example for filling the process parameter:

```
ParamName1 Value = "NoSubscription"  
ParamValue1 Value = VI_BuildITShopLink_Unsubscribe(values("UID_  
RichMail").ToString())
```

UID_RichMail is determined by the pre-script for generating within the process and passed to the function.

Take implementation examples from base object PersonWantsOrg processes that are triggered by changes to IT Shop requests.

The process parameter is referenced when a hyperlink is entered in the **Address** field:

```
$PC(<ParamName>)$
```

Example:

```
$PC(NoSubscription)$
```

Related Topics

- [Default Functions for Creating Hyperlinks on page 490](#)

Customizing Email Signatures

Configure the email signature for mail templates using the following configuration parameter.

Table 285: Configuration Parameters for Email Signatures

Configuration Parameter	Description
Common\MailNotification\Signature	Data for the signature in email automatically generated from mail templates.
Common\MailNotification\Signature\Caption	Signature under the salutation.
Common\MailNotification\Signature\Company	Company name.
Common\MailNotification\Signature\Link	Link to company website.

The script `VI_GetRichMailSignature` combines the components of an email signature according to the configuration parameters for use in mail templates.

Reports in the One Identity Manager

The One Identity Manager provides the means to create and execute multi-object reports, including totals and other aggregate functions. It is also possible to create groups and graphically represent data. Predefined reports are supplied with the schema installation. You can create and edit custom reports with the Report Editor.

You can also send reports to specified email addresses using scheduled subscriptions. You can create reports for the current state or over a specified period (change history). For every report, you can create different subscribable reports that can be requested by Web Portal users. You can also link the report to the administration tool user interfaces, for example, to view in the Manager.

Detailed information about this topic

- [Creating and Editing Reports on page 498](#)
- [Translating Reports on page 522](#)
- [Example of a Simple Report with Data Grouping on page 518](#)
- [Linking Reports into the User Interfaces on page 524](#)

Working with the Report Editor

The Report Editor is a program for creating and editing reports. The program uses StimulReport.Net components for designing the reports. You can find accurate descriptions and the functionality of individual components in the Stimulsoft online help (www.stimulsoft.com).

To start the Report Editor

- Select **Start | One Identity | One Identity Manager | Configuration | Report Editor.**
- Enter the database connection data and the system ID and log onto the program.

NOTE: When you start the Report Editor for the first time, you can select the configuration type (basic, default or professional) for the report. The configuration type determines the range of properties displayed when editing a report. You can change the configuration type later in the edit view using the context menu in the property view.

Menu Items

Table 286: Meaning of Items in the Menu Bar

Menu	Menu Item	Meaning
Database	New connection...	Creates a new database connection.
	Settings...	For configuring general program settings.
	Exit	Exits the program.
Report	New	Creates a new report.
	Save	Saves the current report in the database.
	Delete	Deletes the current report.
	Edit	Opens the property dialog for the current report.
	Refresh data	Reloads the report data from the database.
	New virtual data source	Opens a dialog box for creating a virtual data source.
	Community	Opens the One Identity Manager community website.
Help	Support Portal	Opens the One Identity Manager product support website.
	Training	Opens the One Identity Manager training portal website.
	Online documentation	Opens the One Identity Manager documentation website.
	Search...	Opens the search dialog box.
	Report Editor help	Opens program help.
	Info...	Shows the version information for program.

Table 287: Meaning of Icons in the General Toolbar

Icon	Meaning
	Creates a new report.

Icon	Meaning
	Deletes the current report.
	Saves the current report in the database.
	Opens a dialog box for editing change labels.
	Defines the current change label as default and applies it automatically.
	Opens the property dialog for the current report.
	Reloads with the newest report data.
	Opens a dialog for creating a new virtual data source.

Table 288: Functions in the Report List Toolbox

Icon	Meaning
	Displays all reports.
	Uses a filter condition to limit the number of reports displayed.
	Runs the filter and shows all reports that satisfy the filter condition. The filter condition is interpreted internally as a 'Like' comparison.
	Updates the report list.

Table 289: Functions in the Report List Context Menu

Context Menu Item	Meaning
New	Creates a new report.
Edit	Opens the property dialog box for the current report.
Edit properties...	Loads the properties dialog box for the selected report.
Copy	Copies the selected report.
Delete	Deletes the current report.

Views in the Report Editor

The Report Editor has several views for editing reports.

Table 290: Report Editor Views

View	Description
Report list	All reports are displayed by category. Uses a filter condition to limit the number of reports displayed.

View	Description
Edit view for reports	Reports are designed with the Report Designer in the edit pane. Using the Report Designer's toolbar, you can place the controls you want on the report form. ① NOTE: Use the online help from Stimulsoft StimulReport.Net (www.stimulsoft.com) as a basis for the report design.
Property dialog box	Use the view edit the properties of the selected report. A default context menu is available for input fields.
SQL log	Database queries are listed in this view. Use query logging to look for errors and to optimize the report during the design phase. For more information, see Logging Database Queries on page 498.

Customizing Program Settings

To change the program settings

- Select **Database | Settings...** from the menu.

Table 291: Program Settings

Setting	Meaning
Language	The initial program login uses the system language for the user interface. Changes to the language settings take effect after the Report Editor has been restarted. This specifies the language globally for all the One Identity Manager programs so that the language does not have to set separately in each program.
Show code tab	Set this option to display the script code edit tab in the Report Editor.
Ask on save without change label	Change labels should be used for changes to reports. Set this option so that an alert box is called when changes are saved without a change label.
Max. number of preview rows	Specify how many data sets

Related Topics

- [Enabling More Languages for Displaying and Maintaining Data on page 126](#)

Logging Database Queries

Use database query logging in the Report Editor to look for errors and to optimize the report during the design phase. The execution time and the command that was run are recorded.

- Open the log window in the bottom part of the application using the option **SQL log**.

Table 292:
Toolbar Functions for Logging Database Queries

Icon	Meaning
	Starts logging database queries.
	Stops database query logging.
	Copies recorded data to the clipboard.
	Save logged data in a file.
	Deletes the recorded data.
	Displays the recorded data in an editor.

Creating and Editing Reports

Create and edit reports with the Report Editor program. Reports are stored in the database table DialogReport. The following steps are required to create a report:

1. Defining report properties, data sources and report parameters
2. Designing the report form with the Report Designer

Predefined reports supplied with the One Identity Manager by default, automatically customized during schema installation. If you need to make changes to a default report:

1. Create a copy of the report.
2. Edit the required report properties.
3. Use the customized report from now on.

When you add or copy a report, the property dialog box opens first, which you use to enter the general data for the report, the data source required and any parameters for the report definition. Then a new report form is created in the edit view with the Report Designer. This

forms the basis of the report design. Using the Report Designer's toolbar, you can place the controls you want on the report form.

NOTE: Use the online help from Stimulsoft StimulReport.Net (www.stimulsoft.com) as a basis for the report design.

To create a new report

- Select **Report | New** from the menu.

To copy a report

- Select the report in the report list and select the menu item **Kopieren**.

This creates a new report and the property dialog box opens. The properties in the new report are taken from the original.

To edit a report

1. Select the report in the report list and open it with double-click or with **Edit** from the context menu.
This opens the report form in the Report Designer.
2. To open the property dialog, select **Report | Edit** from the menu.

To edit the report properties with loading the report in the Report Designer

- Select the report in the report list and select **Edit properties...** from the context menu.
This opens the property dialog.

NOTE: After you have customized a report, you can mark it by setting change labels. Change labels are offered in the program "Database Transporter" as export criteria when a customer transport package is created.

Detailed information about this topic

- [General Report Properties on page 499](#)
- [Creating a Data Source on page 500](#)
- [Editing Report Parameters on page 510](#)
- [Using Virtual Data Sources on page 515](#)
- [Editing the Report Form on page 516](#)

General Report Properties

To edit general report properties

- Select the tab **Properties** in the properties dialog box.

Table 293: General Report Properties

Property	Meaning
Name	Report name Label custom reports with the prefix "CCC_".
Display name	Display name of the report. The display name is available when the report is created as ReportAlias and can, for example, be used to compose the title of the report or the file name when a report is exported to a web front-end. Translate the given text using the  button. The report display name can contain variables, permitted are system variables such as report parameters. The variables are passed using a percent character. Example: Name of report %variable%
Description	Report description. Translate the given text using the  button.
Filter criteria	Filter criteria for displaying the report in the web front-end.
Base table	Basis table for the report.
Category	Category for classifying reports. Permitted values are the categories "Common", "Mail" and "Attestation".
Preprocessor condition	Preprocessor conditions can be added to reports. In this case, a report is only available if the preprocessor condition is fulfilled.
Custom properties	Enter additional company specific information. Enter display names, formats and templates for the input fields (by default Spare field no. 01 to spare field no. 10) with the Designer to meet your requirements.
Extended properties	An extended property is the UID under which the report is stored in the database.

Related Topics

- [Creating a Data Source on page 500](#)
- [Editing Report Parameters on page 510](#)

Creating a Data Source

For each report you need to create a data source from which to read the report data to be displayed. Normally one data source is sufficient for one report. However, you can define several data sources for each report.

To edit a data source

1. Select the **Data source** tab in the properties dialog box.
2. Select the data source from **Defined queries**.
 - OR -
- Click **Add**.
This creates a new data source.
3. Edit the data source properties.

To delete a data source

1. Select the **Data source** tab in the properties dialog box.
2. Select the data source from **Defined queries**.
3. Click **Delete**.

You can test the results while processing a data source.

To test a data query

1. Select the **Data source** tab in the properties dialog box.
2. Select the data source from **Defined queries**.
3. Click the  button next to **Query module**.

The result of a data source is shown in a separate dialog.

 **NOTE:** When a data query is copied to the clipboard, a database query is generated in SQL syntax that you can run on the database with an appropriate query tool. To copy the data query, use the  button next to **Query module**.

Detailed information about this topic

- [Data Retrieval using an SQL Query on page 502](#)
- [Data Retrieval using a Database View on page 502](#)
- [Data Retrieval using an Object on page 503](#)
- [Data Retrieval using Single Object History on page 504](#)
- [Data Retrieval using Multiple Object History on page 506](#)
- [Data Retrieval using Historical Assignments on page 507](#)
- [Data Query for Simulation Data on page 509](#)

Related Topics

- [Using Virtual Data Sources on page 515](#)

Data Retrieval using an SQL Query

Data queries with the query module "SQL" are executed directly on the database without checking user access permissions. This means that a column to be used in the report is displayed even though the user may not have access permission to it.

Table 294: Data Source SQL Properties

Property	Meaning
Name	Name of the data source.
Description	Description of data source.
Query module	Select the query module "SQL".
Query	Full database query SQL syntax. The query must contain all the columns used in the report. You can also use SQL parameters in the query. Add these parameters subsequently to the report by entering them on the tab Parameters .

Example:

The query should return the employees (table Person) assigned to an department. The department (UID_Department) is found with the object key (XObjectKey). This is passed as a parameter to the report. The employee's first name (firstname), last name (lastname) and department name (departmentname) are queried.

```
Select Firstname, Lastname, Departmentname  
      from person join Department  
        on person.uid_Department = department.uid_Department  
       where Department.XObjectKey = @ObjectKeyBase
```

Related Topics

- [Editing Report Parameters on page 510](#)

Data Retrieval using a Database View

You can use query module "View" to create data queries using predefined database views and in this way control user access rights.

Table 295: Data Source View Properties

Property	Meaning
Name	Name of the data source.
Description	Description of data source.
Query module	Select the query module "View".
View name	Name of the database view.
Condition	Condition for limiting the data set returned from the database table. You formulate the condition as a valid WHERE clause for database queries. You may use SQL parameters in the condition. Add these parameters subsequently to the report by entering them on the tab Parameters .
Sort by	The data queries are sorted by these database view columns.

Related Topics

- [Editing Report Parameters on page 510](#)

Data Retrieval using an Object

Data queries with the query module "Object" are created using the object layer and therefore take user access permissions fully into account.

Table 296: Data Source Object Properties

Property	Meaning
Name	Name of the data source.
Description	Description of data source.
Query module	Select the query module "Object".
Parent query	In a parent query, restrictions are applied to the data record that are passed on to subsequent queries, all members of a department, for example. Parameters that are defined in the parent query are also available in subsequent queries.
Table	Select the table to find the object in.
Columns	Columns to use in the report. Some columns are always added to the report definition and must not be explicitly entered here. These include: <ul style="list-style-type: none">• The table's primary key column.

Property	Meaning
	<ul style="list-style-type: none"> • All columns used in the table display template. • "Dummy" columns (_Display and _DisplayLong) supplied by the table's display template. • An additional column (<column>_Display) is also created for the display value for foreign key columns and columns with a list of defined values or multi language entries.
Resolve foreign key	Set this option if the display value of the referenced object should be returned in <column>_Display rather than the UID.
Condition	Condition for limiting the data set returned from the table. You formulate the condition as a valid WHERE clause for database queries. You may use SQL parameters in the condition. Add these parameters subsequently to the report by entering them on the tab Parameters . Columns in a parent query are formatted with the following syntax: <code>@<parent query name>.<parent query column></code>
Sort by	The data queries are sorted by these table columns.

Related Topics

- [Editing Report Parameters on page 510](#)

Data Retrieval using Single Object History

Use data queries with the query module "Single object history" when you want to create reports about a single object, for example, one employee, with its history data.

Table 297: Properties of Data Source Single Object History

Property	Meaning
Name	Name of the data source.
Description	Description of data source.
Query module	Select the query module "Single object history".
Parent query	In a parent query, restrictions are applied to the data record that are passed on to subsequent queries, all members of a department, for example. Parameters that are defined in the parent query are also available in subsequent queries.
Object key	The object key can be queried directly or using a parameter. Add these parameters subsequently to the report by entering them on the tab

Property	Meaning
	<p>Parameters. Columns in a parent query are formatted with the following syntax:</p> <pre><parent query name>.<parent query column></pre>
Min. date or time period	Use the minimum date to specify the point in time that the history data should start from. You can define the date directly or using a parameter. In the case of a parameter, the minimum date of all the effected entries in the history database are found. Add these parameters subsequently to the report by entering them on the tab Parameters .
Resolve foreign key	Set this option if the display value of the referenced object should be returned rather than the UID.

The data query returns the following columns.

Table 298: Columns from a Data Query using Single Object History

Column	Meaning
ChangeID	Unique identifier (UID) for the record.
ObjectKey	Object key or the record.
ObjectUID	Unique identifier (UID) for the modified objects.
User	Name of user that caused the change.
ChangeTime	Time of change
ChangeType	Type of change (Insert, Update, Delete).
Columnname	Name of column whose value has changed.
ColumnDisplay	Display name of column whose value has changed
OldValue	Old column value.
OldValueDisplay	Old column display value. Only if the option Resolve foreign key is set.
NewValue	New column value.
NewValueDisplay	New value display value. Only if the option Resolve foreign key is set.

Related Topics

- [Editing Report Parameters on page 510](#)

Data Retrieval using Multiple Object History

Use data queries with the query module "Multiple object history" to create reports about multiple objects with historical data that have additional restricting criterion, for example all employees with the last name "Miller".

Table 299: Properties of Data Source Multiple Object History

Property	Meaning
Name	Name of the data source.
Description	Description of data source.
Query module	Select the query module "Multiple object history".
Table	Select the table to find the object in.
Min. date or time period	Use the minimum date to specify the point in time that the history data should start from. You can define the date directly or using a parameter. In the case of a parameter, the minimum date of all the effected entries in the history database are found. Add these parameters subsequently to the report by entering them on the tab Parameters .
Criteria column	Column in the table for limiting the numbers of objects even more.
Criteria value	The value of the criteria column can be queried directly or by parameter. Add these parameters subsequently to the report by entering them on the tab Parameters .

The data query returns the following columns.

Table 300: Columns from a Data Query using Single Object History

Column	Meaning
ChangeID	Unique identifier (UID) for the record.
ObjectKey	Object key or the record.
ObjectUID	Unique identifier (UID) for the modified objects.
User	Name of user that caused the change.
ChangeTime	Time of change
ChangeType	Type of change (Insert, Update, Delete).
Columnname	Name of column whose value has changed.
ColumnDisplay	Display name of column whose value has changed

Column	Meaning
OldValue	Old column value.
OldValueDisplay	Old column display value. Only if the option Resolve foreign key is set.
NewValue	New column value.
NewValueDisplay	New value display value. Only if the option Resolve foreign key is set.

Example

A history of all employees with the last name "Miller" should be created. The report data can be defined in the following way:

Table:	Employee
Min. Date:	MinDate
Criteria column:	Lastname
Criteria value:	Miller

Related Topics

- [Editing Report Parameters on page 510](#)

Data Retrieval using Historical Assignments

Use data queries with the type "historical assignments" to create reports with historical data from object assignments, for example, employee role memberships. This type is used for queries using foreign key relations as well as though assignment tables (many-to-many tables).

Table 301: Properties of Data Source Historical Assignments

Property	Meaning
Name	Name of the data source.
Description	Description of data source.
Query module	Select the query module "Historical assignments".
Parent query	In a parent query, restrictions are applied to the data record that are passed on to subsequent queries, all members of a department, for example. Parameters that are defined in the parent query are also available in

Property	Meaning
	subsequent queries.
Table	Table for the assignment.
Min. date or time period	Use the minimum date to specify the point in time that the history data should start from. You can define the date directly or using a parameter. In the case of a parameter, the minimum date of all the effected entries in the history database are found. Add these parameters subsequently to the report by entering them on the tab Parameters .
Criteria column	Column in the table for linking to the base object.
Criteria value	The value of the criteria column can be queried directly or by parameter. Add these parameters subsequently to the report by entering them on the tab Parameters . Columns in a parent query are formatted with the following syntax: <parent query name>.<parent query column>
Disabling columns	Certain tables contain columns that can disable an object, for example, the column AccountDisable in the table ADSAccount. Enter these column if an assignment should be labeled as "deleted" when disabled and "Added" if enabled.
Additional object columns	Enter the columns from the table that should also be available in the report.

The data query returns the following columns.

Table 302: Columns from a Data Query using Historical Assignments

Column	Meaning
BaseKey	Object key for assignment base object.
BaseUID	Base object unique identifier.
ObjectKey	Assignment object key.
DestinationKey	Object key for assignment target object.
DestinationUID	Target object unique identifier.
Display	Target object display value.
CreationUser	User that created the assignment.
CreationTime	Time of assignment.
DeletionUser	User that deleted the assignment.

Column	Meaning
DeletionTime	Time of deletion.
Type	More detailed specification of the assignment, for example, assignment table name or target system type.

Related Topics

- [Editing Report Parameters on page 510](#)

Data Query for Simulation Data

To select simulation data generated during simulation in the Manager or Manager in a report, use the following query modules:

- Front-end simulation result
You can apply this query module to all parts of a simulation excluding rule violation analysis.
- Front-end simulation result for compliance
You can apply this query module to publish the rule violation analysis in the report.

Table 303: Data Source Front-End Simulation Result Properties

Property	Meaning
Name	Name of the data source.
Description	Description of data source.
Query module	Select the query module "Front-end simulation result".
Parent query	Not used.
Simulation analysis	Defines which part of the simulation analysis is shown in the report. For more information, see Table 304 on page 509.

Table 304: Simulation Analysis Type

Type	Description
Overview	Shows which actions were triggered through changes made during the simulation in an overview.
Changed object	Shows objects and their properties affected by the changes made during simulation.
DBQueue	Shows the calculation tasks for the rDBQueue Processoresulting from

Type	Description
	changes made during simulation.
Trigger changes	Shows all changes made to objects during simulations due to triggering.
Generated process	Shows processes and process steps generated during simulation due to the changes.

Table 305: Data Source Front-End Simulation Result for Compliance Properties

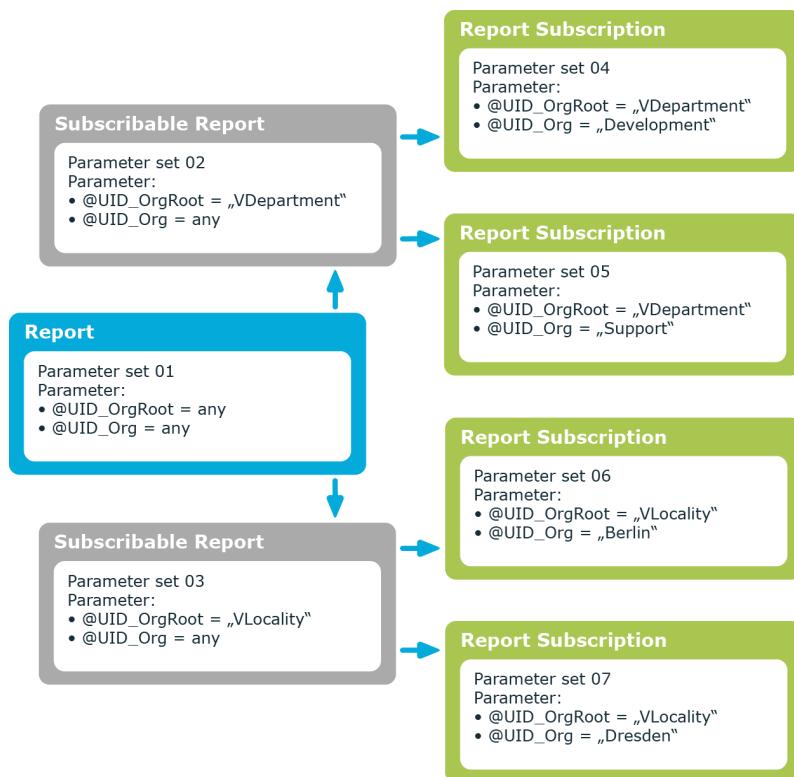
Property	Meaning
Name	Name of the data source.
Description	Description of data source.
Query module	Select the query module "Front-end simulation result".
Parent query	Not used.

Editing Report Parameters

A report can contain several parameters that are determined when the report is created or when an email notification is generated and passed to the report. The generated report is then displayed or send by email to the subscriber corresponding to the report subscription set up. The user can query the report parameters before the report is displayed. This means, you can, for example, limit the time period or pass specific departments for displaying the report.

Report parameters are grouped internally into parameter sets. A separate parameter set is automatically created for every report, every subscribable report and every report subscription. The parameters and their settings are passed down in the sequence *report->subscribable report->report subscriptions*.

Figure 50: Report Parameter Inheritance



You can configure report parameters at several places.

Parameters for Reports

Define the report parameters to use when you create the report in the Report Editor. This is where you specify which report parameters are viewable or writable and which are already predefined in a subscribable report.

Parameters for Subscribable Reports

When you add a subscribable report viewable parameters are displayed in the Manager. You can make further changes to these report parameters assuming they can be overwritten. That means, you specify which report parameters can be viewed or overwritten by Web Portal users and define parameter values.

Parameters for Report Subscriptions

Report parameters labeled as viewable and editable in subscribable reports, are shown to Web Portal users when they are setting up their personal report subscriptions. If the report parameters are editable, Web Portal users can modify the values in them.

NOTE: You must define all report parameters in the report that are to be available to users, for example, when the report is displayed, when subscribable reports are generated in the Manager or in Web Portal report subscriptions.

To edit report parameters

1. Select the **Parameters** tab in the properties dialog box.
 2. Select the report parameter from **Defined queries**.
 - OR -
- Click **Add**.
- Creates a new report parameter.
3. Edit the report parameter properties.

To delete a report parameter

1. Select the **Parameters** tab in the properties dialog box.
2. Select the report parameter from **Defined queries**.
3. Click **Delete**.

Detailed information about this topic

- [General Parameter Settings on page 512](#)
- [Defining Parameter Values on page 513](#)
- [Settings for Calculating Values on page 515](#)

General Parameter Settings

To edit general parameter settings

- Select the **Parameters** tab in the properties dialog box and select the **General** tab.

Property	Meaning							
Parameter Name	Name of the parameter. i NOTE: The name must agree with the name of the parameter in the data query.							
Parameter type	Type of parameter. The following are available: <table border="1"><tbody><tr><td>Fixed</td><td>Fixed parameter values are used.</td></tr><tr><td>User prompt</td><td>The user must select a parameter value through a user prompt.</td></tr><tr><td>Calculation</td><td>The parameter value is calculated at runtime when the report is created.</td></tr></tbody></table>		Fixed	Fixed parameter values are used.	User prompt	The user must select a parameter value through a user prompt.	Calculation	The parameter value is calculated at runtime when the report is created.
Fixed	Fixed parameter values are used.							
User prompt	The user must select a parameter value through a user prompt.							
Calculation	The parameter value is calculated at runtime when the report is created.							
	Other settings are shown or hidden depending on the type.							

Property	Meaning
Display name	User friendly name for the report parameter. Translate the given text using the  button.
Description	Detailed description of the report parameter. Translate the given text using the  button.
Sort order	Position of the report parameter in the subscribable report view and in the Web Portal.
Mandatory parameter	You must enter value in this report parameter.
Viewable	Specifies whether the report parameters is displayed.
Can be overwritten	Specifies whether the report parameter can be overwritten.

Related Topics

- [Defining Parameter Values on page 513](#)
- [Settings for Calculating Values on page 515](#)

Defining Parameter Values

Specify the parameter value and define the parameter value characteristics. Other input is shown or hidden depending on the parameter definition values.

To edit parameter definitions

- Select the **Parameters** tab in the properties dialog box and select the **Value definition** tab.
- NOTE:** The input **Parameter value** and **Default value** are affected by the parameter value definition. You can see this on the one hand, through dynamic customization of the controls for selecting a parameter value or default value and the dynamic customization of selectable values themselves on the other. It is therefore recommended, you edit these values last.

Table 306: Value Definition Properties

Property	Meaning
Data type	Data type for the report parameter.
Value range	Specifies whether the report parameter value has to be within a given range. If "yes", Parameter value (from) , Parameter value (to) and Default value (from) , Default value (to) are displayed.

Property	Meaning
Multivalue	The report parameter can have multiple values. You can select more than one value.
Multiline	The report parameter can have more than one line, which means that line breaks are allowed.
Data source	Type of data source. You can select the values "None", "table", "List of permitted values".
Table column (query)	(Only for data source "Table") Table column for selecting the value. You can select a value from this table column. You can select several values from this column if the report parameter is multi-value as well.
Display Pattern	The display template for displaying table entries in the administration tool result lists are displayed. If a customer specific display template exists it is used instead of the default display template. Syntax: %column name%
Condition (query)	Limiting condition (where clause) for selecting the value through a table column. You can select a value from the result set. You can select several values from this set if the report parameter is multi-value as well. You can reference other report parameters in the condition using the following syntax: \$PC(<Parametername>)\${br/> Example: UID_Database = \$PC(UID)\$ where UID is the name of the referenced report parameter.
List of permitted values	(Only for data source "List of permitted values") Lists the permitted values in this report parameter in the notation value=description. If no = is given, the entry is counted as value AND description.
Overwrite empty value	Specified whether an empty report parameter overwrites the default value.
Example value	The example value is used to create a report preview.
Default value	Report parameter default value. This is used, for example, if the Web Portal user does not specify a parameter value.

Related Topics

- [General Parameter Settings on page 512](#)
- [Settings for Calculating Values on page 515](#)
- [Display Template for Displaying a List on page 268](#)

Settings for Calculating Values

To edit settings for value calculation

- Select the **Parameters** tab in the properties dialog box and select the **Value calculation** tab.

Table 307: Value Calculation Properties

Property	Meaning
Table column (calc.)	Table column for selecting the value. The parameter value is determined at runtime when the report is created.
Condition (calc.)	Limiting condition (where clause) for selecting the value through a table column. The parameter value is determined at runtime when the report is created. If the report parameter is multivalue as well, several values may be found.
Script for finding values	Script in VB.Net syntax for finding the parameter value.
Script for checking values	Script in VB.Net syntax for checking permitted values of parameters.

Related Topics

- [General Parameter Settings on page 512](#)
- [Defining Parameter Values on page 513](#)

Using Virtual Data Sources

You can use virtual data sources when you want to use a data source more than once within a report, but with other limitations or sorted differently.

To create a virtual data source

1. Select the report in the report list and open it with double-click or with **Edit** from the context menu.
This opens the report form in the Report Designer.
2. Select **Report | New virtual data source** from the menu.
Opens a dialog window showing all existing data sources for the report.
3. Configure the properties for the virtual data source.

Related Topics

- [Creating a Data Source on page 500](#)

Editing the Report Form

You can create and edit reports in the edit view of the Report Editor. The Stimulsoft Reports.Ultimate Report Designer is integrated into the edit view. You can find accurate descriptions and the functionality of individual components in the Stimulsoft online help (www.stimulsoft.com).

 **NOTE:** When you start the Report Editor for the first time, you can select the configuration type (basic, default or professional) for the report. The configuration type determines the range of properties displayed when editing a report. You can change the configuration type later in the edit view using the context menu in the property view.

The following functions are appended to the Stimulsoft Reports.Ultimate Report Designer toolbar:

Table 308: Extensions to Stimulsoft Reports.Ultimate Report Designer Toolbar

Icon	Meaning
	Imports a reports (XML format).
	Export a report (XML format).
	Globalization editor. Opens the Report Designer globalization editor.
	Opens the "Translate Text" dialog box.

Detailed information about this topic

- [Adding Data Fields to a Report Form on page 516](#)
- [Translating Reports on page 522](#)
- [Example of a Simple Report with Data Grouping on page 518](#)

Adding Data Fields to a Report Form

Add the control elements for the data you want to appear in the report on the report form and link them to the data source columns. After you created the data sources, they are listed with all the columns used in the Report Designer's dictionary under the entry "Quest". The report parameters are also available under the "Quest" entry.

You can find accurate descriptions and the functionality of individual components in the Stimulsoft online help (www.stimulsoft.com).

To insert data boxes into the report form

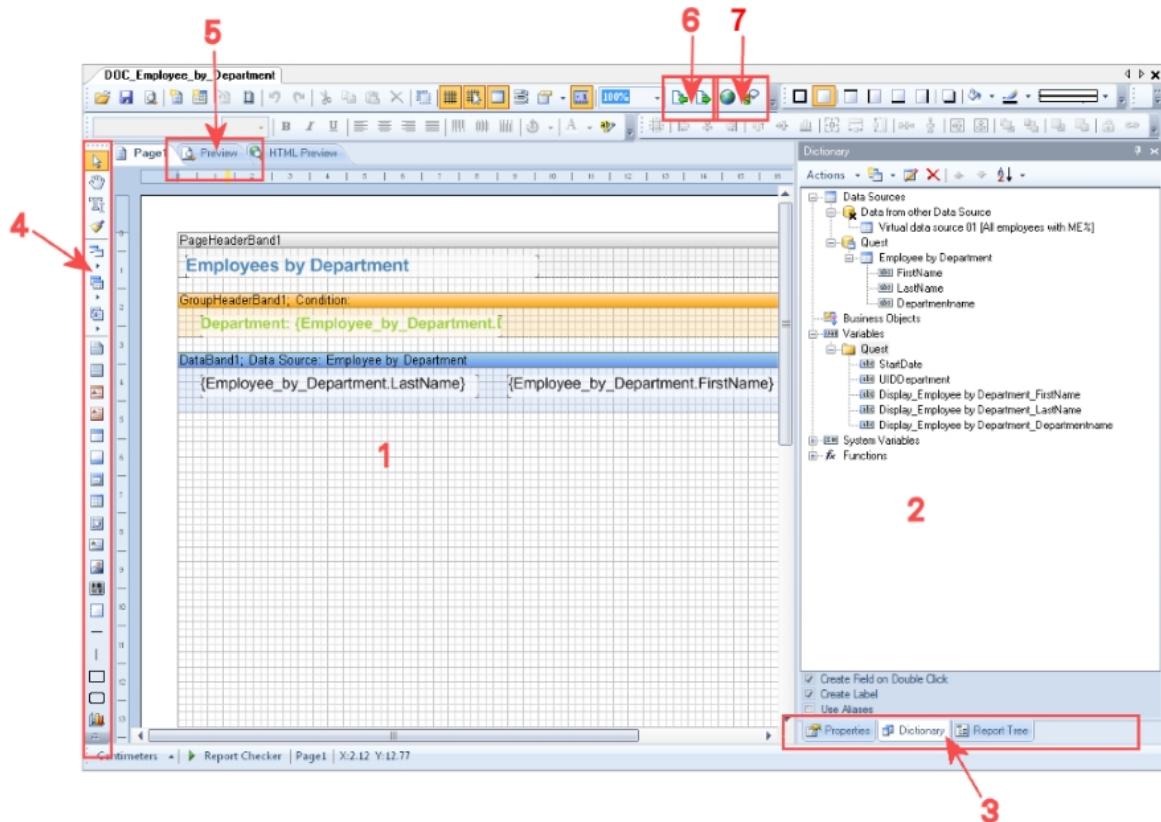
1. Select the column you want to add to the report in the dictionary (tab **Dictionary**).
2. Place the column on the report form using "Drag and drop".

This creates a new control element on the report form which includes some predefined variables.

TIP: You can add other control elements as necessary with the Report Designer tool palette.

3. The Report Designer properties window (tab **Properties**) allows you to customize individual control elements.
4. Use the **Preview** to view the report during editing. The preview takes some sample parameter values to determine the data for the preview display.

Figure 51: Report Designer with Report Form (1), Dictionary/Properties View (2), Tabs for Swapping between Dictionary/Properties (3), Toolbox (4), Preview (5), Import/Export of Report Pages (6), translate report (/)



Related Topics

- Example of a Simple Report with Data Grouping on page 518

Example of a Simple Report with Data Grouping

We want to create a report that lists all employees as grouped in their respective departments.

1. A new report is created to do this.

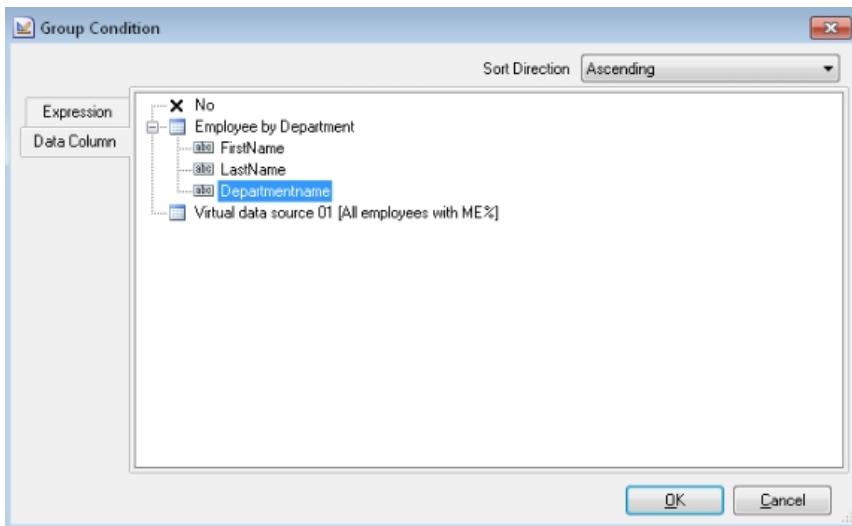
- The report is given the name "CCC_Employee_by_Department". The display name is "Employees by Department %UID%".
- A data source "Employee by department" is created for the report with the query module "SQL". The data query should return the employees assigned to a department. The department is found with the object key (XObjectKey). This is passed as a parameter to the report. The employee's first name (firstname), last name (lastname) and department name (departmentname) are queried.

```
Select Firstname, Lastname, Departmentname
```

```
from person join Department  
on person.uid_Department = department.uid_Department  
where Department.XObjectKey = @UIDDepartment
```

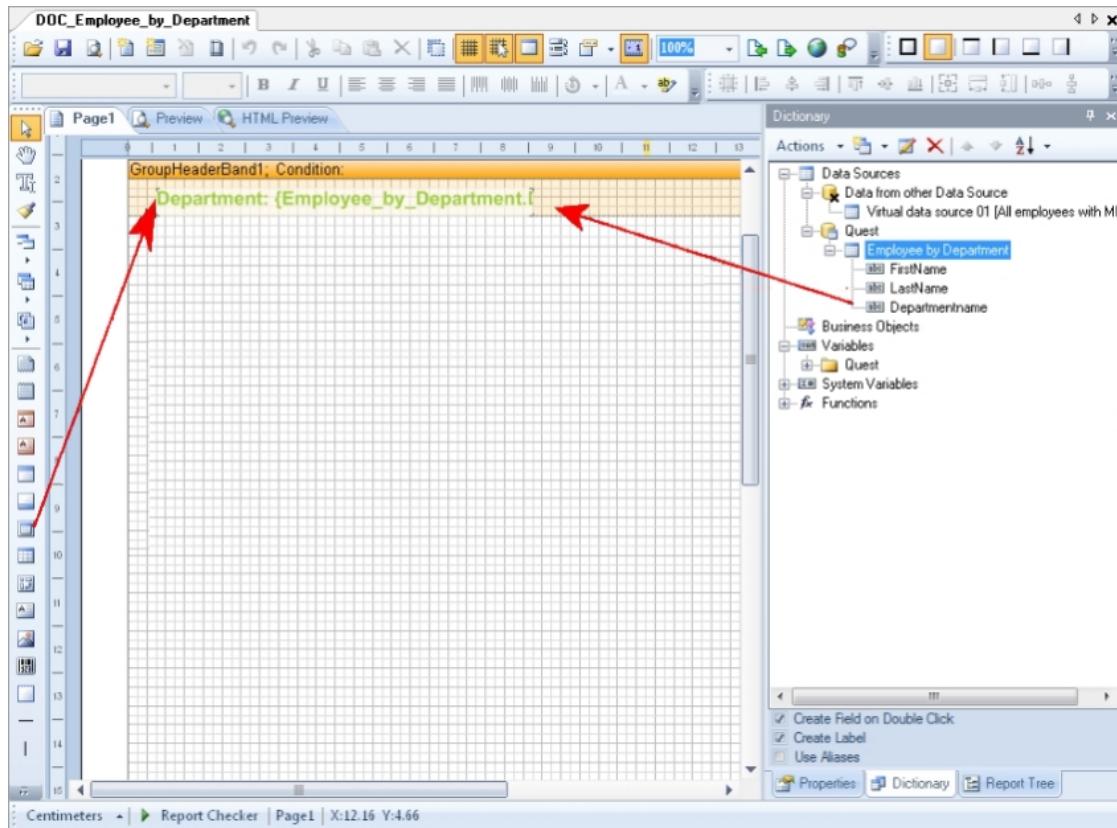
- This adds the parameter "UIDDepartment" to the report and fill the preview with a sample value.
2. Next, the control elements for the database columns are arranged on the report form.
- Add a band of type "group header" from the Report Designer's toolbox to the report form. The column name used for grouping has to entered as "grouping condition", which is Departmentname in the example.

Figure 52: Specifying the Grouping Condition



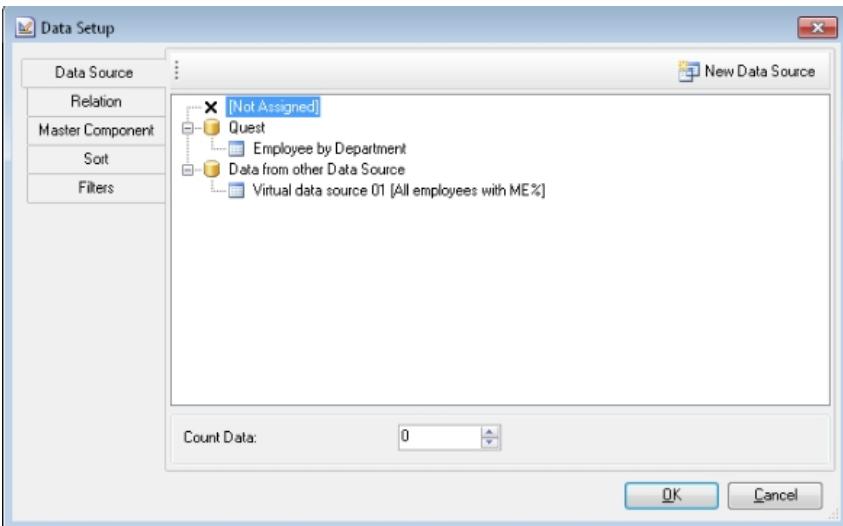
3. Afterwards drag and drop the column Departmentname from the Report Designer's dictionary (tab **Dictionary**) into the group header. This creates a new control element on the report form.

Figure 53: Creating a Group



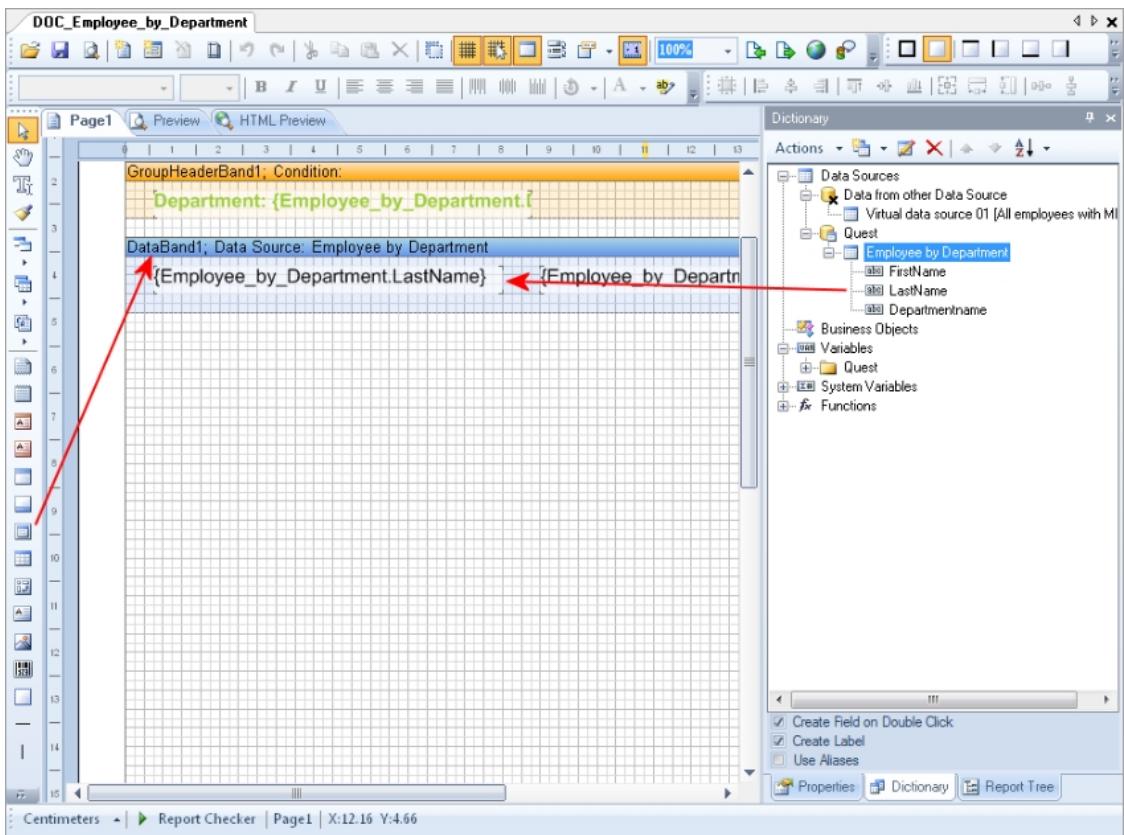
4. To display employees, add a new band of type "data band" to the report form using the Report Designer's toolbox. Specify the data source to be "Employee by Department".

Figure 54: Specifying the Data Source



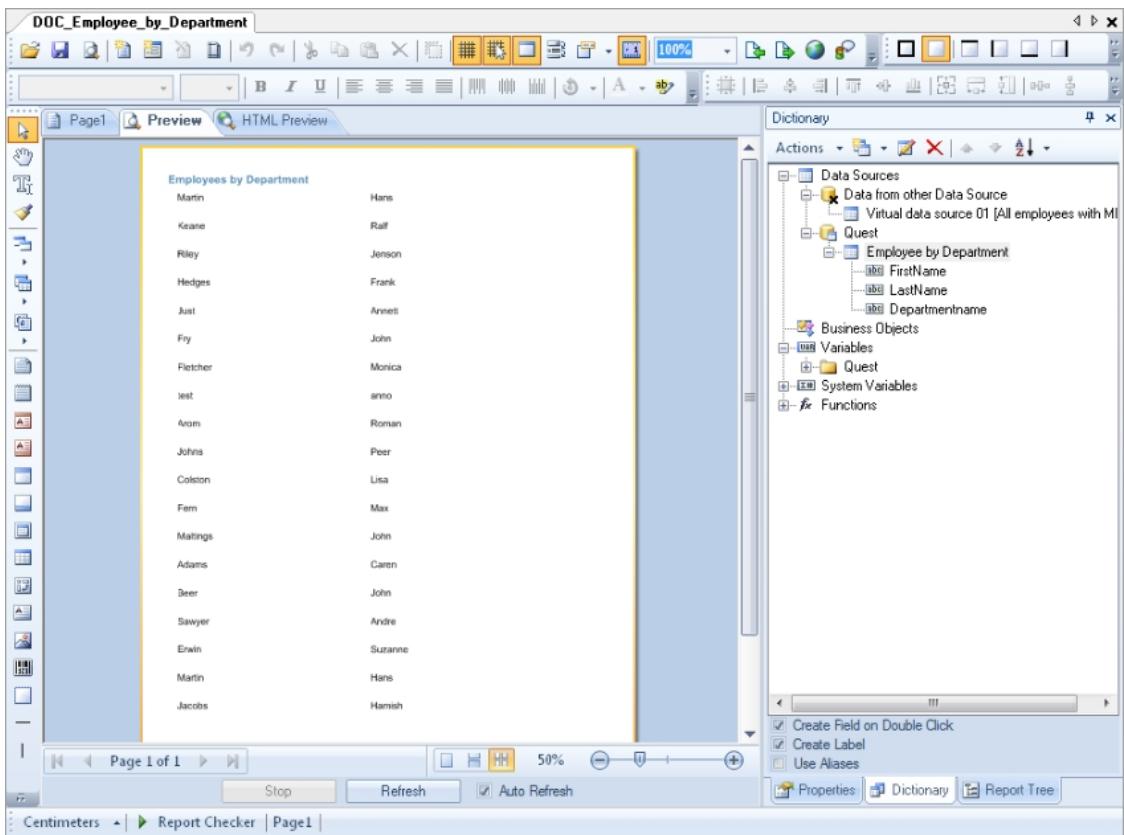
5. Next, drag and drop the columns Lastname and Firstname from the Report Designer's dictionary (tab **Dictionary**) into the data band. This creates the respective control elements on the report form.

Figure 55: Organizing Control Elements on a Report Form



6. Other control elements such as a title (PageHeader) can be added as necessary with the Report Designer. The Report Designer properties window (tab **Properties**) allows you to customize individual control elements.
7. The preview can be used to view the report during setup. The preview uses the sample parameter values in the parameter view of the report edit dialog to determine the data for this.

Figure 56: Report Preview



Detailed information about this topic

- [General Report Properties on page 499](#)
- [Creating a Data Source on page 500](#)
- [Editing Report Parameters on page 510](#)
- [Editing the Report Form on page 516](#)

Translating Reports

A report can contain several elements which require translating in order to display the report in more than one language:

- Database columns used in the report definition.
Translate database columns with the Language Editor in the Designer.
- Display name/ReportAlias

The report's display name used when a report is created as ReportAlias. Enter the display name in the report properties dialog box and translate it there. Translate the given text using the  button.

- Text elements on the report form.

Translate text elements directly in the Report Editor with the Globalization Editor.

To translate all text elements in a report

1. Select the report in the report list and open it with double-click or with **Edit** from the context menu.

This opens the report form in the Report Designer.

2. Start the Globalization Editor.

- Click on the  button in the Report Designer toolbar.

- OR -

- Select the report on the **Properties** tab in the Report Designer's properties view and open the Globalization Editor using **Globalization Strings**.

 **NOTE:** You can only start the Globalization Editor from the Report Designer's properties view when you have selected the configuration type "Professional". You can change the configuration type later in the edit view using the context menu in the property view.

3. Ensure **Auto localize report on run** is set.

This means the report is generated in the current language.

4. Enter a culture for the language using **Add culture** and translate each entry.

To translate single captions

1. Select the report in the report list and open it with double-click or with **Edit** from the context menu.

This opens the report form in the Report Designer.

2. Select the caption on the report form.

3. Open the dialog box using the  in the Report Designer toolbar.

4. Translate the text and confirm the changes with **OK**.

Related Topics

- [Editing Translations in the Language Editor on page 333](#)
- [General Report Properties on page 499](#)

Linking Reports into the User Interfaces

In order to present a report in a One Identity Manager administration tool, such as the Manager, you need to link in the report as a custom interface form. Create the interface form in the program Designer.

To create a user interface form

1. Select **User Interface | Forms | User interface forms** in the Designer.
2. Select **Form | Insert** from the menu.
3. Edit the user interface form master data.

Take the following cases into account:

- Use the form definition "VI_Report".

This form definition is configured for displaying in the graphical user interface and in web applications. You only need to set up one interface form for this. Which form template will be used to display the interface form is decided dynamically, depending on usage.

- In the configuration data the name of the report to run and the special report parameters are passed to the interface form in "SpecialSheetData".

Syntax:

```
<DialogSheetDefinition FormatVersion="1.0">  
    <SpecialSheetData>report name|parameter1=value1|parameter2=value2 .  
    ...  
    </SpecialSheetData>  
</DialogSheetDefinition>
```

Example:

```
<DialogSheetDefinition FormatVersion="1.0">  
    <SpecialSheetData>CCC_Employee_by_Department|UIDDepartment=%UID_  
    Department%</SpecialSheetData>  
</DialogSheetDefinition>
```

4. Assign the form to the application. Use the **Program** view to do this.
5. The form also needs to be linked to a permissions group. Use the **Object relations** view to do this.
6. Then you must assign the form to a permissions group. Use the **Permissions groups** view to do this. Read Effects of Object Definitions when Displaying Interface Forms about controlling the display of forms through object definitions.
7. Assign the menu item to the user interface form, if it is to be displayed in the user interface depending on this menu item. Use the **Menu assignments** view to do this. Use the **to** assign a menu item to the interface form.

Reports can be created and exported on a cyclical basis or event controlled. This functionality is provided in the form of a process component. You can create a report with the process component "ReportComponent" and export it in various file formats. The following formats are supported: HTML, PDF, RTF, TEXT, XLS, TIFF, XML, CSV, XPS, DOCX and XLSX . To use this functionality, create a custom process.

 **NOTE:** Use the default report server as executing server in the processes.

Related Topics

- [Editing Interface Forms on page 279](#)
- [Editing Menu Items on page 258](#)
- [Defining Processes on page 391](#)

Custom Schema Extensions

The One Identity Manager database distinguishes between reference data and meta data. Reference data is specified by the application data model and meta data by the system data model. Meta Data includes data for specifying the application data model, for regulating access rights and to influence and control system behavior such as modifying One Identity Manager administration tools to meet user defined requirements. The object technology implemented in One Identity Manager makes it possible to add customer specific columns and tables to the existing application data model at database level. These are therefore available at object level with all the corresponding tasks. Custom extensions to the system data model are not permitted.

- ❶ **IMPORTANT:** Use the program, "Schema Extension" to extend the One Identity Manager data model. Schema extensions are added to the database using "Schema Extension" and the necessary extensions are made in the One Identity Manager data model.
- ❷ **IMPORTANT:** Run the consistency checks after the schema extensions and apply the repair methods. You must subsequently edit table and column definitions in the Designer.

You can make the following extensions in the Schema Extension:

- Create new tables
- Create new assignment tables
- Create new columns
- Create new views
- Create new indexes

Additional steps for extending the schema are:

- Issuing permissions for schema extensions
- Specifying change labels
- Add schema extensions to the database

You cannot create custom functions, triggers or database procedures with the program Schema Extension. If you require custom functions, triggers or database procedures, add them with a suitable query tool into the database. Keep to the following conventions for name database components.

- Name begin with the string `ccc_`.
- All names are a maximum of 30 characters long.
- It is recommended to use UpperCamelCase for as notation for the names.

Detailed information about this topic

- [Creating a New Table on page 527](#)
- [Creating New Assignments Tables on page 537](#)
- [Extending a Table on page 529](#)
- [Creating a Read-Only Database View on page 534](#)
- [Creating a Union View on page 536](#)
- [Creating Indexes on page 538](#)
- [Permissions for Schema Extensions on page 539](#)
- [Specifying Change Labels for Schema Extensions on page 540](#)
- [Adding Schema Extensions on page 541](#)

Related Topics

- [The One Identity Manager Data Model on page 161](#)
- [Basics of the Data Model on page 162](#)
- [General Advice for Editing Table and Column Definitions on page 165](#)
- [Adding in Custom Extensions to the User Interface on page 542](#)

Creating a New Table

To create a simple table in the Schema Extension

1. Open Launchpad and select **One Identity Manager Schema Extension**. This starts the program "Schema Extension".
2. Click **Next** on the start page.
3. Enter connection credential for the One Identity Manager database on the **Database connection** page and click **Next**.
4. Select the extension method **New table** table on the **Extension method** page and click **Next**.

This creates a table with table type "T".

5. Enter the table properties on the page **Create new table** and click **Next**.

Table 309: Table Properties

Property	Description
Table	Table name. The name of the new table must begin with the prefix "CCC". The table name is formatted as CCC<table name>.
Display name	The display name is used, for example, to identify the table in a database search or for error output.
Description:	You can enter a comment on a table about using it in the data model.

6. Next step: You create a new columns on the **Configure columns** page. [For more information, see Defining Columns on page 529](#).

 **NOTE:** The following columns are automatically created:

- Primary key column
The primary key column is automatically taken to be the UID. The name of the primary key column is formatted as UID_CCC<table name>.
- X Columns (XUserInserted, XUserUpdated, XDateInserted, XDateUpdated, XTouched, XObjectKey, XMarkedForDeletion)

Detailed information about this topic

- [Defining Columns on page 529](#)
- [Creating a Simple Column on page 530](#)
- [Creating a Foreign Key Column on page 531](#)
- [Creating New Columns for Database Views on page 531](#)
- [Configuring Column Properties on page 532](#)

Related Topics

- [Extending a Table on page 529](#)
- [Creating New Assignments Tables on page 537](#)
- [Creating a Read-Only Database View on page 534](#)
- [Creating a Union View on page 536](#)
- [Mapping Table Definitions on page 172](#)

Extending a Table

To extend an existing table in the Schema Extension

1. Open Launchpad and select **One Identity Manager Schema Extension**. This starts the program "Schema Extension".
2. Click **Next** on the start page.
3. Enter connection credential for the One Identity Manager database on the **Database connection** page and click **Next**.
4. Select the extension method **Extend table** on the **Extension method** page and click **Next**.
5. On the **Extend table** page, select the table for you want to extend and click **Next**.
6. Next step: You create a new columns on the **Configure columns** page. **For more information, see Defining Columns on page 529.**

Detailed information about this topic

- [Defining Columns on page 529](#)
- [Creating a Simple Column on page 530](#)
- [Creating a Foreign Key Column on page 531](#)
- [Creating New Columns for Database Views on page 531](#)
- [Configuring Column Properties on page 532](#)

Related Topics

- [Creating a New Table on page 527](#)
- [Creating New Assignments Tables](#)
- [Creating a Read-Only Database View on page 534](#)
- [Creating a Union View on page 536](#)
- [Mapping Table Definitions on page 172](#)

Defining Columns

On the **Define columns** page in the Schema Extension, you can see which columns already exists for the selected table and how many resources are free for new columns.

 **NOTE:** Take the maximum size allowed for a table into account when extending.

If the table you are extending is a simple table (table type "T") or a base table (table type "B") then you can decide whether to create a simple or a foreign key column.

If the table to be extended is a "V" (view) type, only the custom columns of the underlying base table (type "B") are shown in the selection menu. You can only select custom columns from the base table that are not yet used in the view.

The following functions are available to you:

Table 310: Functions

Icon	Meaning
	Adds a new column.
	Deletes a newly inserted column definition.
	Displays the configuration options for column properties.

Detailed information about this topic

- [Creating a Simple Column on page 530](#)
- [Creating a Foreign Key Column on page 531](#)
- [Creating New Columns for Database Views on page 531](#)
- [Configuring Column Properties on page 532](#)

Related Topics

- [Mapping Table Definitions on page 172](#)
- [Mapping Column Definitions on page 188](#)

Creating a Simple Column

To add a simple column in the table in the Schema Extension

1. Click on the **Define column** page.
2. Select **Simple column**.
3. Enter the name of the column in **Column name**.
If a custom table is extended, you can give it any name. If a default table is extended, the column must begin with the prefix "CCC". The column name is formatted as CCC_<column name>.
4. Click **OK**.
5. Configure more column properties.

Related Topics

- [Creating a Foreign Key Column on page 531](#)
- [Creating New Columns for Database Views on page 531](#)

- [Configuring Column Properties on page 532](#)

Creating a Foreign Key Column

The following restrictions apply if the new column is a foreign key:

- The name of the new column must be unique in the table.
- The referenced table has a one column primary key.
- The value type and some other properties are taken from the referenced table primary key column.
- Validation of referential integrity is done by DLL or triggered.
- The table and column relations are created automatically. Column relations follow the naming convention:
`CCC-<database ID>-<4 digit sequential number>`
- If a database view is extended by a foreign key, the column relations following the naming convention:
`CCC-<database ID>-<4 digit sequential number> <Name of referenced table>`
- If a column is referenced from a base table (table type "B"), the table and column relations are also created for the underlying base table.

To add a foreign key column in the table in Schema Extension

1. Click  on the **Define column** page.
2. Select **Foreign key column**.
3. Select the referenced table in **From table**.
4. Enter the name of the column from the referenced table in **Column name**.
5. Click **OK**.
6. Configure more column properties.

Related Topics

- [Creating a Simple Column on page 530](#)
- [Creating New Columns for Database Views on page 531](#)
- [Configuring Column Properties on page 532](#)
- [Mapping Table Relations on page 208](#)

Creating New Columns for Database Views

If the table to be extended is a "V" (view) type, only the custom columns of the underlying base table (type "B") are shown in the selection menu. You can only select custom columns

from the base table that are not yet used in the view.

- First extend the base table by a new column (simple column or foreign key column).
- Then you extend the database view with the new columns.

To create a new column for a database view in the Schema Extension

1. Click  on the **Configure columns** page.
2. Select the base table column to be inserted into the database view from **Base column**.
3. Enter the name of the column in **Column name**.
4. Click **OK**.
5. Configure more column properties.

Related Topics

- [Creating a Simple Column on page 530](#)
- [Creating a Foreign Key Column on page 531](#)
- [Configuring Column Properties on page 532](#)
- [Database Views of Type "View" on page 181](#)

Configuring Column Properties

To edit column properties

1. Select the column on the **Define column** page and click the  button.
2. Configure the column properties.
3. Click **OK**.

You can configure the following settings for a column.

Table 311: Configuring the Column Properties

Property	Remarks
Name	Technical identifier for the column.
Data type	Permitted .Net data types are listed in a pop-up menu. These are represented internally as SQL data types.
Length	A length is only given for the .Net data type "String". A length of 38 is entered for a UID.
Column contains UIDs	Marks a column as a UID (table <code>DialogColumn</code> , column <code>IsUID</code>). This option is only permitted for columns with the .Net data type "String" and length 38.

Property	Remarks
Column contains unicode	Marks the column as unicode. This option is only permitted for .Net data types "String" and "Text".
Primary key	The primary key is used.
Compulsory field	Labels column as compulsory (table DialogColumn, column CustomMinLen).
Display name	Column name for displaying (table DialogColumn, column Caption).
Display in Filter Designer	The column is displayed in the Filter Designer or the Rule Editor for creating request.
Do not auto extend permissions	Permissions for predefined permissions groups are not issued automatically for this custom schema extension on a predefined table, even if the configuration parameter "Common\AutoExtendPermissions" is set.
Comment	Enter a comment, which provides information about using the new column. (table DialogColumn, column CustomComment).
Initial value	Specifies an initial value for the column. This is transferred to the existing data set in the extended table. The initial value for numerical data types is "0". The initial value for the data type "Bool" is "False".
Sort order	The sort order specifies the position for displaying the column on the generic form and the custom tab of the default form. Columns with a value less than one are not displayed.
Foreign key	This is a foreign key column.
From table	Only for foreign key tables: references tables by foreign key relations.
Delete restrictions	Only for foreign keys: restrictions for testing referential integrity when an object is deleted.
Insert restrictions	Only for foreign keys: restrictions for testing referential integrity when an object is inserted.

Only value types that are already in the One Identity Manager data model are permitted.

Table 312: List of Permitted Data Types

.Net Data Types	SQL Server data type
Binary	Image
Bool	Bit

.Net Data Types	SQL Server data type
Date	Datetime
Double	Float
Int	Int
Long	Bigint
String	Varchar/nVarchar and Char (only String(38))
Text	Text, nText

Table 313: Permitted Restrictions for Testing Referential Integrity

Restriction	Meaning
DeleteNotRestricted (D)	Dependencies are not taken into account on deletion.
DeleteRestrict (DR)	The object can only be deleted when no more references to other objects exist.
DeleteCascade (DC)	All dependent objects are deleted when this object is deleted.
DeleteSetNULL (DS)	All links to other objects are deleted when the object is deleted (SetNULL).
InsertNotRestricted (I)	Dependencies are not taken into account on insertion.
InsertRestrict (IR)	Checks for the referenced object when the object is added.

Related Topics

- [Creating a Simple Column on page 530](#)
- [Creating a Foreign Key Column on page 531](#)
- [Creating New Columns for Database Views on page 531](#)
- [Mapping Column Definitions on page 188](#)
- [Mapping Table Relations on page 208](#)

Creating a Read-Only Database View

Views with the table type "read only" can be parts but also unions of the underlying tables. Views with type "read only" are predefined. Templates and formatting rules cannot be defined for columns in these views.

Database views of type "read only" are mainly used to edit the user interface and for creating reports.

To create a database view of type "read-only" in the Schema Extension

1. Open Launchpad and select **One Identity Manager Schema Extension**. This starts the program "Schema Extension".
2. Click **Next** on the start page.
3. Enter connection credential for the One Identity Manager database on the **Database connection** page and click **Next**.
4. Select the extension method **New view** table on the **Extension method** page and click **Next**.
5. Enter the properties for the database view on the **Create view** page and click **Next**.

Table 314: Database View Properties

Property	Description
Table	Name of the table. The name of the new table must begin with the prefix "CCC". The table name is formatted as CCC<table name>.
Display name	The display name is used, for example, to identify the table in a database search or for error output.
Description	You can enter a comment on a table about using it in the data model.
View definition	Enter the database query as a SELECT instruction. The first column in the database query is used as primary key column in the database view.

NOTE: It is recommended you reference the primary key column for the queried table as the first column in view definition. If this is not possible, then at least select a unique characteristic.

6. Next step: Create foreign key relations on the page **Create FK relations for view**. For more information, see [Creating Foreign Key Relations for Views on page 535](#).

Related Topics

- [Database Views of Type "Read-only" on page 187](#)
- [Creating a Union View on page 536](#)

Creating Foreign Key Relations for Views

If a database view contains a foreign key column, you specify which destinations tables should be referenced. The table and column relations are generated automatically.

To create a foreign key for a view in the Schema Extension

1. You create foreign key relations on the page **Create FK relations for view**.
 - a. Double-click on  in front of the column name to release the "Destination table" column.
 - b. Select the destination table in the column "Destination table".
 - c. Click **Next**.
2. Next step: You define the column properties on the **Configure columns** page. **For more information, see Configuring Column Properties on page 532.**

Related Topics

- [Creating a Read-Only Database View on page 534](#)
- [Creating a Union View on page 536](#)
- [Creating a Foreign Key Column on page 531](#)
- [Database Views of Type "Read-only" on page 187](#)
- [Database Views of Type "Union" on page 183](#)

Creating a Union View

Database views with the table type "Union" are views of the union of different tables and supply a grouping of different object types with the same context. Thus the union view QERAccProductUsage determined, which service items are used in which products, for example.

Views of type "Union" are predefined. Templates and formatting rules cannot be defined for columns in these views. The object key column (XObjectKey) must be referenced in the view definition. This makes it possible to create single object with its valid permissions.

"Union" views are mainly used for editing the user interface and for creating reports.

To create a database view of type "union" in the Schema Extension

1. Open Launchpad and select **One Identity Manager Schema Extension**. This starts the program "Schema Extension".
2. Click **Next** on the start page.
3. Enter connection credential for the One Identity Manager database on the **Database connection** page and click **Next**.
4. Select the extension method **New union** table on the **Extension method** page and click **Next**.

- Enter the database view properties on the **Create union view** page and click **Next**.

Table 315: Database View Properties

Property	Description
Table	Name of the table. The name of the new table must begin with the prefix "CCC". The table name is formatted as CCC<table name>.
Display name	The display name is used, for example, to identify the table in a database search or for error output.
Description	You can enter a comment on a table about using it in the data model.
View definition	Enter the database query as a SELECT instruction. You must reference the object key (XObjectKey) as the first column in the database query. Using the object key makes faster access to individual objects with valid permissions possible.

- Next step: Select the permissions groups to obtain permissions for the schema extensions on the page, **Access permissions**. For more information, see [Permissions for Schema Extensions on page 539](#).

Related Topics

- [Database Views of Type "Union" on page 183](#)
- [Creating a Read-Only Database View on page 534](#)

Creating New Assignments Tables

To create a new assignment (many-to-many) table in the Schema Extension.

- Open Launchpad and select **One Identity Manager Schema Extension**. This starts the program "Schema Extension".
- Click **Next** on the start page.
- Enter connection credential for the One Identity Manager database on the **Database connection** page and click **Next**.
- Select the extension method **New relation** table on the **Extension method** page and click **Next**.
- Enter the properties for the assignment table on the **Create a relation table** page and click **Next**.

Table 316: Assignment Tables Properties

Property	Description
Table	Name of the table. The name of the new table must begin with the prefix "CCC". The table name is formatted as CCC<table name>.
Display name	The display name is used, for example, to identify the table in a database search or for error output.
Description	You can enter a comment on a table about using it in the data model.
Create XOrigin column (for assignment requests)	You can create the origin column (XOrigin) optionally. The origin of an assignment is stored in this column as a bit field. Each time an entry is made in the assignment table the bit position is changed according to the assignment type.
Related tables	Use Left table and Right table to specify which tables are involved in the relation table.
Column names	Enter the relevant columns in Column name for each side of the table. Select the table's primary key column.

NOTE: Table and column relations are created automatically.

6. Next step: You define the column properties on the **Configure columns** page. [For more information, see Configuring Column Properties on page 532.](#)

NOTE: The columns XObjectKey and XMarkedForDeletion are created automatically.

Related Topics

- [Creating a New Table on page 527](#)
- [Creating a Read-Only Database View on page 534](#)
- [Creating a Union View on page 536](#)
- [Mapping Table Definitions on page 172](#)

Creating Indexes

Define indexes to optimize access to database columns. An index can contain one or more database columns.

NOTE: Tables that you create with the program "Schema Extension" are automatically indexed for primary key columns and the object key column (XObjectKey).

NOTE: Creating bitmap indexes is not supported under Oracle by the program "Schema Extension". It is not recommended to generate bitmap indexes manually.

To create a new index in the Schema Extension

1. Open Launchpad and select **One Identity Manager Schema Extension**. This starts the program "Schema Extension".
2. Click **Next** on the start page.
3. Enter connection credential for the One Identity Manager database on the **Database connection** page and click **Next**.
4. Select the extension method **New index** on the **Extension method** page and click **Next**.
5. On the **Extend table** page, select the table for which you want to create an index and click **Next**.
6. Specify the columns for the index definition on the **Create index** page and click **Next**.
 - a. Click .
This opens a dialog box where you can define the columns for the index. You can see all the columns in the table on the right-hand side of the dialog window. The columns on the left-hand side of the window belong to the index.
 - b. Enter the index name in **Index name**.
A name is already suggested. You can change this as required.
 - c. Select the column you want to add to the index on the right-hand side of the dialog window.
 - d. Add the column to the index using the add button, .
Change the order of the column in the index definition if you want or remove a column from the index with the appropriate button.
 - e. Confirm with **OK**.
7. Next step: Assign the schema extension to a change label on the **Define change label** page. [For more information, see Specifying Change Labels for Schema Extensions on page 540](#).

Permissions for Schema Extensions

Specify the permissions group to be given permissions for the schema extension. This make initial access to the schema extensions possible with One Identity Manager administration tools.

You can allocate more permissions with the Permissions Editor in the Designer after these changes have been made in the database.

To specify permissions groups in the Schema Extension

1. Select the permissions groups to obtain permissions for the schema extensions on the page, **Access permissions**.

Use the menus to select:

- Permissions groups with read and write permissions
- Permissions groups which are only read-only

2. Next step: Assign the schema extension to a change label on the **Define change label** page. [For more information, see Specifying Change Labels for Schema Extensions on page 540.](#)

Related Topics

- [Editing Permissions for One Identity Manager Schema Tables and Columns on page 231](#)

Specifying Change Labels for Schema Extensions

Select a change label under which your schema extension will be grouped. The change labels appear as export criterion in the program "Database Transporter" when a customer transport packet is set up.

To assign a change label in the Schema Extension

1. Assign the schema extension to a change label on the **Define change label** page.

These options are available:

- No change label
- Add new change label
 - Enter the name of the change label in **Change label**.
- Use existing change label
 - Select a change label from the **Change label list**.

2. Next step: The schema extensions are displayed on the **Schema modifications** page. You can export the extensions to file. [For more information, see Adding Schema Extensions on page 541.](#)

Related Topics

- [Working with Change Labels on page 69](#)

Adding Schema Extensions

In this step, you copy all the schema extensions, such as new tables, columns or indexes into the database.

To add schema extensions in the Schema Extension

1. Changes to the schema are displayed on the page **System modifications**. This displays all the Data Definition Language (DDL) statements that are used by the extensions.

- a. If you want to save statements in a file

- Set the option **Attach statements to existing file** to add the statements to an existing file.
 - Select **Save to file** in the repository path and enter a file name. The statements are saved as an SQL file.
 - To create the file, click **Save**.
 - Click **Next**.
 - Confirm the security prompt with **Yes**.

- b. If you do not want to save the extensions in a file

Confirm the security prompt with Yes

- Click **Next**.
 - .

The schema extensions are added to the database and the necessary extensions are made to the One Identity Manager system data model. This make take some time.

2. Start compilation on the **Compiling** page. This make take some time. Click **Next** after compilation is complete.
3. New calculation tasks are queued in the DBQueue Processor due to the schema extension. The current DBQueue Processor calculation tasks are displayed on the **System queue** page. After the calculation tasks have finished processing, click **Next**.
4. On the last page, you return to the beginning of the wizard to enter more extensions or click **Finished** to end the program.

After completing the schema extensions, you can access them with One Identity Manager tools and make further changes.

Adding in Custom Extensions to the User Interface

After you have extended the schema with a custom table or column, other steps are required to display the extensions in the Manager user interface.

1. Edit the table, column and table relation properties.

Properties include, for example, display names, descriptions, display patterns for tables and columns as well as templates, formatting, mandatory field definitions. You already specify some of the properties when you extend the schema with the program "Schema Extension". Use the Designer Schema Editor to make more changes to the tables and columns.

For more information, see [Mapping Table Definitions on page 172](#) and [Mapping Column Definitions on page 188](#).

2. Grant editing permissions

You already grant permissions for permissions groups when you extend the schema with the program Schema Extension. You can carry on editing permissions in the Designer Permissions Editor and also create permissions groups with the User & Permissions Group Editor.

For more information, see [Granting One Identity Manager Schema Permissions on page 215](#) and [Editing Permissions Groups and System Users on page 219](#).

3. Creating object definitions

The data in the user interface is represented by objects. A generally valid object definition without limited selection criteria has already been created with the program "Schema Extension". You can create other object definition constraints in addition. You create object definitions in the Designer.

For more information, see [Object definitions for the User Interface on page 247](#).

4. Editing navigation

Extend the navigation in order to display data in the Manager. Use Designer's User Interface Editor to create menu items for navigation and result lists.

For more information, see [User Interface Navigation on page 251](#).

5. Creating the forms for the user interface

Create or extend forms for displaying and editing in the Manager.

- To edit custom table, create an interface form with the form definition "VI_Generic_MasterData" using Form Editor in the Designer.
- Default forms can be used to customize column extensions on default tables under certain conditions.
- Create the corresponding overview form with the Overview Form Editor in the Designer.

- One Identity Manager provides a set of form templates and definitions in the default installation. These can be used for easily creating your own forms.

For more information, see [Editing Interface Forms on page 279](#), [Forms for Custom Extensions on page 287](#) and [Working with Overview Forms on page 297](#).

6. Creating method definitions

If you want to provide specific tasks in the Manager, you can create method definitions in the Designer.

For more information, see [Task Definitions for the User Interface on page 322](#).

7. Creating analyzes

Create statistic definitions and reports to analyze data and add these into the user interface.

For more information, see [Statistics in the One Identity Manager on page 306](#) and [Reports in the One Identity Manager on page 494](#).

8. Localizing text

Use the Language Editor in the Designer to translate text for multilingual captions in the Manager, for example, column names, comments, menu items, form names.

For more information, see [Language Dependent Data Representation on page 328](#).

Transporting One Identity Manager Schema Customizations

Automatic version control is integrated into the One Identity Manager, ensuring that One Identity Manager components are always consistent with each other and with the database. If program extensions that change the structure are implemented, for example, table extensions, the database needs to be updated.

You need to update the database if hotfixes and service packs for your installed version of the One Identity Manager are available or complete version updates. In addition, you are repeatedly required to transfer custom changes from a development database into the live database.

The One Identity Manager schema is customized by loading so-called 'transport packages'. One Identity Manager recognizes the following types of transport packages that can be copied to the database depending on requirements.

Table 317: Transport Package

Transport Package Type	Description	Tool Used
Migration package	Migration packages are provided by for the initial database schema installation, for service pack and complete version updates. A migration package contains all the necessary tables, data types, database procedures and the default One Identity Manager configuration.	Configuration Wizard
Hotfix package	Hotfix packages are provided to load individual corrections to the default configuration such as templates, scripts, processes or files into the database. ① NOTE: If a hotfix only contains modified files, load these files into the database using the "Software Loader" program.	Database Transporter Software Loader
Custom configuration	A custom configuration package is used to exchange customer specific changes between the development, test	Database Transporter

Transport Package Type	Description	Tool Used
package	and productive system database. This transport package is created by the customer and loaded into the database.	

NOTE: If more custom configuration adjustments are made to a One Identity Manager database, then create a custom configuration package and import this transport package in the target database with the Database Transporter. There is no support for merging a hotfix package with a custom configuration package into one transport package.

Detailed information about this topic

- [Basics for Transport of One Identity Manager Schema Modifications on page 545](#)
- [Creating a Transport Package with the Database Transporter on page 546](#)
- [Displaying Contents of a Transport Package on page 555](#)
- [Importing a Transport Package with the Database Transporter on page 556](#)
- [Displaying Transport History on page 558](#)

Related Topics

- [Files for Software Update on page 151](#)
- [One Identity Manager Installation Guide](#)

Basics for Transport of One Identity Manager Schema Modifications

Prerequisite for transporting modifications between One Identity Manager databases:

- Source and destination database have the same underlying database system.

Different methods are implemented for transporting modifications.

- Transport of single objects is done through the object layer. This means, permissions, templates and Customizer are taken into account when a transport package is imported.

This method is implemented, for example, if you create and import custom configuration packages with the program "Database Transporter" which contain modifications to system users, modifications from a specific date or single objects.

- The transport of the entire system configuration is done through a transfer buffer. All relevant tables are checked when creating the transport package. The condition

applied to the table, defines which objects are transported. The primary key is used to establish whether the transport entry has a GUID module and whether it is transferred to the source database transfer buffer. The transfer buffer is read and transport package is created. When importing into the target database, the contents of the transport package is transferred to the target database's transfer buffer. The information is then transferred to the target tables.

This method is used if custom configuration packages, which contain the entire system configuration, are created and imported with the program "Database Transporter". This method is also used when installing and updating the One Identity Manager schema with the Configuration Wizard.

When a transport package is imported into a One Identity Manager database the following operations are carried out:

- Paste

No object was found in the destination database using the primary key or alternative key, therefore a new object is created with this key value.

- Refresh

If an object is found in the destination database using the primary key or an alternative key, this object is updated. The update is done using the configuration buffer.

If transporting modifies a default configuration, the default configuration is moved into the configuration buffer. You can retrieve changes from the configuration buffer and restore the default configuration in this way.

If, during a One Identity Manager version upgrade, the default configuration is changed by a service pack, a complete version upgrade or by loading a hotfix package, a check is made to see if it has already been customized. In this case, the modified default configuration is copied to the configuration buffer. This ensures that customizations do not go missing.

- Delete

Objects that are no longer needed are deleted. This operation is always executed if the entire system configuration is transported.

Related Topics

- [Creating a Transport Package with the Database Transporter on page 546](#)
- [Importing a Transport Package with the Database Transporter on page 556](#)

Creating a Transport Package with the Database Transporter

Create a custom configuration package, which you import into the target database, in order to exchange customizations between the development database, test database and the

production database.

You can specify restricting export criterion for creating custom configuration packages. System user modifications, modifications as from a defined date, or individual objects can be exported. A limited custom configuration package is recommended for transporting individual changes from a development database to a test database. However, you should create a transport of the entire system configuration package to transfer changes from the test database to the productive database.

- NOTE:** The current user must be entitled to use the program function "Allows transport packages to be imported into the database." in order to import a transport package with the Database Transporter. (Transport_Import).

To create a transport package

1. Start the Launchpad and select **Transport custom modifications**. This starts the program "Database Transporter".
2. Select **Create a transport file** on the start page.
3. Enter the connection credentials for the One Identity Manager database on the **Select the database connection** page.
4. Enter the information about the transport file on the **Define file name** page.
 - a. Enter the name of the transport file and change the output directory as required.
 - b. To create an export log file, set the option **Create a log file for data export**.The log file is saved in the output directory of the transport file.
5. Enter a description of the transport data on the **Show and define transport parameters** page.
6. Select the export criteria on the **Define transport data** page.

- NOTE:** You can combine several export criteria.

Table 318: Export Criteria

Export Criteria	Description
Run SQL statements before data import	You can integrate SQL statements into the custom configuration package, which are executed before the data is imported.
Transport of favorite objects	All modified processes, scripts, report and mail templates from a specific time period are available for preselection.
Transport by change label	Transport modifications to objects or object properties, which are grouped under a change label.
Transport by change	Limit transportation data by user, time period and

Export Criteria	Description
information	database tables.
Transporting Schema Extensions	Transport custom schema extensions, like tables, columns, database procedures, functions, triggers, views and indexes.
"Transport selected objects and their dependencies	Select single objects and their dependencies for transporting.
System Configuration Transport	Transport the complete system configuration.
Transports system files	Transport single files. These files are distributed with the automatic software update.
Run SQL statements after data import	You can integrate SQL statements into the custom configuration package, which are executed after the data is imported.

7. Click **Next** to start exporting.

The program determines the data to export and displays the progress of the export in the dialog box. The export procedure can take some time.

8. Click **Finish** on the last page to end the program.

The export date, the export description, database revision and the name of the export file in the source database transport history are recorded when a transport package is created with the Database Transporter.

Detailed information about this topic

- [Transporting SQL Statements on page 549](#)
- [Transporting Favorite Objects on page 549](#)
- [Transport by Change Label on page 550](#)
- [Transport by Change Information on page 551](#)
- [Transporting Selected Objects and their Dependencies on page 553](#)
- [Transporting Schema Extensions on page 552](#)
- [Transporting System Configuration on page 554](#)
- [Transporting System Files on page 555](#)
- [Transporting SQL Statements on page 549](#)

Related Topics

- [Displaying Contents of a Transport Package on page 555](#)
- [Importing a Transport Package with the Database Transporter on page 556](#)

- [Displaying Transport History on page 558](#)

Transporting SQL Statements

You have the option to integrate SQL statements into the custom configuration package. These SQL statements are run either before or after the data is imported. For example, after a schema extension has been transported an SQL statement may be required for filling initial data in the new columns.

NOTE: To create transport packages, the current user must have permissions for the program function "Enables integration of SQL statements in a transport file. (Transport_SQL)".

To run SQL statements within a transport package

1. Select the export criteria for running SQL statements. The following export criteria are available:
 - Run SQL statements before data import
 - Run SQL statements after data import
2. Create the SQL statement using the **Edit** button. Differentiate between SQL statements for system data transport and user data transport.
 - Enter the SQL statements directly.
 - OR -
 - Load a *.sql file with the  button.
 - Save them with the  button.

Related Topics

- [Creating a Transport Package with the Database Transporter on page 546](#)

Transporting Favorite Objects

This method of transport shows modified processes, scripts, reports and mail templates during a specific time period.

To transport favorite objects

1. Select the export criteria **Transport of favorite objects**.
2. Use the **Select** button to open a selection dialog box where you select the objects you want to transport.
3. Enter the time period for the object selection in the input field **Object changed in last ... days**. All the objects are displayed with the change date and user.

If you also want to add processes, scripts, reports or mail templates to the other objects in the display, use the "Load all.." option to load them.

4. Select the objects you want and add them to the transport. All the objects you have selected and their dependencies are listed under "Transport objects".

TIP: You can select multiple objects with the **SHIFT+ CLICK** or **CRTL + CLICK**.

Table 319: Functionality of the Object Selection Icons

Icon	Meaning
→	Selected objects and dependencies are added to the transport.
←	Selected objects and dependencies are removed from the transport.

Related Topics

- [Creating a Transport Package with the Database Transporter on page 546](#)

Transport by Change Label

Several changes to objects or objects properties are grouped together under a change label and can be swapped between source database and target database in this way. When a custom configuration package is imported with change labels, new data records are added to the target database and existing data records are updated. In addition, data records that are marked for deletion in the change labels are deleted from the database.

TIP: Use a change label to mark changes to single objects properties as well as entire objects.

Note the following:

- If an object property is modified through a transport, you must ensure that the object to modify is present in the target database. In certain circumstances, the object must be added to the change label.
- If the entire object is labeled with a change label, the object is added to the custom transport package with the object properties valid at the time of export.

TIP: There are no change labels available after initial schema installation.

To transport by change label

1. Select the export criteria **Transport by change label**.
2. Select a change label in the list.

TIP: You can group together several change labels in one custom configuration package. To add other change labels, use the  button.

3. To display the contents of a change label, click **Show**.
Objects and changes are displayed, which belong to the change label.
 - If there are still references in the change label to objects that do not exist anymore in the database, remove the assignment with **Repair**.
4. Click **Options** for additional setting for transporting changes labels.

Table 320: Additional Transport Settings

Option	Description
Close change label after export	This closes the change label after the transport has completed. No more changes can be booked to this change label.
Add dependent objects to transport file	Objects that are dependent on the selected objects and are not marked with a change label are also added to the transport.
Show with closed change labels	Change labels, which are already closed, are also available for selection.

Related Topics

- [Creating a Transport Package with the Database Transporter on page 546](#)
- [Working with Change Labels on page 69](#)

Transport by Change Information

Use transport by change information to limit transportation data by user, time period and database tables.

To transport by change information

1. Select the export criteria **Transport by change information**.
2. Specify which changes you want to transport.

Table 321: User Selection

Entry	Description
me	Only your changes are added.
all users	Changes are added from all users.
selected users	Changes are added from selected users. <ul style="list-style-type: none"> • A selection list of system users is displayed from which you can

Entry	Description
	<p>make a selection in the case of specified users.</p> <ul style="list-style-type: none"> You can also specify additional users directly in an input field or by using a selection dialog, which you open with
	<p>TIP: You can select multiple objects with the SHIFT+ SELECT or CRTL + SELECT.</p>

3. Use the date filter to export changes for the selected user(s) from a specified date.

To specify the date, you can select "today", "yesterday", "day before yesterday", "this week" and "Last database migration" from the quick select menu. However, you can also choose another time period for the transport. To do this, select "period" and enter the time period in the "From:" and "To:" positions.

4. You can limit transportation data even further by selecting database tables.

Table 322: Table Selection

Entry	Description
whole system	Changes are added from all tables.
system data	Changes to system components in the tables are added.
user data	Changes to user data components in the tables are added.
selected tables	Changes are added from specific tables.

5. To display objects that fulfill the specified export criteria, click **Show**.

TIP: You can exclude individual objects from the transport at this point. To do this, disable the corresponding objects.

Related Topics

- [Creating a Transport Package with the Database Transporter on page 546](#)

Transporting Schema Extensions

Custom schema extensions, like tables, columns, database procedures, functions, triggers, views and indexes that you want to add, must be distinguished by a custom prefix. This customer prefix must be given for the main database. Furthermore, only custom database procedures, functions, triggers, views and indexes that are not encoded and are smaller than 64 kb are included. Custom database procedures, functions, triggers and views are always exported in their entirety. Entries corresponding to custom tables and columns are generated in the One Identity Manager schema when the transport package is imported (tables DialogTable, DialogColumn, QBMRelation).

The following procedure is recommended to transport all schema extensions in their entirety, from a test database into a production database

1. Create a transport of schema extensions in the test database and import these into the production database.
2. Create a transport of the system configuration in the test database and import these into the production database.

Use the transport options to transport single customizations by change label, change information or selected objects .

to transport schema extensions

- Select the export criteria **Transport schema extensions**.

 **NOTE:** Use the **Show** button to display the schema extensions.

Related Topics

- [Creating a Transport Package with the Database Transporter on page 546](#)
- [Managing Custom Database Objects within a Database on page 159](#)

Transporting Selected Objects and their Dependencies

Select single objects and their dependencies for transport. You can add objects dependent on the object you want to transport without having to select them individually.

To transport single objects and their dependencies

1. Select the export criteria **Transport selected objects and their dependencies**.
2. Use the **Select** button to open a selection dialog box where you select the objects you want to transport.
3. To select the objects, select the database table in the "Tables" list that you want to take the objects from that will be added to the custom configuration package.

 **NOTE:** All tables that are not labeled with the option **No DB transport** are in the list. If object from other tables should also be transported, disable this option in the program "Designer".

4. ChildRelation (CR), ForeignKey (FK) and many-to-many relations for the selected database table are displayed in "Relations".
 - Enable the relations you want. The objects linked through these relations are added to the transport when an object is selected.
5. All the objects in the selected table are shown in the "Object" list. You can limit the selection with a search filter. Select the objects you want and add them to the

transport.

TIP: You can select multiple objects with the **SHIFT+ CLICK** or **CRTL + CLICK**.

Table 323: Functionality of the Object Selection Icons

Icon	Meaning
	Selected objects and dependencies are added to the transport. There is no post-processing after data import.
	Selected objects and dependencies are added to the transport. Redundant objects are deleted after data import.
	You can define a search filter. The search filter displays all matching objects.

6. All the objects you have selected and their dependencies are listed under "Transport objects".

TIP: To remove individual object from the transport, select **Remove**.

Related Topics

- [Creating a Transport Package with the Database Transporter on page 546](#)
- [Mapping Table Definitions on page 172](#)

Transporting System Configuration

NOTE: You should only use a full custom configuration package to transfer all customizations from the test database into an initial production database. Use change labels to change information or selected objects to transport configuration data into an existing live database.

CAUTION: This procedure overwrites the entire configuration data in the target database! This also applies to the configuration parameter settings. You can prevent individual properties from being overwritten.

To transport the system configuration

- Select the export criteria **Transport system configuration**.

The transport of the entire system configuration is done through a transfer buffer. All relevant tables are checked when creating the transport package. The condition applied to the table, defines which objects are transported. The primary key is used to establish whether the transport entry has a GUID module and whether it is transferred to the source database transfer buffer. The transfer buffer is read and transport package is created.

When importing into the target database, the contents of the transport package is transferred to the target database's transfer buffer. The information is then transferred to the target tables. Creating and importing this transport is very time consuming.

When a full custom configuration package is imported, new objects are added to the destination database and existing objects are updated. In addition, redundant objects are deleted from the target database.

Related Topics

- [Creating a Transport Package with the Database Transporter on page 546](#)
- [Advice on Importing the Entire System Configuration on page 557](#)

Transporting System Files

Use this export criteria to transport single files. These files are distributed with the automatic software update.

To transport new or modified One Identity Manager files

1. Select the export criteria **Transport system configuration**.
2. Click **Select** and specify the files to transport.

Related Topics

- [Creating a Transport Package with the Database Transporter on page 546](#)

Displaying Contents of a Transport Package

You can display the contents of a transport package with the Database Transporter before you import.

NOTE: To import transport packages with the Database Transporter, the current user must be authorized to use the program function "Allows transport packages to be imported into the database (Transport_Import)".

To display the contents of a transport package

1. Start the Launchpad and select **Transport custom modifications**. This starts the program "Database Transporter".
2. Select **Show transport file** on the start page.
3. Select the transport package file browser and click **Open**.

4. Click **Next** on the **Select transport file** page.
5. The contents of the transport file are displayed on the **Show transport file**.
 - To display the order for importing objects
 - a. Use + to open an entry in the transport file and select **Sort in import order** from the context menu.
 - b. Click **OK** and enter the database connection data. This step is only required when you established the first in the order.
The order in which the entry's objects are imported into the database is found.
 - c. Repeat this step for all other entries for which you want to determine the import order.
6. Click **Finish** on the last page to end the program.

Related Topics

- [Importing a Transport Package with the Database Transporter on page 556](#)

Importing a Transport Package with the Database Transporter

Before importing a transport package, you can protect individual properties from being overwritten in the target database. For example, you might want to block processing:

- Configuration parameters and their values should not be overwritten when a test environment is transported to a productive system.
- Server configurations should neither be overwritten in the test environment nor the productive system during a transport.
- Passwords of administrative user accounts should neither be overwritten in the test environment nor the productive system during a transport.

To lock single properties for editing

1. Open the object you want to lock, in the Designer or Manager.
2. Click on the property name and select the context menu **Prohibit modification**.
The input field is locked and grayed-out.
3. To unlock the property again, click the property's name and select **Permit modification**.

NOTE: To import transport packages with the Database Transporter, the current user must be authorized to use the program function "Allows transport packages to be imported into the database (Transport_Import)".

To import a transport package

1. Start the Launchpad and select **Transport custom modifications**. This starts the program "Database Transporter".
2. Select **Import transport file** on the start page.
3. Enter the connection credentials for the One Identity Manager database, into which you want to import the transport package, on the **Select the database connection** page.
4. Select the transport package file browser and click **Open**.
5. Specify your import options on the **Select transport file**.
 - a. To create an import log file, set the option **Create import log file**.
The log file is saved in the output directory of the transport file.
 - b. To import individual objects, set the option **Import objects singly and ignore errors**.
Errors, which might occur during importing are ignored and displayed when importing is complete. If this option is not set, the import is stopped when an error occurs.
6. Import steps and import progress are displayed on the **Importing transport data** page. The import procedure can take some time. Calculation tasks are queued for the DBQueue Processor on termination.
7. If changes have been made to the system configuration, for example, processes or scripts imported, you have to compile the database after the tasks have been processed. Compilation is started automatically once importing is complete.
8. Click **Finish** on the last page to end the program.

 **NOTE:** Use the  button to save errors that occur whilst importing.

When you import a transport package with the Database Transporter, the import date and description, the database version and the transport package name are recorded in the transport history of the target database.

Related Topics

- [Advice on Importing the Entire System Configuration on page 557](#)

Advice on Importing the Entire System Configuration

 **IMPORTANT:** Take note of the following advice when you import a transport package into a target database that was created with the export criteria **Transporting system configuration**.

⚠ CAUTION: This procedure overwrites the entire configuration data in the target database! This also applies to the configuration parameter settings. You can prevent individual properties from being overwritten.

Before importing a transport package.

- Prevent individual properties in the target database from being overwritten.
 - a. Open the object you want to lock, in the Designer or Manager.
 - b. Click on the property name and select the context menu **Prohibit modification**.
The input field is locked and grayed-out.

After importing a transport package.

- Check the staging level of the target database.
 - a. Select the category **Base Data | General | Databases** in the Designer.
 - b. Select the database and check the value of the property **Staging level**.
- Check the configuration settings in the target database. Check at least the configuration settings for the DBQueue Processor. The settings are specified through the database staging level and configuration parameters.
- Unlock the properties for editing again.
 - a. Open the object you want to lock, in the Designer or Manager.
 - b. Click on the property name and select the context menu **Permit modification**.

Related Topics

- [Configuring a One Identity Manager Database for Testing, Development or Production on page 116](#)
- [Configuring the DBQueue Processor for Test and Development on page 645](#)
- [Importing a Transport Package with the Database Transporter on page 556](#)
- [Transporting System Configuration on page 554](#)

Displaying Transport History

The export date, the export description, database revision and the name of the export file in the source database transport history are recorded when a transport package is created with the Database Transporter.

When you import a transport package with the Database Transporter, the import date and description, the database version and the transport package name are recorded in the transport history of the target database.

To display transport history

- Start the Designer and select the menu item **Help | Transport history**.

Importing Data

Data Import has a program called "One Identity Manager" which provides a simple solution for importing data from other systems. The program supports importing from files and importing directly from other database systems. You can import data immediately. You also have the option to import data from customized processes using the import scripts that are created. The import definition is saved so that you can use it for future data imports.

The steps in the program are as follows:

1. Load export definitions
 2. Select the import method
- Two methods are supplied, "Import CSV file" and "Import from database".
3. Configure the import
 4. Create an import definition
 5. Create an import script
 6. Start the import

Detailed information about this topic

- [Importing Data from CSV Files on page 560](#)
- [Importing Data from a Database on page 565](#)
- [Import Configuration on page 568](#)
- [Using Import Definition Files on page 573](#)
- [Importing Data on page 573](#)

Importing Data from CSV Files

The data structure of the import file needs to fulfill the following requirements:

- The data is separated by a delimiter.
- The data records are separated by a new line.
- Data that contains a new line is marked with a text qualifier.
- The data in the import files is already sorted accordingly to resolve object dependencies in larger CSV imports.

NOTE You can use the Data Import sort options for csv imports with small data sets.

To import data from CSV files into the One Identity Manager database

1. Start the Data Import and log in to the program.
2. Load the import definition file, if available.
- NOTE:** Leave this field empty if you want to create a new import definition.
3. Select the import method "Import CSV files".
4. Load the import file and enter any additional data.
5. Specify how the file is structured.
6. Specify how the row structure is set up.
7. Specify a condition for the rows to import.
8. Configure the import.
 - a. Assign the data to target tables and columns in the One Identity Manager database.
 - b. Specify the key columns.
 - c. Specify the data hierarchy for the import.
 - d. Specify options for handling the data.
 - e. Define the variables to be set during the import.
9. Save the import definition file and the import script.
10. Start the import.
11. End the program or start another import.

Detailed information about this topic

- [Loading Import Files on page 562](#)
- [Specifying the File Structure on page 562](#)
- [Specifying a Line Condition on page 565](#)
- [Assigning Target Tables and Columns on page 568](#)
- [Specifying the Hierarchy on page 570](#)
- [Options for Handling Records on page 571](#)
- [Specifying Connection Variables on page 572](#)

- [Importing Data on page 573](#)
- [Using Import Definition Files on page 573](#)

Loading Import Files

Enter the following data to load an import file:

- Import file

Enter the path for the file that contains the import data. Use the ... button next to text box to browse for the file and open it.

- File encoding

CSV file character encoding. Encoding of the character set is determined from the character set on your workstation when the import file is loaded. Change the setting if the file was created with another character set.

- File culture

File language for the file. The language is required in order to read local character formats correctly, for example, dates.

- Time zones

If the date and time data is imported, select the time zone for the data. The time zone is required for converting the data to UTC.

Specifying the File Structure

Specify how the file is structured. Enter the following data:

- Number of lines in header

Enter the number of lines in the CSV file header. The header is not imported.

- Columns identified by

- Select the option **Delimiter** if the data is separated by a semi-colon, comma, space, tab, pipe or other character. Specify the line structure.

- Select the option **Fixed width** if all the data in the columns has the same length. Specify the line structure.

Detailed information about this topic

- [Specifying Line Format for Data with Delimiters on page 563](#)
- [Specifying Line Format for Data with Fixed Width on page 564](#)

Specifying Line Format for Data with Delimiters

If you selected the option **Columns identified by delimiter** then you need to specify the following settings in this step:

- Delimiter

Delimiter used to separate the data in the file. You can choose between "semi-colon", "comma", "space", "tab" and "pipe".

If the data is separated by another character select "other" and enter the delimiter in the field next to the list.

- Text qualifier

Character enclosing the column text. This text is treated as one value on import, even if the text contains the delimiter given as above.

 **NOTE:** The delimiters are masked by doubling them up.

Example:

Delimiter:	Comma (,)
Text qualifier:	Quotation mark ("")
Value in file:	"Smith,Bill"
Value after import:	Smith,Bill

Delimiter:	Comma (,)
Text qualifier:	Not given or other character:
Value in file:	"Smith,Bill"
1st value after import:	"Smith
2nd value after import:	Bill"

- Mask delimiter by doubling

Specifies whether the data is separated by several of the same delimiters. Data that contains a new line must be marked with a text qualifier.

Example:

Delimiter:	Comma (,)
Mask delimiter by doubling:	Enabled
Value in file:	Smith,,Bill
Value after import	Smith,Bill

Delimiter:	Comma (,)
Mask delimiter by doubling:	Not set
Value in file:	Smith,,Bill
1st value after import:	Smith
2nd value after import:	
3rd value after import:	Bill

- Multiple values in / delimited by

Specifies whether the import contains a multivalued property column (MVP) and the column should not be imported directly. Individual values are entries in another table and should be linked through a many-to-many table.

Enter the affected column using the menu. In **Delimited by:** enter the values' delimiter. The column values are split up. A new line is generated for each value although the rest of the columns remain the same.

Example:

The line

Hans;Müller;Org1|Org2|Org3

becomes the import source

Hans;Müller;Org1

Hans;Müller;Org2

Hans;Müller;Org3

Related Topics

- [Specifying the File Structure on page 562](#)
- [Specifying Line Format for Data with Fixed Width on page 564](#)

Specifying Line Format for Data with Fixed Width

If you have selected the option **Columns identified by fixed width**, define the width of the columns in this step.

- Click on the ruler in the Data Import preview to set the separation points. A separation mark is inserted.
- When you click again on a fixed separation point, the separation mark is deleted.

Related Topics

- [Specifying the File Structure on page 562](#)
- [Specifying Line Format for Data with Delimiters on page 563](#)

Specifying a Line Condition

To exclude specific data records from being imported, you can enter a condition. Format the condition in VB.Net syntax. The columns are accessed with dollar notation.

Access using Column Indexing (0...n)

Example:

Do not import a data record if the first column contain the values "OLD".

```
Value = $0$<>"OLD"
```

Access using Column Identifier

If a header is defined, you can use the column identifier for access.

Example:

Import a data record if the column with the identifier "NewData" contains the value "True".

```
Value = $NewData:Bool$
```

Related Topics

- [Using Dollar \(\\$\) Notation on page 455](#)

Importing Data from a Database

To import data from an external database into the One Identity Manager database

1. Start the Data Import and log in to the program.
2. Load the import definition file, if available.

 **NOTE:** Leave this field empty if you want to create a new import definition.

3. Select the import method "Import from database" for this.
4. Specify the connection data to the external database.
5. Formulate the source data query.

6. Configure the import.
 - a. Assign the data to target tables and columns in the One Identity Manager database.
 - b. Specify the key columns.
 - c. Specify the data hierarchy for the import.
 - d. Specify options for handling the data.
 - e. Define the variables to be set during the import.
7. Save the import definition file and the import script.
8. Start the import.
9. End the program or start another import.

Detailed information about this topic

- [Selecting an External Database on page 566](#)
- [Source Data Query on page 567](#)
- [Assigning Target Tables and Columns on page 568](#)
- [Specifying the Hierarchy on page 570](#)
- [Options for Handling Records on page 571](#)
- [Specifying Connection Variables on page 572](#)
- [Importing Data on page 573](#)
- [Using Import Definition Files on page 573](#)

Selecting an External Database

To set up a connection with an external database

1. Select the external database provider in the section "Connection type".
A list of the various database providers available is shown.

Supported Data Providers

Odbc Data Provider

OleDb Data Provider

OracleClient Data Provider

SQLClient Data Provider

dotConnector for Oracle

Microsoft SQL Server Compact Data Provider

When you use another database provider, select it using the ... button next to the input field. Use the  button to remove a provider entry.

2. Enter the connection data for the external database in the section "Connection data". Select ... to this and enter the connection data. Use the  button to encrypt the connection data.
Refer to the documentation of the database provider implemented, for the connection parameters.
3. Click **Check** to test the database connection.
If the database can be successfully connected, you can continue.
4. If the date and time data is imported, select the time zone for the data. The time zone is required for converting the data to UTC.

Source Data Query

Formulate the source data query either by selecting the table and columns or with an SQL statement.

Selecting Table and Columns

Select the table and columns with contents to transfer.

- Table

Select the table from the menu.

- Columns

Enter the column name directly into the input field or open a dialog window to select the column with the ... button.

- WHERE clause

Use a condition to limit the data to import. Format the condition as a valid where clause for a database query.

- Order by

The sort order is required if the data records have to be transferred in a defined sequence, for example, as in hierarchical structures. Format the sort order as a valid order by statement for a database query.

SQL statement

Formulate the data set selection as database query in SQL syntax.

Import Configuration

An import configuration includes the following steps:

1. Assigning the data to target tables and columns in the One Identity Manager database.
2. Specifying the data hierarchy for the import.
3. Specifying options for handling the data.
4. Defining variables that are set on import.

Detailed information about this topic

- [Assigning Target Tables and Columns on page 568](#)
- [Specifying the Hierarchy on page 570](#)
- [Options for Handling Records on page 571](#)
- [Specifying Connection Variables on page 572](#)

Assigning Target Tables and Columns

Specify how the data is stored in the One Identity Manager database.

- Target table

Select the target table into which to import the data.

- Target columns and key

Columns and their values are displayed.

- Select the target column of the target table. If there is no target column assigned you will see "not assigned" displayed in the column header.
- Specify which columns are used as key columns. Key columns are labeled with a key icon in front of the column ID.

TIP: Use the  button in the section "Target table" to automatically sort the target columns and keys. Assigns a column if one is found in the target table whose name matches the name in the source column. It is strongly recommended that you check the assignment afterward.

TIP: Use the assignment wizard to select the target system columns to insert the data into.

Related Topics

- [Assigning Target Columns with the Assignment Wizard on page 569](#)

Assigning Target Columns with the Assignment Wizard

To assign target columns and keys with the wizard

1. Open the wizard in the section "Target columns and keys" using the arrow button in the column header of any column.

Table 324: Functions in Assignment Wizard

Icon	Meaning
<	Moves to previous column.
>	Moves to next column.
⚡	Automatic assignment of target column. Assigns a column if one is found in the target table whose name matches the name in the source column. It is strongly recommended that you check the assignment afterward.
+	Inserts a column. A column is inserted with a fixed value.
-	Deletes a column. Deletes a column with a fixed value.

2. Enter the following data for each column:

- TIP:** Use the option **Show caption** to swap between the display name and the technical column name.
- TIP:** You can see a preview of the values in the section "Data preview".

Table 325: Target Column and Key Properties

Property	Description
Use as key column	Specifies whether the column is used as a key column. More than one key columns can be defined. The data records to import into the database are determined based on key columns. Data records should be uniquely identified with these key columns.
Conversion script	Use the conversion script to modify source column values to match the permitted value of the target column. This is required, for example, if a list of permitted values is defined for the target columns. Write the conversion script in VB.Net syntax. You access the values with the variable <code>value</code> . Use dollar notation to access the source columns.
Target columns	Select the target columns to be imported into the data. All columns from the target table are displayed with their data type. Following

Property	Description
	<p>applies:</p> <ul style="list-style-type: none"> Compulsory data is labeled with a blue triangle in front of the data type. Columns without sufficient permissions are displayed in gray. Columns, deactivated by preprocessor condition, are not shown. <p>TIP: If a column is found in the target table whose name matches the name in the source column, the  button suggests another column. You should always verify this suggestion!</p>

- After completing your assignments, close the wizard by click on a point outside the wizard.

Inserting Columns with Fixed Values

You can also add additional columns with fixed values to the import data and import it to a defined column.

To insert columns with fixed values

- Start the assignment wizard.
- Add a new column using the  button in the assignment wizard.
- Enter the value you want in **Fixed value**.
- OR -
Enter a conversion script if the value should be taken from the source column.
- Assign the target column.

Related Topics

- [Using Dollar \(\\$\) Notation on page 455](#)

Specifying the Hierarchy

If an import contains inter-dependent data, you must ensure that the reference targets are handled before the reference sources.

For example, child departments (`Department.UID_Department`) are imported after parent departments (`Department.UID_ParentDepartment`).

NOTE: Sorting data in a hierarchical structure can take up a lot of memory. Therefore, only use this procedure for imports with small amounts of data.

To sort data hierarchically

1. Set the option **Sort by hierarchy**.
2. Select the **Key columns** in which to map the data (for example, Department.UID_Dept) and **Parent key column** (for example, Department.UID_ParentDepartment)

NOTE: Sort the data in the import files accordingly beforehand, to resolve object dependencies in larger CSV imports. For larger imports from external data sources, use the "ORDER BY" clause to sort the data.

Related Topics

- [Source Data Query on page 567](#)

Options for Handling Records

Specify how new and existing data sets are handled when they are imported. During import, the source data sets (case C) are compared with the entries in the database (case A).

You can restrict the number of relevant database entries by using the **Condition for target objects** (case B). The condition is tested when importing begins. There is a wizard available through the  button next to the input field, to help you formulate your condition.

The import needs to take several cases into account and react accordingly to each one:

- The data set from the source data does not exist in the database (case E) yet. The entry will be added if the option **Insert new data set** is set.

IMPORTANT: If the option **Insert new data set** is set, the entries that are in the source data but are not in the range of relevant database entries due to the **Condition for target objects**, are treated as a new data set and are inserted into the database (case G). This can lead to errors (double data sets).
- There is an entry in the database that corresponds to the source data set (case D). If the option **Update existing data set** is set, the entry in the database is updated.
- There are several entries in the database that correspond to the source data set. An error is logged.
- There is an entry in the database that does not exist in the source database (case F). If the option **Delete sentences that no longer exist** is set, the entry is deleted from the database.

Figure 57: Example for Handling Data Sets

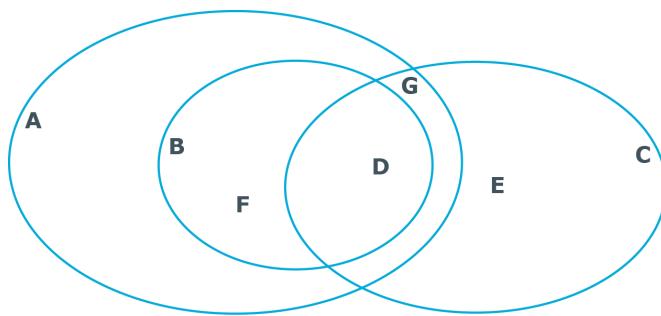


Table 326: Key for Handling Data Sets

Case	Description
A	All objects in the database.
B	Database set restricted by condition.
C	Entry in source data.
D	All entries in the database and in the source data. Typical action: update all entries in the database.
E	Entries that are only in the source data but not in the database. Typical action: add new entry in the database.
F	Entries that are in the database but not in the source data. Typical action: clean up entries in the database.
G	Entries that are in the source data but not in range selected in the database. These entries are treated as in case E although adding entries may cause conflicts in certain circumstances.

Specifying Connection Variables

Connection variables are set when the import is run immediately and are also added to the generated import script. You can use the variables in customized processes or templates that are executed after importing.

To define a connection variable

1. Click the button.
2. Click the "Name" entry and enter the variable name.
3. Click the "Value" entry and enter the variable value.

To delete a connection variable

- Click the button.

Importing Data

The following methods are available to you to import data:

- Start the data import manually in the Data Import. The data records that are processed during import are logged.
- To execute data imports on a regular basis, create an import script.

You can use the import script in custom processes, for example. Use the process task "DataImport" of the process component "ScriptComponent" to create custom processes for importing.

To start importing immediately

- Set the option **Import data** in the step "Save the Import Definition".
- Select the **Next** to start importing the data.

After importing has finished the processing result are displayed. If errors occur during the importing process you can view them with the **Show** button.

TIP: Save the import log with the context menu **Save log as file...**

To create an import script

- Set the option **Create import script** in the step "Saving the import definition".
- Enter a name for the import script in **Import script name**.
Only the VB name are permitted. If a character is not permitted, the text box is highlighted in red.
- Select a change label in **Add script to tag** menu. Use the ... button to create a new change label.
- Select the **Next** button so that the import script is created.
- Compile the script library after saving the script. Click **Yes** to start the compiler

NOTE: The import script is saved in the One Identity Manager database. You can only add import scripts to the database if they have permissions to use the program function "Import scripts can be added in the wizard for data import." (DataImport_CreateScript).

Using Import Definition Files

The import definition provides you with configuration settings for future data imports. You create the import definition file after setting up the import. You can load an existing import definition file in the Data Import when you set up an import.

To load an existing import definition

- Select the file containing the import definition in the "Load an import definition file" step using the ... button.
- Use the  button to remove the selected file again.

 **NOTE:** Leave this field empty if you want to create a new import definition.

To save an import definition

- Set the option **Save import definition file** in the step "Save the Import Definition".
- Select the file path with ..., enter the file name and click **Save**.

The import definition is saved as an XML file.

Web Service Integration

The One Identity Manager offers you the option to integrate web services. For example, you can use web services to write data to applications, which cannot be connected to One Identity Manager as a default target system.

Data for external applications can be originate from any of the One Identity Manager schema's tables. They can, for example, be mapped as custom target systems.

Example

The general data for a telephone system should be found from personnel data in One Identity Manager. The telephone system is mapped in One Identity Manager as a custom target system. One extension in the telephone corresponds a user account in One Identity Manager.

Once a new employee has been added in One Identity Manager, a new extension should become available in the telephone system. A new user account is added for each account definition. A web service passes the user account's master data onto the telephone system, where a new participant and telephone number is added. The web service passes this telephone number to the One Identity Manager as the return value. The telephone number should be transferred to the employee's master data.

Proceed as follows

1. Set up a custom target system in One Identity Manager.
 - Select the value "Synchronization by script" for the property "Synchronized by".
2. Set up the server for provisioning the data.
 - Enter the server as the synchronization server in the custom target system.
3. Set up an account definition for automatic administration of user accounts in this target system.
4. Enter the required IT operating data.
5. Bind the web service to One Identity Manager. Use the generic web service call for this.

The web service integration wizard helps you to create scripts for provisioning data for the default events Insert, Update and Delete. The provisioning processes are supplied automatically through One Identity Manager.

6. Create additional scripts and processes for handling the web service return value.

TIP: When you insert, change or delete containers, user accounts and groups in a custom target system, the return values are saved by default as GUID objects in the database.

Create a process to add the telephone in the object GUID to the employee's master data.

Detailed information about this topic

For more detailed information about setting up a custom target system, about account definitions, IT operating data and setting up a server, see the One Identity Manager Target System Base Module Administration Guide.

- [Generic Web Service Call on page 576](#)
- [Creating a Web Service Solution with the Web Service Integration Wizard](#)
- [Changing a Web Service Solution](#)
- [Handling Processes in the One Identity Manager](#)
- [One Identity Manager Scripts](#)

Binding a Web Service

Create a custom script for integrating a web service into One Identity Manager. There is a wizard available to assist you. The Web Service Integration Wizard finds all the methods used by the web service and creates scripts to call the required methods. The data from One Identity Manager is passed as parameters to the method. Which operations in the external application can be executed, is determined through the methods defined in the web service. The wizard created new entries in the tables DialogWebService and DialogScript.

The Web Service Integration Wizard supports different types of method calls. Each type supports the method call definition and, therefore, script creation to different degrees.

Generic Web Service Call

You use the generic web service call to publish data from a custom target system to an external application through a web service. The Web Service Integration Wizard queries all the required parameters and generates scripts from them.

Prerequisites

The external application data is mapped in One Identity Manager as a custom target system.

- A custom target system is set up (table UNSRootB). The property "Synchronized by" has the value "Synchronization by script".
- A server for provisioning data is set up and stored as synchronization server in the custom target system.

For more detailed information about setting up script controlled provisioning, see the One Identity Manager Target System Base Module Administration Guide.

Default Processes

One Identity Manager supplies default processes for provisioning data from custom target system to a web service.

To use these processes, the scripts you generated with the Web Service Integration Wizard must follow the naming convention: <Customer prefix>_<table>_<Ident_UNSRoot>_<event>.

IMPORTANT: If your target system contains a hyphen ("–") in its name, you must remove it from the script function in the part <Ident_UNSRoot>. Otherwise, error may occur during script processing.

Some of these processes handle the web service return values.

Table 327: Default processes for Synchronizing by Script

Object in custom target system (table)	Process	Saving the return value
Container (UNSContainerB)	VI_UnsContainer_Generic	UNSContainerB.ObjectGUID
User accounts (UNSAccountB)	VI_UnsAccountB_Generic	UNSAccountB.ObjectGUID
	VI_UnsAccountInGroup_Generic_Del	-
	VI_UnsAccountInGroup_Generic_Add	-
Groups (UNSGroupB)	VI_UnsGroup_Generic	UNSGroupB.ObjectGUID
	VI_UNSGroupBInUNSGroupB_Generic_Del	-
	VI_UnsGroupBInUNSGroupB_Generic_Add	-
Permissions controls	VI_UnsItem_Generic	-

Object in custom target system (table)	Process	Saving the return value
(UNSIItemB)	VI_UnsGroupHasItem_ Generic_Del	-
	VI_UnsGroupHasItem_ Generic_Add	-
	VI_UnsAccountHasItem_ Generic_Del	-
	VI_UnsAccountHasItem_ Generic_Add	-

Direct Web Service Call

The Web Service Integration Wizard finds all parameters that are defined in the method and from it, generates the script code. The parameters are passed in the function call. You can modify the parameters.

To run a script

- Create custom processes and pass the scripts and parameters to the process step.
[For more information, see Handling Processes in the One Identity Manager on page 384.](#)

Self-Defined Web Service Call

The Web Service Integration Wizard finds all the parameters, which are defined in the method. You define how the parameter is passed.

To run a script

- Create custom processes and pass the scripts and parameters to the process step.
[For more information, see Handling Processes in the One Identity Manager on page 384.](#)

Creating a Web Service Solution with the Web Service Integration Wizard

Prerequisite

- To bind to a web service with the service type "WCF", the file SvcUtil.exe must exist in the One Identity Manager installation directory.

Refer to Microsoft for information about where you can purchase this file.

- To bind a web service to the service type "SOAP", the file WSDL.exe must be on the server, which ran provisioning.

Refer to Microsoft for information about where you can purchase this file.

To integrate a new web service

1. Select the category **Base Data | General | Web services** in the Designer.
2. Select **Integrate new web service** from the task view.
This starts the Web Service Integration Wizard.
3. On the start page, click **Next..**
4. Enter the access data and general web service properties on the **Integrate new web service** page.

Table 328: General Web Service properties

Properties	Description
Web service name	Display name for the web service in One Identity Manager.
Description	Spare text box for additional explanation.
.Net namespace for the proxy code	Unique identifier for the .NET namespace.
Web service URL	URL under which the web service is run.
WSDL files URL	URL, under which the web service's WSDL.exe can be reached. If the WSDL.exe is not publicly available, it can also be saved locally.



NOTE: If the web service operator changes the WSDL file, run the Web Service Integration Wizard again in order to make use of the changes.

Properties	Description
Service type	Type of web service.
Locked	Specifies whether the web service can be used.
User name	User name for logging in to the web service.
User domain	User's domain.
User password and password confirmation	Password data for logging in to the web service.
Proxy code generator	<p>Path and file name for the proxy code generator.</p> <ul style="list-style-type: none"> • If the service type "WCF" is selected, path to file SvcUtil.exe. • If the service type "SOAP" is selected, path to file WSDL.exe.

Table 329: Web Services extended properties

Property	Description
Proxy server URL	URL of the proxy server, if communication is diverted over a proxy server.
Proxy server user name	User name for logging in to the proxy server.
Proxy server domain	Domain of the proxy server.
Proxy server password and password confirmation	Password and password confirmation for logging in to the proxy server.
Timeout for WSDL.exe	Timeout for accessing the WSDL file.
User defined command line	<p>Command line for calling the proxy code generator. The command line can be extended by another parameter if required.</p> <p>Default command:</p> <pre>/nologo /language:VB "/namespace:%Namespace%" "/out:{0}" %WsdlUrl%</pre> <p>Example:</p> <pre>/nologo /language:VB "/namespace:EnricoHolidayWebservice" "/out:{0}" http://kayaposoft.com/enrico/ws/v1.0/index.php?wsdl</pre>

- a. Click **Check...**
This tests access to the web service.
 - b. If the test is successful, click **Next**.
5. The generated proxy code is shown on the page, **Create proxy code**.
The proxy code contains all web service methods, which are defined in the WSDL file and makes them available to the One Identity Manager script components.
6. Select the web service methods you want to use in One Identity Manager on the **Select method calls** page. A script is generated for each of the selected methods in the next step.
 7. A script is generated to call the selected method on the **Generate web service call** page. Enter all the required parameter and properties for this.
 - Click to specify the type of method call.

Table 330: Method Call Type

Type	Description
Self-defined web service call	For more information, see Self-Defined Web Service Call on page 578 .
Direct web service call	For more information, see Direct Web Service Call on page 578 .
Generic web service call	For more information, see Generic Web Service Call on page 576 .

Table 331: Script properties

Property	Description
Script name	Name of the script. Label custom scripts with the prefix "CCC_". Script names for the generic method call must follow the following format:<customer prefix>_<table>_<target system>_<event>. Select the table, target system and event to create the script for. Parameter, value type and data table are automatically determined from the selected table.
Parameter	Name of the parameter.
Value type	Data type of the parameters.
Data table	Data table, which contains the data to be passed to the web service.
Return value	Data type of the parameter that contains the return value.

Table 332: Data mapping

Property	Description
Parameter	Parameter passed to the web service.
Value type	Data type of the parameters.
Mapped from	Parameter from the defined script properties. Open the menu and assign the associated parameters. If necessary, select the column from the data table which contains the value to be passed.

You will see the generated script in the script code view. You can use extended edit mode to edit the script.

TIP: The script call the function VID_GetWcfWebService, which in turn, uses the functions GetWcfBinding and GetWcfEndpointAddress. These three functions can be overwritten.

8. To end the Web Service Integration Wizard, click **Finish**.
9. Save the changes.
10. Compile the database.

Related Topics

- [One Identity Manager Scripts on page 453](#)
- [Support for Scripting on page 51](#)
- [Overriding Scripts on page 472](#)

Changing a Web Service Solution

You can change or extend an existing web service solution at any time. This overwrites the existing script or adds new scripts.

To extend a web service solution

1. Select the category **Base Data | General | Web services** in the Designer.
2. Select the web service in the List Editor.
3. Select **Create web service call** in the task view.

This start the Web Service Integration Wizard.

4. Follow the wizard's instructions.
5. Save the changes.
6. Compile the database.

To edit a web service solution

1. Select the category **Base Data | General | Web services** in the Designer.
2. Select the web service in the List Editor.
3. Select **Edit web service <web service name>** from the task view.
This starts the Web Service Integration Wizard.
4. Follow the wizard's instructions.
5. Save the changes.
6. Compile the database.

Detailed information about this topic

- [Creating a Web Service Solution with the Web Service Integration Wizard](#)

Deleting a Web Service Solution

To delete a web service solution from the database

1. Delete the web service.
2. Delete all associated custom scripts.
3. Determine all other custom elements of your web service solution and delete them.
4. Save the changes.
5. Compile the database.

SOAP Web Service

The One Identity Manager's SOAP Web Service provides an SOAP interface for accessing the One Identity Manager object model. The SOAP Web Service manages a connection pool. Not every call opens a new connection. Not all object layer functions are support by the SOAP Web Service. The SOAP Web Service supplies methods for single objects, object lists and function call.

Table 333: Methods for Single Objects

Method	Description
CreateSingleObject	Adds a new single object.
GetCompleteSingleObject	Loads a single complete object from the database with all parameters.
GetCompleteSingleObjectEx	Functionality analog to GetCompleteSingleObject with support for a multi-column primary key.
GetSingleObject	Loads a single object from the database.
GetSingleObjectEx	Functionality analog to GetSingleObject with support for a multi-column primary key.
ChangeSingleObject	Saves changes to a single object.
ChangeSingleObjectEx	Functionality analog to ChangeSingleObject with support for a multi-column primary key.
DeleteSingleObject	Deletes a single object.
DeleteSingleObjectEx	Functionality analog to DeleteSingleObject with support for a multi-column primary key.
Exists	Does a specific single object exist?
ExistsEx	Functionality analog to Exists with support for a multi-column primary key.
GetSingleProperty	Get a single value from an object.
GetSinglePropertyEx	Functionality analog to GetSingleProperty with support for a multi-column primary key.

Table 334: Methods for Object Lists

Method	Description
GetListObject	Loads a list of objects.
GetListObjectWithDisplays	Loads a list of objects with data additional to the primary key about the columns to load.

Table 335: Methods for Function Calls

Function	Description
InvokeCustomizer	Call a customizer method for an object.
InvokeCustomizerEx	Functionality analog to InvokeCustomizer with support for a multi-column primary key.
InvokeDialogMethod	Calls a dialog method for a dialog object.
InvokeDialogMethodEx	Functionality analog to InvokeDialogMethod with support for a multi-column primary key.
FireGenEvent	Generates processes of a specific event.
FireGenEventEx	Functionality analog to FireGenEvent with support for a multi-column primary key.

Detailed information about this topic

- [Installing and Configuring the SOAP Web Service on page 585](#)
- [Examples of Calls on page 592](#)

Installing and Configuring the SOAP Web Service

To install the SOAP Web Service, you must provide a server on which the following software is already installed:

- Windows operating system

Following versions are supported:

- Windows Server 2008 (non-Itanium based 64-bit) Service Pack 2 or later
- Windows Server 2008 R2 (non-Itanium based 64-bit) Service Pack 1 or later
- Windows Server 2012

- Windows Server 2012 R2
 - Windows Server 2016
 - Microsoft .NET Framework Version 4.5.2 or later
- NOTE:** Microsoft .NET Framework version 4.6 is not supported.
- Microsoft Internet Information Service 7, 7.5, 8, 8.5 or 10 with ASP.NET 4.5.2 and Role Services:
 - Web Server > Common HTTP Features > Static Content
 - Web Server > Common HTTP Features > Default Document
 - Web Server > Application Development > ASP.NET
 - Web Server > Application Development > .NET Extensibility
 - Web Server > Application Development > ISAPI Extensions
 - Web Server > Application Development > ISAPI Filters
 - Web Server > Security > Basic Authentication
 - Web Server > Security > Windows Authentication
 - Web Server > Performance > Static Content Compression
 - Web Server > Performance > Dynamic Content Compression

Required Permissions

The user account that the Internet Information Service runs under, needs write access (MODIFY) to the installation directory.

Detailed information about this topic

- [Installing the SOAP Web Service on page 586](#)
- [Configuring the SOAP Web Service on page 589](#)
- [Displaying the Status of a SOAP Web Service on page 591](#)
- [Uninstalling the SOAP Web Service on page 592](#)

Installing the SOAP Web Service

IMPORTANT: Start the SOAP Web Service installation locally on the server.

To install the SOAP Web Service

1. Execute the program autorun.exe from the root directory on the One Identity Manager installation medium.
2. Go to the **Installation** tab and select the entry **Web based components** and click **Install**. Starts the Web Installer.

3. Select **Install Web Installer** on the SOAP Web Service start page and click **Next**.
4. Enter connection credential for the One Identity Manager database on the **Database connection** page and click **Next**.
5. Configure the following settings on the **Select setup target** page and click **Next**.

Table 336: Settings for the Installation Target

Setting	Description
Application name	Name used as application name, as in the title bar of the browser, for example.
Target in IIS	Internet Information Services web page on which to install the application.
Enforce SSL	Specifies whether insecure websites are available for installation. If the option is set, only sites secured by SSL can be used for installing. This setting is the default value. If this option is not set, insecure websites can be used for installing.
URL	The application's Uniform Resource Locator (URL).
Install dedicated application pool	Specifies whether an application pool is installed for each application. This allows applications to be set up independently of one another. If this option is set, each application is installed in its own application pool.
Application pool	<p>The application pool to use. This can only be entered if the option Install dedicated application pool is not set.</p> <p>The application pool is formatted with the following syntax, if the default value "DefaultAppPool" is used.</p> <p><application name>_POOL</p>
Identity	<p>Permissions for executing an application pool. A default identity or a user defined user account can be used.</p> <p>The user account is formatted with the following syntax, if the default value "ApplicationPoolIdentity" is used.</p> <p>IIS APPPOOL\<application name>_POOL</p> <p>If you want to authorize another user, click ... next to the text box and enter the user and password.</p>
Web Authentication	<p>Specifies the type for authentication against the web application. You have the following options:</p> <ul style="list-style-type: none"> • Windows Authentication (Single Sign-On) <p>The user is authenticated against the Internet Information Services using their Windows user account and the web application logs in the employee assigned to</p>

Setting	Description
	<p>the user account as role-based. If single sign-on is not possible, the user is diverted to a login page. You can only select this authentication method if Windows authentication is installed.</p> <ul style="list-style-type: none"> • Anonymous <p>Login is possible without Windows authentication. The user is authenticated against the Internet Information Services and the web application anonymously and the web application is directed to a login page.</p>
Database authentication	<p>NOTE: You can only see this section if you have selected an SQL database connection in Database connection.</p>

Specifies the type for authentication against the One Identity Manager database. You have the following options:

- Windows authentication

The web application is authenticated against the One Identity Manager database using the Windows account under which your application pool is running. Login is possible with a user defined user account or a default identity for the application pool.

- SQL Authentication

Login is only possible through a user defined user accounts. Authentication is done using user name and password. This access data is saved in the web application configuration as computer specific encrypted.

- Specify the user account for automatic updating of the application server on the **Set update credentials** page.

The user account is used to add or replace files in the application directory.

- Set the option **Use IIS credentials for update** if you want to use the user account, under which the application is run, for updates.
- Set the option **Use other credentials for updates** if you want to use another user account and enter the domain, user name and password for the user.



NOTE: The following permissions are required for automatic updating:

- The user account for updating required write permissions for the application directory.
- The user account for updating requires the local security policy "Log on as a batch job".
- The user account, under which the application pool runs, requires the local security policies "Replace a process level token" and "Adjust memory quotas for a process".

7. Installation progress is displayed on the **Setup is running** page. Once installation is complete, click **Next**.

The Web Installer generates the web application and the corresponding configuration files (web.config) for each folder.

8. Click **Finish** on the last page to end the program.

Configuring the SOAP Web Service

The SOAP Web Service configuration is found in the XML file Web.config in the installation directory. You can use any text editor to edit this file.

Table 337: Configurable Options in the Configuration File "Web.config"

Section	Option	Permitted Values	Meaning
connectionString			Database connection parameter.
runtimedirs	key="Cache"	value = "<path>"	Directory for storing the cache directory. Default: value="C:\inetpub\wwwroot\<web service name>\App_Data\Cache\DB"
	key="AssemblyCache"	value = "<path>"	Directory for storing the cache directory. Default: value="C:\inetpub\wwwroot\<web service name>\App_Data\Cache\Assemblies"
settings	key="timeout"	value-e="<time>"	Timeout for connections in the application pool. Default: value="00:05:00"

Section	Option	Permitted Values	Meaning
	key="maxconnectionlifetime"	value-e=" <time>"</time>	Maximum length of time to maintain the connections. After this time limit has expired, all the connections are closed even if the timeout has not expired yet. Default: value="00:05:00"
	key="usepropertybag"	value = "True" value = "False"	Specifies whether a property bag is used. A property bag is used in order to maintain a particular fill order when object properties are populated.
	key="ignoreinvisiblevalues"	value = "True" value = "False"	Default: value="True" Specifies whether values that the user is not permitted to see are not returned.
			Table 338: Permitted Values
Value	Meaning		
False	Values are set in the object in the order in which they were given.		
True	The fill order is taken from the meta data.		
			Table 339: Permitted Values
Value	Meaning		
False	An error message sent if the user is not permitted to see the values.		

Section	Option	Permitted Values	Meaning
		Value	Meaning
		True	Values that the user is not permitted to see are not returned. If this value is set, the user is issued an error message.
			Default: value="True"
	key = logdirectory	value = "<path>"	Log directory. Default: value = "C:\inetpub\wwwroot\<web service name>\App_Data\Logs"
	key = allowwebservicemethods	value = "List of methods"	semicolon delimited list of permitted web service methods.
	key = allowfunctions	value = "List of functions"	List of the permitted functions for each CallFunction method. If no other function is given, all functions are permitted.

Displaying the Status of a SOAP Web Service

The SOAP Web Service can be reached over a browser under:

`http://<server>/<application name>`

`https://<server>/<application name>`

TIP: You can open the web server's status display in Job Queue Info. Select the menu item **View | Server state** in the Job Queue Info and display the web server's state on the **Web servers** tab by using **Open in browser** in the context menu.

In addition, API documentation is available here.

Uninstalling the SOAP Web Service

To uninstall a web application

1. To uninstall a web application, use the Web Installer.
 - a. Execute the program autorun.exe from the root directory on the One Identity Manager installation medium.
 - b. Go to the **Installation** tab and select **Web-based components** and click **Install**. This starts the Web Installer.
- OR -
- c. Start the Web Installer from **Start | One Identity | One Identity Manager | Configuration | Web Installer**.
2. Select **Uninstall a One Identity Manager web application** on the Web Installer start page and click **Next**.
3. All installed web applications are displayed on the page, **Uninstall a One Identity Manager web application**.
 - a. Select the web application you want to remove by double-clicking on it.
 - b. Select the authentication module in the **Authentication method** section and authenticate yourself.
 - c. Click **Next** to start uninstalling.
 - d. Confirm the security prompt with **Yes**.
4. The uninstall progress is displayed on the **Setup is running** page. After installation is complete, click **Next**.
5. Click **Finish** on the last page to end the program.

Examples of Calls

You will find an overview of supplied methods under [SOAP Web Service on page 584](#). In the following there are some examples of a web service client calls in the programming language C#.

Preparation

Authentication is carried out through an authentication string which contains an authentication module and the login data to use. You must create an instance of the web service and the object for the login data to log in to the system. The login data is passed to following calls.

Example:

```
var svc = new Q1IMServiceSoapClient();  
var login = new LoginInformation
```

```
{ AuthString = "Module=DialogUser;User=viadmin;Password=" };
```

Table 340: Examples of Authentication

Authentication Module	Example
System user	Module=DialogUser;User=<user name>;Password=<password>
Employee	Module=Person;User=<central user account>;Password=<password>
Active Directory user account (role based)	Module=RoleBasedADSAccount
Active Directory user account (manual input/role based)	Module=RoleBasedManualADS;User=<AD user name>;Password=<AD password>

You can find an overview of the authentication module under [One Identity Manager Authentication Module on page 74](#).

GetListObject

This method returns an array of objects, which correspond to the given WHERE clause. The returned array contains the object's primary key and a special key, [DISPLAY], which contains the object's display value.

Example:

```
Q1IMService.KeyValuePair[][] objects = svc.GetListObject(login, "Person",
    "FirstName like 'Hal%'");
```

GetListObjectWithDisplays

This method works in the same way as GetListObject and allows you to enter details of additional columns to be loaded.

Example:

In the example, the columns FirstName and LastName are available.

```
Q1IMService.KeyValuePair[][] objects = svc.GetListObjectWithDisplays(login, "Person",
    "FirstName like 'Hal%'",
    new [] {"FirstName", "LastName"});
```

GetCompleteSingleObject

All the properties of the object, defined through the primary key, are loaded by the method.

Example:

```
Q1IMService.KeyValuePair[] singleValues = svc.GetCompleteSingleObject(login,
    "Person", "UID_Person", "746a5662-054b-4531-a889-1c135dad4c05");
```

GetSingleObject

Properties of a single object are loaded with this method.

Example:

In the example, the columns FirstName and LastName and the display value are loaded. The display value is given in the key [DISPLAY].

```
Q1IMService.KeyValuePair[] values = svc.GetSingleObject(login, "Person",
    "UID_Person", "746a5662-054b-4531-a889-1c135dad4c05",
    new[] { "FirstName", "LastName" });
```

ChangeSingleObject

This method changes individual properties of an object.

Example:

In the example, the column Description of the employee with the corresponding UID_Person is modified.

```
var values = new[]
{
    new Q1IMService.KeyValuePair
    {
        Key = "Description",
        Value = "Created by webservice"
    }
};

svc.ChangeSingleObject(login, "Person", "UID_Person",
    "746a5662-054b-4531-a889-1c135dad4c05", values);
```

ChangeSingleObjectEx

Modifying an object with this method is done in the same way as with ChangeSingleObject, but here the primary key value is passed as a Key-Value-Pair-Array.

Example:

```
var values = new[]
{
    new Q1IMService.KeyValuePair
    {
        Key = "Description",
        Value = "Created by webservice"
    }
};
```

```

};

var keys = new[ ]
{
    new Q1IMService.KeyValuePair
    {
        Key = "UID_Person",
        Value = "746a5662-054b-4531-a889-1c135dad4c05"
    }
};

svc.ChangeSingleObjectEx(login, "Person", keys, values);

```

DeleteSingleObject

This method deletes an object.

Example:

In this example, the employee with the corresponding UID is deleted from the database.

```
svc.DeleteSingleObject(login, "Person",
    "UID_Person", "746a5662-054b-4531-a889-1c135dad4c05");
```

DeleteSingleObjectEx

It is possible to delete objects with multicolumn primary key (as from M:N tables) with this method.

Example:

```
svc.DeleteSingleObjectEx (
    login,
    "OrgHasApp",
    new []
    {
        new Q1IMService.KeyValuePair { Key = "UID_Org", Value = <UID> },
        new Q1IMService.KeyValuePair { Key = "UID_Application", Value = <UID> }
    });

```

CreateSingleObject

A new object is created in the database with this object.

Example:

In this example, the employee "Jon Doe" is created.

```
var values = new[ ]
```

```

    {
        new Q1IMService.KeyValuePair {Key = "FirstName", Value = "John"},
        new Q1IMService.KeyValuePair {Key = "LastName", Value = "Doe"}
    };
    svc.CreateSingleObject(login, "Person", values);
}

```

exists

This method checks the existence of an object.

Example:

```
bool exists = svc.Exists(login, "Person",
    "UID_Person", "746a5662-054b-4531-a889-1c135dad4c05");
```

GetSingleProperty

This method can be implemented to find a single property.

Example:

```
string description = svc.GetSingleProperty(login, "Person",
    "UID_Person", "746a5662-054b-4531-a889-1c135dad4c05",
    "Description");
```

InvokeCustomizer

The SOAP Web Service support a method "InvokeCustomizer", which calls a function for an object in the database. The first three parameters specify the object on which the method is called. The parameter customizerName provides the function name. An array of strings follows which contains the fully qualified name of the parameter data types. These are passed to the calling function. The following array of strings contains textual representation of the parameter.

How the function works

- First, the database is opened and gets the object specified by objectType, pkName and pkValue.
- Ten runtime data types, given by parameterTypes are determined.
- After that, text representations of the parameters are converted from the value array to the corresponding runtime data types.
- The function is called with these values.

If the calling function does not have any parameter, the function for the parameter "parameterTypes" and "parameter" "null" can be passed.

Example:

In this example, the method "TestMethod" is called for a Person type object with the primary key UID_Person and the given value. In this case, both parameters of type "System.String" and "System.Int32" with the values "Foo" and "4711" are passed.

```
svc.InvokeCustomizer (login, "Person",
    "UID_Person", "0000644F-C139-4B25-8D1C-5ECB93067E79",
    "TestMethod",
    new [] {"System.String", "System. Int32"},
    new [] {"foo", "4711"});
```

InvokeDialogMethod

The method can call a dialog method on an object. Dialog methods do not have any parameters and no return values. The call is similar to the InvokeCustomizer call.

Example:

In this example, the method "TestDialogMethod" is called for a specific person. "TestDialogMethod" is the name of the method corresponding to DialogMethod.MethodName.

```
svc.InvokeDialogMethod (login,
    "Person",
    "UID_Person", "746a5662-054b-4531-a889-1c135dad4c05",
    "TestDialogMethod");
```

FireGenEvent

A specific event is generated by this method. There is the option to enter other generating parameters.

```
public void FireGenEvent(
    string objectType, string pkName, string pkValue,
    string[] columnName, KeyValuePair[] parameters);
```

Example:

In this example, the event "EXPORT_DATA" is generated without additional parameters.

```
svc.FireGenEvent(login, "Person",
    "UID_Person", "746a5662-054b-4531-a889-1c135dad4c05",
    "EXPORT_DATA", new QIIMService.KeyValuePair[] { });
```

CallFunction

This method calls a One Identity Manager script function.

Example:

In the example, the script VI_BuildInitials is called.

```
svc.CallFunction(login, "VI_BuildInitials",
    new string [] {"John", "Doe"});
```

One Identity Manager as SPML Provisioning Service Provider

The One Identity Manager enables data exchange with other vendor's systems using SPML. SPML stands for Service Provisioning Markup Language and defines a standardized interface for exchanging provisioning information. SPML version 2 (SPMLv2) was published in April 2006 by the Organization for the Advancement of Structured Information Standards (OASIS, www.oasis-open.org). The interface provides a means to simplify and standardize data exchange in the context of complex provisioning solutions and environments.

The One Identity Manager can be implemented as SPML client or as SPML provider. At this point we shall only go into the One Identity Manager configuration as SPML provider. The SPML Provider supports the entire One Identity Manager schema. The objects and relations to be administrated through the SPML provider can be configured to meet customer requirements.

Detailed information about this topic

- [SPML Web Service on page 598](#)
- [Installing and Configuring SPML Web Services on page 599](#)
- [Configuring the One Identity Manager Schema on page 606](#)
- [Testing SPML Web Service Functionality on page 608](#)

SPML Web Service

A web service called the SPML Web Service, is provided for the function of SPML service provider. SPML Web Service conforms to SPMLv2 and its implementation is based on the OASIS publication. It makes the main operations such as adding, deleting and changing objects available as well as extensions for searching and referencing objects.

SPML Web Service supports the SPMLv2 functions defined in the following:

Table 341: SPMLv2 Supported Functions

Function	Description
listTargetsRequest	Returns the provider target system with its specific schema. The SPML provider supports the One Identity Manager schema exclusively.
addRequest	Adds a new object in the given provider target system with the given properties.
lookupRequest	Returns the properties of a object identified by a key.
modifyRequest	Changes the properties of a key identified object in the given provider target system.
deleteRequest	Deletes a key identified object in the provider target system.
searchRequest	Returns all objects in the provider target system that fulfill the search criterion.
iterateRequest	Returns other data sets from a search assuming not all of search results have been sent to the client.
closeIteratorRequest	Closes an active search and informs the provider that no further results are required.

The extension "reference" makes it possible to maintain references between different provider target system objects. There are two different types of references for this.

- Reference type "owner"
References of type "owner" result in One Identity Manager in foreign key relations.
- Reference type "memberOf"
Reference of type "memberOf" result in One Identity Manager many-to-many assignments.

Installing and Configuring SPML Web Services

To install SPML Web Service, a server has to be made available on which the following software is already installed:

- Windows operating system

Following versions are supported:

- Windows Server 2008 (non-Itanium based 64-bit) Service Pack 2 or later
- Windows Server 2008 R2 (non-Itanium based 64-bit) Service Pack 1 or later

- Windows Server 2012
 - Windows Server 2012 R2
 - Windows Server 2016
- Microsoft .NET Framework Version 4.5.2 or later
- NOTE:** Microsoft .NET Framework version 4.6 is not supported.
- Microsoft Internet Information Service 7, 7.5, 8, 8.5 or 10 with ASP.NET 4.5.2 and Role Services:
 - Web Server > Common HTTP Features > Static Content
 - Web Server > Common HTTP Features > Default Document
 - Web Server > Application Development > ASP.NET
 - Web Server > Application Development > .NET Extensibility
 - Web Server > Application Development > ISAPI Extensions
 - Web Server > Application Development > ISAPI Filters
 - Web Server > Security > Basic Authentication
 - Web Server > Security > Windows Authentication
 - Web Server > Performance > Static Content Compression
 - Web Server > Performance > Dynamic Content Compression

Required Permissions

The user account that the Internet Information Service runs under, needs write access (MODIFY) to the installation directory.

Detailed information about this topic

- [Installing the SPML Web Service on page 600](#)
- [Configuring the SPML Web Service on page 603](#)

Installing the SPML Web Service

IMPORTANT: Start the SPML web service installation locally on the server.

To install the SPML web service

1. Execute the program autorun.exe from the root directory on the One Identity Manager installation medium.
2. Go to the **Installation** tab and select the entry **Web based components** and click **Install**. Starts the Web Installer.

3. Select the **Install SPML web service** on the Web Installer start page and click **Next**.
4. Enter connection credential for the One Identity Manager database on the **Database connection** page and click **Next**.
5. Configure the following settings on the **Select setup target** page and click **Next**.

Table 342: Settings for the Installation Target

Setting	Description
Application name	Name used as application name, as in the title bar of the browser, for example.
Target in IIS	Internet Information Services web page on which to install the application.
Enforce SSL	Specifies whether insecure websites are available for installation. If the option is set, only sites secured by SSL can be used for installing. This setting is the default value. If this option is not set, insecure websites can be used for installing.
URL	The application's Uniform Resource Locator (URL).
Install dedicated application pool	Specifies whether an application pool is installed for each application. This allows applications to be set up independently of one another. If this option is set, each application is installed in its own application pool.
Application pool	<p>The application pool to use. This can only be entered if the option Install dedicated application pool is not set.</p> <p>The application pool is formatted with the following syntax, if the default value "DefaultAppPool" is used.</p> <p><application name>_POOL</p>
Identity	<p>Permissions for executing an application pool. A default identity or a user defined user account can be used.</p> <p>The user account is formatted with the following syntax, if the default value "ApplicationPoolIdentity" is used.</p> <p>IIS APPPOOL\<application name>_POOL</p> <p>If you want to authorize another user, click ... next to the text box and enter the user and password.</p>
Web Authentication	<p>Specifies the type for authentication against the web application. You have the following options:</p> <ul style="list-style-type: none"> • Windows Authentication (Single Sign-On) <p>The user is authenticated against the Internet Information Services using their Windows user account</p>

Setting	Description
	<p>and the web application logs in the employee assigned to the user account as role-based. If single sign-on is not possible, the user is diverted to a login page. You can only select this authentication method if Windows authentication is installed.</p>
	<ul style="list-style-type: none"> • Anonymous <p>Login is possible without Windows authentication. The user is authenticated against the Internet Information Services and the web application anonymously and the web application is directed to a login page.</p>
Database authentication	<p>NOTE: You can only see this section if you have selected an SQL database connection in Database connection.</p>
	<p>Specifies the type for authentication against the One Identity Manager database. You have the following options:</p>
	<ul style="list-style-type: none"> • Windows authentication
	<p>The web application is authenticated against the One Identity Manager database using the Windows account under which your application pool is running. Login is possible with a user defined user account or a default identity for the application pool.</p>
	<ul style="list-style-type: none"> • SQL Authentication <p>Login is only possible through a user defined user accounts. Authentication is done using user name and password. This access data is saved in the web application configuration as computer specific encrypted.</p>

6. Specify the user account for automatic updating of the application server on the **Set update credentials** page.

The user account is used to add or replace files in the application directory.

- Set the option **Use IIS credentials for update** if you want to use the user account, under which the application is run, for updates.
- Set the option **Use other credentials for updates** if you want to use another user account and enter the domain, user name and password for the user.

- NOTE:** The following permissions are required for automatic updating:
- The user account for updating required write permissions for the application directory.
 - The user account for updating requires the local security policy "Log on as a batch job".
 - The user account, under which the application pool runs, requires the local security policies "Replace a process level token" and "Adjust memory quotas for a process".
7. Installation progress is displayed on the **Setup is running** page. Once installation is complete, click **Next**.
The Web Installer generates the web application and the corresponding configuration files (web.config) for each folder.
 8. Click **Finish** on the last page to end the program.

Configuring the SPML Web Service

The SPML Web Service configuration is found in the XML file Web.config in the installation directory. You can use any text editor to edit this file.

- NOTE:**
- After the default installation, make any changes required to the option AuthenticationString in the section configuration\application.
 - Create the schema files QOIM_Schema.xsd and QOIM_SpmlTargetSchema.xsd with the Schema Editor in the Designer. [For more information, see Creating the Schema File on page 608](#). Add the schema files to the SPML Web Service directory (by default in the Schema directory of the install directory) and declare the storage location of the schema files in the configuration file using the options ProviderSchema and SpmlTargetSchema.
 - If the SPML Web Service should only be available over an encoded SSL connection, configure this in the Internet Information Services setting for each respective application. Look at your Internet Information Services documentation for further information.

Table 343: Configurable Options in the Configuration File "Web.config"

Section	Option	Permitted Values	Meaning
connectionString			Database connection parameter.
runtime-	key="Cache"	value =	Directory for storing the cache directory.

Section	Option	Permitted Values	Meaning
dirs		"<path>"	Default: value="C:\inetpub\wwwroot\<web service name>\App_Data\Cache\DB"
	key="AssemblyCache"	value = "<path>"	Directory for storing the cache directory. Default: value="C:\inetpub\wwwroot\<web service name>\App_Data\Cache\Assemblies"
application	key = "ProviderSchema"	value = "<path>"	Relative path to SPML schema (QOIM_Schema.xsd). The schema defines all objects and properties the can be administered using the web service. The file is created by Designer. All requests made to the web service are verified against this file. Default: value=".\\Schemas\\QOIM_Schema.xsd"
	key = "SpmlTargetSchema"	value = "<path>"	Relative path to SPML target schema (QOIM_SpmlTargetSchema.xsd). The schema defines the response to the list Target Request. The file is created by Designer. Default: value=".\\Schemas\\QOIM_SpmlTargetSchema.xsd"
	key = "MaxConnections"	value = "<Integer>"	Number of possible simultaneous connections (number of clients). Default: value ="1"
	key = „AuthenticationString”	value= "Module=;User=;Password="	Authentication module and login data for carrying out login and all operations of the web service. Default: value="Module=DialogUser;User=DIALOGUSER;Password=PASSWORD"
	key = "DebugMode"	value = "True" value = "False"	Extended data in the log. Default: value="true"
	key =	value =	Always log queries.

Section	Option	Permitted Values	Meaning
	"LogAllRequests"	"True" value = "False"	Default: value="false"
	key = "LogDirectory"	value = "<path>"	Log directory. Default: value=".\\Log"
	key = "MaxSearchResults"	value = "<Integer>"	Maximum number of search results permitted for the iteration. Default: value="10000"
	key = "ConcurrentSearchResponseObjects"	value = "<Integer>"	Number of objects per iteration that may be returned to the client by the search operation. Default: value="10"
	key = "CheckForUnusedResultsInterval"	value = "<Integer>"	Interval in seconds for scanning orphaned search results. Default: value="30"
	key = "KeepSearchResultsFor"	value = "<Integer>"	Interval in seconds the client has to iterate the result set before it is discarded. Default: value="60"
	key = logdirectory	value = "<path>"	Log directory. Default: value = "C:\\inetpub\\wwwroot\\<web service name>\\App_Data\\Logs"

NOTE: To encrypt the connection parameter (ConnectionString), use aspnet_regiis.exe.

Calling example:

```
c:\\windows\\Microsoft.NET\\Framework\\v4.0.30319\\aspnet_regiis.exe -pe "application" -app "/<web service name>" -prov "DataProtectionConfigurationProvider"
```

where: <web service name> = web service path on the Internet Information Services

Uninstalling the SPML Web Service

To uninstall a web application

1. To uninstall a web application, use the Web Installer.
 - a. Execute the program autorun.exe from the root directory on the One Identity Manager installation medium.
 - b. Go to the **Installation** tab and select **Web-based components** and click **Install**. This starts the Web Installer.
- OR -
 - c. Start the Web Installer from **Start | One Identity | One Identity Manager | Configuration | Web Installer**.
2. Select **Uninstall a One Identity Manager web application** on the Web Installer start page and click **Next**.
3. All installed web applications are displayed on the page, **Uninstall a One Identity Manager web application**.
 - a. Select the web application you want to remove by double-clicking on it.
 - b. Select the authentication module in the **Authentication method** section and authenticate yourself.
 - c. Click **Next** to start uninstalling.
 - d. Confirm the security prompt with **Yes**.
4. The uninstall progress is displayed on the **Setup is running** page. After installation is complete, click **Next**.
5. Click **Finish** on the last page to end the program.

Configuring the One Identity Manager Schema

The SPML Web Service supports the entire One Identity Manager schema. It is necessary to define the objects and properties to be managed as well as the relations in the One Identity Manager schema in order to manage objects and their relations using the SPML web service. The SPML web service cannot be used until the objects and properties as well as references have been defined in the One Identity Manager schema as being managed with SPML. After the definition has been made, two schema files are created that are needed for validation by the SPML Web Service. The files should be exchanged in the appropriate SPML Web Service directory.

Detailed information about this topic

- Preparing the One Identity Manager Schema for Exporting to the SPML Schema on page 607
- Creating the Schema File on page 608

Preparing the One Identity Manager Schema for Exporting to the SPML Schema

For administration of single objects with their properties and for relations between different object types with SPML Web Service, label the corresponding tables, columns and the One Identity Manager schema table relations to be exported to the SPML schema.

Edit the table and column definitions and the table relations with Schema Editor's Designer. We shall only go into the minimum amount of data required at this point.

To manage objects and their properties with the SPML web service

1. Select the category **One Identity Manager Schema** in the Designer.
2. Select the table and start the Schema Editor with the task **Show table definition**.
3. Set the option **Export for SPML schema**.
4. Select the column in the Schema Editor and swap to **More** tab.
5. Set the option **Export for SPML schema**.

NOTE: If references between different One Identity Manager schema object types should be managed with the SPML Web Service, both objects effected have to be marked with the SPML administration option that means both tables have to be labeled with **Export for SPML schema**.

References between object types are mapped by foreign key relations and many-to-many assignments in the One Identity Manager.

- It is sufficient to mark the respective column in the One Identity Manager schema with the option **Export for SPML schema** in order to handle foreign key relations with SPML.

NOTE: Only one foreign key relation can be managed between two object types with SPML. Thus the business role manager (Org._UID_PersonHead) can be maintained with SPML, but not simultaneously with the deputy manager (Org._UID_PersonHeadSecond).

- For the configuration of many-to-many relations for use with SPML, select the respective many-to-many tables and label the table relation with the option **Export for SPML schema**.

Related Topics

- [Mapping Table Definitions on page 172](#)
- [Mapping Column Definitions on page 188](#)
- [Mapping Table Relations on page 208](#)

Creating the Schema File

Once you have labeled all tables, columns and table relations that should be managed using SPML, you need to create the necessary schema file for SPML Web Service.

Before exporting, ensure that you have committed all the changes in the Designer in the main database and that all open calculation tasks for the DBQueue Processor have been processed.

To create a schema file

1. Start the Schema Editor in the Designer in the category **One Identity Manager Schema**.
2. Select **Schema | Export SPML schema information...** in the menu.
3. Enter the directory that the schema file is going to be created in.
4. Click **OK**.

This starts the export. The export can take some time depending on the number of changes.

5. Click **OK**.

IMPORTANT: If you change other SPML relevant settings on the One Identity Manager schema at a later date, you must recreate the schema file.

Place the schema files "QOIM_Schema.xsd" and "QOIM_SpmITargetSchema.xsd" in the SPML Web Service directory (by default directory "Schema" in the installation directory). Enter the storage location for the schema files in the SPML Web Service configuration file.

Related Topics

- [Configuring the SPML Web Service on page 603](#)

Testing SPML Web Service Functionality

A simple test front-end is supplied in order to test the basic functionality of SPML Web Service. Prerequisite for using the test front-end is that SPML Web Service is correctly installed and configured.

Use a browser to check whether SPML Web Service is functioning and correctly installed.

The SPML web service can be reached over a browser under:

```
http://<server>/<application name>  
https://<server>/<application name>
```

TIP: You can open the web server's status display in Job Queue Info. Select the menu item **View | Server state** in the Job Queue Info and display the web server's state on the **Web servers** tab by using **Open in browser** in the context menu.

Detailed information about this topic

- [SPML Test Front-end Configuration on page 609](#)
- [Using the SPML Test Front-end on page 610](#)

SPML Test Front-end Configuration

This configuration setting provides the SPML Web Service specific URL for use in the SPML test front-end.

1. Copy the file "VI.SPMLTestFrontend.exe" from the installation medium, directory QBM\dvd\AddOn\SPML\Testfrontend.
2. Create a configuration file "VI.SPMLTestFrontend.exe.config" with the following contents in the same directory as "VI.SPMLTestFrontend.exe" in order to post the configured web service in test front-end:

```
<?xml version="1.0" encoding="utf-8" ?>  
<configuration>  
<configSections>  
<sectionGroup name="applicationSettings"  
type="System.Configuration.ApplicationSettingsGroup, System, Version=2.0.0.0,  
Culture=neutral, PublicKeyToken=b77a5c561934e089" >  
<section name="VI.SPML.Properties.Settings"  
type="System.Configuration.ClientSettingsSection, System, Version=2.0.0.0,  
Culture=neutral, PublicKeyToken=b77a5c561934e089" requirePermission="false" />  
</sectionGroup>  
</configSections>  
<applicationSettings>  
<VI.SPML.Properties.Settings>  
<setting name="SPMLTestFrontend_SPMLService_AESPMService" serializeAs="String">  
<value>http://<Servername>:<Port>/<web service name>/AESPMService.asmx</value>  
</setting>  
</VI.SPML.Properties.Settings>
```

```
<applicationSettings>  
</configuration>
```

3. Save the configuration. Then start the application "VI.SPMLTestFrontend.exe".

Using the SPML Test Front-end

It is possible with the SPML test front-end to test and analyze SPML Web Service functionality. The front-end is used exclusively to analyze SPML Web Service and to test the functionality.

 **NOTE:** Long term usage of the front-end for controlling the SPML web service is not planned.

To test the SPML web service functions

1. Start the SPML test font-end using the file VI.SPMLTestFrontend.exe.
2. Select the SPML web service function to test from the **Choose Request** list.
The corresponding XML queries are displayed in the text field, **SPML Request (XML)**.
You can edit the XML request before sending it to the SPML Web Service. Always check the predefined section of the XML request and modify the schema defined in the target system for SPML support. The predefined sections are supposed to provide help for formulating SPML compliant requests.
3. Set the option **Increment request IDs automatically**, if the request ID passed to the XML queries should be incremented. This option is disabled by default and the given request ID is used.
4. Send the query to the SPML web service using the **Access** button.
The result is displayed in the text field **SPML Response (XML)**.

If a new object is added in the target system, its key is added to the **Known Objects** list. An iterator is returned if a search is carried out with a limited result set and the result list cannot be returned in its entirety. This iterator is added to the list of known iterators (**Known Iterators**). The search can be continued with this iterator.

You will find detailed error messages in the log file. This is stored in the directory that you specified in the option "LogDirectory" in the SPML Web Service configuration file.

Related Topics

- [Configuring the SPML Web Service on page 603](#)

Searching for Errors in the One Identity Manager

The One Identity Manager provides various options for localizing errors. These include:

- Monitoring Process Handling with Job Queue Info on page 611
- Error Messaging in the Error Message Window on page 627
- Displaying Messages in the Error Log View on page 629
- Logging Messages in the Database Journal on page 632
- Writing the One Identity Manager Log on page 634
- One Identity Manager Service Logging on page 636

Monitoring Process Handling with Job Queue Info

The Job Queue Info tool supports control of the current state of services running in a One Identity Manager network. It enables a detailed and comprehensive overview of the requests in the Job queue and various One Identity Manager Service requests on the servers. This program makes it easier to work with processes, supplies status information during run-time and allows errors to be quickly recognized and debugged.

Detailed information about this topic

- Monitoring Process Execution on page 617
- Displaying Details of Process Handling on page 618
- Displaying Details of Process Step Handling on page 619
- Displaying Details of a Process Step Parameter on page 620
- Reinstating Process Steps and Processes on page 621
- Enabling Extended Logging of Process Steps

- Determining the State of the Server on page 623
- Displaying Entries in the DBQueue on page 624
- Displaying Messages in the Database Journal on page 624
- Displaying the Job Queue Sequence on page 626
- Stopping the System (Emergency Stop) on page 626

Working with Job Queue Info

The Job Queue Info tool supports control of the current state of services running in a One Identity Manager network. It enables a detailed and comprehensive overview of the requests in the Job queue and various One Identity Manager Service requests on the servers. This program makes it easier to work with processes, supplies status information during run-time and allows errors to be quickly recognized and debugged.

Menu Items

Table 344: Meaning of Items in the Menu Bar

Menu	Menu Item	Meaning	Key Combination
Database	New connection...	Creates a new database connection.	Ctrl + Shift + N
	Close connection	Closes the current database connection.	
	Change password...	The logged in user can change the password.	
	Settings...	For configuring general program settings.	
	Exit	Exits the program.	
Filter	Define filter...	The WHERE clause wizard is opened to assist in defining a filter.	
	Delete filter	Deletes the filter.	

Menu	Menu Item	Meaning	Key Combination
View	Job queue	Hides or shows job queue view.	
	Job server	Hides or shows job server view.	
	Process History	Hides or shows process history view.	
	Base object	Hides or shows base object view.	
	Process	Hides or shows process view.	
	Process step	Hides or shows process step view.	
	Parameter	Hides or shows parameter view.	
	Progress	Hides or shows progress view.	
	Server state	Hides or shows server state view.	
	DBQueue	Hides or shows DBQueue view.	
	Database journal	Opens the database journal view.	
	Refresh view	Updates the view.	F5
Help	Layout	Restores the default layout of the program's graphical interface.	
	Community	Opens the One Identity Manager community website.	
	Support Portal	Opens the One Identity Manager product support website.	
	Training	Opens the One Identity Manager training portal website.	
	Online documentation	Opens the One Identity Manager documentation website.	
	Search...	Opens the search dialog box.	
	Emergency stop	Displays a dialog from which the system can be stopped.	
	Job Queue Info help	Opens program help.	
	Info...	Shows the version information for program.	

Table 345: Meaning of the Icons in the Toolbar

Icon	Meaning
	Creates new database connection.
	Closes current database connection.
	Hides or shows job queue view.
	Hides or shows job server view.
	Hides or shows process history view.
	Hides or shows process view.
	Hides or shows process step view.
	Hides or shows parameter view.
	Opens WHERE clause wizard to set up a filter.
	Deletes filter.
	Reload the data.

Customizing Program Settings

To change the program settings

- Select **Database | Settings...** from the menu.

Table 346: Program Settings

Setting	Meaning
Language	Language of the user interface. The changes come into effect after the program has been restarted. This specifies the language globally for all One Identity Manager programs so that the language does not have to be set separately in each program.
Result limit	Number of entries to load and display for processes or process steps.
Polling interval	Specifies the number of seconds between data requests. The views are updated at the end of every interval. If the value is 0, the views are not updated. In this case, use F5 to update.
Job server HTTP Port	HTTP port configured for the One Identity Manager Service for querying the Job server status. The default value is port 1880.
Status	Maximum delay for status queries. Job servers that do not respond within this

Setting	Meaning
query timeout (s)	time limit are considered unavailable.
Only show process errors	Limits the process history display to processes with errors. The setting does not effect how the process history is recorded, only how it is displayed.

Updating the Views

To update the views press **F5**. If the view focus is on a base object then the whole display is updated and the hierarchy tree is closed. This update refreshes the contents of all views. This update also refreshes the contents of other views.

The views can only ever display a snap-shot of the queue because the contents of the job queue is continually changing. Therefore, when a node is opened or the view is updated, the necessary information may have already been deleted from the job queue. If this is the case, the corresponding entry in the hierarchical display is deleted or the corresponding element is not shown.

Column Configuration

You can specify which columns will be displayed in each of the program's views.

To specify which columns to display

- Select a node in the hierarchical display and select **Configure columns** from the context menu.
Select the columns you want to display by moving through the list and accepting with the arrow buttons, then change the order in which they are displayed.

To change the width of the columns on display

- The column is adjusted to the optimal size by double-clicking on the column sizing bar.
- You can optimize the size of all columns at once by double-clicking on the column delimiter whilst holding down the **Shift** key.

Job Queue Info Views

The Job Queue Info has several views for displaying and editing processes and process steps in the Job queue.

Table 347: Job Queue Info Views

View	Description
Job queue	This view shows the contents of the Job queue grouped by processes. In the first level of the hierarchy, all the processes are shown with a process count. If a process node is opened, all the processes are shown with start times. The complete process with its hierarchy is displayed under a process node. Each process step contains its success and failure branches as sub elements.
Job server	This view shows the Job queue contents sorted by executing servers. At the first hierarchy level, all Job servers are displayed, with their counts of the different processes, that exist in the Job queue for the Job server. If a Job server node is opened, the process tasks are listed and the number of process step per process task is shown. The process steps are listed by start time under the process task node.
Process History	The shows the contents of the table JobHistory. The course of the process execution is displayed by sorted processes. You can limit the list of processes in the process history to only processes with errors in the program settings. If you select a failed process step, the entire error message is shown in a tooltip.
Base objects	The process history entries and the current job queue entries are summarized here in this view for the object being processed. If an error occurs during processing which stops process handling (execution state "Frozen" or "Overlimit"), you can use this view to analyze the processing flow up until this point. Once all processes have been successfully handled for this object the error messages are removed from the view.
Process	This view gives an overview of how process steps are linked within a process. In this way, the execution sequence of individual process steps for large processes can be monitored better. After selecting a process, all its process steps are displayed.
Process step	In this view detailed information is displayed for each process step. The view shows the data structure for a process step at compilation time. After selecting a process step, specific information from the Job queue is mapped as well as each parameter of the selected process step with its values.
Parameters	After selecting a process step, the passing parameters of the process step are displayed with their names and their values. If the selected node does not represent a process step, the parameter view is cleared.
Progress	This view displays the number of entries in the Job queue is queried. The current value is represented by a number and inserted, at the same time, into a bar graph. The process step progress state is shown in different colors.
Server state	This view gives you a faster overview of all the Job servers and Web servers available in the network.

View	Description
DBQueue	Calculation tasks in the table DialogDBQueue used for DBQueue Processor processing are displayed in this view. The number, sort order and name of the queued requests are displayed.
Database journal	Displays entries in the database journal.

Monitoring Process Execution

To monitor the process information

- Select the **Job queue** or **Base objects** process and select **Monitor process** in the context menu.
The process information is updated regularly.

In order to improve the overview, the execution progress of a process step is mirrored in the color of the text.

Table 348: Job Queue Display - Meaning of the Colors

Color	Meaning	Execution state
Orange	This process step is being processed.	Processing
Yellow	This process step is loaded for processing.	Loaded
Green	This process step is ready for processing.	True
Blue	This process step has already been processed.	Finished
Black	This process step is not ready for processing.	False
Red	The process step being dealt with cannot be processed. You can re-enable process steps with a progress state of "Frozen" and therefore present them again for processing. The error message is shown in a tooltip.	Frozen
Purple	The process step being dealt with cannot be processed. You can re-enable process steps with a progress state of "Overlimit" and therefore present them again for processing. The error message is shown in a tooltip.	Overlimit
Light purple	The process step cannot be found.	Missing

TIP: Use **CTRL + F2** you can mark process steps with a bookmark. Use **F2** or **SHIFT + F2** can swap between the marked process steps.

Related Topics

- [Reinstating Process Steps and Processes on page 621](#)

Displaying Details of Process Handling

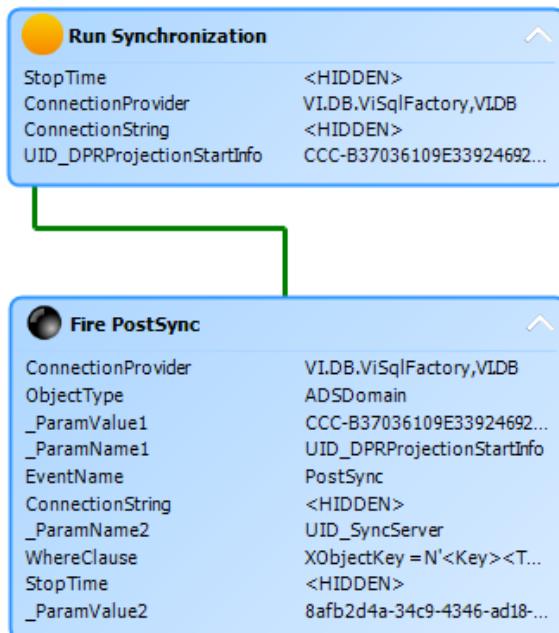
This view gives an overview of how process steps are linked within a process. In this way, the execution sequence of individual process steps for large processes can be monitored better.

To display details of the process handling

- Select a process and select **View | Process** from the menu.
All the process steps of the selected process are displayed.

The process step and its properties are displayed through a special control element. The process step name is displayed in the control's header. The progress state of the process step is clarified by the use of a color icon (●). All other entries represent the parameters for this process step. You can hide or show the parameter list by clicking on the icon ↕ in the header of the control element.

Figure 58: The Process View



Each control element entry has a tooltip.

The process step's tooltip displays the following information:

- Name of the executing queue
- Name of the process component

- Name of the process task name
- Execution state
- Start time of the process step
- Error Message

A parameter's tooltip show the following information:

- Parameter name
- Parameter value

Table 349: Displaying Process's Process Steps - Meaning of the Colors

Color	Meaning	Execution state
Orange	This process step is being processed.	Processing
Yellow	This process step is loaded for processing.	Loaded
Green	This process step is ready for processing.	True
Blue	This process step has already been processed.	Finished
Black	This process step is not ready for processing.	False
Red	The process step being dealt with cannot be processed. You can re-enable process steps with a progress state of "Frozen" and "Over limit" and therefore present them again for processing.	Frozen/Overlimit/unknown

Displaying Details of Process Step Handling

In this view detailed information is displayed for each process step. The view shows the data structure for a process step at compilation time. After selecting a process step, specific information from the Job queue is mapped as well as each parameter of the selected process step with its values.

To display details of the process step handling

- Select a process step and select **View | Process step** from the menu.

Figure 59: Process Step View

Process step	
UID_Job	0A4349C3-454A-46F7-BD32-02386291
BasisObjectKey	
ComponentAssembly	HandleObjectComponent
ComponentClass	VI.JobService.JobComponents.Hand
DeferOnError	False
ErrorNotify	False
ErrorMessages	
ExecutionType	INTERNAL
GenProcID	7BA8D4CF-568F-4EF0-8FBB-E01E587
IgnoreErrors	False
IsRootJob	True
IsSplitOnly	False
IsToFreezeOnError	False
JobChainName	Proc: vid_InsertForHandleObject Obj
LimitationCount	0
MaxInstance	0
MinutesToDefer	0
NotifyAddress	
NotifyAddressSuccess	
NotifyBody	
NntifvRnrvSuccess	

Table 350: Process Step View - Meaning of Icons

Icon	Meaning
	Selection of a process step and its parameters.
	Displays a column from the Jobqueue table and the value.
	Displays a process step parameter and the value.

TIP: You can copy the currently selected data in the view into the clipboard with the key combination **CTRL + C**. The data format is column name = value.

Displaying Details of a Process Step Parameter

After selecting a process step, the passing parameters of the process step are displayed with their names and their values. If the selected node does not represent a process step, the parameter view is cleared.

To display process step parameters

- Select a process step and select **View | Parameters** from the menu.

TIP: You can copy the currently selected data in the view into the clipboard with the key combination **CTRL + C**. The data format is column name = value.

Displaying OUT Parameters

Parameters of type "OUT" or "INOUT" are parameters that can be used by process components to output a value. This value is then available to subsequent process steps in the process and can be used as a value for IN parameters.

Job Queue Info cannot determine technically when or for which process step this parameter applies. Therefore, out parameters are added to a parameter list for a process step (marked in blue).

You cannot see the parameters in the process step view under ParamIN because this view shows the data structure of each process step at compiler time and the out parameters are created within the context of the process.

The time at which the process is loaded into Job Queue Info is important. If a parameter is overwritten several times, only the state at the time of data query is displayed.

Example:

Step 1	Out parameter: X=1
Step 2	In parameter: X=1
	Value changes: X=2
	Out parameter: X=2
Step 3	In parameter: X=2

If the process in Job Queue Info is loaded before step 2 is processed, the value "X=1" is shown for the out parameter in Job Queue Info. If the process is loaded after step 2 has been process, The out parameter shows the value "X=2".

You can find more detailed information about each process step and how the parameters are filled, in the One Identity Manager Service log file.

Related Topics

- [One Identity Manager Service Logging on page 636](#)

Reinstating Process Steps and Processes

The maximum number of times a process can appear in the Job queue can be limited in order to prevent mass modifications. When the limit is exceeded the process steps are set to the state "Overlimit" and can no longer be collected for processing. You can re-enable these process steps for execution.

Critical process steps that have failed are set to "Frozen". You can also re-enable these processes after correcting the error.

To re-enable process steps

- Select the process step and select the context menu item **Re-able process step**.

NOTE: Use **SHIFT + SELECT** or **CTRL + SELECT** to select and re-enabled multiple process steps.

To re-enable a process step

- Select the process step and select the context menu item **Restart process**.

IMPORTANT: All process step are handled again if you restart the process. All previously handled processes up to the the point at which the error occurred are run again. This can lead to data inconsistencies in certain circumstances.

Sometimes a rerun of the failed process step is not desired. This might occur when the action to be carried out by the process has been carried out manually, for example, an expected directory has been manually added in the meantime. Even so, it may just happen that the process should be rerun even though the error has not been fixed, for example, for a rollback of already processed steps. In this case, to continue with the process, the next process step in the success or failure branch can be handled.

To run the subsequent process step

- Select the failed process step and, in the context menu, select **End with success** or **End with error**.

NOTE: Both entries are only visible if there is a success/failure successor and the process step is in the "Frozen" state.

NOTE: Use **SHIFT + SELECT** or **CTRL + SELECT** to select multiple process step to continue process handling.

Enabling Extended Logging of Process Steps

Success and error messages from process handling are written to the One Identity Manager Service log file. In order to test your processes, you can enable logging mode for process steps in the Job Queue Info. This writes progress messages from the process step with the severity level "debug" in a separate log. You can view the log in Job Queue Info as well as in the One Identity Manager Service log.

To enable process step logging mode

- In order to log messages on success and failure, Select the process step **Job queue** view and select **Execution log | Create always** in the context menu.
- In order to log messages only on failure, Select the process step in the **Job queue** view and select **Execution log | On error** in the context menu.

NOTE: You can set the log mode by default for separate process steps. To do this, edit the process step in the Process Editor.

To display the log in Job Queue Info

- Select the process step in the **Job queue** view and select **Execution log | Show** in the context menu.

This displays the log in a separate window. If a process step was executed more than once, for example, if it is re-enabled more than once, several log are displayed.

To display the log in the One Identity Manager Service log file

- Select the Job server in the **Server status** view and select **Open in browser** in the context menu.
- The log is labeled with a link "Log written to Job_<UID_Job>_<yyyymmdd>_<Timestamp>.log". Click the link to display the log.

The files are stored in the One Identity Manager Service log directory.

Repository structure:

```
<log directory>\JobLogs\<first 4 digits of the UID_Job>\Job_<UID_Job>_<yyyymmdd>_<Timestamp>.log
```

To end log mode

- Select the process step in the **Job queue** view and select **Execution log | Disable** in the context menu.

Related Topics

- [One Identity Manager Service Logging on page 636](#)
- [Creating and Editing Process Steps on page 399](#)

Determining the State of the Server

This view gives you a faster overview of all the Job servers and Web servers available in the network. One Identity Manager Service configurations of each Job server stored in the database are used to get more detailed results of Job server status queries. This is especially required if the HTTP server port has been set individually or a Job server processes several queues.

NOTE: Configure and enable the schedule "Get configuration file from the Job server and write in the Job server configuration" in the Designer to load the One Identity Manager Service Job server configuration into the database.

To determine the server status

- Select **View | Server state** from the menu.

To query the status of all the existing Job servers in the database

- Use **F5**.

To query the status of a single Job server

- Use **Get status** in the context menu.

NOTE: Set the HTTP port to be queried and the maximum response time in the program settings.

If the server responds, the system time, the One Identity Manager Service version and the One Identity Manager Service account name are determined and displayed. The software update status as well as the current version of the software are also displayed.

TIP: Use **Refresh server list** or **F6** to reload the list of servers.

To display a Job server's services

- Select the Job server and select **Open in browser**.

The One Identity Manager Service HTTP server for the Job server is queried and the varying One Identity Manager Service services are displayed.

Related Topics

- [Customizing Program Settings on page 614](#)
- [Displaying the One Identity Manager Service Log File on page 638](#)

Displaying Entries in the DBQueue

When changes to inheritance relevant data, such as changes to assignments or when changes are made to certain system data, like changes to the user interface for a system user, the resulting data needs to be recalculated in the One Identity Manager. The requests are queued in the DBQueue and processed by the DBQueue Processor.

To display DBQueue entries

- Select **View | DBQueue** in the menu.

Calculation tasks in the table DialogDBQueue used for DBQueue Processor processing are displayed in this view. The number, sort order and name of the queued requests are displayed. The display is updated at fixed time intervals of 2 seconds.

Displaying Messages in the Database Journal

The database journal is used to store information, warning and error messages from different components of One Identity Manager, for example, DBQueue Processor, Configuration Wizard or One Identity Manager Service. Actions in the program "Job Queue Info", such as re-enabling process steps, are also written to the database journal.

To display items from database journal

- In the Job Queue Info, go to **View | System log**.

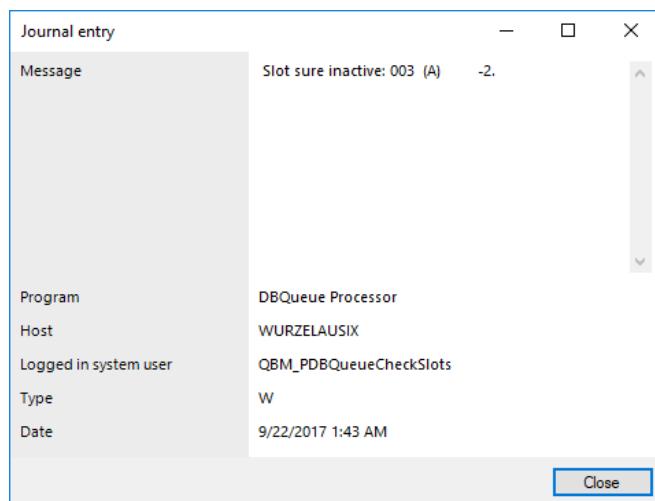
TIP: You can apply different filters to limit the information being displayed. Click the arrow in the column header and select a filter. The  icon in the log toolbar shows whether a filter is active.

Table 351: Displaying Messages in the Database Journal

Icon	Meaning
	Information is written to the error log/database journal.
	A warning has been written to the database journal.
	An error has been written to the database journal.

TIP: Double-click a message to open more details in a separate dialog box.

Figure 60: Detailed Information about a Message



The following information is displayed about a message. The range of information depends on the type of message.

Table 352: Information about a message

Detail	Description
Message	Logged message.
Program	One Identity Manager component from which the message was sent.
Host	Computer from which the action was started.

Detail	Description
Logged in system user	System user that triggered the action.
Type	Type of message. (W= Warning, I = Info, E = Error, T = Trace)
Date	Time and date of the log entry.

Related Topics

- [Logging Messages in the Database Journal on page 632](#)

Displaying the Job Queue Sequence

To display the Job queue sequence

- Select **View | DBQueue** from the menu.

This queries the number of entries in the Job queue. The current value is represented by a number and inserted, at the same time, into a bar graph. The process step progress state is shown in different colors. The display is updated every 5 seconds.

The tooltip shows the timestamp and the number of process steps in the Job queue at this point.

Table 353: Progress View - Meaning of the Colors

Color	Meaning	Progress state
Black	Number of process steps that are not read for processing.	False
Green	Number of process steps ready for processing.	True
Yellow	Number of process steps loaded for processing.	Loaded
Blue	Number of process step that have completed processing	Finished
Red	Number of process steps with an unknown progress state	Frozen/Overlimit/Missing

Stopping the System (Emergency Stop)

In certain circumstances, situations can occur in the system that require processing by One Identity Manager Service and processing of tasks by the DBQueue Processor to be stopped. Changes in One Identity Manager can, for example, sometimes cause the system to become overloaded by making mass entries in the job queue or the DBQueue. To analyze

this situation and to take the necessary steps to solve the problem where necessary, the system can be stopped in Job Queue Info and started again once the problem has been fixed.

To temporarily halt process handling of a single Job server

- Select the menu **View | Server state** from the menu and select the Job server.
- Select **Process handling** from the context menu.
- Use **Start processing** to continue processing the queue.

To stop processing entirely

- Select **Help | Emergency stop**.
- To stop DBQueue processing, click the **DBQueue Processor** button.
From this point on no new calculations are carried out in the database.
 **NOTE:** After the problem has been fixed, the DBQueue Processor can be started again using the same button.
- Click the One Identity Manager Service button to stop collection of process steps for all **One Identity Manager Services**.
Process steps that have already been collected are still processed but no new process step are sent to the services.
 **NOTE:** After the problem has been fixed, the service can be started again using the same button.

The following icons are displayed in the status bar of all administration tools to inform the user that DBQueue Processor processing and services have been stopped.

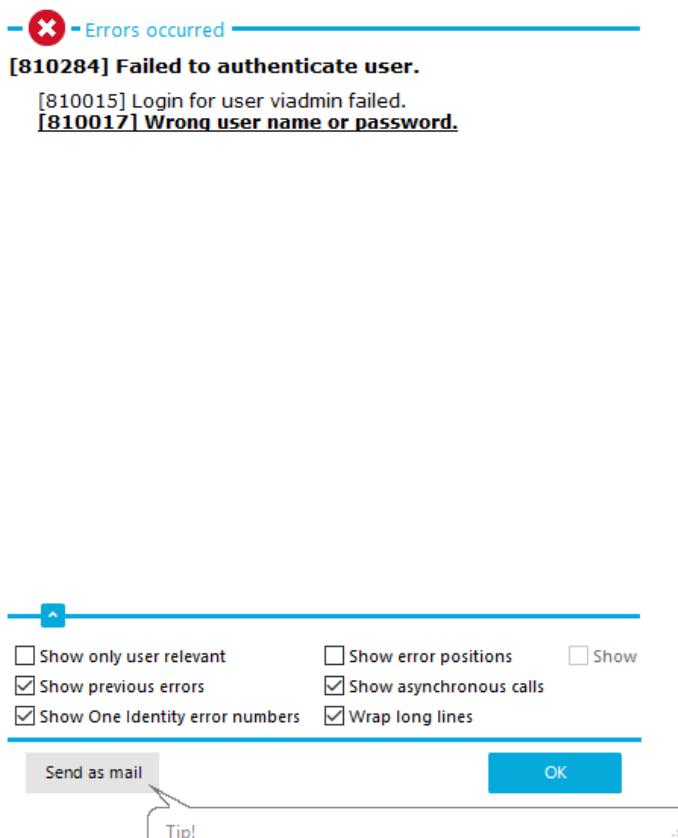
Table 354: Special Icon in the Status Bar for System Stop

Icon	Meaning
	The DBQueue Processor has been stopped.
	The server services have been stopped.

Error Messaging in the Error Message Window

Error messages are shown in a separate window in One Identity Manager tools. In addition, a more detailed description of the error is displayed.

Figure 61: Error Message Window



Configure the amount of information to be displayed using the options in the error message window.

To change options

- Open the configuration with the button and enable or disable the options you want.
- NOTE:** The button **Send as mail** creates a new email message in the default mail program and copies over the error text.

Table 355: Options for Displaying Error Messages

Option	Meaning
Show previous errors	Specifies whether all previous errors that lead to the current error, should also be shown.
Show One Identity error numbers	Specifies whether internal error numbers are shown.
Show error positions	Specifies whether error position are also shown in the program code.

Option	Meaning
Wrap long lines	Specifies whether long error messages are wrapped.
Show only user relevant	Specifies whether all error messages or only message classified as "user relevant" are shown.
Show asynchronous calls	Specifies whether error messages in asynchronous method calls are shown.
Show crash report	Specifies whether error messages from the crash recorder are shown.

Related Topics

- [Enabling the Crash Recorder on page 635](#)

Displaying Messages in the Error Log View

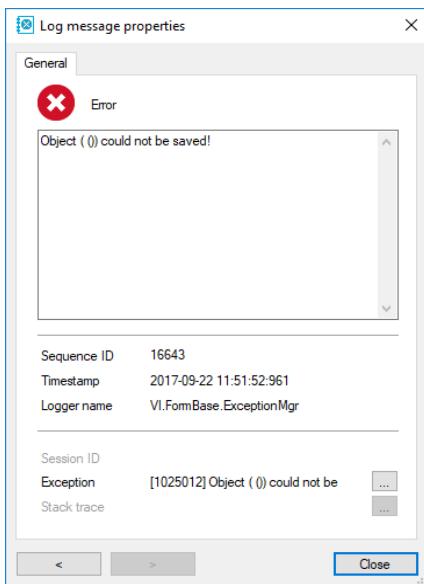
A program's error log, as in the Manager for example, collects all the messages, such as error messages and warnings, that have occurred since the program started. The error log is reinitialized when the program is restarted.

To display items from error log

1. In the Manager, go to **View | Error log**.
2. Enable the  view in the error log toolbar.

 **TIP:** Double-click a message to open more details in a separate dialog box.

Figure 62: Detailed information about a message



You can configure how the messages are displayed in the error log. To do this, switch the error log to advanced mode by clicking on the right of the column headers. Here you have the possibility to debug individual actions.

Figure 63: Simple Error Log (above) and Advanced Error Log (below)

Table 356: Meaning of Icons in the Error Log

Icon	Meaning
!	Logs all critical error messages. (Severity level = Fatal)
!	Logs all information. (Severity level = Info)
!	Logs all warnings. (Severity level = Warning)
!	Logs all error messages. (Severity level = error)
!	Logs debugger output. This setting should only be used for testing. (Severity level = Debug)

Icon	Meaning
	Logs highly detailed information. This setting should only be used for analysis purposes. The log file quickly becomes large and cumbersome. (Severity level = Trace)
	Adds a custom filter condition.
	Deletes filter condition.
	Searches for term.
	Searches next term.
	Marks all messages with a specific term.
Buffer size	Sets the message buffer size. The buffer's level is displayed next to the text box.
	Deletes the buffer contents.
	Stops logging.
	Starts logging.
	Saves log to file.
	Specifies which column are displayed in the error log.
	Copies selected messages to the clipboard.
	Opens the error log with a text editor.

The following information is displayed about a message. The range of information depends on the severity level of a message.

Table 357: Information about a message

Detail	Description
Severity level	Level of information supplied for the message.
Timestamp	Time and date of the log entry.
Logger name	One Identity Manager component from which the message was sent.
Message	Logged message.
Error Message	Detailed error message.
Data	Additional data about the message.
Sequence ID	Number of the line in the error log.
Stack trace	Complete stack trace for the error message.
Session ID	Session identification number.

Detail	Description
 NOTE:	If there is a filter set on the session ID, only the messages for this session are displayed, for example, loading collections and single objects. If no filter is set, action outside the connection, for example, loading table definitions or configuration parameters are displayed.
 TIP:	You can apply different filters to limit the information being displayed. Click the arrow in the column header and select a filter. The  icon in the log toolbar shows whether a filter is active.

Related Topics

- [Logging Messages using NLog on page 634](#)

Logging Messages in the Database Journal

Table 358: Configuration Parameters for Logging in the Database Journal

Configuration parameter	Meaning
Common\Journal	General parameter for configuring the database journal.
Common\Journal\LifeTime	Use this configuration parameter to specify the maximum amount of time (in days) that a database journal entry can be stored in the database. Older entries are deleted from the database.
Common\Journal\LifeTime\D	This configuration parameter contains retention period (in days) for entries with type 'Debug'.
Common\Journal\LifeTime\E	This configuration parameter contains retention period (in days) for entries with type 'Error'.
Common\Journal\LifeTime\I	This configuration parameter contains retention period (in days) for entries with type 'Info'.
Common\Journal\LifeTime\T	This configuration parameter contains retention period (in days) for entries with type 'Trace'.
Common\Journal\LifeTime\W	This configuration parameter contains retention period (in days) for entries with type 'Warning'.

Configuration parameter	Meaning
Common\Journal\LoginAudit	Logs successful One Identity Manager logins.
Common\Journal\Delete	This configuration parameter allows configuration of deletion behavior for system messages.
Common\Journal\Delete\BulkCount	This configuration parameter contains the number of entries to be deleted in an operation.
Common\Journal\Delete\TotalCount	This configuration parameter contains the total number of entries to be deleted in one processing run.

The database journal is used to store information, warning and error messages from different components of One Identity Manager, for example, DBQueue Processor, Configuration Wizard or One Identity Manager Service. Actions in the program "Job Queue Info", such as re-enabling process steps, are also written to the database journal.

To log error in process handing in the database journal

- Set the option **Log errors to journal** in the process steps.
 1. Open the process in the Process Editor.
 2. Click on the element for the process step in the process document.
 3. Set the option **Log errors to journal** on the "Error handling" tab

To log One Identity Manager successful logins

- Set the configuration parameter "Common\Journal\LoginAudit" in the Designer.

To delete log entries in the database journal

- Set the configuration parameter "Common\Journal\LifeTime" in the Designer and enter the maximum retention time for entries in the database journal. Use the configuration sub parameters to specify the retention period for each warning level.
- If there is a large amount of data, you can specify the number of objects to delete per DBQueue Processor operation and run in order to improve performance. Use the configuration parameter "Common\Journal\Delete\BulkCount" and "Common\Journal\Delete\TotalCount" to do this.
- Configure and set the schedule "Delete journal" in the Designer.

Related Topics

- [Creating and Editing Process Steps on page 399](#)
- [Monitoring Process Handling with Job Queue Info on page 611](#)
- [Displaying Messages in the Database Journal on page 624](#)
- [Globallog.config on page 655](#)

Writing the One Identity Manager Log

One Identity Manager provides various options for extending its log. The log can be configured for each One Identity Manager component.

Detailed information about this topic

- [Logging Messages using NLog on page 634](#)
- [Enabling the Crash Recorder on page 635](#)

Logging Messages using NLog

Each One Identity Manager component supports message logging using the integrated NLog functionality. For an exact description and functionality, see the online help (<http://nlog-project.org/>).

Setting for logging with Nlog are made in the "nlog" section in the One Identity Manager component's configuration files. Use the variable `appName` to pass the One Identity Manager component names. The configuration of the logs is defined in the global configuration file `globallog.config`. This file is referenced in the One Identity Manager component's configuration files.

In the One Identity Manager default installation, the log files are written to the directory `%LocalAppData%\One Identity\One Identity Manager\<appName>` under the name `<appName>.log`, where `appName` is the name of the One Identity Manager component. All messages with a minimum information level of "info", are recorded in the `<appName>.log`. The files are kept for 7 days and backed up daily. All messages with the information level "fatal" are additionally recorded in the event log for the source `One Identity Manager <appName>`.

Example of a Configuration File

```
<configuration>
    <configSections>
        ...
        <section name="nlog" type="NLog.Config.ConfigSectionHandler, NLog"/>
    </configSections>
    ...
    <nlog autoReload="true" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
        <variable name="appName" value="Manager"/>
        <include file="${basedir}/globallog.config" ignoreErrors="true"/>
    </nlog>
```

```
...
</configuration>
```

Related Topics

- [Globallog.config on page 655](#)
- [*.exe.config on page 658](#)

Enabling the Crash Recorder

The crash recorder saves the last 128 message from the "Debug" level and shows them in an error message window. The crash recorder is configured through the One Identity Manager tool's configuration file.

Example: Configuration file entry for a One Identity Manager component

```
<configuration>
    <configSections>
        ...
        <section name="connectionbehaviour" type="System.Configuration.
            NameValueSectionHandler" />
    </configSections>
    ...
    <appSettings>
        <add key="CrashRecorderBuffer" value="128" />
        <add key="CrashRecorderLevel" value="Error" />
    </appSettings>
    <connectionbehaviour>
        ...
    </connectionbehaviour>
    ...
</configuration>
```

If the variable CrashRecorderBuffer is set to the value 0, the crash record functionality is disabled. Permitted values for the CrashRecorderLevel are "Debug", "Error", "Fatal", "Info", "Off", "Trace" and "Warn".

Related Topics

- [Error Messaging in the Error Message Window on page 627](#)
- [*.exe.config on page 658](#)

One Identity Manager Service Logging

Success and error messages from process handling are written to the One Identity Manager Service log file. Messages can also be written to a server's event log. A severity level can be configured for output to this log file.

Detailed information about this topic

- [Configuring the One Identity Manager Service Log File on page 636](#)
- [Displaying the One Identity Manager Service Log File on page 638](#)
- [Extended Debugging in the One Identity Manager Service on page 640](#)
- [Output of Extended Return Values from Individual Process Components on page 641](#)
- [Outputting Custom Messages in the One Identity Manager Service Log File on page 641](#)
- [HTTPLogPlugins Log File on page 643](#)

Configuring the One Identity Manager Service Log File

To create a log file, modify the module "FileLogWriter" in the One Identity Manager Service configuration file for each One Identity Manager Service.

Following parameters are available:

- Log file (OutPutFile)

The parameter contains the name of the log file including its directory. Log information for the One Identity Manager Service is written to this file.

IMPORTANT: Ensure that the given directory exists. If the file cannot be created, no error output is possible. In this case, the error messages appear in the Windows event log or, under Linux, in /var/log/messages.

- Renaming interval for the log file (LogLifeTime)

In order to avoid unnecessarily large log files, the module supports the functionality of exchanging the log file with a history list. The LogLifeTime specifies the maximum life of a log file before it is renamed as backup. If the log file has reached its maximum age, the file is renamed (i.e. as JobService.log_20040819-083554) and a new log file is started.

Timeout format:

day.hour:minutes:seconds

- Process step log duration (JobLogLifeTime)

Use this parameter to specify the length of time process step logs are kept. After this expires, the logs are deleted.

Timeout format:

day.hour:minutes:seconds

For test purposes, you can enable logging of individual process steps in the Job Queue Info. The process step's processing messages with the NLog warning level "Debug" are written to a separate log. The files are stored in the log directory.

Repository structure:

```
<Protokollverzeichnis>\JobLogs\<first 4 digits of the UID_Job>\Job_<UID_Job>_<yyyymmdd>_<Timestamp>.log
```

- Max.number of archived log files (HistorySize)

This attribute limits the number of log files. If several log files exist, the oldest backup file is deleted when a new log file is created so that the limit is not exceeded.

- Max .log file size (MB) (MaxLogSize)

Use this parameter to specify the maximum size for the log file. Once the log file has reaches the limit, it is renamed into a backup file and a new log file is created.

- Max. length of the parameter (ParamMaxLength)

This parameter defines how many character can be in a job so that it is still written to the log file.

- Severity level (LogSeverity)

Specifies the warning level for logging messages. Only warnings and fatal errors are logged by default.

Table 359: Warning Levels for Logging

Severity level	Description
Info	All messages are written to the log file. The log file quickly becomes large and cumbersome.
Warning	Only warnings and exception errors are written to the log file (default).
Serious	Only exception errors are written to the log file.

Related Topics

- [Logging Messages in the Event View on page 642](#)
- [One Identity Manager Service Configuration on page 353](#)
- [Enabling Extended Logging of Process Steps on page 622](#)

Advanced Logging in the One Identity Manager Service

To implement advanced logging for the One Identity Manager Service, configure the log file's repository in the One Identity Manager Service configuration file in the module "Connection".

NOTE: The given directory must exist and the One Identity Manager Service user account must have write permissions to the directory.

The following parameters are available.

- Directory for generating logging (JobGenLogDir)

Log files are created in this directory that record process generation instructions from One Identity Manager Service.

Related Topics

- [The Connection Module on page 378](#)

Displaying the One Identity Manager Service Log File

The One Identity Manager Service log file can be displayed in a browser.

Prerequisites for Displaying the Log File

- The module "FileLogWriter" is configured in the One Identity Manager Service configuration file.
- A user must have the appropriate permissions in order to open an HTTP server. The administrator must grant URL approval to the user to do this. This can be executed with the following command line call:

```
netsh http add urlacl url=http://*:<port number>/ user=<domain>\<user name>
```

If the One Identity Manager Service has to run under the Network Service (NT Authority\NetworkService) user account, explicit permissions for the internal web service must be granted under Windows Server 2008 (R2). This can be executed with the following command line call:

```
netsh http add urlacl url=http://<IP address>:<port number>/ user="NT AUTHORITY\NETWORKSERVICE"
```

The result can also be verified using the following command line call:

```
netsh http show urlacl
```

- The port for displaying services is configured in the "Configuration" module in the

One Identity Manager Service configuration file. The default value is port 1880.

- An authentication method for displaying the log file must be set up.

Use the HTTP authentication module to specify authentication on HTTP servers to access services, for example, for displaying the log file or the status display.

The following module types may be selected:

- BasicHttpAuthentication

To access the HTTP server with this authentication type, enter a specific user account (user) and the associated password (password).

- WindowsHttpAuthentication

Use this authentication type to specify an Active Directory group, whose users can be authenticated on the HTTP server. Either an SID or the Active Directory group name can be entered into the Job server domain. If the Active Directory groups are not in the Job server domain, you must use the SID.

NOTE: If no model is given, no authentication is required. All users can access the service.

To display the One Identity Manager Service log file in a browser

- You call up the log file with the appropriate URL.

`http://<server name>:<port number>`

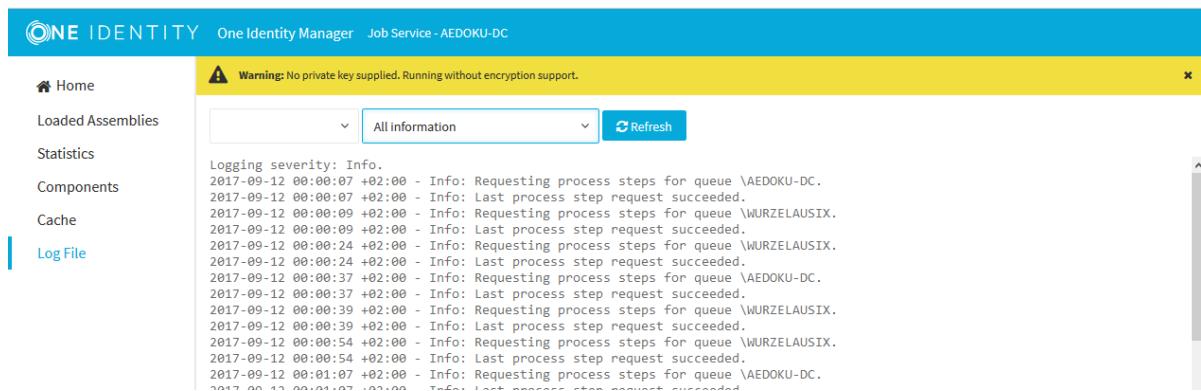
The default value is port 1880.

To open the One Identity Manager Service log file in Job Queue Info

- Select the Job server in the **Service status** view and select **Open in browser** in the context menu.

The One Identity Manager Service HTTP server for the Job server is queried and the varying One Identity Manager Service services are displayed.

Figure 64: One Identity Manager Service Log File



The messages to be displayed on the web page can be filter interactively. There is a menu on the website for this. Only text contained in the log file can be displayed in this case. If, for example, the message type is set to "Warning", no "Info" messages can be shown even if the appropriate filter is chosen.

The log output is color coded to make it easier to identify.

Table 360: Log File Color Code

Color	Meaning
Green	Processing successful.
Yellow	Warnings occurred during processing.
Red	Fatal errors occurred during processing.

NOTE: If you want to retain the color information to send by mail, you need to save the complete web page.

Related Topics

- [Monitoring Process Handling with Job Queue Info on page 611](#)
- [Configuring the One Identity Manager Service Log File on page 636](#)
- [The Configuration Module on page 373](#)
- [The HTTP Authentication Module on page 379](#)

Extended Debugging in the One Identity Manager Service

There are two parameters available in the One Identity Manager Service configuration module that you can use to extend debugging functionality:

- `DebugMode`
- Component debugging mode (`ComponentDebugMode`)

One Identity Manager Service writes more detailed data into the log file if the parameter "DebugMode" is set, for example, all parameters that are passed to a component as well as the processing results together with OUT parameters.

Individual One Identity Manager Service process components can output additional process data to the One Identity Manager Service log file. To do this you set the parameter "ComponentDebugMode" in the configuration module. You should only use "ComponentDebugMode" for localizing errors because the effect on performance means that it is not recommended for normal use.

Related Topics

- [The Configuration Module on page 373](#)
- [One Identity Manager Service Configuration on page 353](#)
- [One Identity Manager Configuration Files on page 655](#)

Output of Extended Return Values from Individual Process Components

Table 361: Configuration Parameter for Outputting Extended Return Values

Configuration parameter	Meaning
Common\Jobservice\DoReturnOutput	The entire output of the parameter is written to the One Identity Manager Service log file when a error occurs in the case of process task that supply an extended return value.

Individual process components have process tasks with parameters that supply extended return values (OUT). The entire output of the parameter is written to the One Identity Manager Service log file when a error occurs. For example, when a command or program is executed using the process component "CommandComponent", the output text for the command or program can be returned.

To log return values

- Set the configuration parameter "Common\Jobservice\DoReturnOutput" in the Designer.

Related Topics

- [Displaying OUT Parameters on page 621](#)

Outputting Custom Messages in the One Identity Manager Service Log File

You can use the script engine methods `RaiseMessage` and `AppData.Instance.RaiseMessage` from within process steps to write output messages to the One Identity Manager Service log file. Use the process component "ScriptComponent" to run the script.

Messages are marked in color in the log file depending on the severity level (parameter "MsgSeverity").

Figure 65: Example Output of Messages to a One Identity Manager Service Log File

```
2007-08-10 12:48:58 - Warning: Example warning message
2007-08-10 12:48:58 - Info: Example Info message
2007-08-10 12:48:58 - Serious: Example error message
```

RaiseMessage:

The output is consolidated with other messages and logged at the end of processing the process step.

Syntax:

```
RaiseMessage (MsgSeverity, "string")
```

Example:

```
RaiseMessage (MsgSeverity.Warning, "Example warning message")
```

```
RaiseMessage (MsgSeverity.Info, "Example Info message")
```

```
RaiseMessage (MsgSeverity.Serious, "Example error marked message")
```

AppData.Instance.RaiseMessage

This output is written immediately during processing; not taking into account the end of the process step.

Syntax:

```
AppData.Instance.RaiseMessage (MsgSeverity, "string")
```

Example:

```
AppData.Instance.RaiseMessage (MsgSeverity.Warning, "Example warning message")
```

```
AppData.Instance.RaiseMessage (MsgSeverity.Info, "Example Info message")
```

```
AppData.Instance.RaiseMessage (MsgSeverity.Serious, "Example error marked message")
```

For more examples of One Identity Manager Service log file output, see the script example on the installation medium in the directory QBM\dvd\AddOn\SDK\ScriptSamples.

IMPORTANT: You should never use the VB.Net functions MsgBox and InputBox on servers. Use the functions VID_Write2Log, RaiseMessage or AppData.Instance.RaiseMessage.

Logging Messages in the Event View

To write One Identity Manager Service messages in the server's event view, the module "EventLogLogWriter" has to be modified in the One Identity Manager Service configuration file.

Following parameters are available:

- Event log (EventLog)

Enter the name for the event log to which the messages should be written. The messages are written to the application log if the default value "Application" is used.

NOTE: If several One Identity Manager Service write event logs on a server, ensure that the first 8 letters of the log name are unique.

- Severity level (LogSeverity)

Specifies the warning level for logging messages. By default, only warnings and serious errors are logged.

Table 362: Warning Levels for Logging

Severity level	Description
Info	All messages are written to the event log. The event log quickly becomes large and confusing.
Warning	Only warnings and exception errors are written to the event log (default).
Serious	Only exception errors are written to the event log (exceptions).

- Event ID

Define an ID with which messages are written in the event log.

- Category

Define a category with which messages are written to the event log.

- Source

Define a name for the source with which messages are written to the event log.

Process handling error can also be written to a server's result log. To do this use the process component "LogComponent".

Related Topics

- [EventLogLogWriter on page 374](#)
- [One Identity Manager Service Configuration on page 353](#)

HTTPLogPlugins Log File

If the HTTPLogPlugin is configured in the One Identity Manager Service, a log file with HTTP is created with One Identity Manager Service requests. The file is written in Apache HTTP Server Combined Log Format.

Example Entry

```
172.19.2.18 - - [03/Feb/2005:14:55:48 +0100] "GET /resources/JobService.css HTTP/1.x"
OK - "http://<server name>:<port>/status/LogWriter/Config""Mozilla/5.0 (Windows; U;
5.1; en-US; rv:1.7.5) Gecko/20041108Firefox/1.0"
```

Table 363: Meaning of each Entry

Entry	Meaning
172.19.2.18	IP address before the request.
-	Client user name using IDENT protocol (RFC 1413)-
-	Client user name conforming to HHTP authentication
[03/Feb/2005:14:55:48 +0100]	Time that the request is processed on the server
GET /resources/JobService.css HTTP/1.x"	Request
OK	Status code-
-	Size of data sent back to the browser
"http://<server name>:<-port>/status/LogWriter/Config"	URL from which the page can be accessed
"Mozilla/5.0 (Windows; U; Windows NT 5.1; de-DE; rv:1.7.5) Gecko/20041108Firefox/1.0"	Browser name

Related Topics

- [HTTPLogPlugin on page 380](#)
- [One Identity Manager Service Configuration on page 353](#)

Displaying One Identity Manager Application Server Status

The application server can be reached over a browser under:

`http://<server>/<application name>`

`https://<server>/<application name>`

TIP: You can open the web server's status display in Job Queue Info. Select the menu item **View | Server state** in the Job Queue Info and display the web server's state on the **Web servers** tab by using **Open in browser** in the context menu.

You will see different status information. Status information for the application server are displayed as performance counters.

In addition, API documentation is available here.

Processing DBQueue Tasks

The tasks queued in the DBQueue are the result of triggering, modifications to configuration parameters (for example, changes to a configuration parameter concerning inheritance) or executing scheduled tasks. The DBQueue Processor processes tasks in the DBQueue. The DBQueue Processor uses several slots for executing tasks in parallel.

Detailed information about this topic

- [Configuring the DBQueue Processor for Test and Development on page 645](#)
- [Initializing the DBQueue Processor on page 646](#)
- [Controlling Processing of DBQueue Processor Tasks on page 648](#)
- [Processing DBQueue Processor Tasks on page 649](#)
- [Reactivating DBQueue Processor Tasks on page 650](#)
- [Bulk Processing in DBQueue Processor on page 651](#)
- [How the Central Dispatcher Communicates with Individual Slots on page 651](#)
- [Scheduled Maintenance Tasks on page 653](#)

Configuring the DBQueue Processor for Test and Development

DBQueue ProcessorA number of Use the staging level of the One Identity Manager database to specify whether a test database, development database or a live database is being dealt with. configuration settings are controlled by the staging level. If you modify the database staging level the configuration settings are changed.

To modify a database staging level

1. Select the category **Base Data | General | Databases** in the Designer.
2. Select the database and change the value of the property **Staging level** to "Test environment", "Development system" or "Development system".
3. Select the **Database | Commit to database...** and click **Save**.

Table 364: DBQueue Processor Database Settings for Development, Test and Live Environments

Setting	Database Staging Level		
	Development Environment	Test Environment	Live Environment
Maximum DBQueue Processor runtime	20 minutes	40 minutes	120 minutes
Maximum number of slots for DBQueue Processor	3	5	Maximum number of slots according to the hardware configuration

The DBQueue Processor configuration settings are configured for normal operations and must not be modified normally. The number of configuration settings is reduced in the case of test and development environments because there may be more databases on one server.

If you have to change the settings for test or development environments on performance grounds, you must modify the following configuration parameter settings in the Designer.

Table 365: Configuration Parameters for the DBQueue Processor

Configuration parameter	Meaning
QBM\DBQueue\CountSlotsMax	This configuration parameter specifies the number of maximum slots available. Enter the value 0 to use the maximum number of slots according to the hardware configuration.
QBM\DBQueue\KeepAlive	This configuration parameter regulates the maximum runtime of the central dispatcher. Tasks on slots currently in use are still processed when the timeout expires. Then the slot database schedules are stopped and the central dispatches exits. The lowest permitted value for runtime is 5 minutes; the highest value is 720 minutes.

Related Topics

- [Database Connection Properties](#)

Initializing the DBQueue Processor

IMPORTANT: Do not change or delete predefined database schedules as it may lead to unexpected errors.

DBQueue Processor initialization takes place once during schema installation. The following database schedules are generated during the initialization phase:

- QBM_PWatchDog on <database>

This database schedule assumes several functions in the One Identity Manager.

- It checks whether DBQueue Processor's central dispatcher is active and restarts it.
- It starts a database schedule to remove complete processes from the DBQueue.
- It controls validation and starting of schedules.
- It checks at regular intervals, whether database single-user mode is still required and resets the setting if necessary.

This database schedule has an active schedule with a 1 minute interval.

- QBM_PDBQueueProcess_Main on <database>

This database schedule is the DBQueue Processor's central dispatcher. The central dispatcher assumes control of processing and distributes DBQueue tasks to individual slots. Each time the central dispatcher is executed, the number of currently available slots required for the current run is found. The central dispatcher starts database schedules for the currently available slots just once.

Only one database schedule at most is started for the central dispatcher. The central dispatcher's database schedule does not have an active schedule, but is started by the database schedule QBM_PWatchDog on <database>.

- QBM_PDBQueueProcess<SlotNumber> on <database>

The maximum number of available slots is determined during the DBQueue Processor initialization phase. The maximum number of slots depends on the number of processors on the database server. An associated database schedule QBM_PDBQueueProcess<SlotNumber> on <database> is set up for each slot. Each database schedule is set up with a process which executes the DBQueue tasks for exactly this slot. The database schedules associated to each slot do not have any active schedules. They are started by the central dispatcher.

- QBM_PDBQueueProcess_Del on database

This database schedule removes processed DBQueue tasks. The database schedule does not have an active schedule, but is started through the database schedule QBM_PWatchDog on <database>.

Related Topics

- [DBQueue Processor Reinitialization on page 648](#)

DBQueue Processor Reinitialization

- You must execute the procedure QBM_PDBQueuePrepare once manually when the server hardware has been extended or custom DBQueue Processor tasks have been created.
- You must execute the procedures QBM_PDBQueuePrepare and QBM_PWatchDogPrepare once when you set up a reference database for test and development.

Run the following procedures once in the reference database using a suitable query tool.

SQL Server

```
exec QBM_PWatchDogPrepare  
exec QBM_PDBQueuePrepare 0,1
```

Oracle Database

```
exec QBM_GWATCHDOG.PPREPARE(0);  
exec QBM_GDBQUEUE.PDBQUEUEPREPARE(0);
```

Controlling Processing of DBQueue Processor Tasks

The database schedule for the central dispatcher is started with the database schedule QBM_PwatchDog on <database>. The central dispatcher assumes control of processing and distributes DBQueue tasks to individual slots.

Each time the central dispatcher is executed, the number of currently available slots required for the current run is found first. The more load there is on the database, the less slots there are to use. At least two slots are used.

The number of currently available slots results from:

The number of currently available slots = maximum number of available slots - sum of all own database processes - sum of processes of other databases on the server

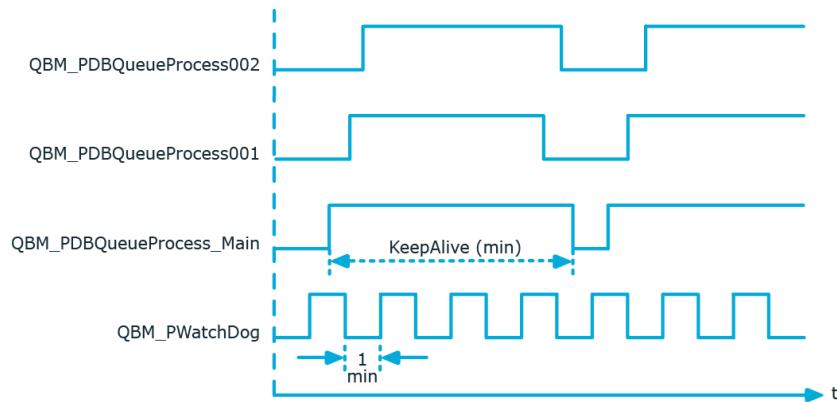
The central dispatcher starts database schedules for the currently available slots just once. Each database schedule is set up with a process , which executes tasks for exactly this slot.

Once tasks in the DBQueue are entered, the central dispatcher is notified. The central dispatcher distributes tasks to individual slots and notifies the slot processes that there are tasks waiting to be processed. Each process processes the tasks queued for its slot. Once the task is complete, each process sends a message to the central dispatcher and waits for new tasks.

The central dispatcher checks at defined intervals whether the slots are still active and distributes new tasks to them. If there are no more tasks in the DBQueue, the central dispatcher goes into a wait state and waits for new task notifications.

Tasks on slots currently in use are still processed when the timeout expires. Then the slot database schedules are stopped and the central dispatches exits. [For more information, see How the Central Dispatcher Communicates with Individual Slots on page 651.](#)

Figure 66: Controlling Processing



Processing DBQueue Processor Tasks

The central dispatcher finds entries in the DBQueue (table DialogDBQueue) and moves the tasks into the table QBMDBQueueCurrent with the assignment tasks per slot.

Example of Entries in the Tables DialogDBQueue and QBMDBQueueCurrent

Table 366: Entries in the table DialogDBQueue (extract)

Task name	Object
OrgRoot	A
OrgRoot	B
ADSAccountInADSGroup	X
ADSAccountInADSGroup	Y
ADSAccountInADSGroup	Z

Table 367: Entries in the table QBMDBQueueCurrent (extract)

Slot number	Task name	Object
001	OrgRoot	A
001	OrgRoot	B

Slot number	Task name	Object
002	ADSAccountInADSGroup	X
002	ADSAccountInADSGroup	Y
002	ADSAccountInADSGroup	Z

Each process processes tasks queued for its own slot in the table QBMDBQueueCurrent. Subsequent tasks resulting from processing are queued in the table DialogDBQueue.

When a process has handled its tasks and no new task are pending, process sets the slot number itself to "0" in the table QBMDBQueueCurrent. The entry initially remains in the table QBMDBQueueCurrent but is no longer taken into account (because slot 0 is not active).

The database schedule QBM_PDBQueueProcess_Del on <database> deletes all entries with slot number "0" from the table QBMDBQueueCurrent in regular intervals.

Table 368: Meaning of Slot Numbers in the Table QBMDBQueueCurrent

Slot number	Meaning
001 - n	Number of slot to be processed by the task.
0	State after the task is completed correctly.
-1	An error occurred during task processing or processing was deferred, for example, because synchronization is running. The central dispatcher re-enables the task.
-2	An error occurred during task processing or processing was deferred, for example, because of blocking. The central dispatcher re-enables the task.
-3	An error occurred during task processing or processing was deferred, for example, because there are still entries in the Job queue. The central dispatcher re-enables the task.

Related Topics

- [Reactivating DBQueue Processor Tasks on page 650](#)

Reactivating DBQueue Processor Tasks

If a task has to be deferred, for example due to an error or if synchronization is running, the process sets the slot number in the table QBMDBQueueCurrent to "-1" on its own. These tasks are re-enabled if there are no more tasks in the DBQueue. At the very latest, these deferred tasks are reinstated into the DBQueue the next time the central dispatcher runs. This means deferred tasks are re-enabled at the latest once the maximum runtime has elapsed.

 **NOTE:** Deferring DBQueue tasks is recorded in the system log.

Related Topics

- [Logging Messages in the Database Journal on page 632](#)

Bulk Processing in DBQueue Processor

Table 369: Configuration Parameter for Bulk Processing in the DBQueue Processor

Configuration parameter	Meaning
QBM\DBQueue\DefaultRuntime	The configuration parameter specifies how long the DBQueue Processor run. The default value is 90 seconds. Some DBQueue Processor procedures are marked for bulk processing to reduce the total time required for processing DBQueue tasks. If a lot of entries are marked for bulk processing in the DBQueue, the DBQueue Processor switches from single to bulk processing. There is a mechanism implemented that is used to decide whether switching to bulk processing as opposed to single processing would result in time savings. To do this, 25 single task processes are run and the processing time is recorded. All other entries for the task are processed in bulk and the minimum and maximum load time required for advantageous bulk processing is defined. A self optimizing calculation procedure updates the load times. As a result of this method, the DBQueue Processor needs to "kick in" after initial schema installation or after system changes, such as increased database memory. You can specify how long the DBQueue Processor runtime interval should be with the configuration parameter "Common\DBQueue\DefaultRuntime". The default value is 90 seconds. This corresponds to the time period that achieves the best load for the calculation procedure.

How the Central Dispatcher Communicates with Individual Slots

The table QBMDQueueSlot is responsible for communication of the central dispatcher with individual slots. The maximum number of slots available is determined during DBQueue Processor initialization. One entry per slot is created in the table QBMDQueueSlot. The table contains information about each slot and its status as well as currently running tasks.

Table 370: Meaning of Status in Table QBMDBQueueSlot

Status	Meaning
0	No activity required. Initial state (through initialization) or end state (set by process).
1	The process is triggered to prepared central temporary tables, for example.
2	Ready for operation. The process has started but the currently no tasks exist. This is the state in which tasks can be queued.
3	Transfer to table QBMDBQueueCurrent. The process has received tasks for processing and needs to begin.
4	The process has recognized the tasks and added them.
5	The process is handling the tasks.
-1	The process was prompted to quit. Stop behavior by maximum timeout or abort on process error.

Related Topics

- [Communication during Processing on page 652](#)
- [Configuring the DBQueue Processor for Test and Development on page 645](#)

Communication during Processing

The following example show the entries in the table QBMDBQueueSlot during processing.

- Slot initialization

Slot number	Status	Task name
001	0	

- Starts the process using the central dispatcher.

Slot number	Status	Task name
001	1	

- The process is operational. Preparations, for example, for temporary tables, are complete. The slot status is regularly tested.

Slot number	Status	Task name
001	2	

- The central dispatcher distributes tasks. The central dispatcher checks slots for readiness and enters the task from table DialogDBQueue in the table QBMDBQueueCurrent with the slot number. The status of each slots is updated once the table QBMDBQueueCurrent has taken over.

Slot number	Status	Name of task
001	3	OrgRoot

- The process recognizes a task on the basis of the status, starts processing and updates its slots' status.

Slot number	Status	Task name
001	4	OrgRoot

- The process has completed processing and set the slot number in the table DialogDBQueueCurrent to "0". The process changes the status of its slots to operational.

Slot number	Status	Name of task
001	2	

Stop behavior by maximum timeout

Once the maximum runtime has expired, the tasks of slots in the table QBMDBQueueCurrent currently in use are still processed. No new tasks are added from the table QBMDBQueue. All slots with a slot status of "2" are set to "-1" in the table QBMDBQueueSlot. This prompts the processes to finish and stop themselves. The central dispatcher checks whether all processes have completed and have stopped.

Scheduled Maintenance Tasks

Some calculation tasks for the DBQueue Processor are scheduled. There are schedules set up for these maintenance tasks, which you can customize as required. It is recommended to run maintenance task outside main working hours of the connected clients.

Table 371: DBQueue Processor Maintenance Tasks

Task	Schedule	Execution
Reduce size of change entries	Reduce logs	Daily
Reduce size of process tracking logs	Reduce logs	Daily
Purge dynamic users	Reduce logs	Daily
Reduce size of process log entries	Reduce logs	Daily
Reduce size of process history	Reduce logs	Daily
Reduce size of database journal	Reduce logs	Daily
Populate calendar	Daily maintenance tasks	Daily
Lock table statistics	Daily maintenance tasks	Daily
Calculate table statistics	Daily maintenance tasks	Daily
Calculate statistics for data contents	Weekly maintenance tasks	Weekly

Related Topics

- [Setting Up and Configuring Schedules on page 134](#)

One Identity Manager Configuration Files

General configuration settings can be preset in a configuration file. The configuration file is kept in the program directory. Each administration tool can take its settings from a configuration file in NET executable format. Valid global configuration settings can also be defined through a configuration file in One Identity Manager's own format.

Detailed information about this topic

- [Globallog.config on page 655](#)
- [*.exe.config on page 658](#)
- [Global.cfg on page 659](#)
- [Jobservice.cfg on page 660](#)
- [viNetworkService.exe.config on page 662](#)

Globallog.config

Configuration setting for logging messages are made by NLog in Globallog.config. Globallog.config is referenced in the One Identity Manager component's configuration files.

IMPORTANT: The settings for globallog.config are global for all One Identity Manager components. Use the application specific configuration file *.exe.config to customize individual components.

Use variables to define names, output path and layout of the log files. Output destinations for the messages are specified in the "targets" section. NLog already has predefined targets that you can use in the configuration file. Use the "rules" section to specify rules for logging messages. For an exact description and functionality of NLog, see the online help (<http://nlog-project.org/>).

In the One Identity Manager default installation, the log files are written to the directory %LocalAppData%\One Identity\One Identity Manager\<appName> under the name

`<appName>.log`, where `appName` is the name of the One Identity Manager component. All messages with a minimum information level of "info", are recorded in the `<appName>.log`. The files are kept for 7 days and backed up daily. All messages with the information level "fatal" are additionally recorded in the event log for the source One IdentityOne Identity Manager `<appName>`.

The variable `appName` is defined in the One Identity Manager component's configuration files.

Example of file structure

```
<nlog autoReload="true" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <variable name="companyName" value="One Identity"/>
    <variable name="productTitle" value="One Identity Manager"/>
    <variable name="logBaseDir"
        value="${specialfolder:LocalApplicationData}/${companyName}/${productTitle}/${ap
        pName}"/>
    <variable name="layout" value="${longdate} ${level:upperCase=true} (${logger}
        ${event-context:item=SessionId}) : ${event-context:item=Indentation}${message}
        ${exception:format=ToString,StackTrace}" />
    <targets async="true">
        <default-wrapper xsi:type="BufferingWrapper" bufferSize="256"
            flushTimeout="2000" />
        <target name="logfile" xsi:type="File"
            fileName="${logBaseDir}/${appName}.log" layout="${layout}" encoding="utf-8"
            archiveFileName="${logBaseDir}/${appName}.#{}.log" maxArchiveFiles="7"
            archiveEvery="Day" archiveNumbering="Rolling"/>
    </targets>
    <targets>
        <target name="eventLog" xsi:type="EventLog" source="${companyName}
            ${productTitle} ${appName}"
            layout="${message}${newline}${exception:format=toString}"/>
    </targets>
    <rules>
        <logger name="*" minlevel="Info" writeTo="logfile"/>
        <logger name="*" level="Fatal" writeTo="eventLog"/>
    </rules>
</nlog>
```

You can enter the severity level through:

- `minlevel`= Messages are logged from this severity level.
- `level`= Message are logged which have exactly this severity level.

Table 372: Permitted Severity Levels

Severity Level	Description
Trace	Logs highly detailed information. This setting should only be used for analysis purposes. The log file quickly becomes large and cumbersome.
Debug	Logs debug steps. This setting should only be used for testing.
Info	Logs all information.
Warning	Logs all warnings.
Error	Logs all error messages.
Fatal	Logs all critical error messages.

By providing logger name, you specify for which One Identity Manager components messages are logged. Messages are logged for all components with the default setting logger name="*". To limit logs to certain components, use the name contained in the log.

Table 373: Component Logger Name

Logger name	Description
FrontendLog	Logs actions in front-ends.
JobGenLog	Logs during process generation.
Jobservice	Logs One Identity Manager Service messages.
ObjectLog	Logs object actions through the object level.
ProjectorEngine	Logs messages from the synchronization engine.
SqlLog	Logs database queries
StopWatch	Logs timings.
SyncLog	Logs actions within synchronization.
SystemConnection	Detailed logging of data communication with the system connection during synchronization, including system configuration and system connectors' data communication.
SystemConnector	Logs system connector data communication during synchronization.
Update	Logs update handling.
WebLog	Logs Web service actions.

Related Topics

- [Logging Messages using NLog on page 634](#)
- [*.exe.config on page 658](#)

*.exe.config

- NOTE:** Use the configuration file globallog.config for global setting that apply to all One Identity Manager components.

The One Identity Manager components, for example Manager or Designer, have a configuration file for .NET executable's with a predefined format for this. The text is case sensitive. There is a configuration section in the file for each of the different modules of a One Identity Manager component.

The root in the XML file is always called configuration. All other sections of the configuration file must be in the mandatory section configSections and their type must be defined.

Format of the Configuration File using .exe.config as an Example

```
<?xml version="1.0" encoding="utf-8" ?>
<configuration>
    <configSections>
        <section name="formprovider"
            type="System.Configuration.NameValueSectionHandler" />
        <section name="formarchives"
            type="System.Configuration.NameValueSectionHandler" />
        <section name="vicontrols"
            type="System.Configuration.NameValueSectionHandler" />
        <section name="connectionbehaviour"
            type="System.Configuration.NameValueSectionHandler" />
        <section name="dialogplugins"
            type="System.Configuration.NameValueSectionHandler" />
        <section name="consistencychecks"
            type="System.Configuration.NameValueSectionHandler" />
        <section name="nlog" type="NLog.Config.ConfigSectionHandler, NLog"/>
    </configSections>
    <dialogplugins>
        <add key="ComplianceRuleSimulation"
            value="VI.DialogEngine.Plugins.ComplianceRuleSimulation,
            AE.DialogEngine.Plugins" />
        <add key="ComplianceRuleSimulationSummary"
            value="VI.DialogEngine.Plugins.ComplianceRuleSimulationSummary,
            AE.DialogEngine.Plugins" />
    </dialogplugins>
    <consistencychecks>
```

```

<add key="AE" value="VI.ConsistencyChecks.AE.dll" />
<add key="Common" value="VI.ConsistencyChecks.Common.dll" />
</consistencychecks>
<formarchives>
<add key="Forms" value="archive:.\\???.Forms*.vif;10" />
<add key="CustomForms" value="archive:.\\AE.CustomForms.*.vif;5" />
<add key="CommonForms" value="archive:.\\Common.Forms*.vif;5" />
</formarchives>
<vicontrols>
<add key="defaultcontroldesign" value="System" />
</vicontrols>
<nlog autoReload="true" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
<variable name="appName" value="Manager"/>
<include file="${basedir}/globallog.config" ignoreErrors="true"/>
</nlog>
</configuration>

```

Related Topics

- [Logging Messages using NLog on page 634](#)
- [Globallog.config on page 655](#)
- [Global.cfg on page 659](#)

Global.cfg

The Global.cfg is an XML configuration file in One Identity Manager's own simplified format. The advantage of this file is that run-time loading is supported. The text is case sensitive. Each of the different modules has its own section allocated within the file.

You can find an example of a configuration file on the installation medium in directory QBM\dvd\AddOn\SDK\ConfigSample. If the file Global.cfg is in the program directory, it is used when the One Identity Manager tools start up.

The root in the XML file is always called configuration. Each configuration file module and its values are defined in a section category respectively.

 **NOTE:** Both the sections and the names of the values must be written in lower case.

Format of Global.cfg

```
<configuration>
```

```

<category name="settings">
    <value name="language">English</value>
    <value name="autoupdateenabled">true</value>
    <value name="connectiontimeout">15</value>
</category>
<category name="connections">
    <value name="database display 1">ConnectionString</value>
    <value name="database display 2">ConnectionString</value>
</category>
</configuration>

```

TIP: Use the program "Config Encryptor" to create the connection string. You will find this program on the installation medium in the directory QBM\dvd\AddOn\ConfigEncryptor.

Related Topics

- [*.exe.config on page 658](#)

One Identity Manager Service Configuration Files

One Identity Manager Service is configured using a configuration file. The configuration file has to be in the same directory as the viNetworkService.exe. Two configuration files are supported:

Detailed information about this topic

- [Jobservice.cfg on page 660](#)
- [viNetworkService.exe.config on page 662](#)

Jobservice.cfg

The file Jobservice.cfg is an XML configuration file in One Identity Manager's own simplified format. The advantage of this file is that run-time loading is supported. The text is case sensitive. There is a configuration section in the file for each of the different modules in the One Identity Manager Service.

The root in the XML file is always called configuration. Each configuration file module and its values are defined in a section category respectively. At the moment the program only supports the section type "System.Configuration.NameValueSectionHandler".

 **NOTE:** Both the sections and the names of the values must be written in lower case.

```
<configuration>
    <category name="serviceconfiguration">
        <value name="jobprovider">VI.JobService.MSSqlJobProvider, jobservice</value>
        <value name="HttpPort">1180</value>
        <value name="logwriter">VI.JobService.FileLogWriter, jobservice</value>
    </category>
</configuration>
```

Example for a simple configuration with:

- direct connection to an SQL Server
- only one job destination (JobProcessor)

```
<configuration>
    <category name="serviceconfiguration">
        <value name="jobprovider">VI.JobService.MSSqlJobProvider, jobservice</value>
        <value name="logwriter">VI.JobService.FileLogWriter, jobservice</value>
    </category>
    <category name="sqlprovider">
        <value name="connectstring">User ID=sa;initial Catalog=<Database>;Data
            Source=<SQL-Server>;Password=<Password></value>
    </category>
    <category name="filelogwriter">
        <value name="loglifetime">0.01:00:00</value>
        <value name="logseverity">Info</value>
    </category>
    <category name="dispatcher" />
    <category name="jobdestinations">
        <value name="queuemx">VI.JobService.JobServiceDestination, jobservice</value>
    </category>
    <category name="queuemx">
        <value name="queue">\%COMPUTERNAME%</value>
    </category>
</configuration>
```

Related Topics

- [viNetworkService.exe.config on page 662](#)

viNetworkService.exe.config

The viNetworkService.exe.config file is the default configuration file for .NET exes and has the specified format. The text is case sensitive. There is a configuration section in the file for each of the different modules in the One Identity Manager Service.

The root in the XML file is always called configuration. All other sections of the configuration file must be in the mandatory section configSections and their type must be defined. At the moment the program only supports the section type System.Configuration.NameValueSectionHandler.

```
<configuration>
  <configSections>
    <section name="sectionname"
      type="System.Configuration.NameValueSectionHandler" />
  </configSections>
  <sectionname>
    ...
  </sectionname>
</configuration>
```

Example for a simple configuration with:

- direct connection to an SQL Server
- only one job destination (JobProcessor)

```
<configuration>
  <configSections>
    <section name="serviceconfiguration"
      type="System.Configuration.NameValueSectionHandler" />
    <section name="sqlprovider"
      type="System.Configuration.NameValueSectionHandler" />
    <section name="filelogwriter"
      type="System.Configuration.NameValueSectionHandler" />
    <section name="dispatcher"
      type="System.Configuration.NameValueSectionHandler" />
    <section name="jobdestinations"
      type="System.Configuration.NameValueSectionHandler" />
```

```

<section name="queueX" type="System.Configuration.NameValueSectionHandler"
/>
<section name="plugins"
type="System.Configuration.NameValueSectionHandler" />
</configSections>
<serviceconfiguration>
<add key="jobprovider" value="VI.JobService.MSSqlJobProvider,jobservice" />
<add key="logwriter" value="VI.JobService.FileLogWriter,jobservice" />
</serviceconfiguration>
<sqlprovider>
<add key="ConnectionString" value="User ID=sa;initial Catalog=<Database>;Data
Source=<SQL-Server>;Password=<Password>" />
</sqlprovider>
<filelogwriter>
<add key="LogLifeTime" value="0.01:00:00" />
<add key="LogSeverity" value="Info" />
</filelogwriter>
<dispatcher />
<jobdestinations>
<add key="QueueX" value="VI.JobService.JobServiceDestination,jobservice" />
</jobdestinations>
<queueX>
<add key="queue" value="\%COMPUTERNAME%" />
</queueX>
</queueX>
</configuration>

```

Related Topics

- [Jobservice.cfg on page 660](#)

Contacting us

For sales or other inquiries, visit <https://www.oneidentity.com/company/contact-us.aspx> or call +1-800-306-9329.

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

Index

#

#LD notation 464

\$

\$ notation 455

 data type 455

*

.CustomForms..vif 296

*.exe.config 655, 658

.Forms..vif 282

A

application server 19

 status display 644

AppServerJobProvider 364

 AuthenticationString 364

 ConnectionString 364

 RequestQueueLimit 364

 ResultQueueLimit 364

assignment form 287

 configuration data 290

authentication module

 account-based system user 74

 Active Directory user account 74

 Active Directory user account
 (dynamic) 74

 Active Directory user account
 (manual entry/role based) 74

 Active Directory user account
 (manual) 74

Active Directory user account (role
 based) 74

component authenticator 74

Crawler 74

default 113

employee 74

employee (dynamic) 74

employee (role based) 74

generic single sign-on (role
 based) 74

HTTP header 74

HTTP header (role based) 74

initial data 104

LDAP user account (dynamic) 74

LDAP user account (role based) 74

OAuth 2.0/OpenID Connect 74

OAuth 2.0/OpenID Connect (rollen-
 basiert) 74

share 103

synchronization authenticator 74

user account 74

User Account (role based) 74

Web Agent authenticator 74

C

calculation schedule

 configure 134

 enable 134

 execution interval 134

 set up 134

 start immediately 134

table 134
time of execution 135
time zone 134
validity period 134
change label 69
 assign 72
 create 70
 edit 70
 transport 550
collection 22
column
 #LD content 330
 alternative primary key 189
 assign to employee 205
 base column 189
 data type 189
 default configuration
 change 55
 restore 55
 default value 189
 display name 189
 dynamic foreign key 189, 204
 edit 188
 allow 55
 lock 55
 encrypted 189
 export for SPML 189, 607
 foreign key 189
 format 189, 200
 formatting script 201
 formatting types 200
 group 189
 hierarchy data 189
 index weighting 189, 206
 language dependent 330
length 189
log 189
mapping 188
multiline 189
multilingual 189, 330
MVP column 189
no DB transport 189
number of decimal places 189
permitted value 202
preprocessor condition 189
primary key 189
properties 188
proxy view 189
recipient 195
recursive key 189
sender 195
sort order 189
syntax 189
table 189
template 189, 195
text buffer 330
threshold 189
translate 330
translation source 330
translation target 330
value list 189, 202
column relation 208
combined log format 643
compile
 database 64
 error message 66
 warning 66
compiling
 conditional 448
 error message 415

configuration file 356
configuration parameter
 create 119
 disable 120
 display name 120
 edit 119
 enable 119-120
 encrypted 120
 option 121
 preprocessor condition 449
 preprocessor Expression 121, 449
 preprocessor relevant 120
 value 119-120
Configuration Parameter Editor 117
configuration repository 55, 545
configurations
 ComponentDebugMode 373, 640
 DebugMode 373, 640
 DoNotProtectCryptedValues 373
 HTTPAddress 373
 HTTPPort 373
 language 373
 LogDestinationAndProviderId 373
 RetriesOnFailedStart 373
 UseSSL 373
 VerboseLogging 373
 WaitTimeOnFailedStart 373
connection
 CacheReloadInterval 378
 CacheType 378
 JobGenLogDir 378, 638
 LogBlobReads 378
 NoReloadBeep 378
 ObjectDumpStackExpression 378
consistency check 57
repair 58
start 58
test method 58
test objects 59
test option 60
test status 59
Consistency Editor 57
country
 edit 129
 enable 127, 129
 language 129
 public holiday 129
 time zone 129
 work hours 129
country information 127
 country 129
 public holiday 133
 state 131
 time zone 128
Crashrecorder 635
custom configuration package
 import 556
 show contents 555
Customizer 19

D

data change
 archiving 441
 delete 441, 445
 record 427
 retention period 442
data import 560
 assign to employee 570
 assignment wizard 569

change label 573
configure 568
connection variable 572
CSV file 560
 column index 565
 column name 565
 culture 562
 delimiter 562-563
 encoding 562
 fixed width 562, 564
 header row 562
 import file 562
 line condition 565
 line structure 563-564
 mask delimiter 563
 text identification character 563
CSV import
 time zone 562
delete data 571
external database 565
 columns 567
 condition (where clause) 567
 connection data 566
 provider 566
 query source data 567
 select 566
 sort (order by) 567
 SQL statement 567
 table 567
 time zone 566
handling quantities 571
 condition 571
 delete data 571
 insert data 571
 reload data 571
hierarchy 570
import definition file 573
 load 573
 save 573
import script 573
insert data 571
log 573
reload data 571
start 573
target column 568-569
 conversion script 569
 fixed value 569
 key column 568-569
 show caption 569
 target table 568
Data Import 560
database
 authentication module 113
 compile 64
 connection data 112-113
 consistency check 57
 customer name 113
 customer prefix 113
 database ID 113
 development database 116, 645
 file groups 212
 language 113
 main database 113
 production database 116, 645
 public key 113
 revision 113
 SQL customization 159
 staging level 113, 116, 645
 test database 116, 645
 transport history 558

Database Compiler 64
database journal 629
 delete 632
 display 624, 629, 632
 log login 632
 record 632
database layer 19
Database Transporter 546, 556
DBQueue 19
 display 624
DBQueue Processor 645
 bulk processing 651
 central dispatcher 646, 651
 communication 651
 configurations 645
 database schedule 646
 duration 645, 651
 GenProcID 434
 initialization 646
 process monitoring 434
 process task 649
 processing 648
 QBM_PDBQueuePrepare 648
 QBM_PDBQueueProcess 646
 QBM_PDBQueueProcess_Del 646
 QBM_PDBQueueProcess_Main 646
 QBM_PWatchDog 646
 QBM_PWatchDogPrepare 648
 reactivate task 650
 slots 645-646, 651
 stop 626
DBSchedulerWatchDogPlugin 380
 interval 380
 provider ID 380
DebugMailPlugin 382
DropFolder 382
deferred deletion 22
Designer 27
 change label 69
 change log 41
 compile 64
 database search 37
 editors 42
 empty cache 31
 filter 38
 adhoc 39
 permanent 39
 help 34
 limit list entries 34
 lists
 filter 35
 search 36
 program setting 31
 save changes 41
 search dialog box 36
 SQLite 27
dispatcher
 IsProxy 377
 ProxyIntervall 377

E

email notification 122
emergency stop 626
encryption
 PrivateKey 365
EntityLogic 19
EntitySource 19, 22
error log 629

event
base object 397
delete 397
edit 397
execute 397
insert 397
object event 397
process information 397, 431
program function 241, 397
sort order 397
update 397

EventLogLogWriter 374
category 374
EventID 374
EventLog 374, 642
LogSeverity 374, 642
source 374

F

file
application group 155
backup 152, 155
edit 155
export 154
import 152
transport 555
version 152

file groups 212

FileJobDestination 368
AutoUpdateSubDirectories 368
BackupFiles 368
CheckInputIndex 368
EventTypes 368
host name 368
InputDirectory 368
MaxListCount 368
OutputDirectory 368
port 368
provider ID 368
SubDirectories 368
TimerInterval 368
UseEncryption 368

FileJobProvider 359
AutoSubDirectories 359
BackupFiles 359
CheckInputIndex 359
EventTypes 359
host name 359
InputDirectory 359
MaxListCount 359
OutputDirectory 359
port 359
SubDirectories 359
TimerInterval 359
UseEncryption 359

FileLogWriter 375, 636
HistorySize 375, 636
JobLogLifeTime 375, 636
LogLifeTime 375, 636
LogSeverity 375, 636
MaxLogSize 375, 636
OutPutFile 375, 636
ParamMaxLength 375, 636

foreign key column 162
dynamic 162

form archive 283

form definition 281-282
configurations 285, 290
form sequence 285
required tables 285

VI_Common_ChildRelation_Grid 287
VI_ElementNavigation 287
VI_Generic_MasterData 287, 289
VI_Report 287, 524
VI_Wizard 287
Form Editor 276
form template 282-283
 form archive 283
 form source type 283
 form type 283
 FrmCommonChildRelationGrid 287
FrmCom-
 monOneChildAndMemberRelation 287
 FrmCommonOneChildRelation 287
 FrmCommonOneDynamicRelation 287
 FrmCommonOneGenericRelation 287
 FrmCommonOneMember
 AndChildRelation 287
 FrmCommonOneMemberRelation 287
 FrmElementNavigation 287
 frmGeneric 287
 ReportForm 287
 usage 283
 WizardForm 287
form type 283
 edit 283
 grid 283
 info 283
 MemberRelation 283, 287
 report 283
 virtual 283
 wizard 283
formatting script 201
 test 479
formatting types 200
FrontendLog 655
FTP Server 361
FTP user 361
FTPJobDestination 370
 AutoUpdateSubDirectories 368
 BackupFiles 368
 CheckInputIndex 368
 EventTypes 368
 FTPPassword 370
 FTPPort 370
 FTPServer 370
 FTPUser 370
 host name 368
 InputDirectory 368
 MaxListCount 368
 OutputDirectory 368
 port 368
 provider ID 368
 SubDirectories 368
 TimerInterval 368
 UseEncryption 368
 FTPJobProvider 361
 AutoSubDirectories 359
 BackupFiles 359
 CheckInputIndex 359
 EventTypes 359
 FTPPassword 361
 FTPPort 361
 FTPServer 361
 FTPUser 361
 host name 359
 InputDirectory 359

MaxListCount 359
OutputDirectory 359
port 359
SubDirectories 359
TimerInterval 359
UseEncryption 359
full text search 37
 configure 206

G

GenProcID 426
 replacement 434
Global.cfg 659
Globallog.config 655
Globally Unique Identifier (GUID) 162,
 179
GUID module 179

H

Hotfix package
 show contents 555
HTTP authentication module
 BasicHttpAuthentication 379
 WindowsHttpAuthentication 379
HTTP Server 373
HTTPJobDestination 372
 ChildPort 372
 provider ID 372
 RemoteDomain 372
 RemotePassword 372
 RemoteUser 372
 retries 372
 RetryDelay 372

HTTPJobProvider 364
ParentPort 364
ParentServer 364
RemoteDomain 364
RemotePassword 364
RemoteUser 364
retries 364
RetryDelay 364

HTTPLogPlugin
 log file 380, 643
LogFile 380

I

info system
 bar chart 311
 line diagram 311
 pie chart 311
 statistic definition 306
 statistics 310
 table 311
 tachometer 311
 thermometer 311
 traffic light 311
input value
 define 269
InstallState.config 150
IsChanged 459
IsDeleted 459
IsLoaded 459

J

Job queue 19
progress 626
queue name 365

Job Queue Info 612, 626
Column Configuration 615
continue processing 626
database journal 624
DBQueue 624
emergency stop 626
execution state 617
Job server
 find state 623
Jobqueue sequence 626
monitor process 617
out parameter 621
polling interval 614
process
 detail 618
 reenable 621
 restart 621
process step
 detail 619
 enable logging 622
 end on failure 621
 end on success 621
 frozen 621
 out parameter 621
 over limit 621
 parameter 620
 reenable 621
program setting 614
server state 623
stop One Identity Manager Service 626
stop processing 626
stop system 626
timeout 614
update 615

Job server
configure 349
continue processing 626
create 340
edit 337, 340
executing server 341
Machine role 344
queue 340-341
server function 340-341, 343
server operating system 341
service account 341
start HTTP request 349
statistics 345
status 349, 623
stop processing 626
transfer configuration 349

Job Server Editor 338

Job Service Configuration
 module type 350
 verification test 352

JobDestination 353
 FileJobDestination 368
 FTPJobDestination 370
 HTTPJobDestination 372

Jobgate 353

JobGenLog 655

JobGenLogDir 378

Jobprovider 353
 AppServerJobProvider 364
 FileJobProvider 359
 FTPJobProvider 361
 HTTPJobProvider 364
 MSSQLJobProvider 358
 OracleJobProvider 359

Jobservice 655

Jobservice.cfg 356, 660
JobServiceDestination
 EncryptionScheme 365
 ExternalSlotEnvironment 365
 ExternalSlotEnvironment32 365
 ExternalSlots 365
 ExternalSlots32 365
 InternalSlots 365
 MaxExternalSlotReuse 365
 PrivateKey 365
 PrivateKeyId 365, 383
 provider ID 365
 queue 365
 RequestTimeout 365
 StartInterval 365
 StatisticInterval 365

K

key file 365

L

language 129, 131
 default language 126
 login language 126
Language Editor 332
language pack
 import 334
Launchpad 319
 action 321
 extend 319
 NavigationNodeState 319
link
 set up 270

List Editor 46
 configurations 48
 multi edit 48
 object relations 49
lists
 condition 267
 display pattern 267-268
 edit 267
 input value 269
 insert values 267
 object 267
 sort by 267

log
 Crashrecorder 635
 error message window 627
 generation log
 One Identity Manager Service 378, 638
 NLog 634, 655
 One Identity Manager Service 638
 One Identity Manager Service event log 374
 One Identity Manager Service log file 375, 636
 Logger name 655
 LogWriter
 EventLogLogWriter 374
 FileLogWriter 375, 636

M

Machine role 150, 344
mail template 483
 base object 486, 488
 confidentiality 486
 copy 485

create 485
design type 486
edit 485
email signature 493
hyperlink 489-490, 492
importance 486
language 486-487
mail body 486-487
mail definition 487
preview 485
report 486
subject 486-487
target format 486
unsubscribe 486
Mail Template Editor 483
 preview 485
maintenance task 653
many-to-many table 173
MarkForDeletion 22
menu item
 condition 262, 264
 configuration switch 262
 copy 258
 create 258
 data dependent 252, 264
 data source 264
 database query 264
 diagram type 310-311
 disable 262
 display text 262
 edit 258
 fixed 252
 free 252
 icon 262
 item type 262
link 252, 270
lists 267
main form element 252
menu category 252, 258
overlay icon 262, 319
preprocessor condition 262
recursive 264, 266
sort by 264
sort order 262
statistics 252, 310
task 252
task category 252
unique 264, 266
variable definition 271, 274
method definition
 behavior 324
 disable 322, 324
 display text 324
 edit 322
 enabled for 324
 icon 324
 name 324
 object 324
 permissions group 322, 324
 program function 241, 322, 324
 script 324
 test 480
 MSSQLJobProvider 358
 ConnectionString 358
 RequestQueueLimit 358
 ResultQueueLimit 358
N
navigation
 copy 260

select 259
NavigationNodeState class 319
NLog 634, 655
 Logger name 655
 severity level 655
notification system 122

O

object
 authorizations 243
 base. 460
 change 22, 25
 delete 22, 25
 discard 22
 EntitySource 22
 handling 22
 insert 22, 25
 IsChanged 459
 IsDeleted 459
 IsLoaded 459
 lifecycle 22
 load 22
 marked for deletion 22
 MarkForDeletion 22
 transaction 25
 UnitOfWork 22
 update 22
 XMarkedForDeletion 22
object class 22
object definition
 condition 247, 250
 display name 250
 display pattern 250
 display text 250
 display text (form) 248
 display text (list) 248
 edit 249
 input value 269
 insert values 250
 object 250
 preprocessor condition 250
 selection script 247, 250
 table 250
 universal 250
Object Editor 44
 multi edit 46
object event 397
 program function 397
object layer 19
ObjectLog 655
One Identity Manager
 data model 161
 software architecture 19
 system configurations 74
 authentication module 74
 calculation schedule 134
 configuration parameter 117
 database connection 112
 documentation 157
 email notification 122
 enable country 127
 language 126
 operating system 156
 report 157
 time zone 128
 TimeTrace 149
One Identity Manager History Database
 database 149
One Identity Manager schema 161
 schema overview 168

One Identity Manager Service
 AppServerJobProvider 364
 ComponentDebugMode 640
 configuration file 356, 660, 662
 configurations 373
 template 350
 configure 349, 356
 connection 378
 DBSchedulerWatchDogPlugin 380
 DebugMailPlugin 382
 DebugMode 640
 dispatcher 377
 event log 374, 642
 EventLogWriter 374
 FileJobDestination 368
 FileJobProvider 359
 FileLogWriter 375, 636
 FTPJobDestination 370
 FTPJobProvider 361
 generation log 378, 638
 HTTP authentication 379
 HTTP Server 638
 HTTPJobDestination 372
 HTTPJobProvider 364
 HTTPLogPlugin 380
 install 346
 JobDestination 365
 JobServiceDestination 365
 language 373
 linux share 382
 log file 375, 636, 640
 display 638
 log file (HTTPLogPlugin) 643
 LogWriter 374
 MSSQLJobProvider 358
 NSProviderTrace.log 640
 OracleJobProvider 359
 out parameter 641
 PerformanceCounterPlugin 381
 plugins 379
 procedure 353
 process collection 357
 process component 353
 queue 340-341
 RaiseMessage 641
 remote install 346
 RequestWatchDogPlugin 381
 ScheduleCommandPlugin 380
 services 638
 set up 353
 ShareInfoPlugin 382
 statistics info 365
 StdioProcessor.log 640
 stop 626

One Identity Manager Service module
 configurations 373
 connection 378
 dispatcher 377
 HTTP authentication module 379
 JobDestination 365
 LogWriter 374
 plugins 379
 private key files 383
 process collection 357
 operating system
 client 156
 server 156
 version 156

 OracleJobProvider 359
 ConnectionString 359

RequestQueueLimit 359
ResultQueueLimit 359
overview form 297
 alignment 303
 background color 303
 columns 300, 303
 create 300
 design view 300
 disable 305
 disable form element 305
 display text 300
 form element 300
 header 303
 lists 303
 main form element 300
 menu item 300
 object 300
 permissions group 300
 preview 300
 product assignment 300
Overview Form Editor 298
 design view 300
 preview 300

P

password policy 138
 assign 146
 character sets 141
 check password 145
 conversion script 142, 144
 default policy 140, 146
 display name 140
 edit 139
 error message 140
 excluded list 145
failed logins 140
generate password 146
initial password 140
name components 140
password age 140
password cycle 140
password length 140
password strength 140
predefined 139
test script 142

PerformanceCounterPlugin 381
 category 381
 CounterType 381
 PollingInterval 381

permissions
 column 240
 condition 239
 copy 231, 260
 delete 231
 determine 233
 edit 231
 insert 231
 rules 233
 table 239

Permissions Editor 235
 filter 238

permissions group
 column permissions 240
 copy 224, 260
 hierarchy 223
 only for role-based login 226
 permissions 215
 predefined 216
 program function 241
 QBM_BaseRights 216

role based 216
set up 219, 226
table permissions 239
vi_4_ADMIN_LOOKUP 216
VI_4_ALLUSER 216
VI_Everyone 216
VI_View 216
vid 216

plugins

- DBSchedulerWatchDogPlugin 380
- DebugMailPlugin 382
- HTTPLogPlugin 380
- PerformanceCounterPlugin 381
- RemoteConnectPlugin 382
- RequestWatchDogPlugin 381
- ScheduleCommandPlugin 380
- ShareInfoPlugin 382

preprocessor condition 449, 451

- column 189
- configuration parameter 449
- evaluate 452
- menu item 262
- object 250
- process 394
- process step 401
- report 499
- statistics 307
- table 173
- user interface forms 281

preprocessor Expression 449

presentation Layer 19

primary key column 162

process

- base object 394
- compare 410

compile 415
copy 392
create 392
do not generate 394
edit 391-392
error checking 413
event 397
export 411
frozen 621
generating condition 394
identifier 394
import 411
monitor 617
notification 397
over limit 397, 621
pre-script for generating 394, 461
preprocessor condition 394
process information 394, 431
simulation 411
table 394
threshold 394, 397
UID 394
validity check 413
variable 396

process automation 416

process component 353, 365, 419

- AutoUpdateComponent 419
- CommandComponent 419
- ComponentDebugMode 640
- ControlFilesComponent 419
- DelayComponent 419
- display name 422
- FileComponent 419
- FtpComponent 419
- HandleObjectComponent 419

instance 422
LogComponent 419
MailComponent 419
PowerShellComponent 419
process task 422
ProjectorComponent 419
ReportComponent 419
return value 641
ScriptComponent 419, 573
SQLComponent 419
SubversionComponent 419
WakeOnLanComponent 419
ZipComponent 419
Process Editor 385
 layout position 389
 process document 389
 process element 389
 process step element 389
process generating
 simulate 411
 test 411
process handling 384
 monitor 611
process history
 archiving 441
 delete 441, 445
 record 433
 retention period 442
process information
 archiving 441
 delete 441, 445
 depth of detail 431
 export 444
 for process 431
 for process step 431
 for result 431
 import 444
 record 429
 retention time 442
process monitoring 425-426
 archiving 441
 data change 427
 DBQueue Processor 434
 delete 441, 445
 enable 426
 process handling 429
 process information 429
 process trigger 426
 retention period 442
process parameter template 422
process plan 416
 base object 418
 calculation schedule 418
 condition 418
 event 418
 execute 418
 parameter 418
 set up 418
process step
 copy 399
 create 399
 edit 399
 execution state 617
 generating condition 401
 identifier 401
 ignore errors 401
 import 399
 log error 401, 632
 log mode 401, 622
 notification 401, 405

parameter 406
edit 407
encrypted 407
hidden 407
identifier 407
IN 407
INOUT 407
OUT 407
out parameter 408, 621, 641
Parametercollection 408
set value 408
template 422
type 407
value template 407
pre-script for generating 401, 461
preprocessor condition 401
priority 401
process information 401, 431
process task 401
retention time 401
retries 401
script for server selection 401, 403
search 410
server 403
server function 401, 403
Split processing 401
stop on error 401
wait mode on error 401
process task 419
identifier 422
instance 422
operating system class 422
process component 422
program function 241, 322, 397
method definition 241
object event 241
permissions group 241
script 241
ProjectorEngine 655
proxy server 377
public holiday
country 129
edit 133
state 131
PWatchDogPrepare 648

Q

QBM_GCommon2.PDiskStorePhysicalSync 212
QBM_GCommon2.PTableMove 212
QBM_GDBQueue.PDBQueuePrepare 212, 648
QBM_GWatchDog.PPrepare 212, 648
QBM_PDBQueuePrepare 212, 648
QBM_PDBQueueProcess 646
QBM_PDBQueueProcess_Del 646
QBM_PDBQueueProcess_Main 646
QBM_PDiskStorePhysicalSync 212
QBM_PTableMove 212
QBM_PWatchDog 646
QBM_PWatchDogPrepare 212

R

RemoteConnectPlugin 382
ADGroupAuthPermittedGroup 382
AuthenticationMethod 382
port 382
report
base table 499
bind 524

copy 498
create 498
data field 516
data query
 historical assignment 507
 multiple object history 506
 object 503
 simulation 509
 single object history 504
 SQL 502
 test 500
 view 502
data source
 delete 500
 edit 500
 virtual 515
display name 499
edit 498
export 516
identifier 499
import 516
load 498
multilingual 522
preprocessor condition 499
repo 516
report parameter
 can be overwritten 512
 condition (calc.) 515
 condition (query) 513
 data source 513
 data type 513
 default value 513
 delete 510
 description 512
 display name 512
edit 510
list of permitted values 513
mandatory parameter 512
multiline 513
multivalue 513
overwrite empty value 513
parameter definition 513
parameter type 512
parameter value 513
script 515
sort order 512
table column (calc.) 515
table column (query) 513
value calculation 515
viewable 512
ReportAlias 499
SpecialSheetData 524
translate 516, 522
user interface forms 524
Report Editor 494
 globalization editor 516
 program setting 497
 SQL log 498
RequestWatchDogPlugin 381
 interval 381
 MinRequests 381

S

ScheduleCommandPlugin 380
 command 380
 interval 380
 LogSeverity 380
 OutputToLog 380
 StartCommand 380
 StopCommand 380

Schema Editor 165
 multi edit 168
 schema overview 168
schema extension
 add 541
assignment table
 column names 537
 connected tables 537
 description 537
 display name 537
 index column 538
 name 537
 origin (XOrigin) 537
 x-column 537
authorizations 539
change label 540
column
 base column 531
 column name 530-531
 comment 532
 compulsory field 532
 configure 532
 create 530-531
 data type 532
 define 529
 delete restrictions 532
 display name 532
 foreign key column 531
 from table 531
 initial value 532
 insert restrictions 532
 name 532
 origin (XOrigin) 537
 permission 532
 primary key 527, 532
 simple column 530
 UID 532
 unicode 532
 x-column 527, 537
create index 538
Creating a New Table 527
custom 526
database view
 column 531
 create 534, 536
DDL statements 541
Extending a Table 529
foreign key column
 column name 531
 create 531
 from table 531
foreign key relation 535
Index 538
 index name 538
permissions group 539
read-only database view
 description 534
 display name 534
 foreign key relation 535
 name 534
 view definition 534
save to file 541
Schema Extension 526
simple column
 column name 530
 create 530
table
 create 527, 537
 description 527
 display name 527

extend 529
name 527
primary key 527
x-column 527
transport 552
union view
 description 536
 display name 536
 name 536
 view definition 536
Schema Extension 526
script 453
 #LD notation 464
 \$ notation 455
accessibility 51, 469
 auto-completion 51, 469
base. 460
compile 473
copy 470
create 470
data type 455
date value 454
edit 470, 478
function 460
message 454
overridable 472
override 472
overwrite 472
pre-script 461
program function 241
RaiseMessage 454, 641
save 478
Session Services 461
syntax 453
test 473-474, 476-477
VID_Write2Log 454
web service call 579
Script Editor 467
script library 467
 load 475
semaphore 147
server
 specify 403
server function 150, 343
session
 Session.Config.GetConfigParm 462
 Session.Source.Exists 462
 Session.Variables 463
Session Services 461
ShareInfoPlugin 382
SOAP Web Service 584
 ChangeSingleObject 584, 592
 ChangeSingleObjectEx 584, 592
 configuration file 589
 configure 589
 CreateSingleObject 584, 592
 DeleteSingleObject 584, 592
 DeleteSingleObjectEx 584, 592
 exists 584, 592
 ExistsEx 584, 592
 FireGenEvent 584, 592
 FireGenEventEx 584, 592
 GetCompleteSingleObject 584, 592
 GetCompleteSingleObjectEx 584, 592
 GetListObject 584, 592
 GetListObjectWithDisplays 584, 592
 GetSingleObject 584, 592
 GetSingleObjectEx 584, 592
 GetSingleProperty 584, 592
 GetSinglePropertyEx 584, 592

install 586
InvokeCustomizer 584, 592
InvokeCustomizerEx 584, 592
InvokeDialogMethod 584, 592
InvokeDialogMethodEx 584, 592
status display 591
web.config 589
software architecture 19
Software Loader 152
software update
 export files 151
 import files 151
SPML Provisioning Service Provider 598
SPML schema
 configure 607
 export 608
SPML Web Service 598
 configuration file 603
 configure 599
 install 599
 schema file 608
 test 608
 web.config 603
SQL Editor 49
SQLite 27
SqlLog 655
Standard GUID 179
state
 edit 131
 enable 131
 language 131
 public holiday 131
 time zone 131
 work hours 131
statistic
 traffic light 311
statistics 306
 aggregate function 307
 bar chart 311
 calculation 307
 calculation schedule 307
 condition 308
 diagram type 311
 display name 307
 edit statistics definition 306
 ElementName 308, 311
 ElementObjectKey 308
 ElementObjectKey2 308
 ElementOrder 308
 ElementValue 308, 311
 history 307
 imported statistics data 307
 instant calculation 307
 line diagram 311
 measurement value 308
 not set 307
 pie chart 311
 preprocessor condition 307
 query 308
 table 311
 tachometer 311
 thermometer 311
 threshold 307-308
StopWatch 655
StudioProcessor.exe 365
SwitchToModuleGuid() 179
SwitchToNormalGuid() 179
SyncLog 655

system
 stop 626
System Debugger 474
 SQL log 481
system user
 administrative user 227
 dynamic 108, 216, 230
 employees 230
 logins 227
 password 227
 permissions group 229
 predefined 216
 read-only permissions 227
 sa 216
 service account 227
 set up 227
Synchronization 216
user 230
viadmin 216
viHelpdesk 216
viITShop 216
SystemConnection 655
SystemConnector 655

T

table
 base table 173
 cache information 173
 Customizer 173
 deferred deletion 173
 display name 173
 display pattern 173
 edit 172
 export for SPML 173, 607
 file group 212

GUID module 173
hierarchy path 173
icon 173
input value 269
logical storage 173, 212
M-to-all table 162
many-to-many table 162, 173
mapping 172
multicolumn uniqueness 173
physical storage 212
preprocessor condition 173
properties 172
proxy 185
proxy view 173
read-only 187
simple table 162
statistic information 173
table scripts 173, 180
type 162
union 183
usage type 173
view 181
view definition 173
worktable 162
table definition
 module GUID permitted 179
 module GUID required 179
table relation 208
table scripts 173, 180
 test 480
task
 disable 322
template
 edit 197
 recipient 195

sender 195
test 479
time zone 128
 country 129
 state 131
TimeTrace 149
transaction 25
transfer buffer 545
translation 328
 basics 329
 column 330
 edit 333
 import 334
 Language Editor 332
 language pack 334
 text buffer 330
translation source 330
translation target 330
transport package
 basics 545
 change data 551
 change label 550
 complete transport 554
 create 546
 custom configuration package 544
 date selection 551
 export 546
 export criteria 546
 Hotfix package 544
 import 556
 migration package 544
 schema extension 552
 show contents 555
 SQL statement 549
 system configurations 554
system file 555
tool select 553
tool select (favorites) 549
transport history 558
user list 551

U

UnitOfWork 19, 22
update 655
use case
 authentication module 326
 configuration data 108-111, 326
 form 326
 menu item 326
 set up 325-326
 start menu item 326
 system user 326
user
 authentication module 244
 dynamic 244
 permissions group 244
 program function 244
 read access 244
 system user 244
User & Permissions Group Editor 219
user interface
 form 274
 icons 327
 images 327
 method definition 322
 navigation 251
 object definition 247
 program function 241
 statistics 310
 translation 328

use case 325-326

User Interface Editor 254

user interface forms

- configurations 281
- disable 279, 281
- display 286
- display text 281
- edit 279, 281
- form definition 281
- form name 281
- generic form 289
- input value 269
- insert values 281
- master data form 289
- object definition 286
- overview form 297
- preprocessor condition 281
- replace 296
- repo 524
- sort order 281

V

variable

- define 271
- DialogUserID 271, 463
- EnvUserName 271, 463
- Feature_ 463
- FullSync 463
- GenProcID 463
- LogonUser 271, 463
- ManageOutstandingOperation 463
- pre-script 396
- process definition 396
- SessionType 271, 319
- ShowCommonDialog 271, 463

W

web service

- create script 579
- integrate 575
- method call

 - direct 578
 - generic 576
 - self-defined 578

- script parameter 579
- SOAP 579
- synchronization by script 576
- WCF 579
- WSDL file 579

web service integration wizard 579

WebLog 655

X

XDateInserted 162

XDateSubItem 162

XDateUpdated 162

XIsInEffect 162

XMarkedForDeletion 22, 162

XObjectKey 162

XOrigin 162

XTouched 162

XUserInserted 162

XUserUpdated 162