Technical Documentation Mavim Manager

NAME	:	Technical Documentation Mavim Manager
VERSION	:	44
DATE	:	10/17/2022
STATUS	:	

TABLE OF CONTENTS

1 CREATING A SCHEMA IN SQL SERVER WITHOUT MIXED MODE

To make use of a SQL Database, a **Schema** has to be created in the SQL database.

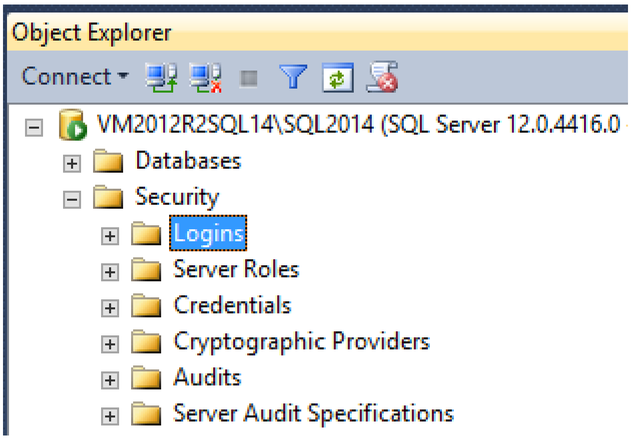
This manual start from a new SQL Server configuration without Mavim databases and users.

For upgrading a SQL Server environment with Mavim database, see the upgrade manual.

The setup of a new Mavim database on SQL Server is as follows:

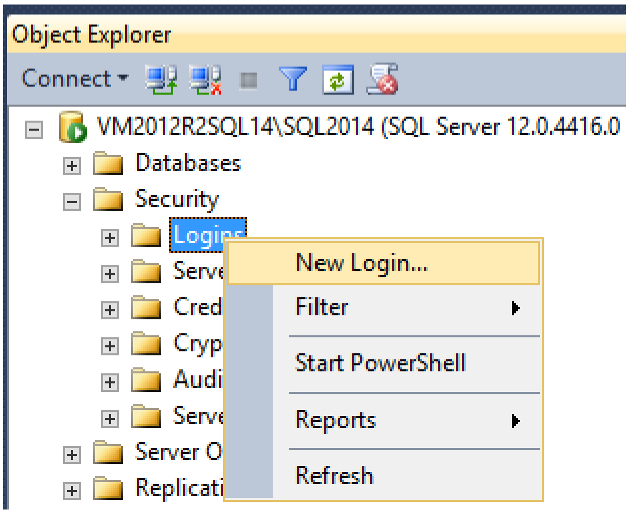
Open SQL Server Management Studio.

Add the Windows users that must have access to the Mavim database to the SQL Server.
Navigate to **Security** and open **Logins**.

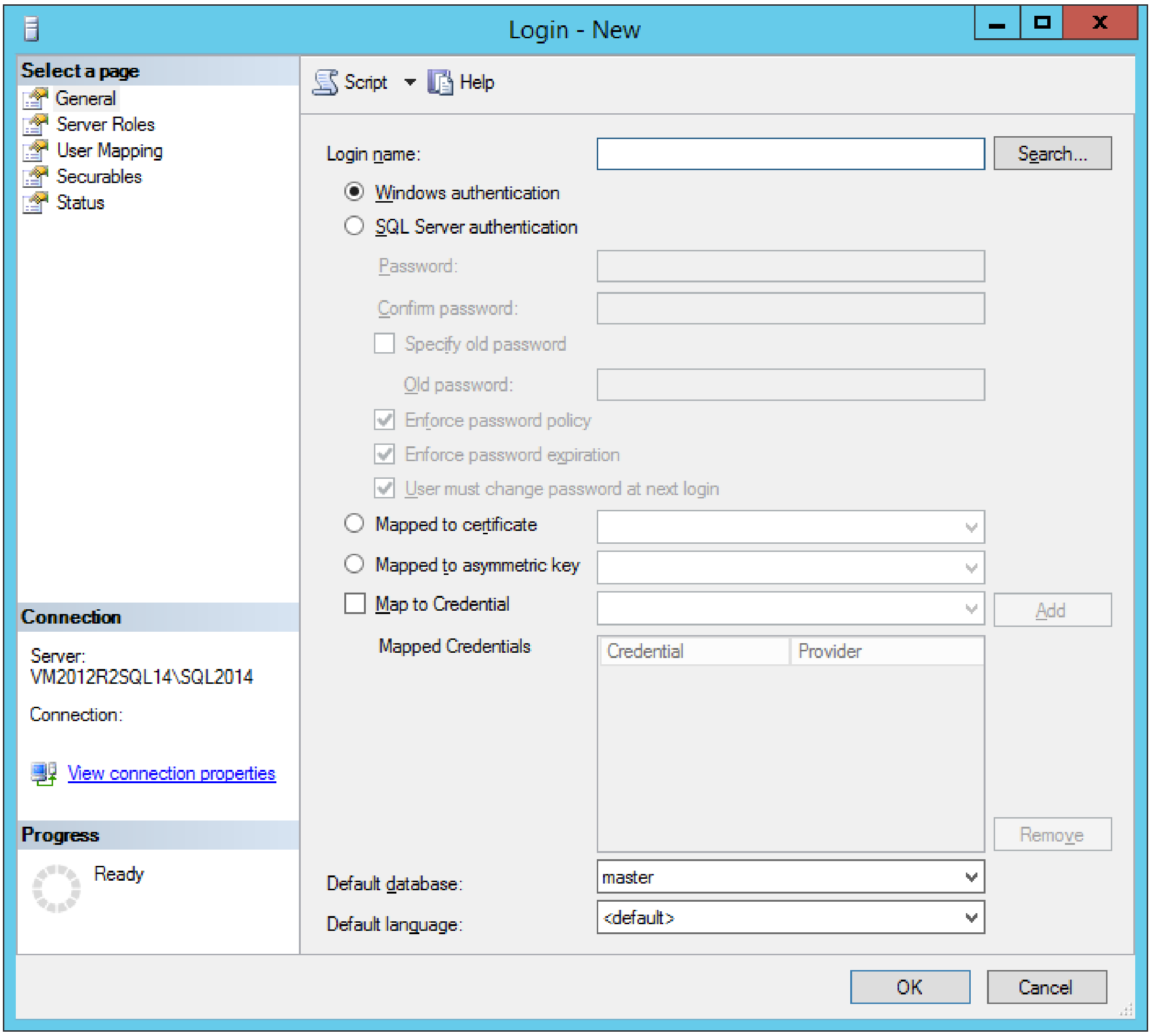


Verify if users that have to have access to the Mavim database are in this list. When this is not the case, add the users (or groups) to the list.

Open the context menu on **Logins** and click on **New Login…**



The following screen will be shown:



Login name:

● Windows authentication

○ SQL Server authentication

Password:

Confirm password:

□ Specify old password

Old password:

☑ Enforce password policy

☑ Enforce password expiration

☑ User must change password at next login

○ Mapped to certificate

○ Mapped to asymmetric key

□ Map to Credential

Mapped Credentials

Credential	Provider

Default database: master

Default language: <default>

Connection

Server:
VM2012R2SQL14\SQL2014

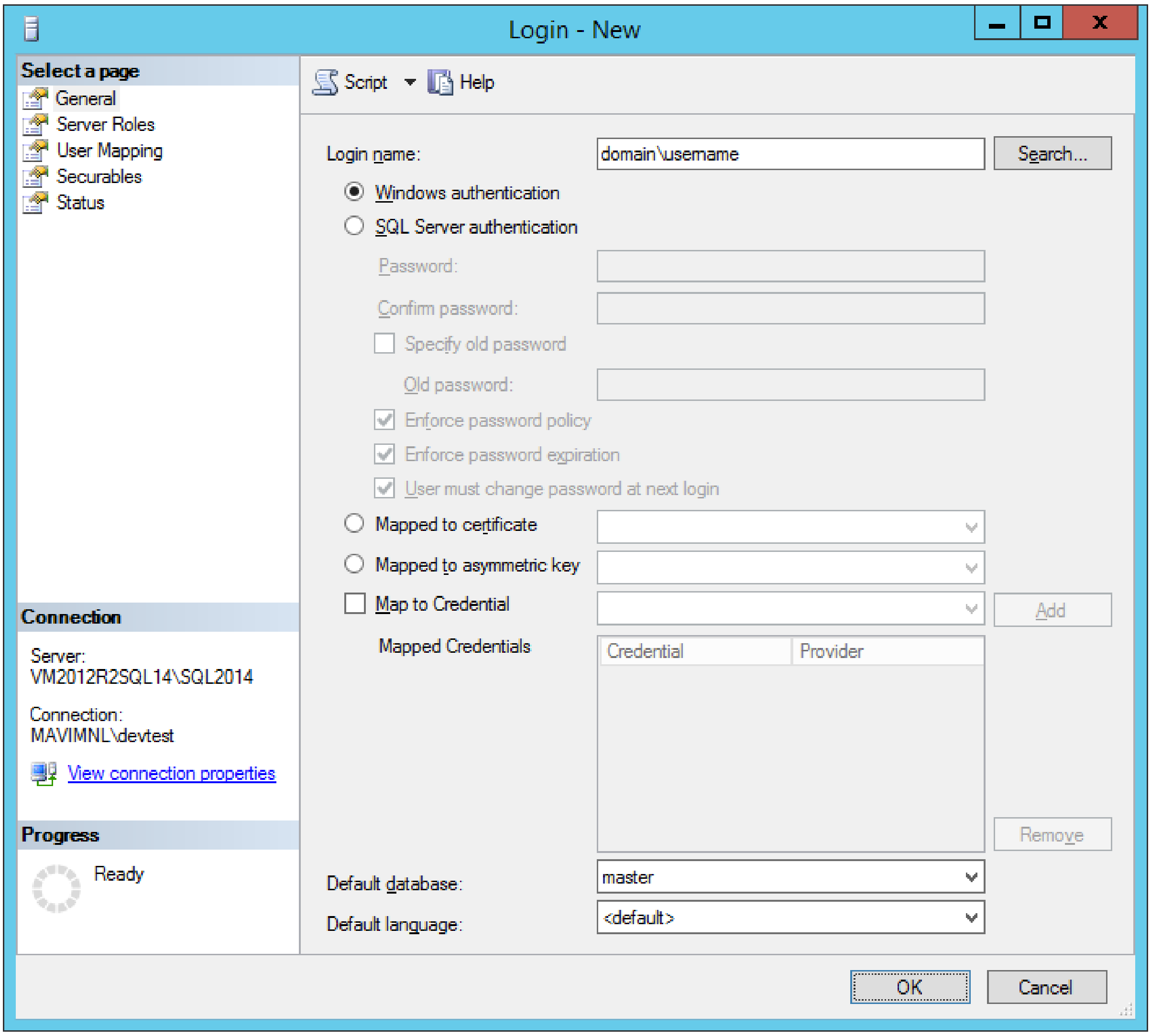
Connection:

View connection properties

Progress

Ready

Enter the domain\username or domain\group. Or click **Search...** to find them and select them.



Login name: domain\username

● Windows authentication

○ SQL Server authentication

Password:

Confirm password:

□ Specify old password

Old password:

☑ Enforce password policy

☑ Enforce password expiration

☑ User must change password at next login

○ Mapped to certificate

○ Mapped to asymmetric key

□ Map to Credential

Mapped Credentials

Credential	Provider

Default database: master

Default language: <default>

Connection

Server:
VM2012R2SQL14\SQL2014

Connection:
MAVIMNL\devtest

View connection properties

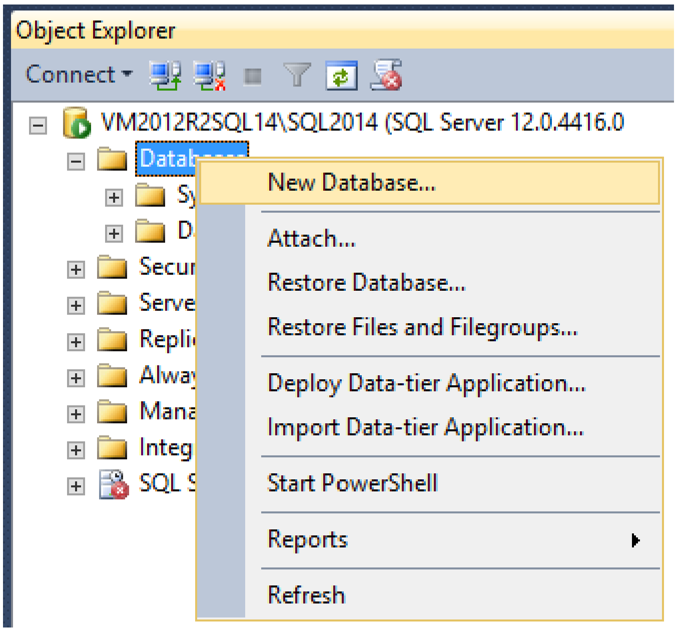
Progress

Ready

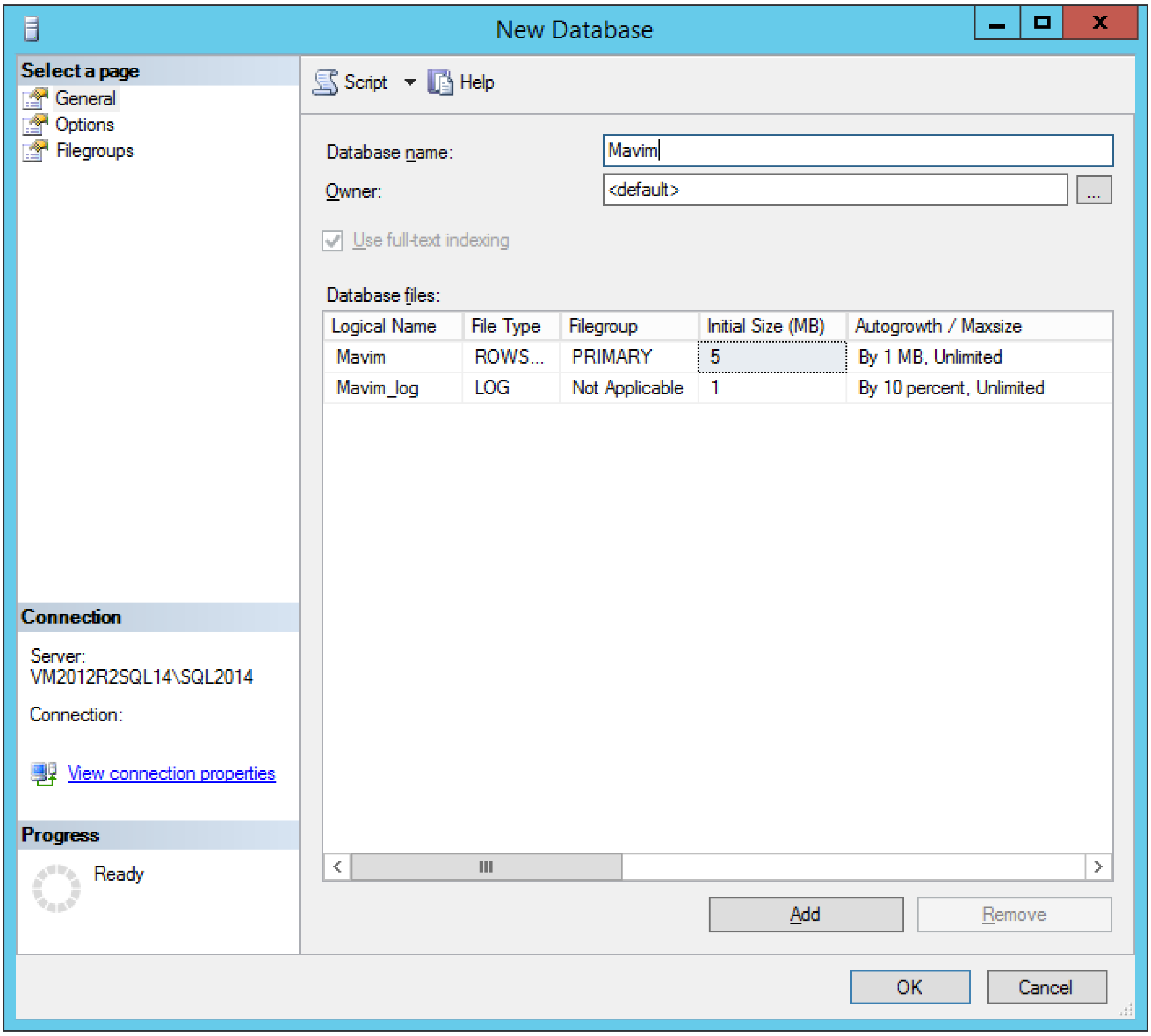
Click OK.

Create a database in SQL Server.

Open the context menu on **Databases** and click on **New Database...**


Object Explorer
Connect
VM2012R2SQL14\SQL2014 (SQL Server 12.0.4416.0
Datab
Sy
D
Secur
Serve
Repli
Alwa
Mana
Integ
SQL S
New Database...
Attach...
Restore Database...
Restore Files and Filegroups...
Deploy Data-tier Application...
Import Data-tier Application...
Start PowerShell
Reports
Refresh

Enter a name for the new database, the default settings can be used.



Database name: Mavim

Owner: <default>

☑ Use full-text indexing

Server:
VM2012R2SQL14\SQL2014

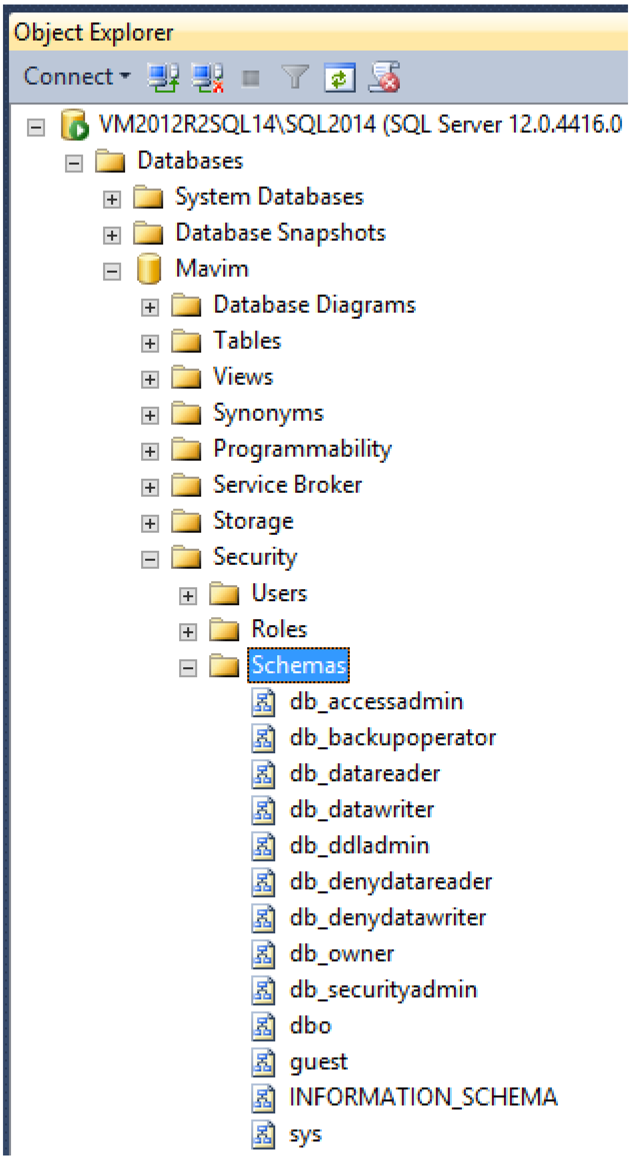
Connection:

Database files:

Logical Name	File Type	Filegroup	Initial Size (MB)	Autogrowth / Maxsize
Mavim	ROWS...	PRIMARY	5	By 1 MB, Unlimited
Mavim_log	LOG	Not Applicable	1	By 10 percent, Unlimited

Click OK.

Open the database **Mavim** and navigate to **Security**, **Schemas**

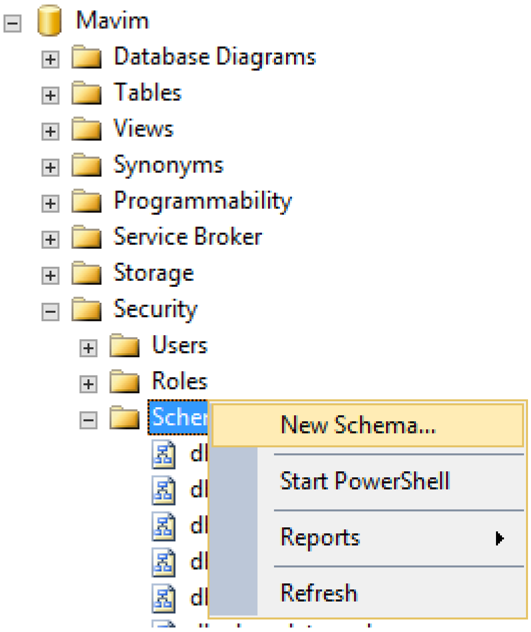


Object Explorer

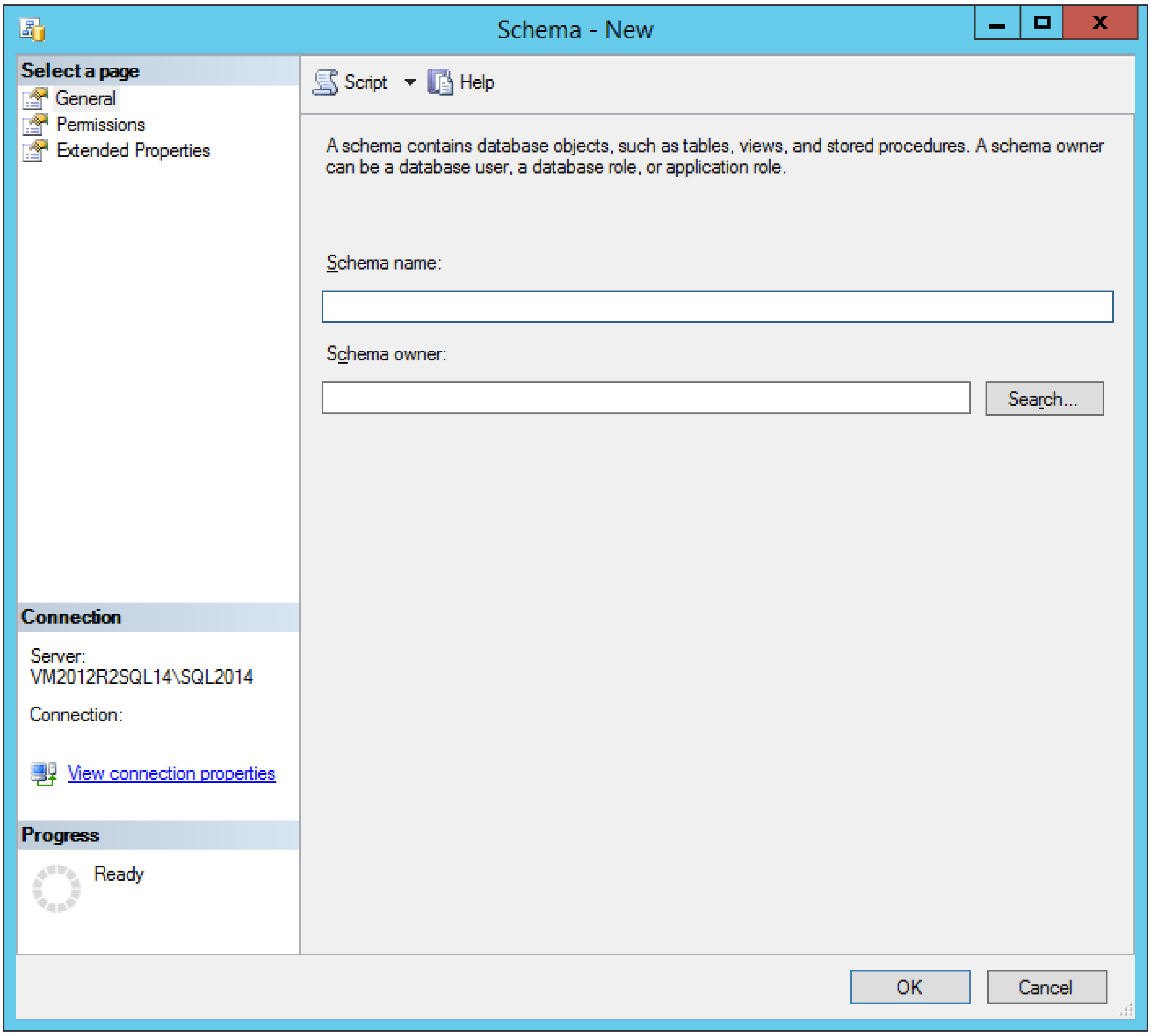
Connect ▾

```
⊟ VM2012R2SQL14\SQL2014 (SQL Server 12.0.4416.0
  ⊟ Databases
    ⊞ System Databases
    ⊞ Database Snapshots
    ⊟ Mavim
      ⊞ Database Diagrams
      ⊞ Tables
      ⊞ Views
      ⊞ Synonyms
      ⊞ Programmability
      ⊞ Service Broker
      ⊞ Storage
      ⊟ Security
        ⊞ Users
        ⊞ Roles
        ⊟ Schemas
            db_accessadmin
            db_backupoperator
            db_datareader
            db_datawriter
            db_ddladmin
            db_denydatareader
            db_denydatawriter
            db_owner
            db_securityadmin
            dbo
            guest
            INFORMATION_SCHEMA
            sys
```

Open the context menu on **Schemas** and select **New Schema…**



The following screen will be shown:



A schema contains database objects, such as tables, views, and stored procedures. A schema owner can be a database user, a database role, or application role.

Schema name:

Schema owner:

Connection

Server:
VM2012R2SQL14\SQL2014

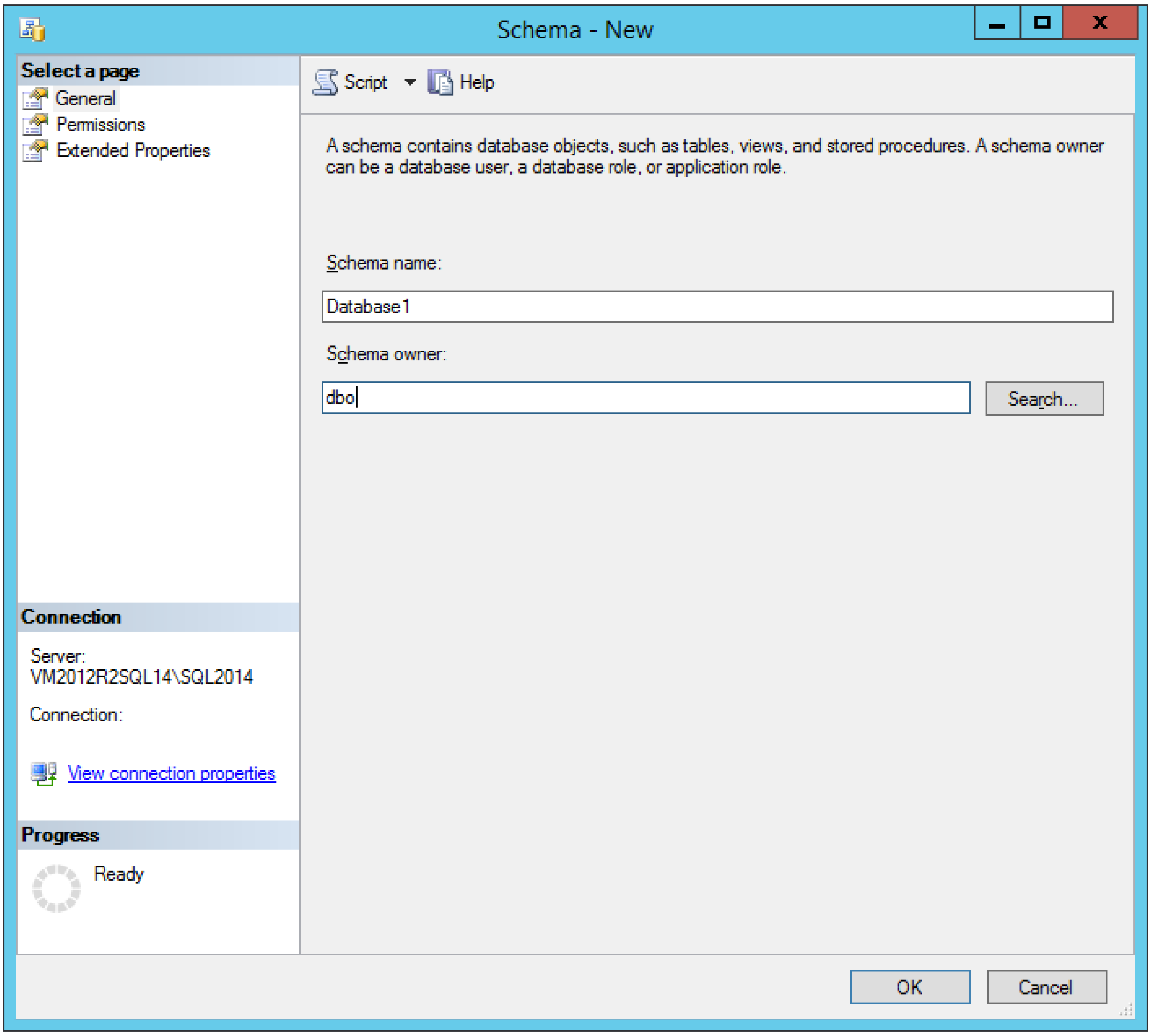
Connection:

Progress

Ready

At **Schema name** enter the name of the new Mavim database. In this example it is **Database1**. This name will appear in the Developer environment of the user.

Enter at **Schema owner** the name **dbo**:



A schema contains database objects, such as tables, views, and stored procedures. A schema owner can be a database user, a database role, or application role.

Schema name: Database1

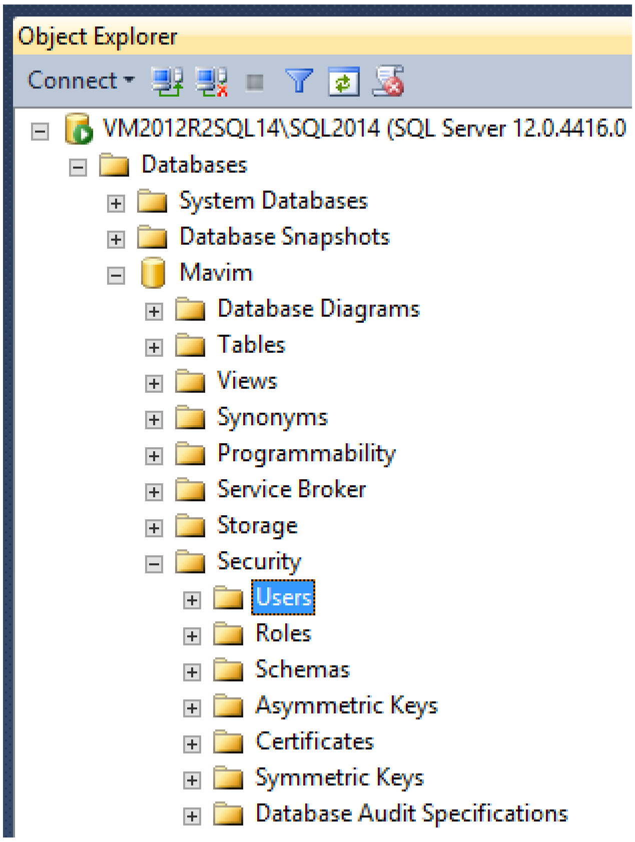
Schema owner: dbo

Server: VM2012R2SQL14\SQL2014

Connection:

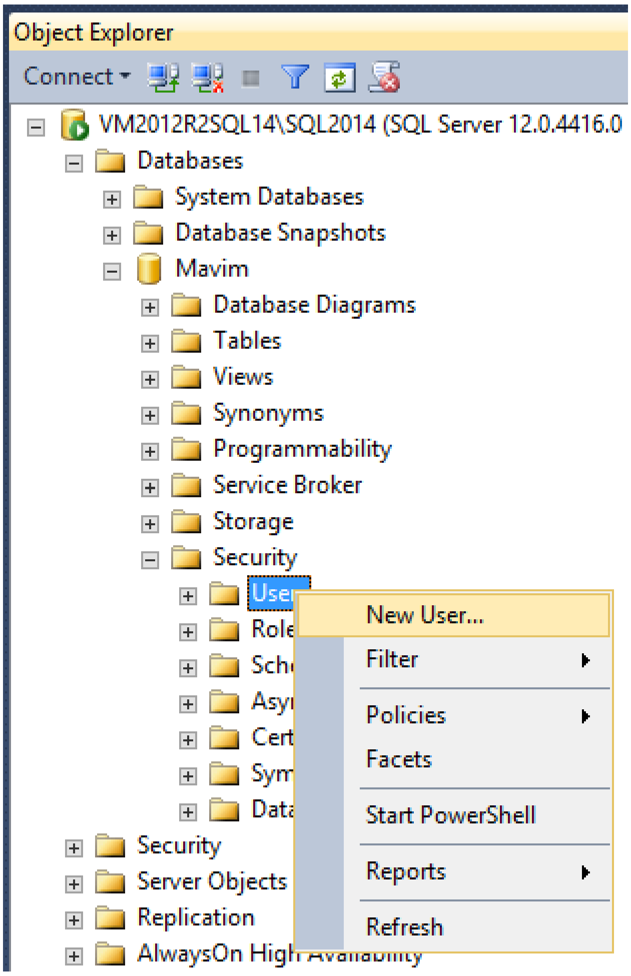
Click OK.

Open the **Users** of **Database1**:

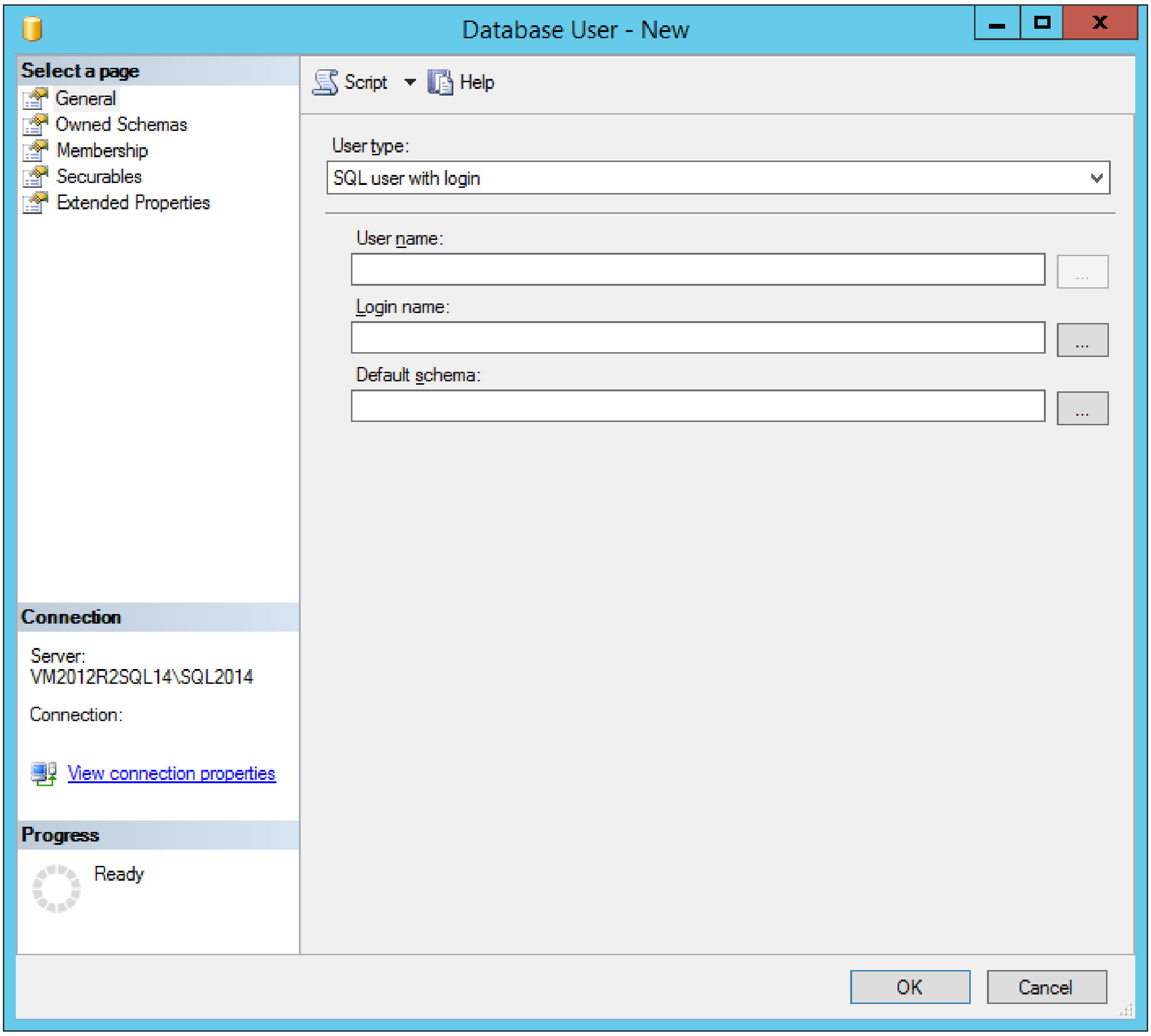


Make sure that user(s) and or group(s) who have to have access to the database, also have access to the SQL Server environment.

Open the context menu and select New User ...



The following screen will be shown:



Database User - New

User type: SQL user with login

User name:

Login name:

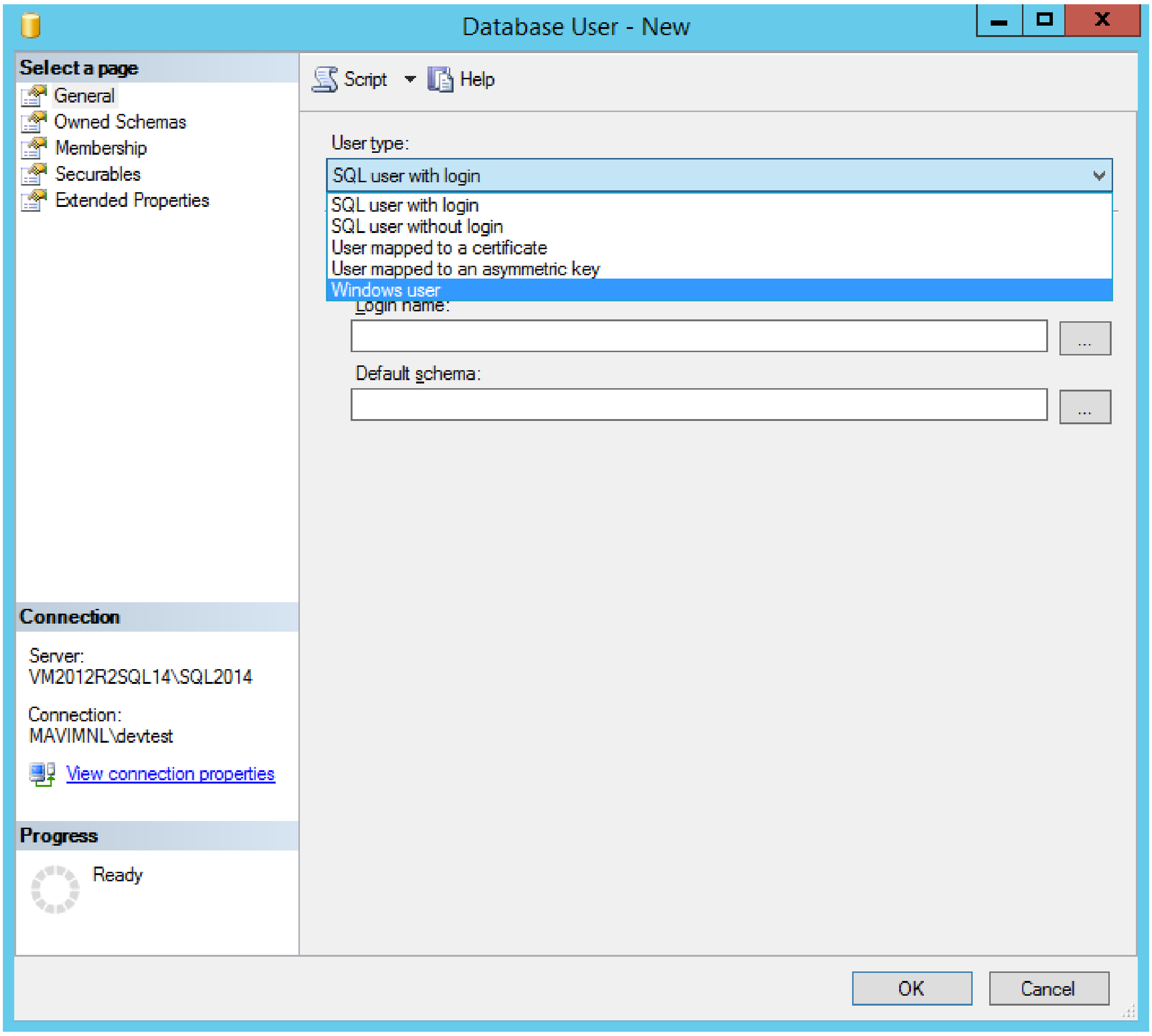
Default schema:

Connection

Server:
VM2012R2SQL14\SQL2014

Connection:

Open the list at **User type** and select **Windows user**:



User type: SQL user with login

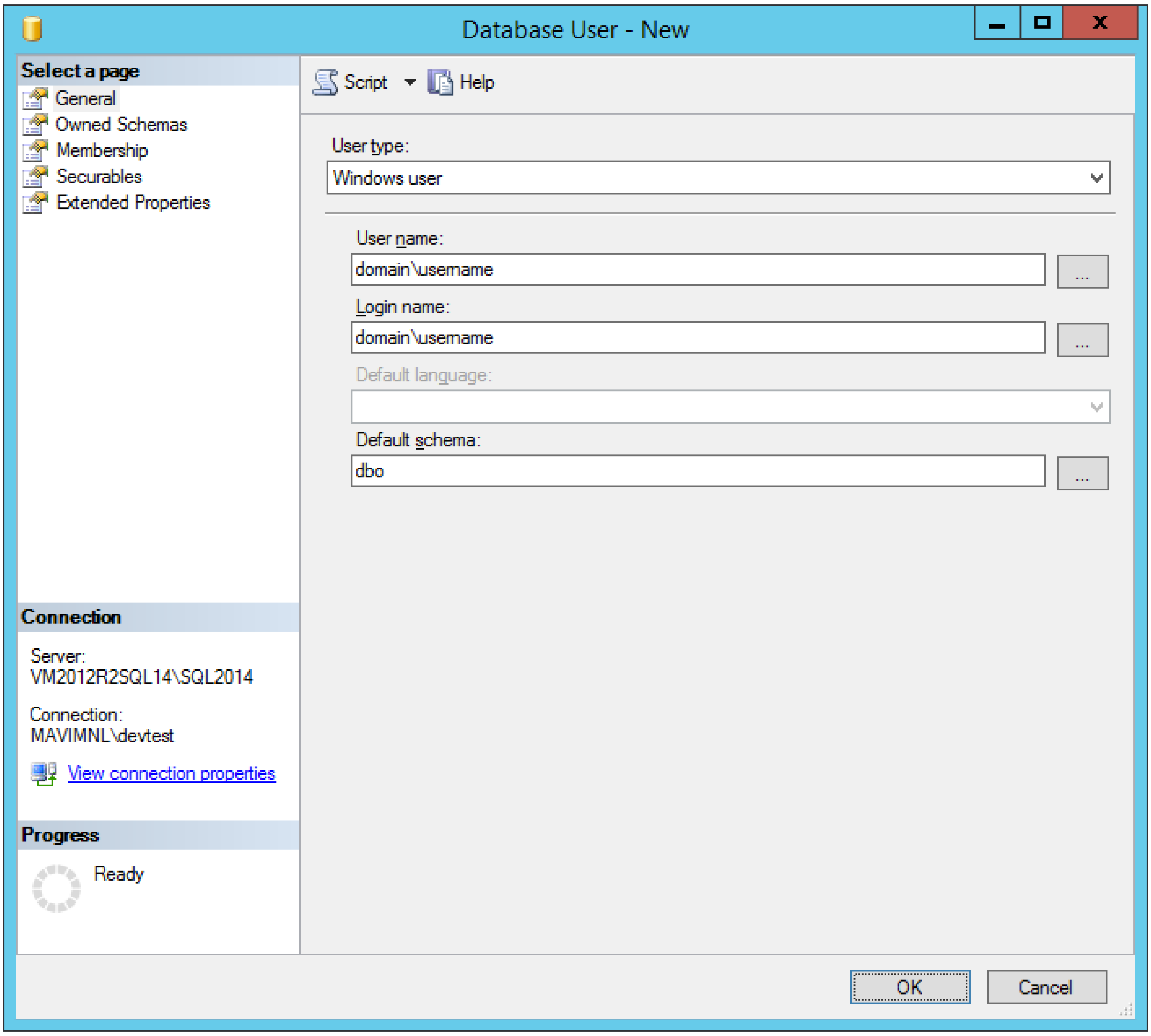
Login name:

Default schema:

Server:
VM2012R2SQL14\SQL2014

Connection:
MAVIMNL\devtest

Select at **User name** and **Login name** the user or group that has access to the schema.
For **Default schema** enter the name **dbo**.



User type: Windows user

User name: domain\username

Login name: domain\username

Default language:

Default schema: dbo

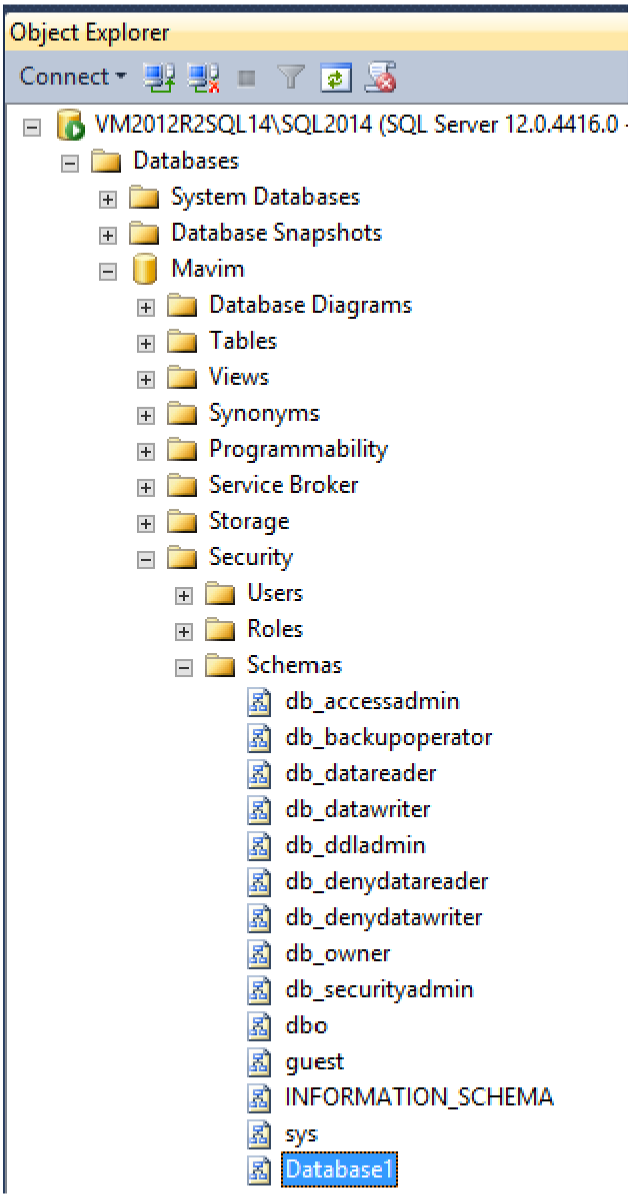
Server:
VM2012R2SQL14\SQL2014

Connection:
MAVIMNL\devtest

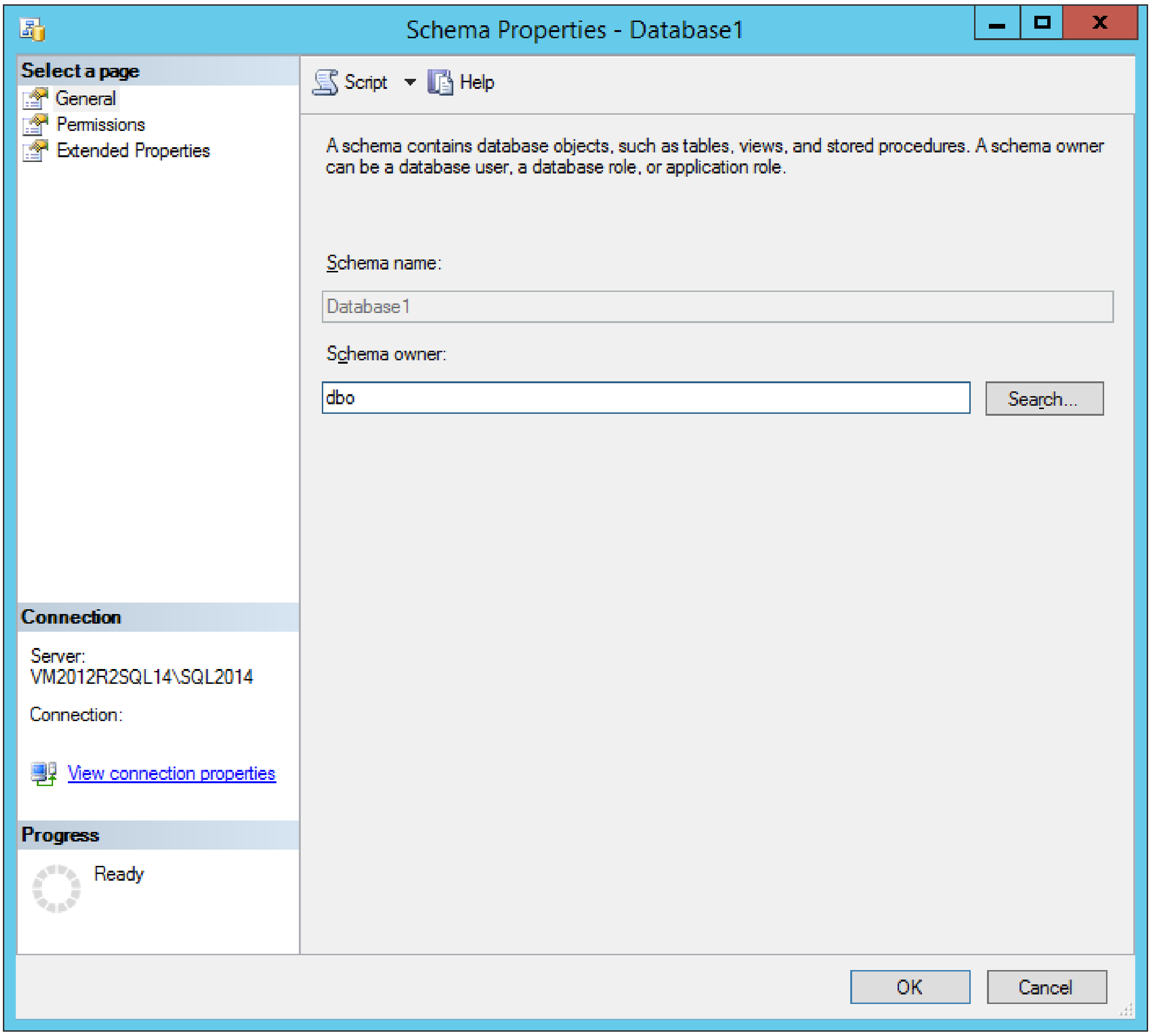
Ready

Click OK.

Open the schema **Database1**



The following screen will be shown:



A schema contains database objects, such as tables, views, and stored procedures. A schema owner can be a database user, a database role, or application role.

Schema name: Database1

Schema owner: dbo

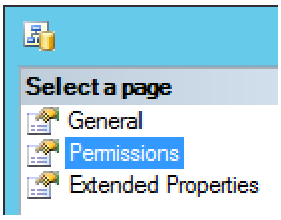
Server: VM2012R2SQL14\SQL2014

Connection:

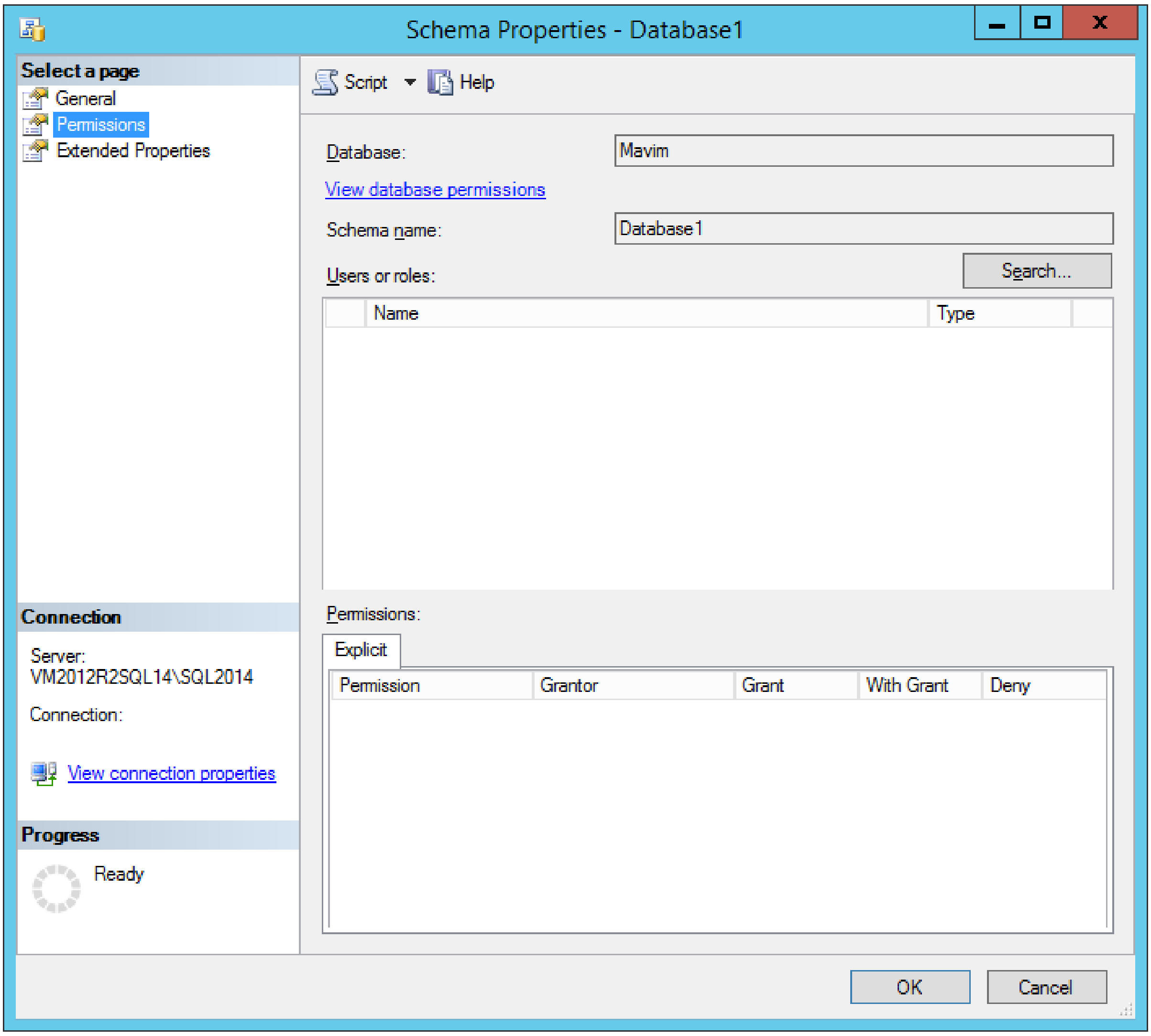
View connection properties

Ready

Click on Permissions



The following screen will be shown:



Database: Mavim

View database permissions

Schema name: Database1

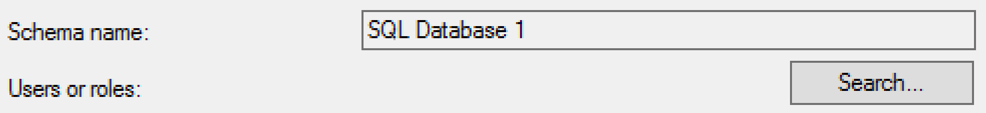
Users or roles:

Server:
VM2012R2SQL14\SQL2014

Connection:

Ready

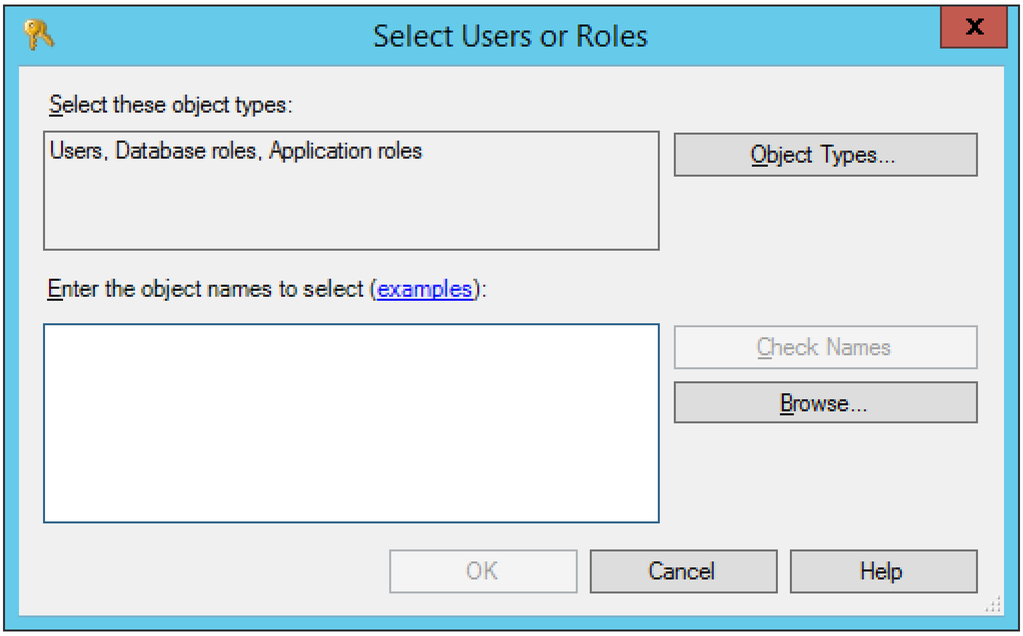
Click on **Search...**



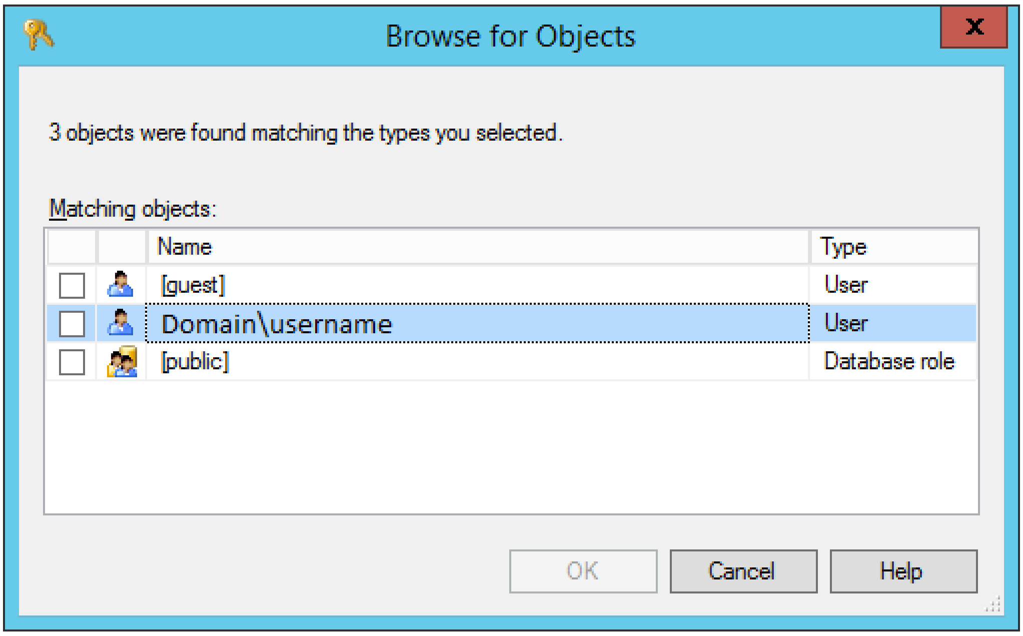
Schema name: SQL Database 1

Users or roles:

The following screen will be shown:


Select Users or Roles
Select these object types:
Users, Database roles, Application roles
Object Types...
Enter the object names to select (examples):
Check Names
Browse...
OK
Cancel
Help

Click on **Browse...**

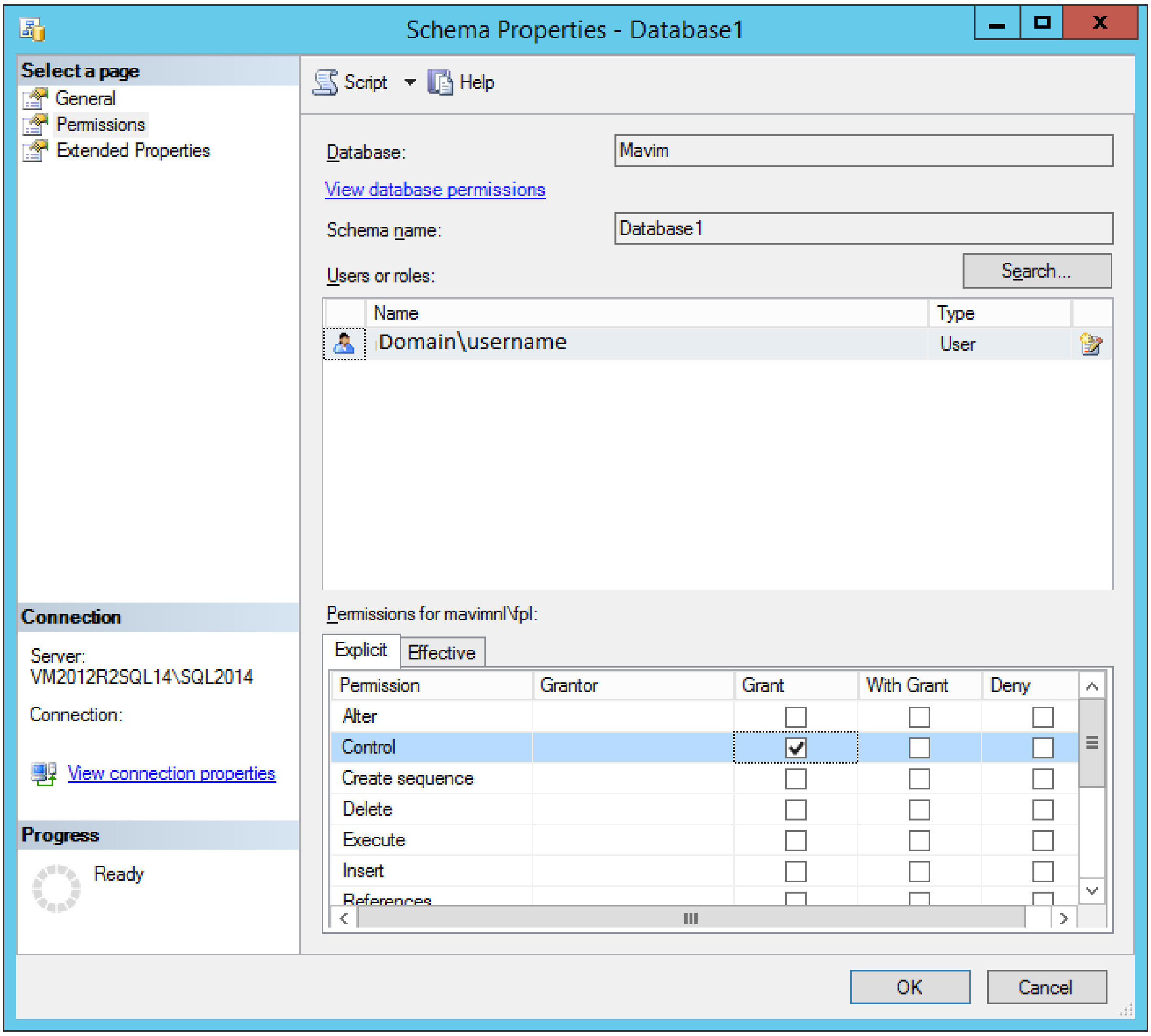


3 objects were found matching the types you selected.

Matching objects:

		Name	Type
□		[guest]	User
□		Domain\username	User
□		[public]	Database role

Select the user who has to have access to the Mavim database and click OK.



Database: Mavim

View database permissions

Schema name: Database1

Users or roles:

Name	Type
Domain\username	User

Permissions for mavimnl\fpl:

Explicit | Effective

Permission	Grantor	Grant	With Grant	Deny
Alter		□	□	□
Control		☑	□	□
Create sequence		□	□	□
Delete		□	□	□
Execute		□	□	□
Insert		□	□	□
References		□	□	□

Server:
VM2012R2SQL14\SQL2014

Connection:

View connection properties

Ready

Give (Grant) the user **Control** right and click OK.

The user that has to be able to connect to the Mavim database, has to have **Connect** rights on the database and **Control** rights on the schema.

This user can now add the Mavim database to the Database Manager and open the database.

When a user has to be able to perform one of the following tasks:

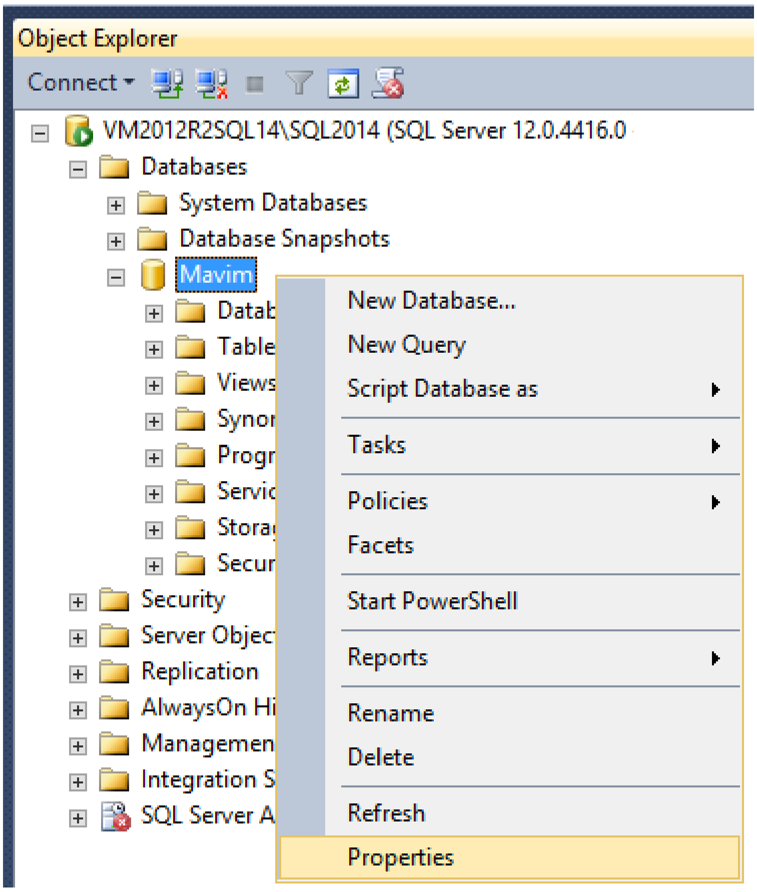
- Create a new (Mavim) database in SQL Server
- Create a (Mavim) database from template in SQL Server
- Check (and repair) the database

The user has to have, next to **Connect** right on the database and **Control** right on the schema, the following **database** rights:

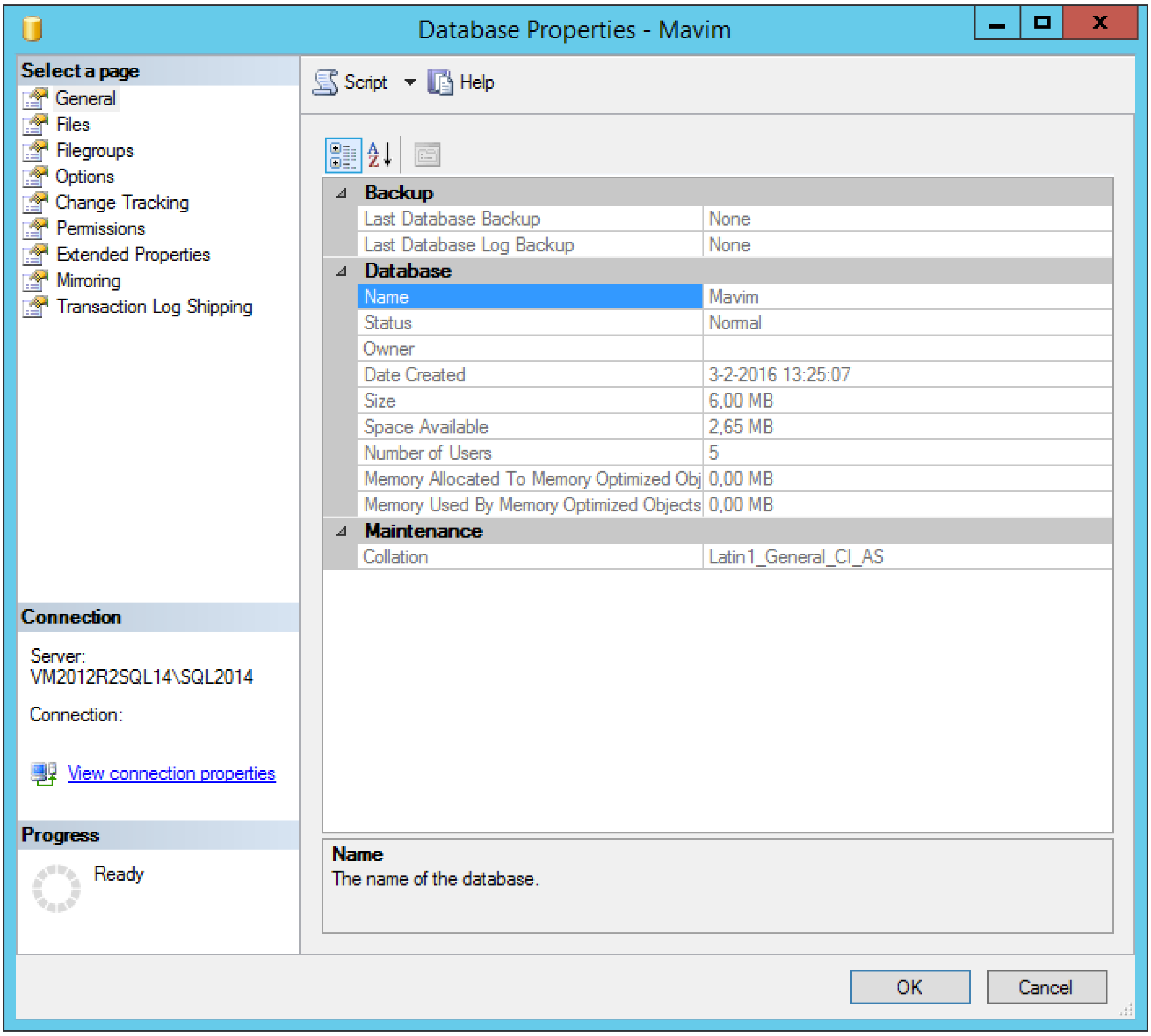
- Create Function
- Create Procedure
- Create Table

Follow the next steps to configure these rights.

Select the database **Mavim**, open the context menu and open **Properties**.



The following screen will be shown:



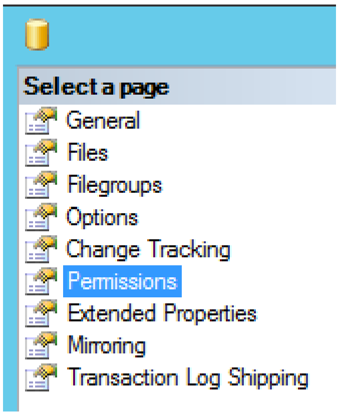
Server:
VM2012R2SQL14\SQL2014

Connection:

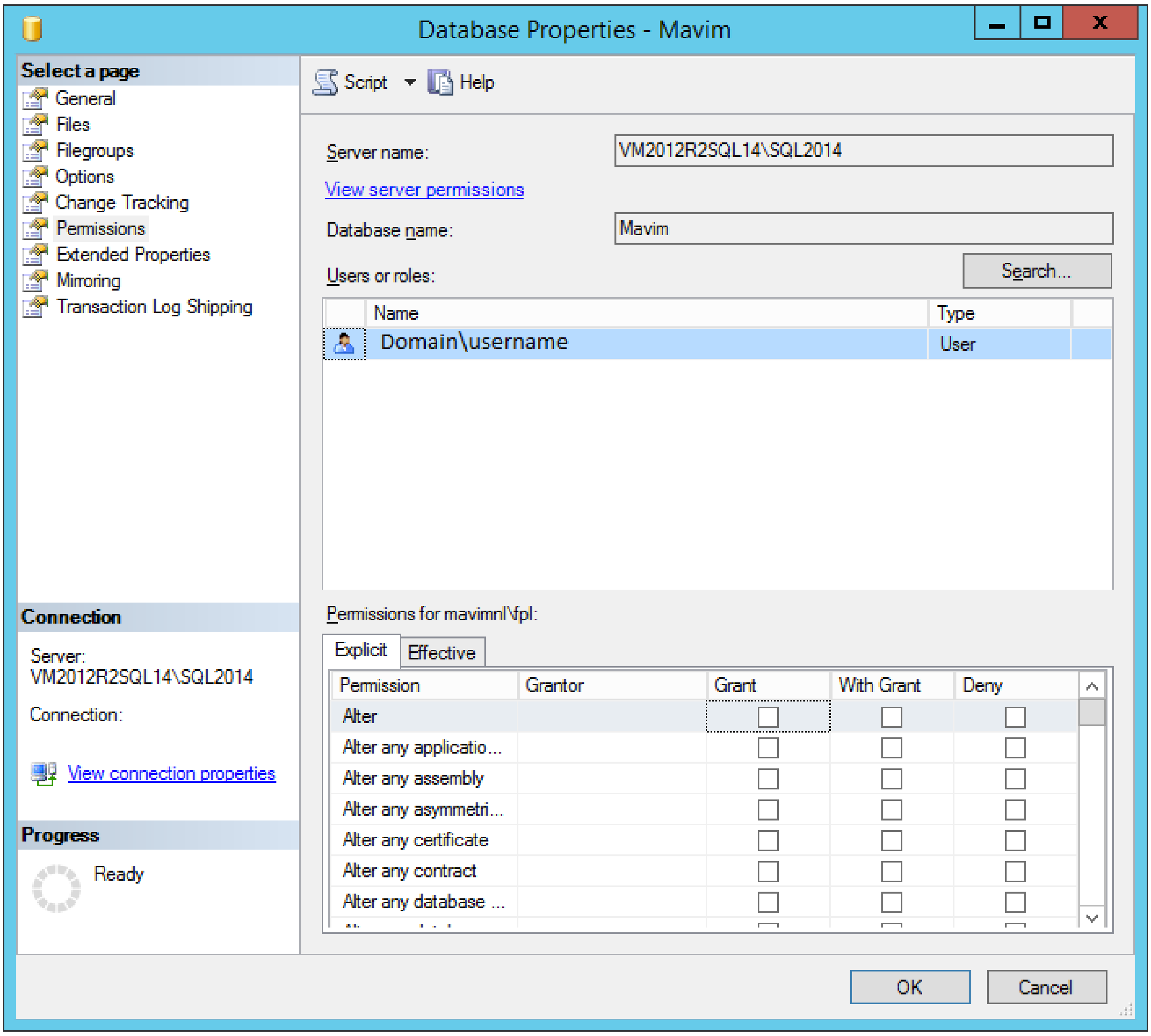
Backup	
Last Database Backup	None
Last Database Log Backup	None
Database	
Name	Mavim
Status	Normal
Owner	
Date Created	3-2-2016 13:25:07
Size	6,00 MB
Space Available	2,65 MB
Number of Users	5
Memory Allocated To Memory Optimized Obj	0,00 MB
Memory Used By Memory Optimized Objects	0,00 MB
Maintenance	
Collation	Latin1_General_CI_AS

Name
The name of the database.

Select Permissions



The following screen will be shown:



Server name: VM2012R2SQL14\SQL2014

Database name: Mavim

Users or roles:

Name	Type
Domain\username	User

Permissions for mavimnl\fpl:

Permission	Grantor	Grant	With Grant	Deny
Alter		☐	☐	☐
Alter any applicatio...		☐	☐	☐
Alter any assembly		☐	☐	☐
Alter any asymmetri...		☐	☐	☐
Alter any certificate		☐	☐	☐
Alter any contract		☐	☐	☐
Alter any database ...		☐	☐	☐

Connection

Server:
VM2012R2SQL14\SQL2014

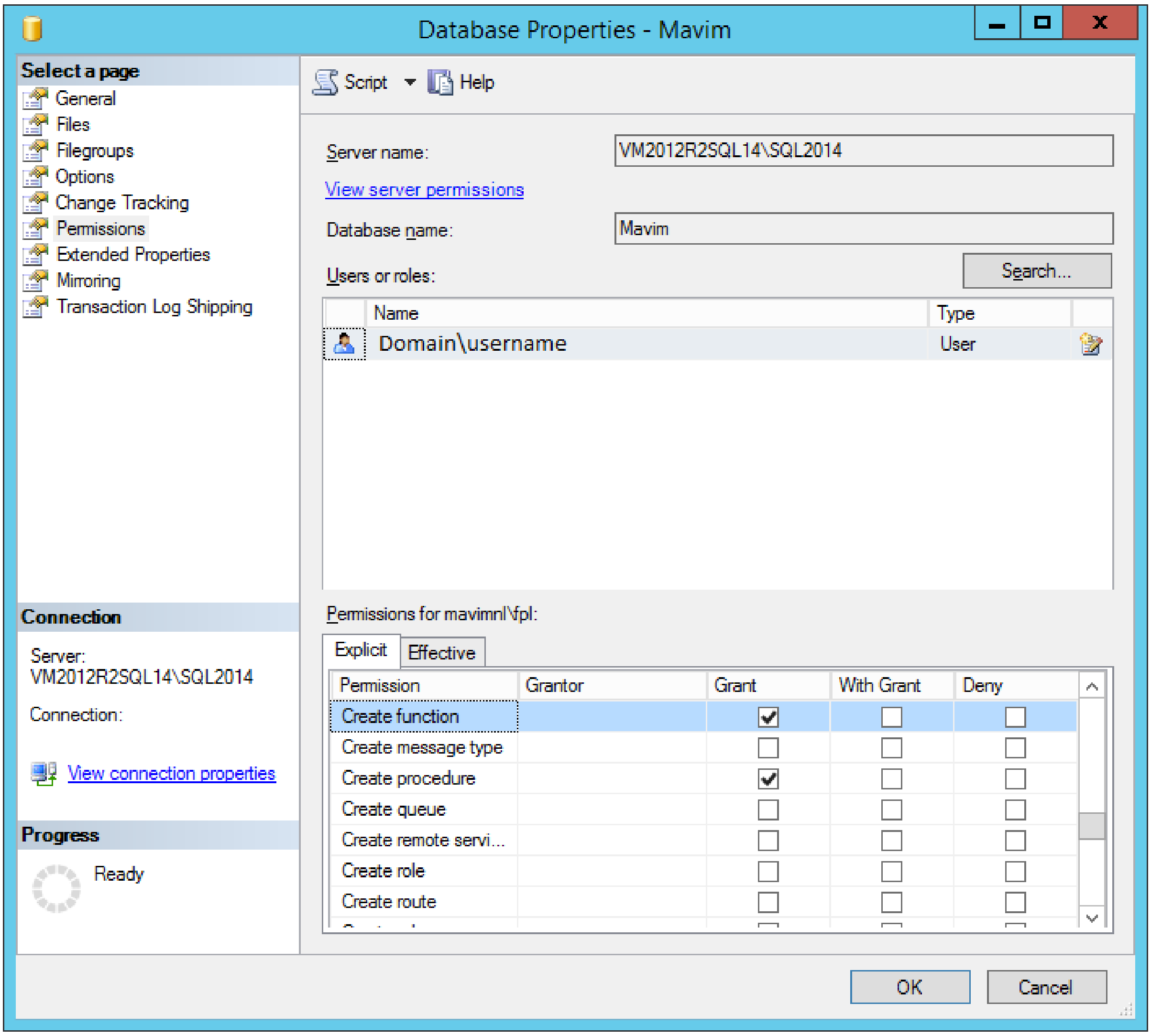
Connection:

Progress

Ready

Give (Grant) the following rights to the user:

- Create Function
- Create Procedure
- Create Table



Server name: VM2012R2SQL14\SQL2014

Database name: Mavim

Users or roles:

Name	Type
Domain\username	User

Server: VM2012R2SQL14\SQL2014

Connection:

Permissions for mavimnl\fpl:

Permission	Grantor	Grant	With Grant	Deny
Create function		☑	□	□
Create message type		□	□	□
Create procedure		☑	□	□
Create queue		□	□	□
Create remote servi...		□	□	□
Create role		□	□	□
Create route		□	□	□

Click OK.

The user now has sufficient rights and is able to execute the following functions from the developer environment:

- Create a new Database in SQL Server
- Create database from template in SQL Server
- Check (and repair) the database

To automatically resolve the error: "*Cannot create a new publication. Multiple users are logged in as <username>. Logout one of the <username> and try again.*".

Assign (Grant) the server permission "View server state" to a user or group. When this user logs in with Mavim Manager to the Mavim database, it automatically resolves the error message as shown above. If at least one user has this permission, this user can resolve the error message for any other blocked user.

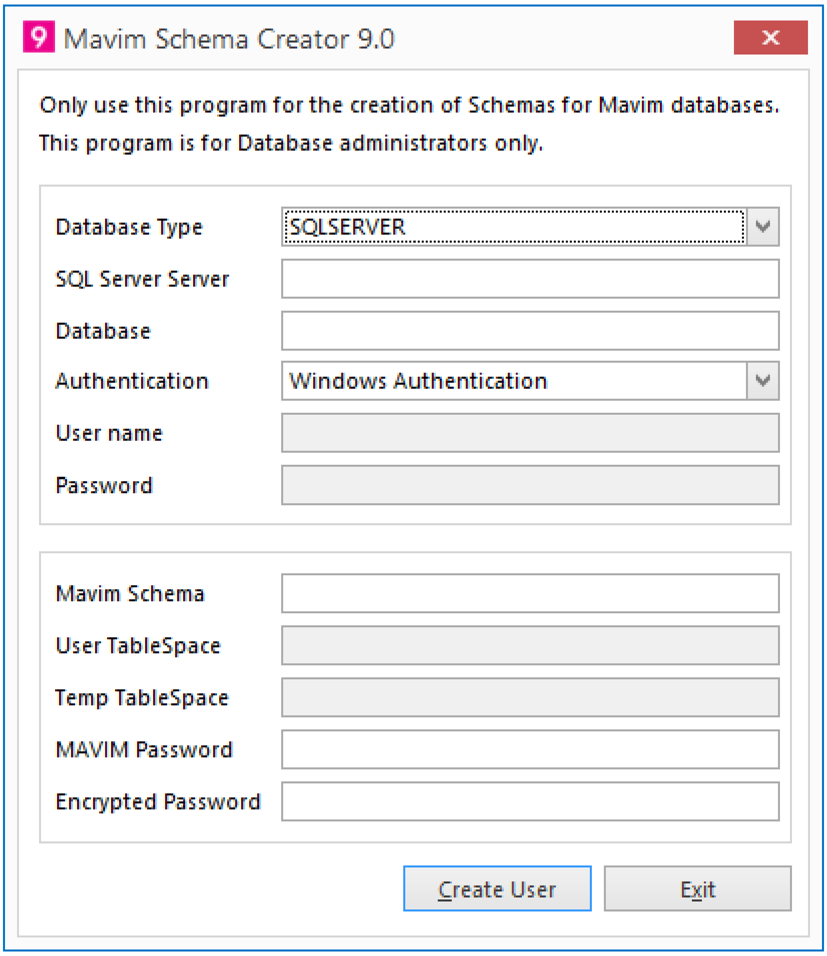
2 CREATING A SCHEMA IN SQL SERVER WITH MIXED MODE OR IN ORACLE

To use a physical database on an SQL Server or Oracle Database Server a Schema must be created. For this you must use a separate tool which is installed separately from Mavim Manager. The name of this tool is "Mavim.SchemaCreator.exe" and it can be found in "C:\Program Files (x86)\Mavim\Manager10\Schema Creator" (64-Bit operating system) of "C:\Program Files (x86)\Mavim\Manager10\Schema Creator" (32-Bit operating system).

For the Mavim database format it is not necessary to create a schema. You can create a new database immediately.

To connect to a Oracle database, an EZCONNECT connection string must be used. The connection string is built up as follows: servername/databasename notice the forward slash and then select the schema.

Depending on whether you work with an SQL server or an Oracle database server, you must make the appropriate selection for the top field. Below is an example for an SQL server:



Only use this program for the creation of Schemas for Mavim databases.
This program is for Database administrators only.

Database Type: SQLSERVER

SQL Server Server:

Database:

Authentication: Windows Authentication

User name:

Password:

Mavim Schema:

User TableSpace:

Temp TableSpace:

MAVIM Password:

Encrypted Password:

The fields are to be filled in as follows:

- SQL Server Server: The server that is running Microsoft SQL Server
- Database: The name of the database that will be used

- DBA User: The database use that will be used to create the schema
- Password: The password for the DBA User
- Mavim Schema: The name of the of the schema. Note that if you are re-creating the schema for an existing database, use must fill in the same schema name here.
- MAVIM Password: The "MAVIM Password" is user-definable. This password is assigned to the user in the physical database. The user created in the database is the "MAVIM Schema".

Click Create User and the schema will be created. The Schema Creator can then be closed, or used to create further schemas.

Result from the Schema Creator

After you use the Schema Creator, the following objects have been created in the physical database:

- An empty schema that will be used for Mavim Manager. The schema user must have enough rights to create the table itself, stored procedures, indexes etc. on its own schema. Mavim Manager logs in as this user and creates these objects itself.
- A MAVIMUSER schema with MAVIMUSER as password and a table called MAVIMUSER. This table is used to store the encoded password. MAVIM logs in as this user to collect the encoded password. The MAVIMUSER must have at least read rights to the table.

  The Schema Creator first checks if these schemas already exist. If the MAVIMUSER table already exists, then only a new record with a new password is added. If the MAVIM schema already exists, then the password is changed.

  Without a record in the MAVIMUSER table, Mavim Manager cannot contact the database.

Changing the password of the MAVIMUSER on SQL Server

The password of the MAVIMUSER on SQL Server is changeable from Mavim Rules version 8.1.338.

After changing the password of MAVIMUSER the Windows user account of the Mavim user has to have read (select) rights on the table MAVIMUSER.MAVIMUSER to open the Mavim database. When these rights are set correct the database can be opened from Rules version 8.1.338. Each Windows user who has to open the Mavim database has to have read (select) rights on the table MAVIMUSER.MAVIMUSER. This read right is needed for reading the password for login on the schema used by the database and to use for further access to the Mavim database.

Only the Mavim Rules versions higher than 8.1.338 can cope with a changed MAVIMUSER password. Previous versions of Mavim Rules will give an error: “Cannot open database: Undefined Error”. In that case, install the Mavim Rules version 8.1.338 or higher or start the Schema Creator older than 8.1.338 to automatically correct the problem. To do this, start the Schema Creator, enter the data for the specific database and click on the button Create user to correct the problem.

3 SETUP SINGLE SIGN ON

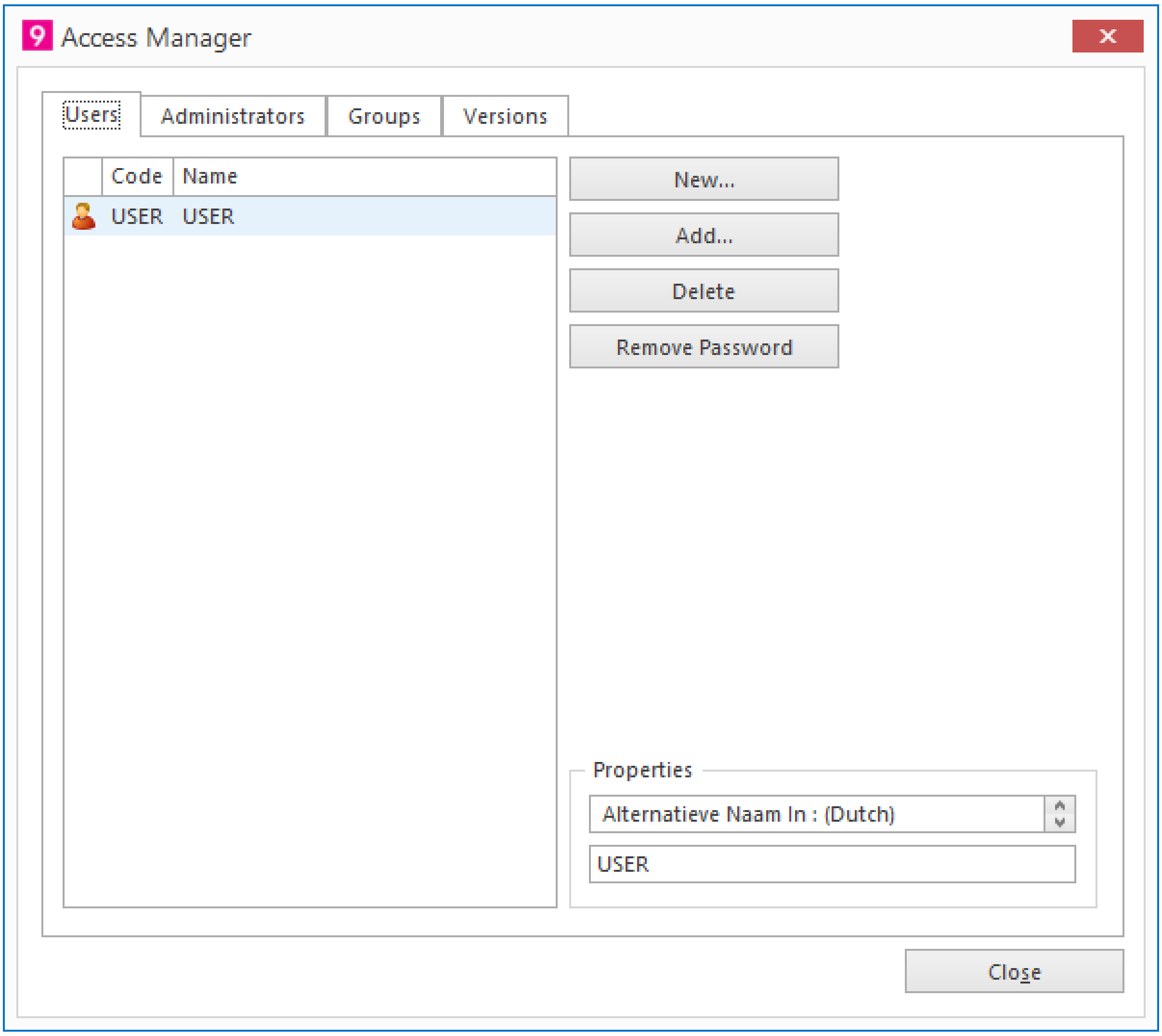
The setup of Single Sign On with Windows Authentication is configured as follows:

> Single Sign On only works when the computer is added to a Windows Domain and the Domain server can be reached.

Open the Mavim database as an Administrator.

Open from the Tab Administration, the option Access Management...

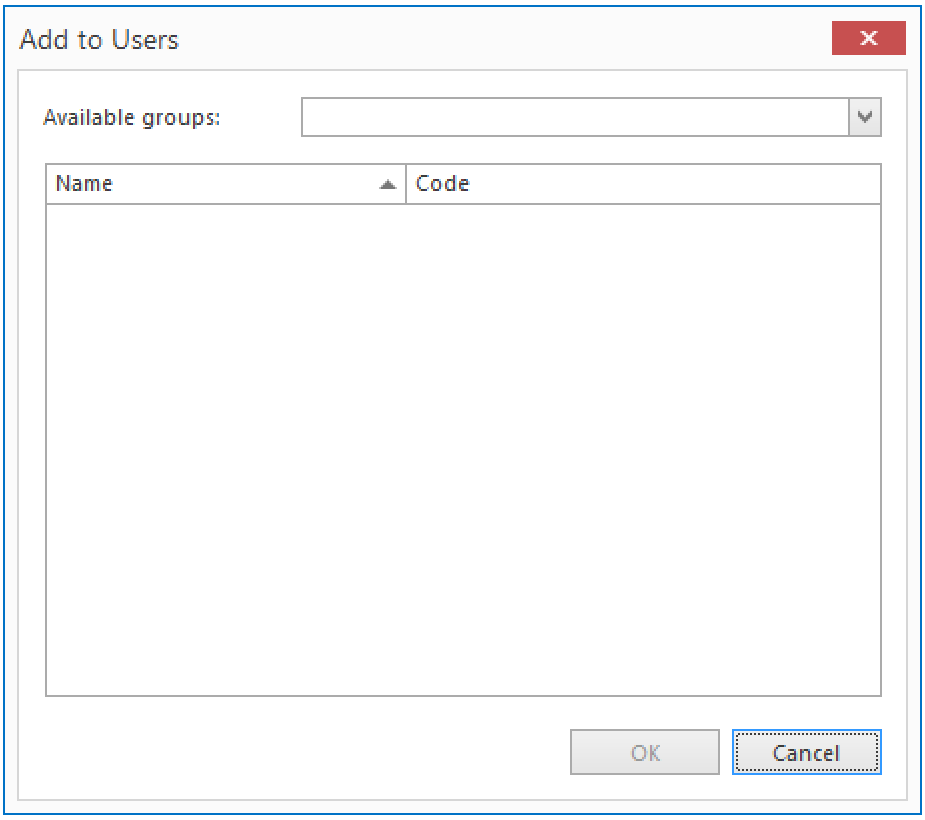
The following screen will be shown:



Click **Add...** to Add users from the Active Directory.

It is not possible to change the Active Directory from the developer environment (for example removing passwords or create new Active Directory users). When New... is clicked, a new user is created in the Database.

The following screen will be shown:



Select from the list of Available groups the group which contains the user.

Select one or more users from the list and click OK.

When a Azure SQL Database is used, you also have the ability to add Azure SQL Users.

To disable this, open the registry at the following location:

HKEY_LOCAL_MACHINE\SOFTWARE\Wow6432Node\Mavim\Manager\10

or

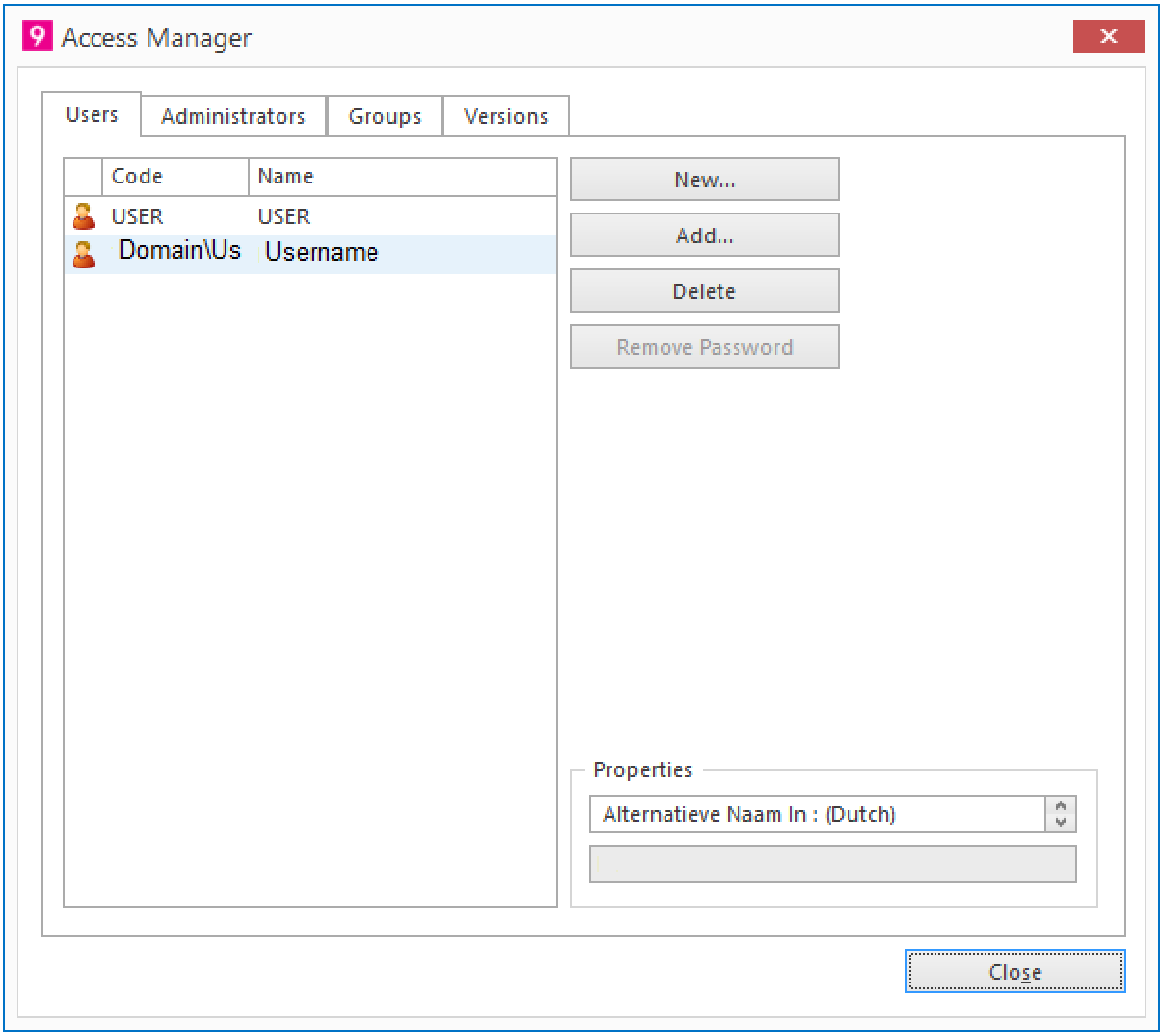
HKEY_LOCAL_MACHINE\Software\Mavim\Manager\10\General

Add the following String Value:

"UseSQLAuthorization"

And set the value to:

"False"



Code	Name
USER	USER
Domain\Us	Username

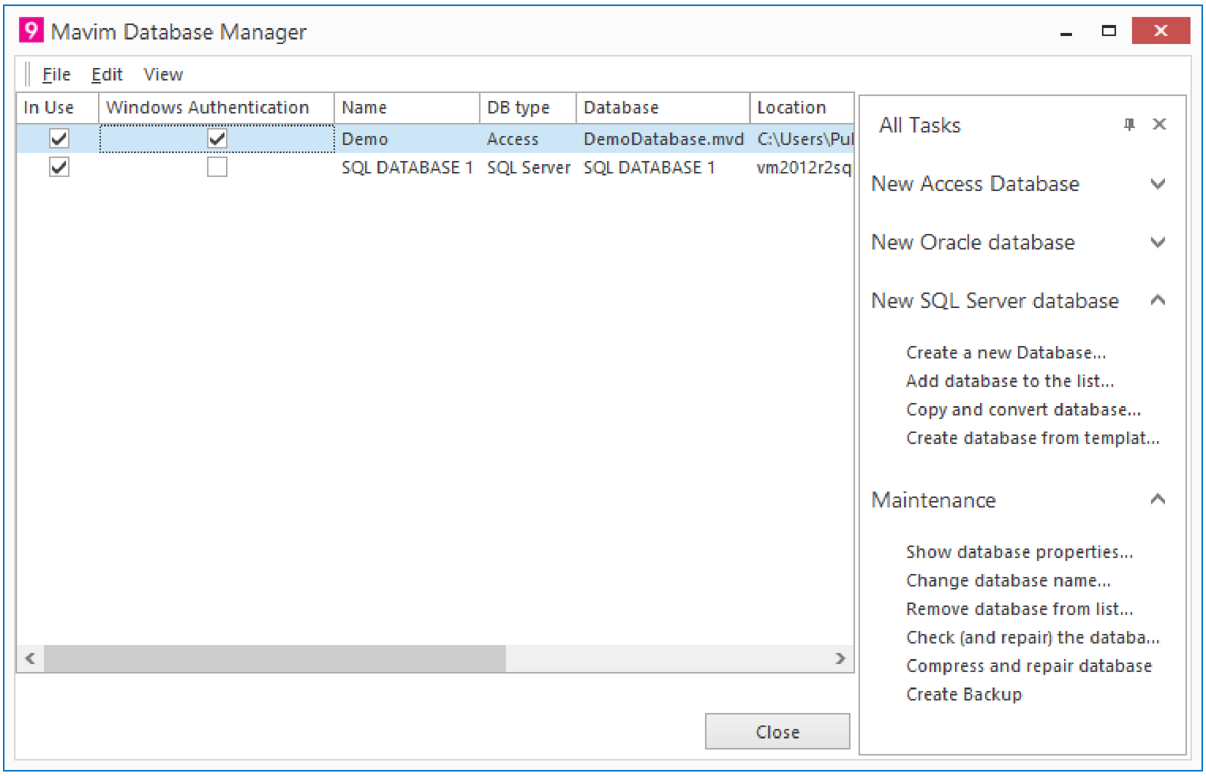
Properties

Alternatieve Naam In : (Dutch)

You can now remove the user account 'USER' so that the database cannot be opened from this account.

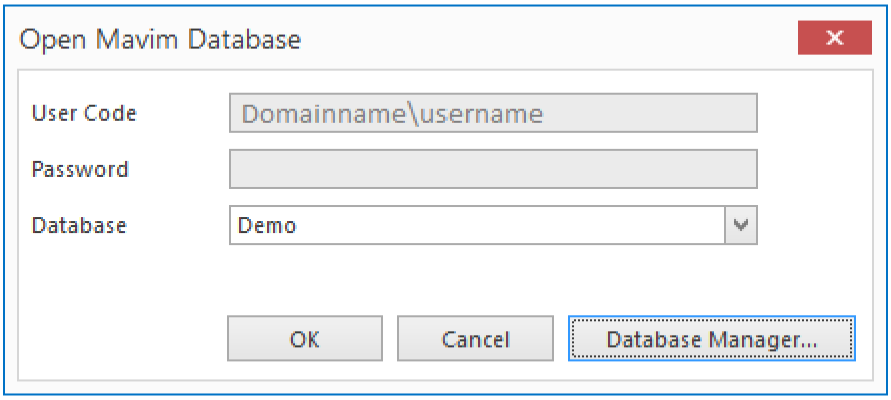
Click Close.

Open the Database Manager and select Windows Authentication.


Mavim Database Manager
File
Edit
View
All Tasks
New Access Database
New Oracle database
New SQL Server database
Create a new Database...
Add database to the list...
Copy and convert database...
Create database from templat...
Maintenance
Show database properties...
Change database name...
Remove database from list...
Check (and repair) the databa...
Compress and repair database
Create Backup
Close

In Use	Windows Authentication	Name	DB type	Database	Location
☑	☑	Demo	Access	DemoDatabase.mvd	C:\Users\Pul
☑	□	SQL DATABASE 1	SQL Server	SQL DATABASE 1	vm2012r2sq

After selecting the option Windows Authentication, the login screen will show the username of the user which is logged in.



User Code: Domainname\username

Password:

Database: Demo

Click OK to login using Single Sign On.

The Windows Authentication setting is stored in the MVMDB.ini file which is located in the profile of the current user.

4 USING AZURE SQL DATABASE

4.1 Introduction

Starting from Mavim Rules 9.2 you can make use of an Azure SQL Database.

To realise a connection to an Azure SQL Database, the following settings are important:

The Network connection to the Azure SQL Database

The Performance level of the Azure SQL Database

4.2 Network connection

The network connection to an Azure SQL Database has to comply with the network connection requirements as specified in the document: 'System and configuration requirements Mavim.pdf'.

It is important that you should only make use of an Azure SQL Database when you are using a Server in Azure where Mavim software is installed on. And both are located in the same Azure datacenter region.

4.3 Performance level

For the most optimal user experience you have to set the correct Performance Level for the Azure SQL database.

For determination of the correct performance level, you should make use of the Microsoft Azure SQL Database DTU Calculator:

http://dtucalculator.azurewebsites.net/

Notice that this only is an advice. While using the Azure SQL Database we advise to gather feedback from the users to reach to most optimal Performance level.

Microsoft advises to monitor and evaluate the Azure SQL Database performance. Read more about this in the following Blog:

https://azure.microsoft.com/en-gb/blog/azure-sql-database-introduces-new-near-real-time-performance-metrics/

4.4 Creating Azure SQL Database in Azure

You can create an Azure SQL Database as follows:

Using the Azure Portal:
As described in the following Microsoft documentation:

https://azure.microsoft.com/en-us/documentation/articles/sql-database-get-started/

Using SQL statements:
As described in the following Microsoft documentation:

https://msdn.microsoft.com/en-us/library/dn268335.aspx

4.5 Create Schema in Azure SQL Database

After creating an Azure SQL Database in Azure, a Schema has to be created for the Mavim Database.

Creating a Schema is described in the following Microsoft documentation:

https://msdn.microsoft.com/en-us/library/ms189462.aspx

The name of the Schema appears in the dropdown list in the development environment of the user.

Example:

Create schema for Mavim Database:

```
CREATE SCHEMA [MavimDatabase] AUTHORIZATION dbo;
```

Grant Control rights for schema

```
GRANT CONTROL ON SCHEMA::[MavimDatabase] TO mavimdbuser;
```

Grant View Database State

```
GRANT VIEW DATABASE STATE TO mavimdbuser;
```

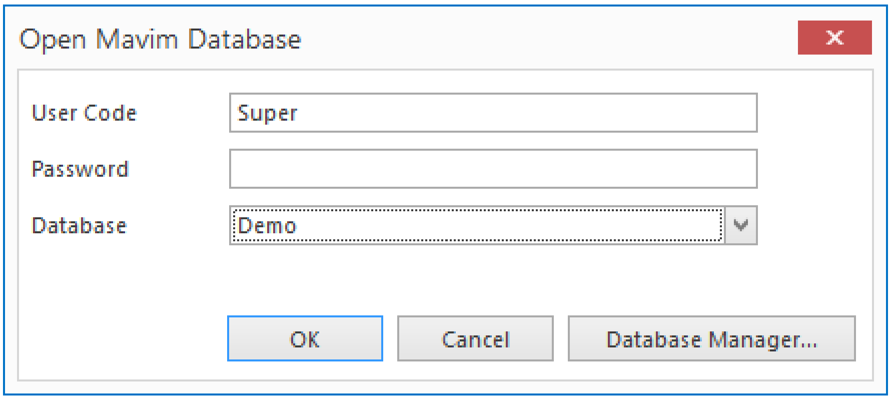
To automatically resolve the error: "*Cannot create a new publication. Multiple users are logged in as <username>. Logout one of the <username> and try again.*".

Assign (Grant) the server permission "View database state" to a user or group. When a user logs in with Mavim Manager to the Mavim database, it automatically resolves the error message as shown above. If at least one user has this permission, this user can resolve the error message for any other blocked user.

4.6 Creating a Mavim Database

When the Azure SQL Database is available, follow the steps below using Mavim Manager software:

1. Start Mavim Manager and open the **Database Manager...**



Open Mavim Database

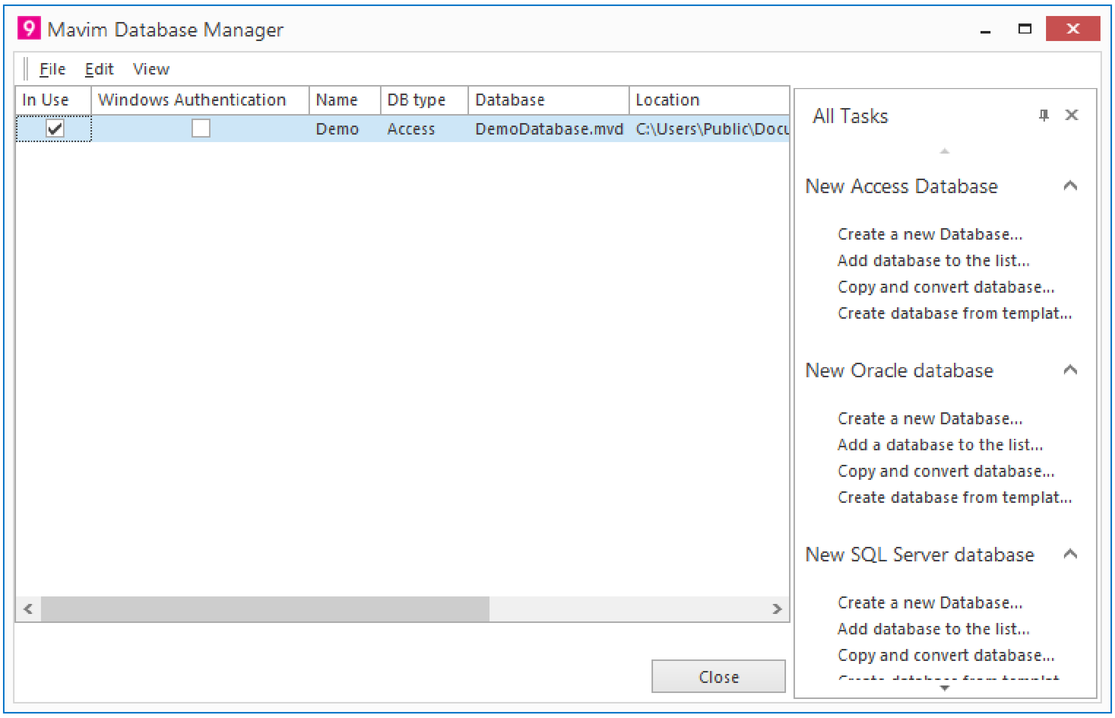
User Code: Super

Password:

Database: Demo

2. Click on **Database Manager...**

3. The following screen will be shown:

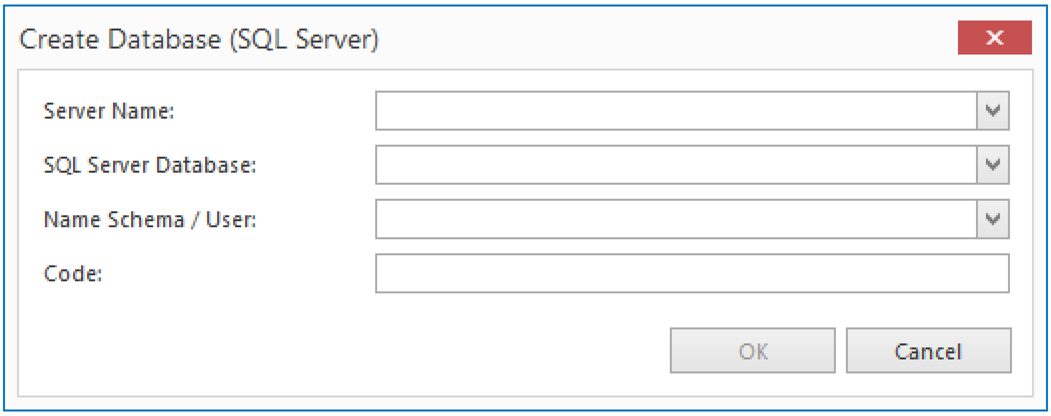


Mavim Database Manager
File
Edit
View
All Tasks
New Access Database
Create a new Database...
Add database to the list...
Copy and convert database...
Create database from templat...
New Oracle database
Create a new Database...
Add a database to the list...
Copy and convert database...
Create database from templat...
New SQL Server database
Create a new Database...
Add database to the list...
Copy and convert database...
Close

In Use	Windows Authentication	Name	DB type	Database	Location
☑	□	Demo	Access	DemoDatabase.mvd	C:\Users\Public\Docu

4. Click at **New SQL Server-database** on **Create a new database ...**

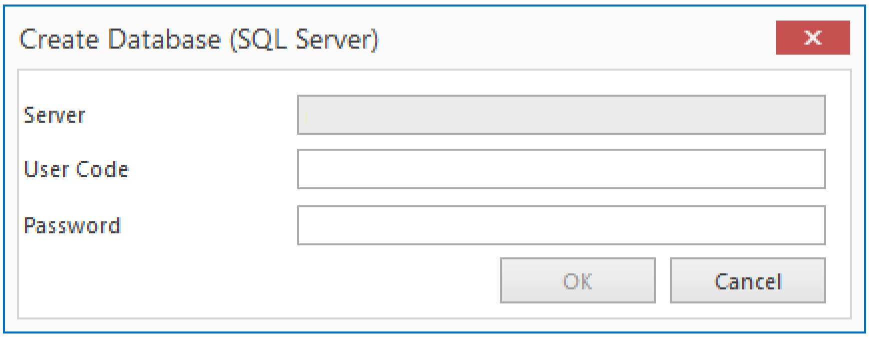
5. The following screen will be shown:



6. In this screen, enter the following information:

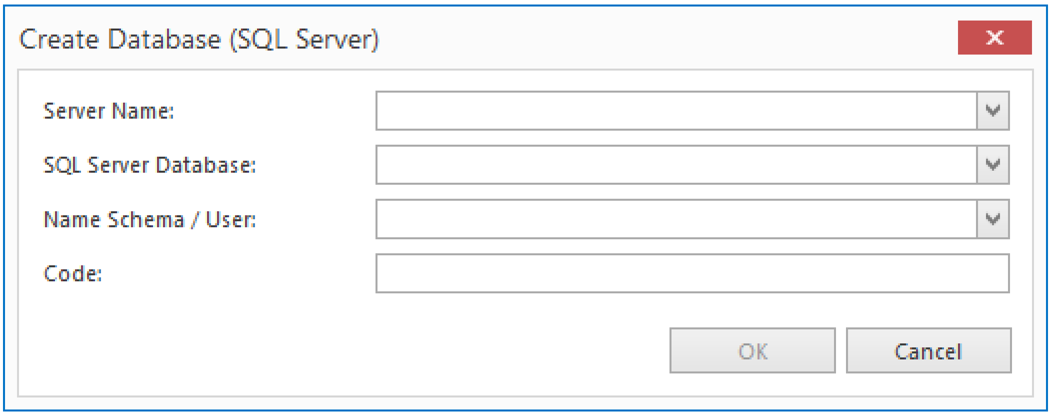
   - Server Name: The name of the Azure SQL Server.
   - SQL Server Database: The name of the Azure SQL Database.
   - Name Schema / User: The empty schema.

   After selecting Name Schema / User a login screen will be shown:



Enter at User Code, the login name and at Password the password for the Azure SQL Database and click OK.

7. The following screen will be shown:



Enter at Code, the code for the new database

8. And click OK

The database is now ready for use.

4.7 Setup configuration file for Single Sign On

Single Sign On can be used in combination with an Azure SQL Database.

> To make use of Single Sign On on a Azure SQL Database, it is mandatory to have the Azure Active Directory synchronised with the On Premise Active Directory.

To enable Single Sign On can be enabled, open the registry at the following location:

HKEY_LOCAL_MACHINE\Software\Mavim\Manager\10\General

or

HKEY_LOCAL_MACHINE\SOFTWARE\Wow6432Node\Mavim\Manager\10\General

Add the following String Value:

"UseAzureIntegratedLogin"

And set the value to:

"True"

Single Sign On is now Enabled. Follow the steps below to finalize the configuration:

Setup Single Sign On

5 MULTI FACTOR AUTHENTICATION

If your account in your Azure environment is set to use Multi Factor Authentication (MFA), you will need to add some registry keys to your local machine to allow you to make use of MFA.

The following keys are needed for SQL databases hosted in Azure:

\HKEY_LOCAL_MACHINE\SOFTWARE\Mavim\Manager\10\General

Key: AzureClientId
Value: You will need to find the ID in the Azure environment

Key: AzureTenant
Value: The URL of your tenant (excluding http://)

Key: UseAzureIntegratedLoginWithMFA
Value: 1

The following keys are needed for a Mavim Portal hosted in Azure:

\HKEY_LOCAL_MACHINE\SOFTWARE\Mavim\Manager\10\Portal

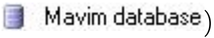
Key: AzureClientId
Value: You will need to find the ID in the Azure environment

Key: AzureTenant
Value: The URL of your tenant (excluding http://)

Key: UseAzureIntegratedLoginMFA
Value: 1

6 TECHNICAL MAINTENANCE OF MAVIM DATABASES

We recommend the following when performing maintenance on the database:

Ensure that the database is **closed**. Do this by opening the Properties of the database (F4 on  Mavim database) and checking the Connections tab to see if anyone currently has the database open. If not, you can close and re-open Mavim. Instead of opening a database select the Database Manager immediately. At this point no database has been opened but you can select the database you want to check.

For large databases, a lot of disk space is used to carry out the checks. The amount of disk space needed depends on the exact size of the database and the options chosen. Assume that the disk space required will be no more than three times the size of the database. Ensure that there is enough free disk space available.

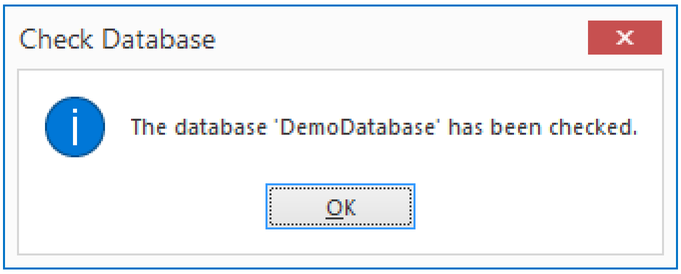
For SQL Server and Oracle, the Database Manager is responsible for maintaining the databases.

6.1 Checking (and repairing) a database

It is possible to check the database contents. This is necessary if the Mavim database contains errors, or if a newer version of the Mavim Manager application has been installed (this option adapts the database to the newer version).

To check the database contents:

1. Start the **Database Manager** (see Database Manager).
2. Select the database you want to check.
3. Select menu **File>Check Mavim Database**.
4. You are presented with a dialog asking if you are sure you want to check the database because this can be a lengthy operation.
5. Click **Yes**.
6. The database will now be checked and repaired if needed. Once the check is complete you will receive a message similar to the one shown below:



Check Database

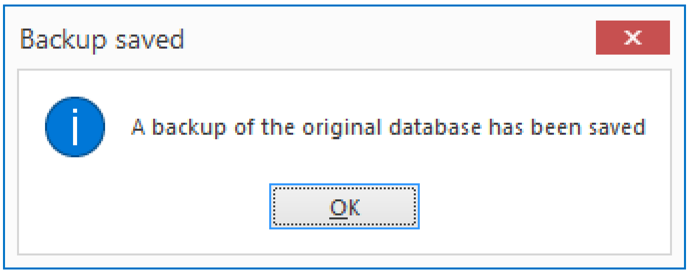
The database 'DemoDatabase' has been checked.

6.2 Creating a Database Backup

It is important to create backups of the database regularly.

You can manually create a backup of an Access database as follows:

1. Open the Database Manager (see: Database Manager).
2. Select the database for which you want to create a backup.
3. Select the **Create Backup** option in the Maintenance section of the Database Manager.
4. Navigate to the location where you want to store the backup and click **Save**.
5. You will receive the following message once the backup is complete:



Backup saved

A backup of the original database has been saved

6. Click OK to return to the Database Manager.

6.3 Converting a database

Mavim Manager provides the option of copying and converting a database for use on another platform. As a result you will have both the original database and a copy of the database.

Aside from creating a copy of a database, you can also create a backup. See:

- Make a backup of a database

6.3.1 Make a backup of a database

It is possible to make a copy of the database from within Mavim (see Creating a Database Backup), but this must not be considered a safe backup method.

The creation of backups must be structured as a procedure. The procedure ensures the automatic periodic copying of the database without human intervention and the errors this can generate.

It is advisable to place Mavim databases on a network drive and include these in a periodic backup procedure.

6.4 Upgrading databases

With the release of new versions of Mavim Manager, databases from previous versions will need to be upgraded. This is needed because for some newer versions of Mavim Manager, adjustments must to be made to the database structure to allow new functionality to work correctly.

From within the Database Manager there are a number of tools available including the Check (and repair) function which is used to upgrade databases. Generally if you try to open a database from a previous version of Mavim Manager you will receive a message stating that the database needs to be upgraded but you can also use the Check (and repair) function from the Database Manager to upgrade your databases.

Before performing an upgrade of your database it is extremely important to **create a backup of the database**. Should something go wrong you will then always be able to revert to the previous version. Please see Make a backup of a database for more information.

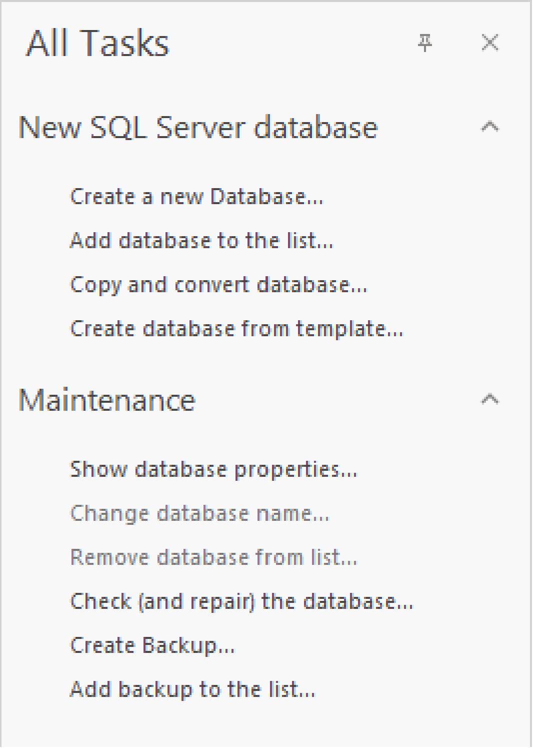
Follow the below steps to upgrade a database from within the Database Manager:

1. Ensure that all users are logged out of Mavim Manager

2. Make sure you have no other instances of Mavim Manager running

3. Start Mavim Manager but instead of logging in, go directly to the Database Manager

4. From the Database Manager, select the database you wish to upgrade (highlight) and click the "Check (and repair) the database" function from the available list on the right:



All Tasks

New SQL Server database

Create a new Database...
Add database to the list...
Copy and convert database...
Create database from template...

Maintenance

Show database properties...
Change database name...
Remove database from list...
Check (and repair) the database...
Create Backup...
Add backup to the list...

Your database will be upgraded and you can continue to work with it.

6.4.1 Sorting order of fields

In older versions of Mavim it was not possible to sort the fields assigned to a topic. As of version 7 this is possible and functionality has been put in place to allow this. It may be apparent that when upgrading a database from a previous version, the order in which the assigned fields are shown for a topic is different in the new version.

To support this feature there are some scenarios to take into consideration. Below is a table showing which actions need to be taken in order to have access to the field sorting functionality when migrating to Mavim Manager 10:

Current Database Version	Upgrade Database	Perform a Check on the Database	Use New Functionality to Sort Fields
Mavim Rules 9			X
Mavim Rules 8			X
Mavim Rules 7			X
Mavim Rules 6.2		X	X

Please note that performing an upgrade of the database automatically executes a check on the database and so the check does not need to be done separately.

7 MAKING DATABASE MANAGER INFORMATION AVAILABLE

The **Database Manager** contains a list of available databases. This list is maintained in the file **MVMDB.ini**. If you want a user to have access to a list of existing databases carry out the followings steps:

1. Place the **MVMDB.ini** file in a place on the network that is accessible to the user(s).
2. Set the location of the **MVMDB.ini** for each user at **Tools**>**Options...** at **Database Manager Information**.

8 RSS FUNCTION

With RSS, the users of your websites can subscribe to specific pages. When these pages are updated, they will be notified via their RSS reader.

To use the RSS function in Mavim Rules websites, it is necessary to install an RSS reader. Most browsers come with a built-in RSS reader.

Examples of commonly used RSS readers are:

Feedreader (http://www.feedreader.com) (Windows; freeware)

Sharpreader (http://www.sharpreader.net) (Windows; freeware)

AmphetaDesk (http://www.disobey.com/amphetadesk/) (Linux, Mac, Windows; freeware)

NewzCrawler (http://www.newzcrawler.com/) (Windows; shareware)

9 WINDOWS REGISTRY SETTINGS

Structure of the registry

User settings can be found under: HKEY_CURRENT_USER\Software\Mavim\Manager\10

Settings that affect the whole Mavim application can be found under:
HKEY_LOCAL_MACHINE\SOFTWARE\Wow6432Node\Mavim\Manager\10
or
HKEY_LOCAL_MACHINE\SOFTWARE\Mavim\Manager\10

Reading from the registry

For easy use of the registry, the following procedure applies to all read actions from the registry: a search is performed for a registry key in HKEY_CURRENT_USER, and if nothing is found there, in HKEY_LOCAL_MACHINE. If nothing is found after searching the second location then a value from Mavim is used.

> Note: If you have made adjustments in the HKEY_LOCAL_MACHINE these may not become active as there is not yet a key for these setting(s) in HKEY_CURRENT_USER.

Writing to the registry

The Mavim application always writes its settings to HKEY_CURRENT_USER\Software\Mavim\Manager\10 and will never write to HKEY_LOCAL_MACHINE. Only the installation program and administrators can change values in HKEY_LOCAL_MACHINE.

9.1 Mavim registry variables

To make the configuration of paths in the registry easier, two variables are provided:

- *MAVIM_APP_DIR* By completing this variable, you assign an alternative directory in which folders will be located instead of one assigned by the Mavim installation.
- MAVIM_APPDATA_DIR This variable pertains to the work folders for Visio Stencils, dictionaries, toolbar settings, Database Manager data, log files etc.
- *MAVIM_USER_DIR* This variable pertains to work folders for generated Word documents, import and export locations for transport databases, published websites etc.

You can use them directly in the registry by referencing them with Windows environment variables: %MAVIM_APP_DIR%, %MAVIM_APPDATA_DIR% and %MAVIM_USER_DIR%.

9.2 Mavim Portal

Limits have been set to the size of reports and the number of cells in a matrix to avoid Portals crashing. These limits can, in extreme cases, be changed or removed using the following registry keys:

[HKEY_CURRENT_USER\Software\Mavim\Manager\10\Portal]

MaxNumberOfCellsInMatrices
Values:

- 100000 (default)
- any number in number of cells
- 0 (no limit)

MaxSizeOfReportsInMb
Values:

- 128 (default)
- any size in MB
- 0 (no limit)

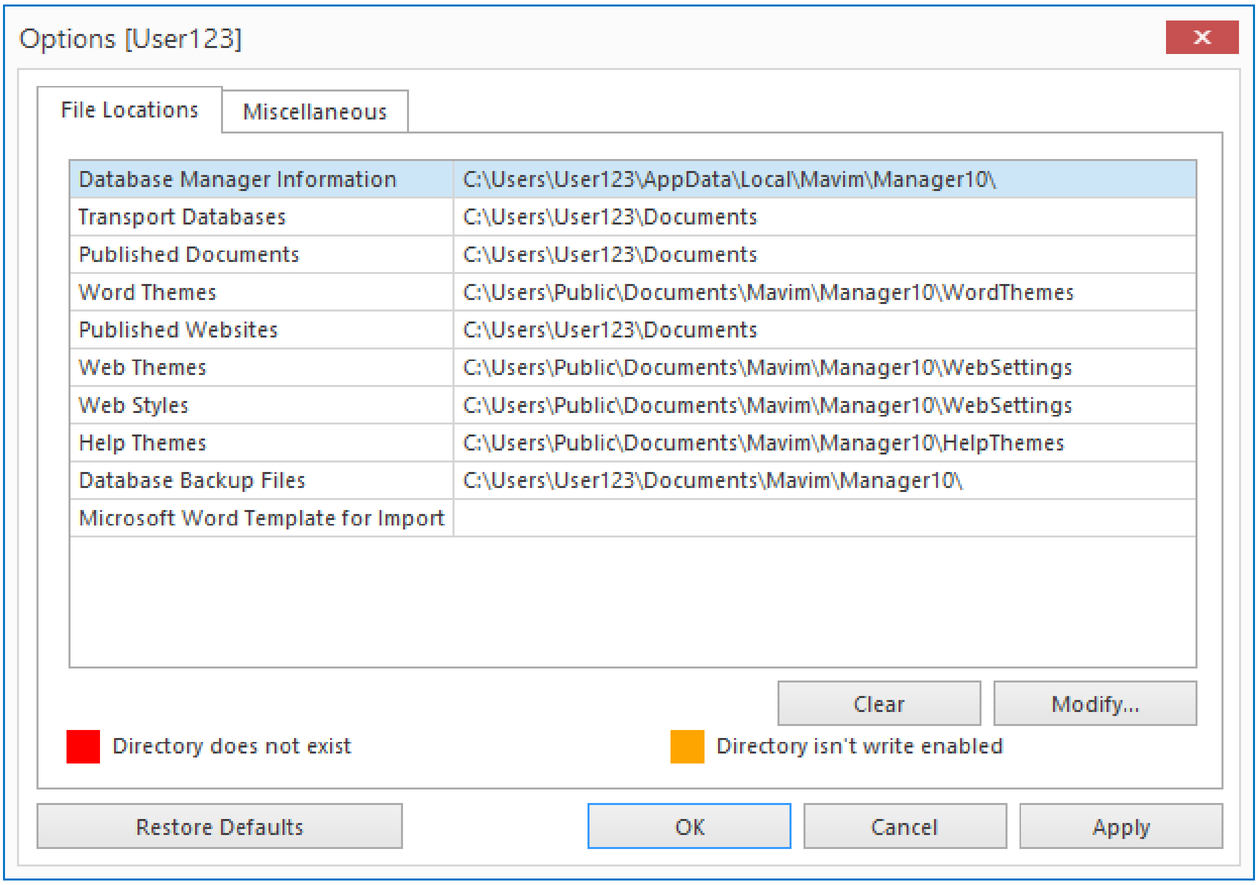
9.3 Changing registry settings

Change values in the registry

The installation program will install the registry keys under HKEY_LOCAL_MACHINE. If you subsequently want to change the settings, you can do this using a registry editor such as "regedit.exe" included with Windows.

Important registry settings

As mentioned above, users can change the user settings in Mavim themselves. The **Options** screen is used to do this (this can be accessed from the **Tools>Options** menu.) The following screenshot shows the **Options** screen with the **File Locations** tab active. This screen uses colours to show if incorrect settings have been made. Red text means that the path or file does not exist, yellow text means that it does exist but is not writable.



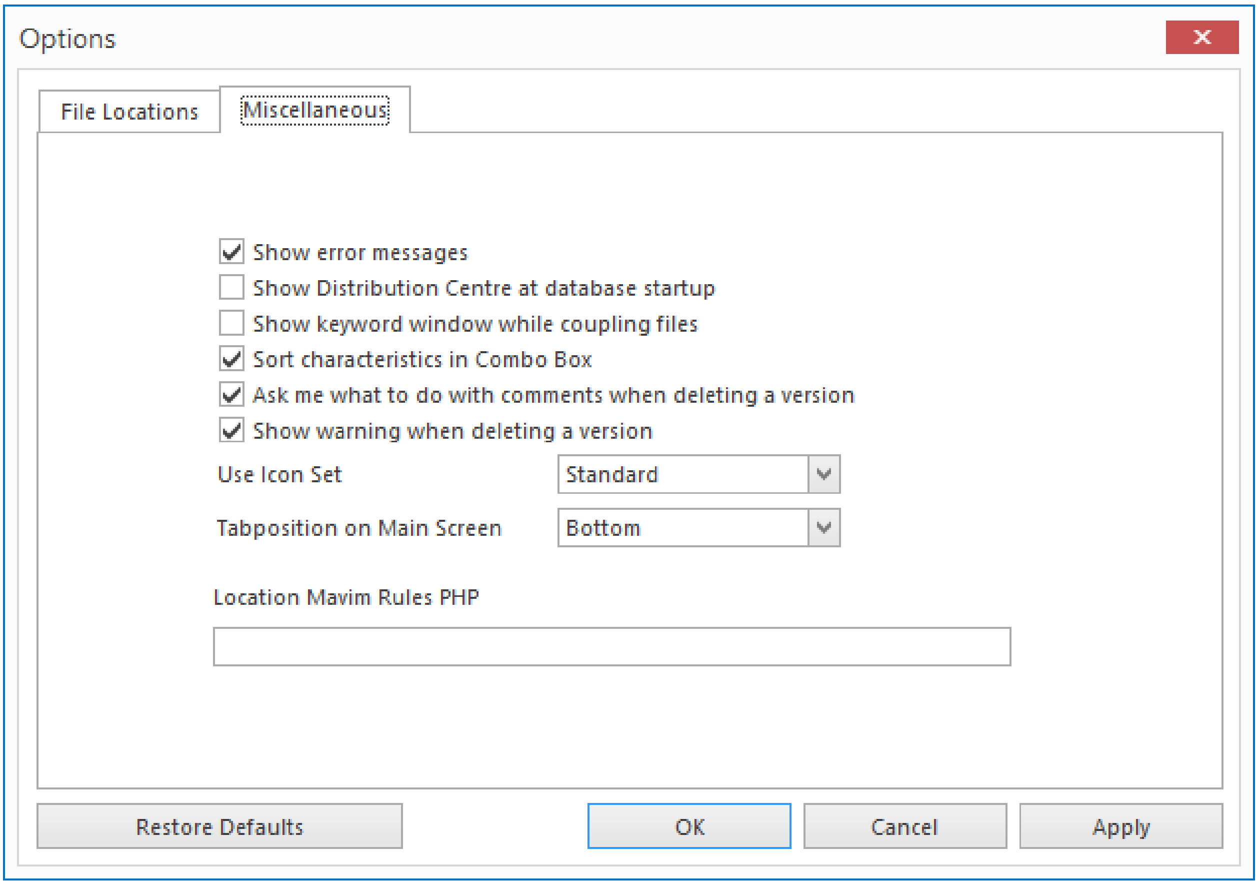
Database Manager Information	C:\Users\User123\AppData\Local\Mavim\Manager10\
Transport Databases	C:\Users\User123\Documents
Published Documents	C:\Users\User123\Documents
Word Themes	C:\Users\Public\Documents\Mavim\Manager10\WordThemes
Published Websites	C:\Users\User123\Documents
Web Themes	C:\Users\Public\Documents\Mavim\Manager10\WebSettings
Web Styles	C:\Users\Public\Documents\Mavim\Manager10\WebSettings
Help Themes	C:\Users\Public\Documents\Mavim\Manager10\HelpThemes
Database Backup Files	C:\Users\User123\Documents\Mavim\Manager10\
Microsoft Word Template for Import	

🟥 Directory does not exist

🟧 Directory isn't write enabled

With the **Miscellaneous** tab the user can choose:

- Whether error messages are shown while working with Mavim
- Whether the **Distribution Centre** should be shown when you open a database (for more information, see Mavim **Help**).
- When coupling files to topics, users can enter key words in the key word screen
- Which icon set (Standard or Classic) should be used for the topics in the tree
- The tab position in the main screen.
- The location of the Web server software.



Options

File Locations | Miscellaneous

☑ Show error messages
□ Show Distribution Centre at database startup
□ Show keyword window while coupling files
☑ Sort characteristics in Combo Box
☑ Ask me what to do with comments when deleting a version
☑ Show warning when deleting a version

Use Icon Set: Standard

Tabposition on Main Screen: Bottom

Location Mavim Rules PHP

The system settings cannot be altered from within Mavim. For this you must use the Windows registry editor (regedit.exe).

9.4 Mavim performance

In some cases, a (trust)certificate is used in your network environment. This can lead to performance issues for Mavim. Certain actions such as:

- Starting Mavim
- Starting Visio or MySite from Mavim

may be slow because the computer in question is attempting to request a certificate but it is unable to complete the transaction. To solve this issue you can set the following registry key:

HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\WinTrust\Trust Providers\Software Publishing

"State"=dword:00023e00

9.5 Limit usergroups shown in Access Management

In the Access Management screen of Mavim, the list of usergroups can be reduced to usergroups defined within a OU (Organizational Unit) of the domain of the user.

This OU is determined by the ‘distinguished name’ of the OU in the Active Directory.
The ‘distinguished name’ has to be entered in the Windows-register als text value of the property ‘OU_distinguishedName’.

For example:

HKEY_LOCAL_MACHINE\SOFTWARE\WOW6432Node\Mavim\Manager\10\General
or
HKEY_LOCAL_MACHINE\SOFTWARE\Mavim\Manager\10\General

"OU_distinguishedName"="OU=NAME,DC=DOMAIN,DC=net"

All groups will be shown when this value does not exist.

9.6 Import from SQL Server

If you would like to import from SQL Server, a registry key has been made available that allows you to configure the connection string:

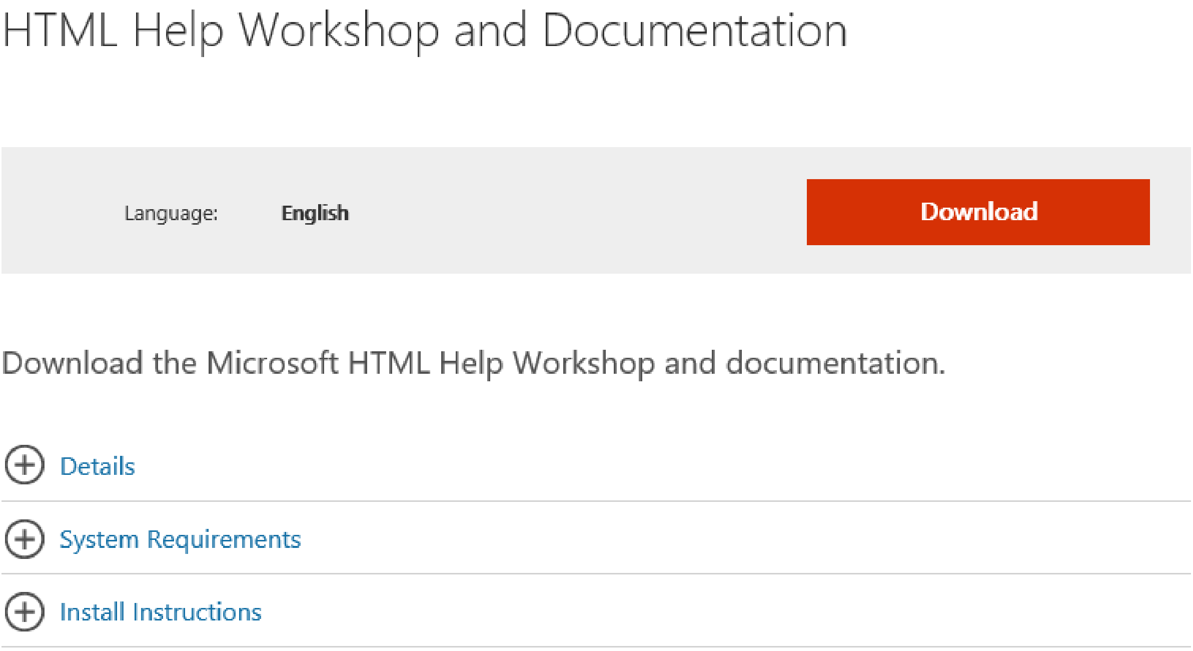
HKEY_LOCAL_MACHINE\SOFTWARE\WOW6432Node\Mavim\Manager\10\Import

DWord: SqlServerImportConnectionString whereby the value is the required connection string.

10 HTML HELP WORKSHOP INSTALLATION

To be able to generate a help file with Mavim Manager, the "HTML Help Workshop" from Microsoft is required. IF this is not installed on your system, it is not possible to publish in this help file format.

This toolkit can be downloaded from the Microsoft website. See HTML Help Workshop and Documentation and follow the instructions.

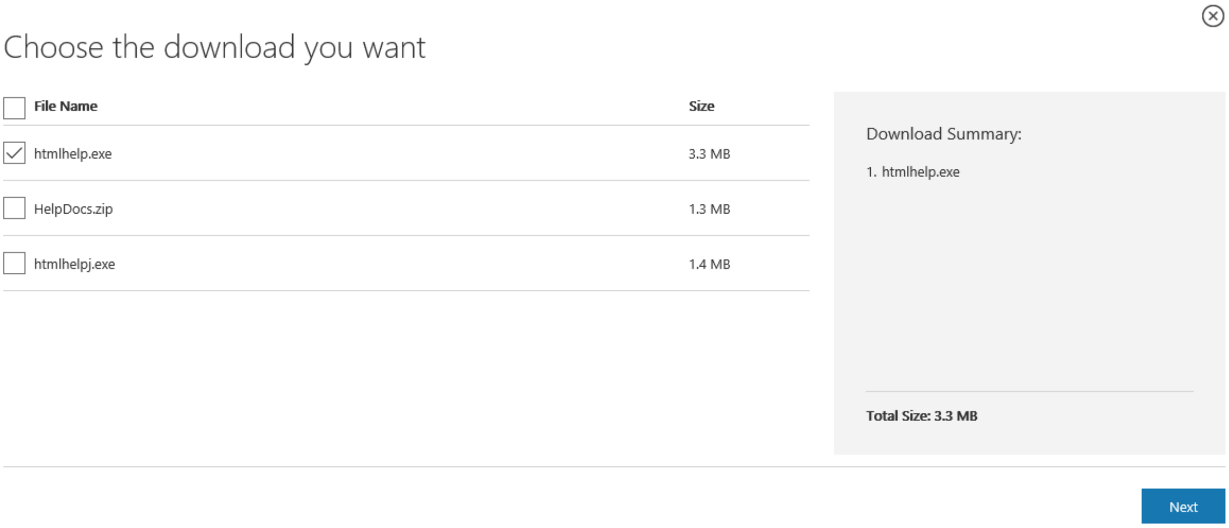


HTML Help Workshop and Documentation

Language:	English

Download the Microsoft HTML Help Workshop and documentation.

In the above window click Download



Choose the download you want

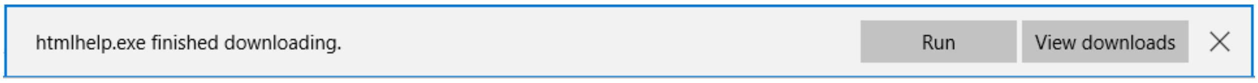
□ File Name	Size
☑ htmlhelp.exe	3.3 MB
□ HelpDocs.zip	1.3 MB
□ htmlhelpj.exe	1.4 MB

Download Summary:

1. htmlhelp.exe

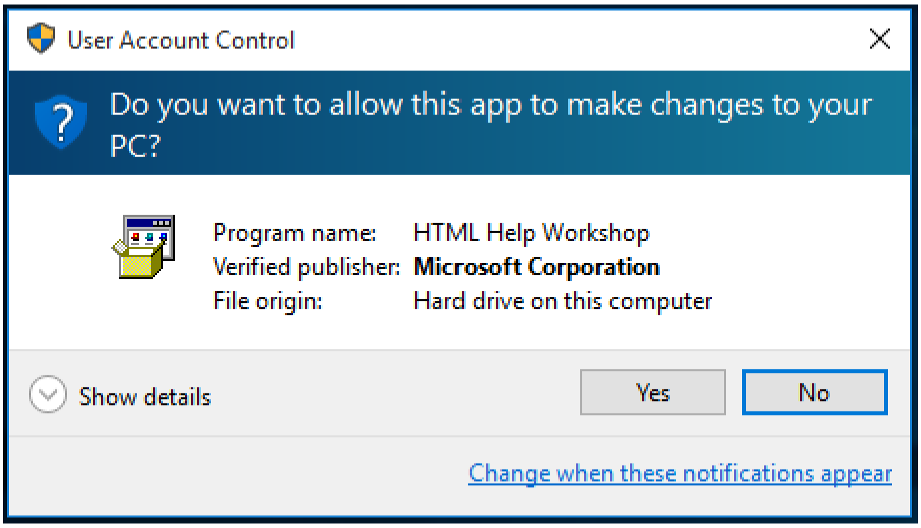
Total Size: 3.3 MB

Select htmlhelp.exe and click Next.



htmlhelp.exe finished downloading.

After the download is completed, click Run

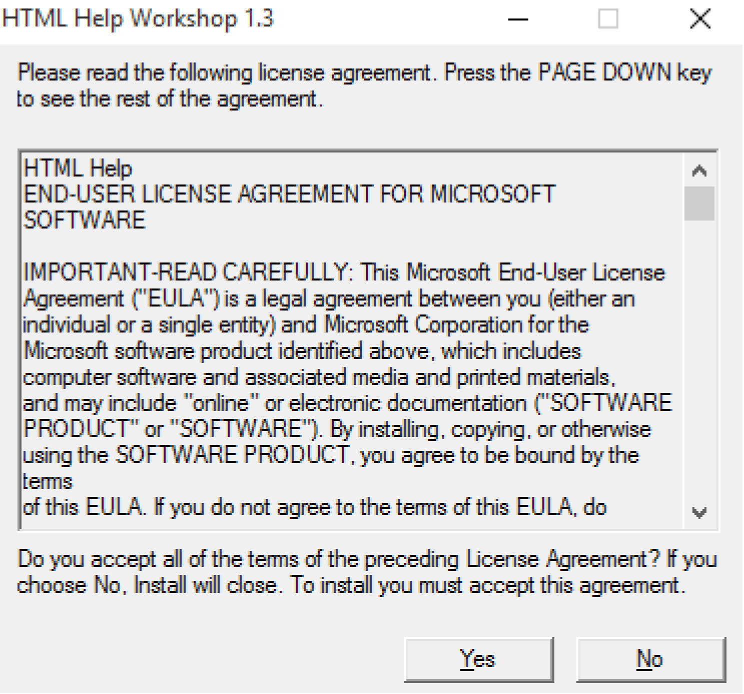


User Account Control

Do you want to allow this app to make changes to your PC?

Program name: HTML Help Workshop
Verified publisher: **Microsoft Corporation**
File origin: Hard drive on this computer

Click **Yes**.



HTML Help Workshop 1.3

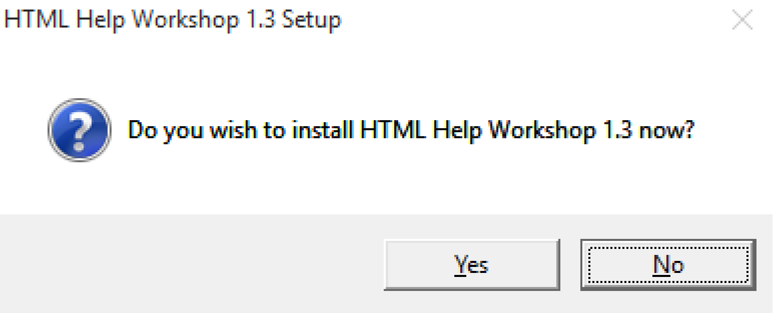
Please read the following license agreement. Press the PAGE DOWN key to see the rest of the agreement.

HTML Help
END-USER LICENSE AGREEMENT FOR MICROSOFT SOFTWARE

IMPORTANT-READ CAREFULLY: This Microsoft End-User License Agreement ("EULA") is a legal agreement between you (either an individual or a single entity) and Microsoft Corporation for the Microsoft software product identified above, which includes computer software and associated media and printed materials, and may include "online" or electronic documentation ("SOFTWARE PRODUCT" or "SOFTWARE"). By installing, copying, or otherwise using the SOFTWARE PRODUCT, you agree to be bound by the terms
of this EULA. If you do not agree to the terms of this EULA, do

Do you accept all of the terms of the preceding License Agreement? If you choose No, Install will close. To install you must accept this agreement.

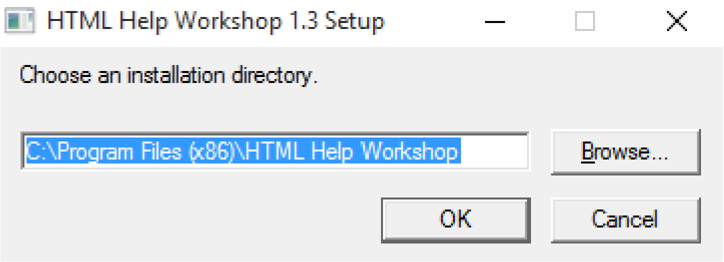
Click **Yes**.



HTML Help Workshop 1.3 Setup

Do you wish to install HTML Help Workshop 1.3 now?

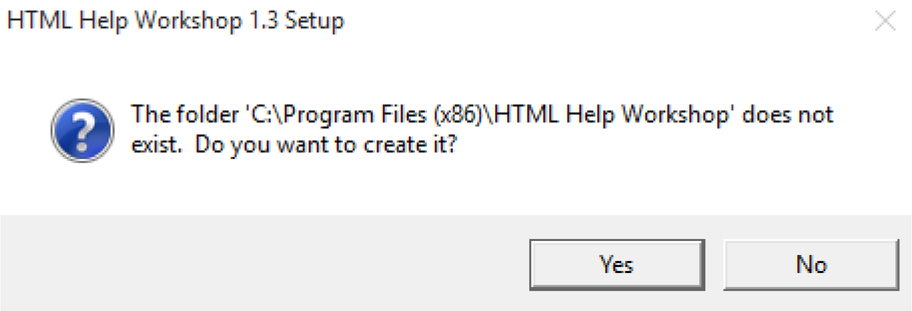
Click **Yes**


HTML Help Workshop 1.3 Setup

Choose an installation directory.

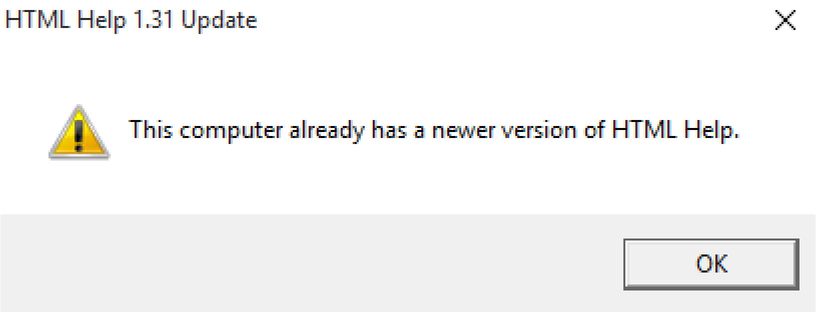
C:\Program Files (x86)\HTML Help Workshop

If you agree with the default directory (C:\Program Files (x86)\HTML Help Workshop), you can click OK


HTML Help Workshop 1.3 Setup

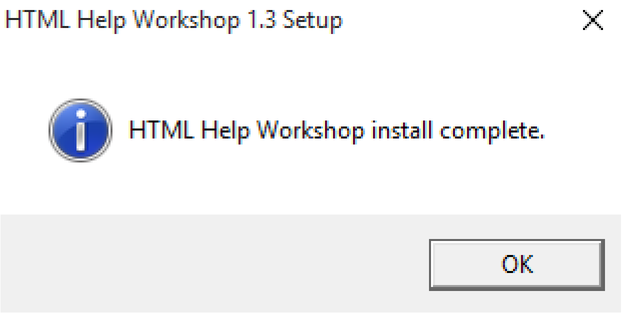
The folder 'C:\Program Files (x86)\HTML Help Workshop' does not exist. Do you want to create it?

Click Yes


HTML Help 1.31 Update

This computer already has a newer version of HTML Help.

Click OK


HTML Help Workshop 1.3 Setup

HTML Help Workshop install complete.

Click OK

The HTML Help Workshop is now installed successfully.

11 USING WINDOWS INSTALLER

To modify the installation or to start up using another method, it is possible to use the Windows Installer (MSIEXEC) 4.5. This program will accept proprietary and Mavim-specific parameters for Mavim Manager.

If you are familiar with the operation of Windows Installer and want to modify the Mavim Manager installation, the following command-line options are available.

Mavim-specific

MAVIMLANGSYSTEM (e.g. MAVIMLANGSYSTEM=F)	Give Language
	F=French
	EN=English (British)
	NL=Dutch
	D=German

REBOOT=ReallySuppress	Absolutely no re-boots
ALLUSERS=0 or 1	0 = installation for current user 1 = installation for all users

Windows Installer-specific

The following are a number of options that can be used with the Microsoft msiexec executable. The full list is much more extensive, see the Windows installer documentation for this.

/i <product.msi>	Install Product
/a <product.msi>	Administrative Installation (for a detailed explanation see the documentation for Microsoft Windows Installer)
/x <product.msi>	Uninstall Product
/l*v c:\logfile.txt	Create Detailed Logfile
/qn	Silent Installation
/qb	Silent Installation with progress dialogs

> When using silent installation, ALLUSERS is 0 by default. If left unchanged, the installation only works for the current user.

Example of a command-line installation

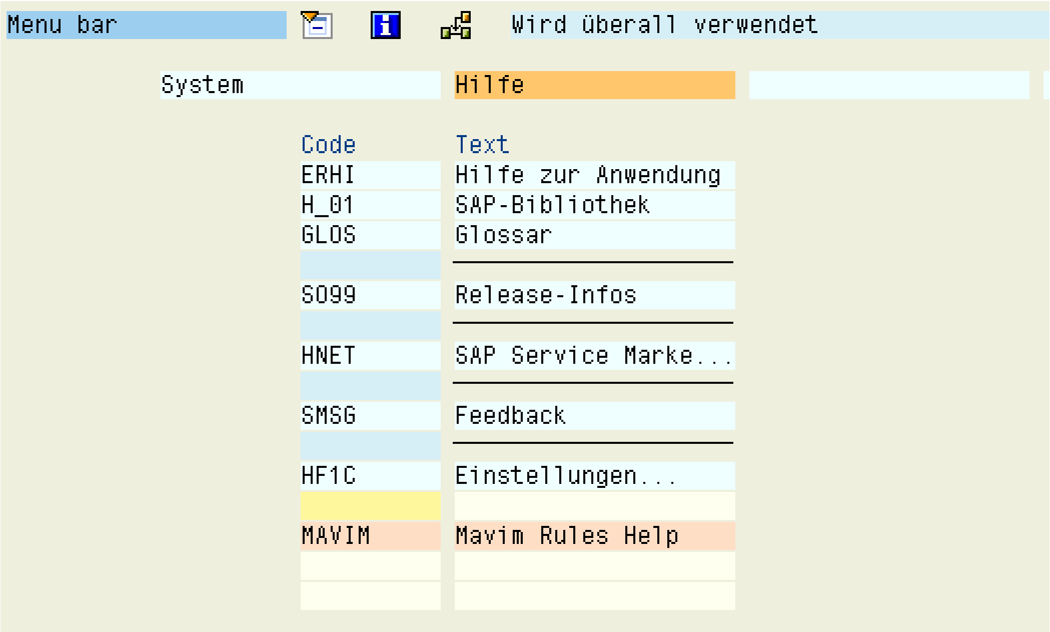
MSIEXEC /i "*path to mavim msi file*" MAVIMLANGSYSTEM=EN ALLUSERS=1 /qb

This will install Mavim Manager in English for all users. It is a silent installation with progress dialogs.

12 CALLING A RULES WEBSITE FROM A SAP TRANSACTION

A topic in a Rules website is called from a SAP transaction as follows:

1. Start transaction: SE41
   Programme: MENUSYST
   GUI Status: MEN
2. Under the **Help** menu item in the **Menu bar** option, an option must be added that references Mavim Manager. The code and description that you use are optional but must be included in the HELP_START function module:

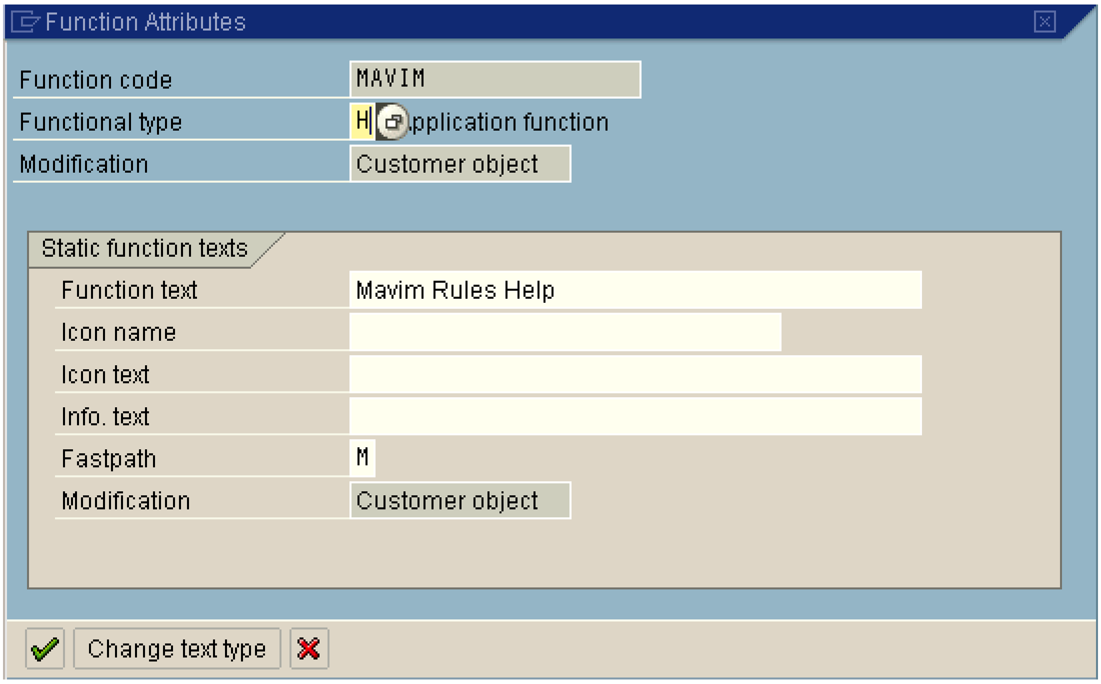


Menu bar Wird überall verwendet

System Hilfe

Code	Text
ERHI	Hilfe zur Anwendung
H_01	SAP-Bibliothek
GLOS	Glossar
S099	Release-Infos
HNET	SAP Service Marke...
SMSG	Feedback
HF1C	Einstellungen...
MAVIM	Mavim Rules Help

3. Double-click the function code:



Function Attributes

Function code	MAVIM
Functional type	H Application function
Modification	Customer object

Static function texts

Function text	Mavim Rules Help
Icon name	
Icon text	
Info. text	
Fastpath	M
Modification	Customer object

4. Enter the value “H” for “Functional Type”
5. Activate the function code by selecting the code and click Function code.
6. Activate the “MENUSYST” user interface.
7. The previous actions will make the Mavim Manager Help option available in the help menu of every SAP screen.
8. The HELP_START function module must then include the start-up of Mavim Manager.
9. This change is a SAP modification that requires the acquisition of an access key via OSS.

Example:
Je nach Aufrufsart (wird vom Dynprozessor übergeben) werden die entsprechenenden Funktionen angesteuert.
 case help_infos-call.
 1. Fall: Hilfe-Menü

```
   when 'H'.
     case help_infos-menufunct.
       when 'GLOS'.               " Glossarliste
……………..
……………..
……………..
```

add code such as

```
*{   INSERT
when 'MAVIM'.
     call function 'CALL_BROWSER'
       exporting
           url          = 'http://172.1.8.110/rulesphp/FastSearch.
php?method=or&keywords=ZWMMG005&upperCaseKeyWords=ZWMMG005&search_type=g
lobal&only_latest_versions=true'.

*}   INSERT
```

This must be made flexible by reading the proper transaction code from sy-tcode and including this in the URL (as opposed to ZWMMG005 in the example above). This must also be changed on the Web server.

13 ADMINISTRATOR TOOLS

The following topics cover the available Administration tools used to control settings and licensing of Mavim Manager 10.

13.1 Admin Tool

The Mavim Administrator tool is used to change the Mavim Manager settings without starting up Mavim Manager. You can choose to set these for all users (Local Machine), or for the user currently logged in (Current User). The following is an explanation of which paths must be stored for the current user and which paths can be set for all users.

Setting	
Configuration Toolbars	Obligatory personal setting
Data Database Manager	Obligatory personal setting
Log Files	Obligatory personal setting
Lib	Can be a group or personal setting
Dictionaries	Obligatory personal setting
Transport Databases	Can be a group or personal setting
Published Documents	Can be a group or personal setting
Published Websites	Can be a group or personal setting
Word Themes	Can be a group or personal setting
Web Themes	Can be a group or personal setting
Help Themes	Can be a group or personal setting
Web Styles	Can be a group or personal setting
Database Backup Files	Can be a group or personal setting
Word Template of importing	Can be a group or personal setting

The Administrator Tool can be found in: C:\Program Files\Mavim\Manager10\AdminTool\Mavim.Manager.AdminTool.exe
(this is the default location)

On a 64Bit operating system the location is: C:\Program Files (x86)\Mavim\Manager10\AdminTool\Mavim.Manager.AdminTool.exe

(this is the default location)

To change the paths:

1. Start the **Administrator tool**.
2. Under Registry Part choose whether you want to configure the setting for Local Machine or Current User.
3. Enter the required paths and click OK.

On the **Miscellaneous** tab you can configure the following:

- Show or hide errors screen
- Show Distribution Centre at start-up
- Adjust the maximum size of the Access database
- Use Icon Set Standard or Classic for icons in the tree
- Standard tab position on the Main Screen
- Location Mavim Rules PHP: location of the Mavim Rules PHP scripts. This must be a relative path (i.e. /<sub-folder>/RulesPhp). By default the path is: /RulesPhp even when this is not entered.

14 BATCH OPERATIONS

14.1 Versions and Website Publications

For the configuring the batch operation for versions and website publications, please refer to the information found in the functional help. Once the configuration has been created, you can automate the creation of versions and website publications by creating a batch file and scheduling a Windows Scheduled Task to execute it.

1. Create a new batch file in a known location.

2. The contents of the batch file is to be filled with the necessary command. As an example:

   "C:\Program Files\Mavim\Manager10\Mavim.Manager.Application.exe" /DB:<database name>.mvd /LOC:D:\Databases /USR:SUPER /ACTION:BATCHPUBLISH

   The configuration of the batch operation is stored in the database and does not need to be supplied in this command.

3. Now create a new Windows Scheduled Task and select to execute the batch file you just created at the desired time and interval.

14.2 Reports

Reports that exist in your database can be scheduled to be generated at a certain time using the standard Windows Scheduled Tasks feature. The following topics will describe how you can set this up.

14.2.1 Creating the batch file

Generating reports using a batch file simply requires the correct command and parameters. Below is a list of the available parameters and a description of what they do:

Parameter	Description
/DB	Used to define the database to be used. This is basically the full file name of the database file (i.e. <database_name.MVD>)
/LOC	Used to define the location of the database file.
/USR	With this parameter you can configure a user account to log in to the database.

/PWS	This parameter is used to configure the appropriate password for the user account used.
/DATALANG	This parameter is used to set the data language for the reports that will be exported.
/DCV	This parameter is used to configure which reports will be executed in the batch. Multiple DCV codes can be added separated by a space. Not setting this parameter will cause all reports in the database to be exported. Entering an incorrect DCV code will cause the export to fail.
/PATH	This parameter defines the path where the exported Excel sheets will be placed.
/ACTION:PUBLISHREPORT	This is basically the command to invoke the creation and export of the configured reports.
/CREATEFOLDERS:TRUE/FALSE	If this parameter is TRUE Rules will create a new folder for the created reports in the folder specified in /PATH. If FALSE all reports will be placed in the folder specified in /PATH. The default is TRUE, even when not specified.

An example commandline could look like this:

```
"C:\Program Files\Mavim\Manager10\Mavim.Manager.Application.exe" /DB:<database name>.mvd
/LOC:C:\Users\Public\Documents\Mavim\Manager10\Data /DBE:2 /USR:SUPER
/ACTION:PUBLISHREPORT /PATH:C:\TEMP /DCV:d12950883c1293v0 /DATALANG:nl
/CREATEFOLDERS:FALSE
```

In this case a DCV code has been provided. When a DCV code is not provided, all reports in the database will be exported.

14.2.2 Setting up a scheduled task

Exporting reports using a batch file means it is very easy to setup a standard Windows Scheduled Task to automate the process. For information on setting up a scheduled task in Windows, please refer to the Windows help.

15 XML

The Mavim XML format is outlined below. You will also find information on importing and exporting data from and to Mavim Manager. It is assumed that you have knowledge of XML, XSD (charts) and XSLT (transformations) and also we assume that you have a basic knowledge of Mavim Manager itself. For more information on XML see http://www.w3schools.com for specifications and tutorials. This site provides detailed information on writing XML, XSD and XSLT files. You can also retrieve XML examples from Rules by exporting and opening XML files.

15.1 Mavim XML format

Using XML files, data can be exchanged between Mavim Manager and other applications. You may only want to export data from Rules to another application, or you may only want to import data from another application into Mavim Manager. Mavim Manager can also be used as an external application. In this case you can import and export XML as an alternative to using Mavim Transport Databases (or MTDs). However, MTDs are more suitable for this than XML. This is because there are a number of limitations in XML compared to MTDs. (see Limitations).

If you want to exchange data with Mavim Manager, it is important to know the documentation possibilities of the Mavim XML format. Other applications can also import and export XML. However, they do this in a different XML format with a different XML schema. To allow data to be exchanged with Mavim Manager, conversions will be necessary. Using an XSLT file, the formats can be changed (converted) to accommodate the application that will be importing the XML file. (see Conversion).

A good source of information is http://www.mavim.com/ns/mxl/mxl9.xsd which explains all of the properties and limitations of the Mavim XML format. This schema is also used by Rules to check the validity of XML files to be imported. If you ensure that your XML data conforms to the Mavim Manager XML schema then you will probably be able to import the XML file into Mavim Manager. The opposite is also true. Mavim Manager will export valid XML files.

15.1.1 Encoding

```
<?xml version="1.0" encoding="UTF-8"?>
```

The Mavim Manager XML format opens with the line stating that this is an XML document and that UTF-8 coding has been used.

```
<mxl:exchange xsi:schemaLocation="http://www.mavim.com/ns/mxl
http://www.mavim.com/ns/mxl/mxl9.xsd" mxl:version="9"
xmlns:mxl="http://www.mavim.com/ns/mxl" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance">
```

The second line contains information about the layout, namespace and format of the document. It also contains the reference to the Mavim Manager XML schema (xsd).
The mxl:exchange is the first element of the XML file.

The 'xsi:schemaLocation' attribute specifies the Mavim Manager namespace 'mxl' and references the Mavim Manager XSD file. This contains a number of conditions which the XML file must fulfil that are described later in this document.

The exchange element knows only one attribute that is specified by Mavim, the version number of the Mavim XML file format. This is 'mxl:version' and is supported by Mavim Rules 6.2 version "2.0".

Apart from the XSD file there are further conditions which the document must fulfil. During an import into Mavim Manager a check is performed against the XSD file and to determine whether the following conditions are met:

- Whether the XML files conforms to the description of the schema (XSD file)
- Whether the topics and relationships have a unique ID and that the references to these IDs are also correct/present.

The XML file is closed with the following line:

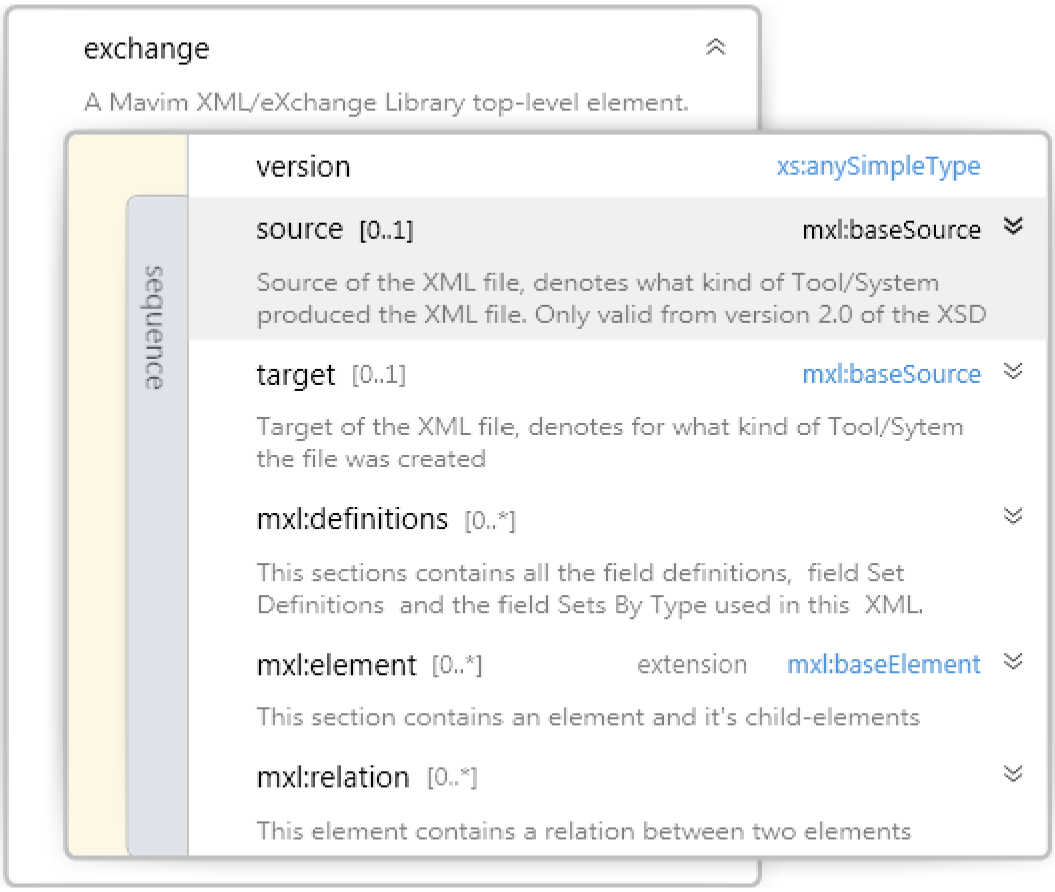
```
</mxl:exchange>
```

15.1.2 <mxl:exchange> - root element

Within the *mxl:exchange* element you will find the actual format that is described in the XML schema. The format consists of five base sections:

- source (see <mxl:source> - source application information)
- target (see <mxl:target> - Target application information)
- definitions (see <mxl:definitions> - Definitions)
- element (see <mxl:element> - Mavim Manager topic)
- relation (see <mxl:relation> - Mavim relationship)

The order of the elements is documented in the XML schema and must be adhered to in the Mavim Manager XML file. None of the mxl:exchange elements are required.


exchange
A Mavim XML/eXchange Library top-level element.
sequence
version
xs:anySimpleType
source [0..1]
mxl:baseSource
Source of the XML file, denotes what kind of Tool/System produced the XML file. Only valid from version 2.0 of the XSD
target [0..1]
mxl:baseSource
Target of the XML file, denotes for what kind of Tool/Sytem the file was created
mxl:definitions [0..*]
This sections contains all the field definitions, field Set Definitions and the field Sets By Type used in this XML.
mxl:element [0..*]
extension
mxl:baseElement
This section contains an element and it's child-elements
mxl:relation [0..*]
This element contains a relation between two elements

15.1.2.1 *<mxl:source> - source application information*

Within this element, information is stored about the source of the XML file. This information can be used to identify the set of data for updating topics.

Mavim Manager creates a transport topic in which these data are saved. When a new version of this XML file is imported, Mavim will recognise this file and then offer the option of updating the previously imported topics.

The *<mxl:source>* element has the <mxl:baseSource> type, and has the following properties:

- <mxl:id> element
  This element consists of a string of 30 characters. The source of this XML file provides the identification and saves it so that it can be re-used for updated versions of this XML file.
- <mxl:type> element
  This element consists of a string. It can contain the name of the source application. This value has no other meaning.
- <mxl:name> element
  This value is the name of the Transport topic. Element <mxl:name> has a language attribute 'mxl:lang' which is limited by the Mavim Manager data languages. See Mavim data languages.
- <mxl:description> element
  As the Transport topic can have a description in Mavim Manager, a description element <mxl:description> is present. For the description see <mxl:description> - Description

15.1.2.2 *<mxl:target> - Target application information*

If you export previously imported topics in Mavim and use the transport topic that was created with the previous import, then the data of the source are included in the XML file using the target element <mxl:target>.
Using these data, the target application can recognise that it relates to topics that originate from the target application itself and have been changed. With this information, the target application can update these topics.

The element <mxl:target> has the <mxl:baseSource> type, and has the following properties:

- <mxl:id> element
  This element consists of a string of 30 characters. The target application of this XML file uses the ID for recognition so that the topics can be updated.
- <mxl:type> element
  This element consists of a string. This can contain the name of the previous source application. This value has no other meaning.
- <mxl:name> element
  This value is the name of the Mavim Manager Transport topic. The element <mxl:name> has the language attribute 'mxl:lang' that is limited by the Mavim Manager data languages. See Mavim data languages.
- <mxl:description> element
  Because a Mavim Manager Transport topic can have a description, an <mxl:description> element is present. For a description of this, see <mxl:description> - Description

15.1.2.3 <mxl:definitions> - Definitions

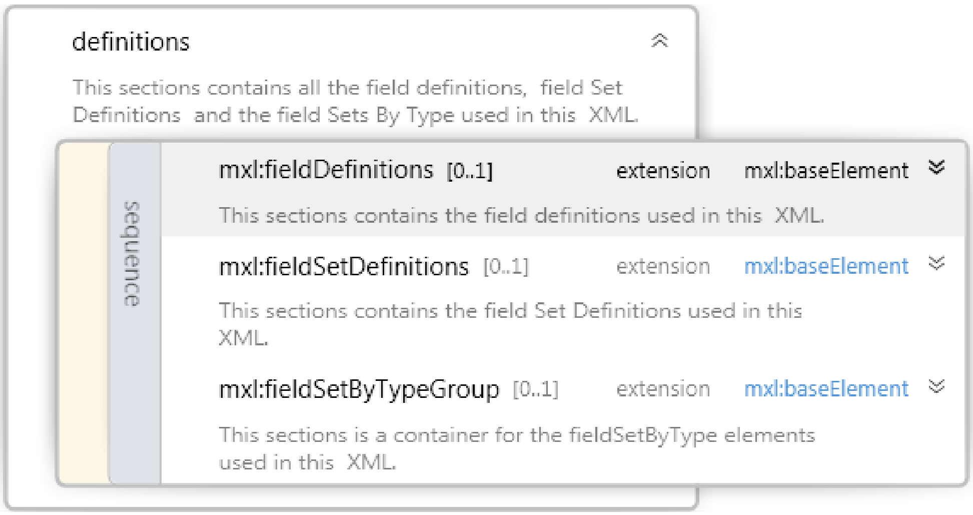
Mavim Manager XML accepts three definition types:

- Field definitions
- Field set definitions
- Topic type definitions

These are all meant to support the use of fields for topics. Also, the order of the elements is documented in the XML schema and must be maintained in the Mavim Manager XML file.

The three definitions are described in:

- <mxl:fieldDefinitions> - Field definitions
- <mxl:fieldSetDefinitions> - Field set definitions
- <mxl:fieldSetByTypeGroup> - Type definitions



definitions

This sections contains all the field definitions, field Set Definitions and the field Sets By Type used in this XML.

mxl:fieldDefinitions [0..1] extension mxl:baseElement

This sections contains the field definitions used in this XML.

mxl:fieldSetDefinitions [0..1] extension mxl:baseElement

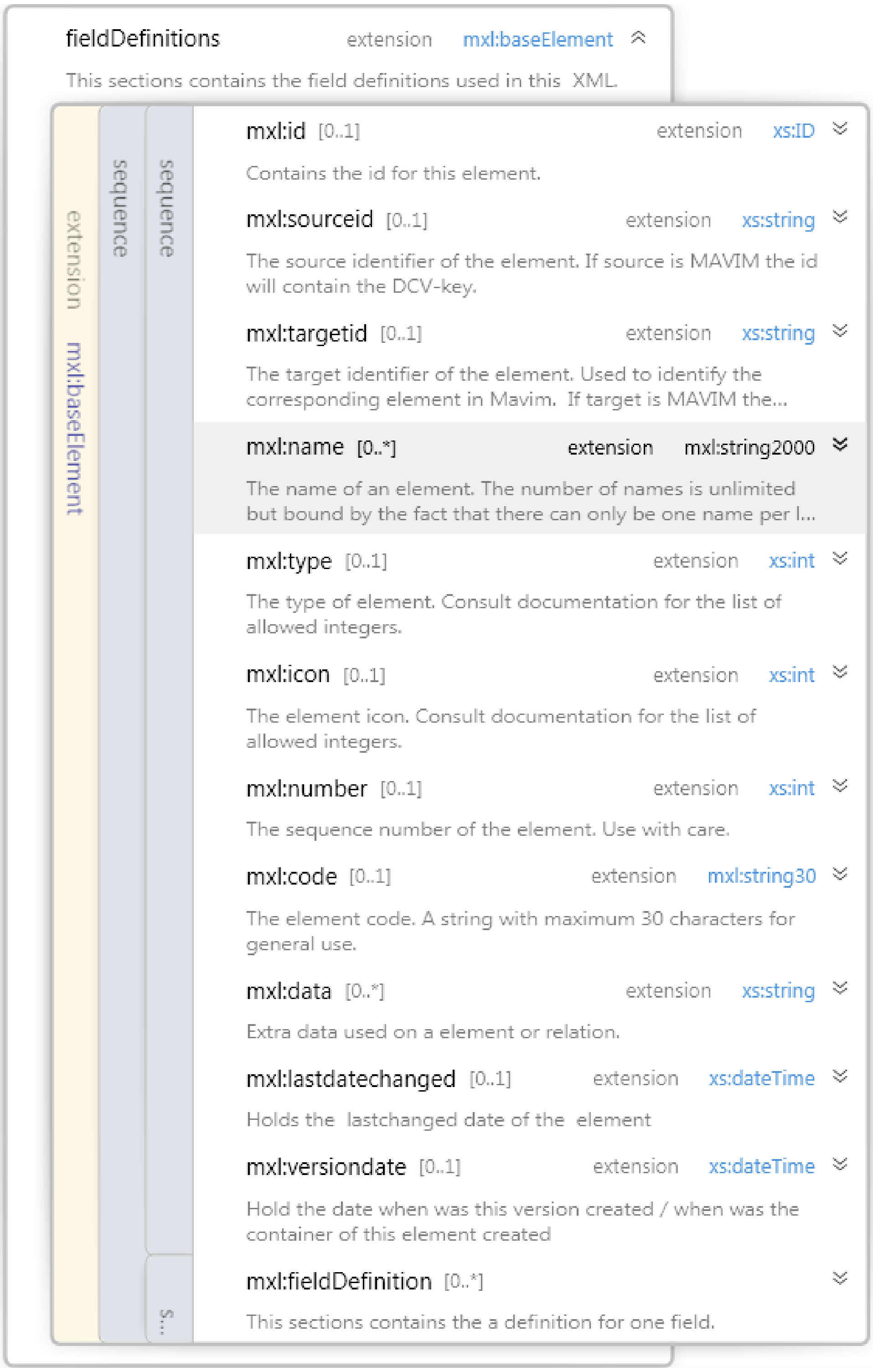
This sections contains the field Set Definitions used in this XML.

mxl:fieldSetByTypeGroup [0..1] extension mxl:baseElement

This sections is a container for the fieldSetByType elements used in this XML.

15.1.2.3.1 <mxl:fieldDefinitions> - Field definitions

This element contains the collection of elements of type <mxl:fieldDefinition> which is described in <mxl:fieldDefinition> - Field definition. There can be only one element of type <mxl:fieldDefinitions> within the XML file.


fieldDefinitions extension mxl:baseElement

This sections contains the field definitions used in this XML.

mxl:id [0..1] extension xs:ID

Contains the id for this element.

mxl:sourceid [0..1] extension xs:string

The source identifier of the element. If source is MAVIM the id will contain the DCV-key.

mxl:targetid [0..1] extension xs:string

The target identifier of the element. Used to identify the corresponding element in Mavim. If target is MAVIM the...

mxl:name [0..*] extension mxl:string2000

The name of an element. The number of names is unlimited but bound by the fact that there can only be one name per l...

mxl:type [0..1] extension xs:int

The type of element. Consult documentation for the list of allowed integers.

mxl:icon [0..1] extension xs:int

The element icon. Consult documentation for the list of allowed integers.

mxl:number [0..1] extension xs:int

The sequence number of the element. Use with care.

mxl:code [0..1] extension mxl:string30

The element code. A string with maximum 30 characters for general use.

mxl:data [0..*] extension xs:string

Extra data used on a element or relation.

mxl:lastdatechanged [0..1] extension xs:dateTime

Holds the lastchanged date of the element

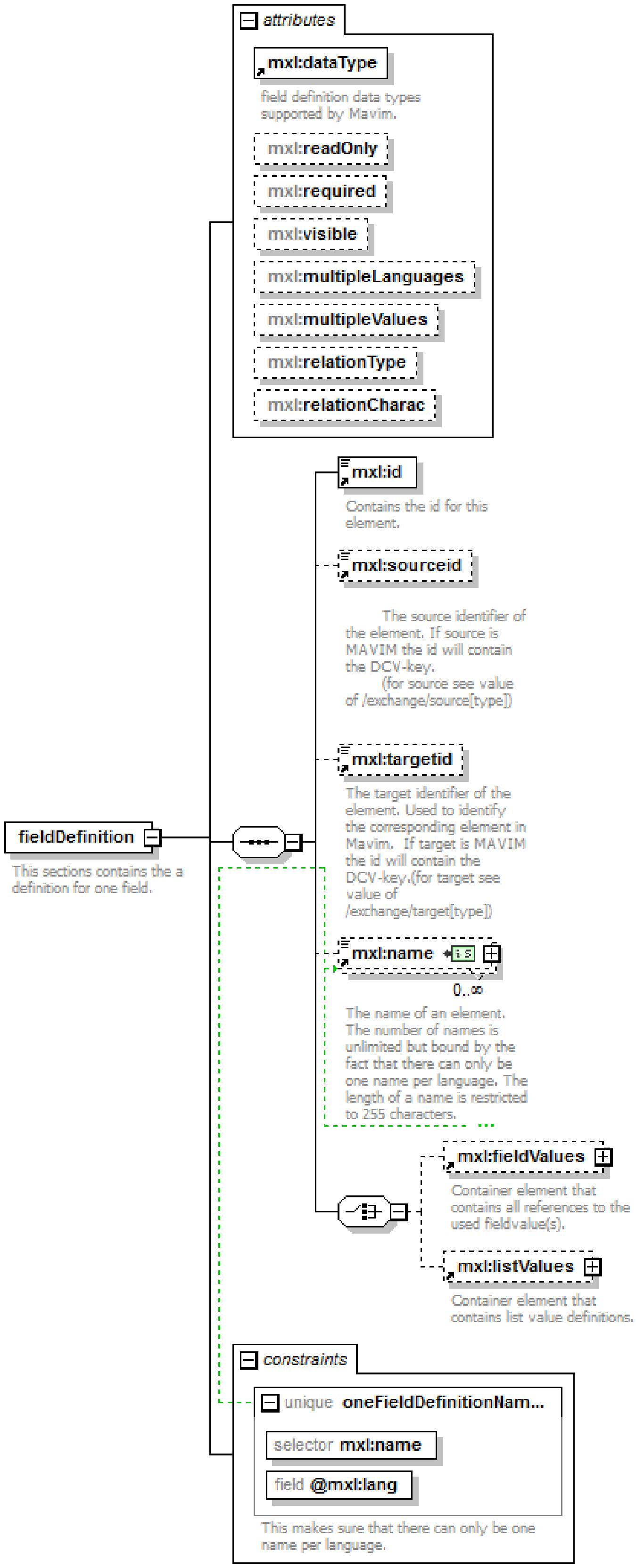
mxl:versiondate [0..1] extension xs:dateTime

Hold the date when was this version created / when was the container of this element created

mxl:fieldDefinition [0..*]

This sections contains the a definition for one field.

15.1.2.3.1.1 <mxl:fieldDefinition> - Field definition


attributes
mxl:dataType
field definition data types supported by Mavim.
mxl:readOnly
mxl:required
mxl:visible
mxl:multipleLanguages
mxl:multipleValues
mxl:relationType
mxl:relationCharac
fieldDefinition
This sections contains the a definition for one field.
mxl:id
Contains the id for this element.
mxl:sourceid
The source identifier of the element. If source is MAVIM the id will contain the DCV-key. (for source see value of /exchange/source[type])
mxl:targetid
The target identifier of the element. Used to identify the corresponding element in Mavim. If target is MAVIM the id will contain the DCV-key.(for target see value of /exchange/target[type])
mxl:name
0..∞
The name of an element. The number of names is unlimited but bound by the fact that there can only be one name per language. The length of a name is restricted to 255 characters.
mxl:fieldValues
Container element that contains all references to the used fieldvalue(s).
mxl:listValues
Container element that contains list value definitions.
constraints
unique oneFieldDefinitionNam...
selector mxl:name
field @mxl:lang
This makes sure that there can only be one name per language.

Within the element <mxl:fieldDefinition>, the properties of a field are described. The attributes of this element make it clear which properties these are:

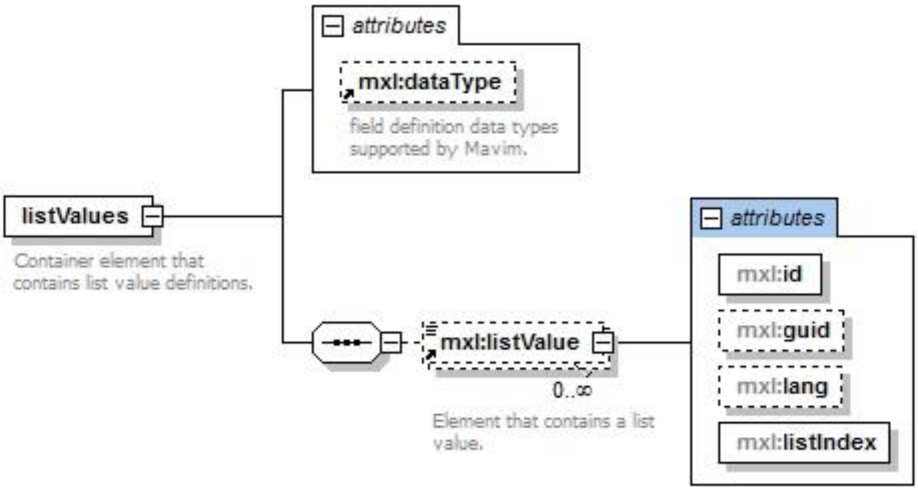
- < mxl:dataType> element
  This is a required attribute and indicates the type of data saved (text, integer, boolean). For possible values see Field types
- <mxl:readOnly> element
  This indicates that the field cannot be changed within Rules. Possible values are ‘true’ and ‘false’.
- <mxl:required> element
  This element indicates whether the field is required or not. Possible values are ‘true’ and ‘false’.
- <mxl:visible> element
  This element indicates whether the field is visible or not. Possible values are ‘true’ and ‘false’.
- <mxl:multipleLanguages> element
  This element determines whether the field value can vary per language. Possible values are ‘true’ and ‘false’.
- <mxl:multipleValues> element
  This element indicates whether the field can have multiple values. Possible values are ‘true’ and ‘false’.
- <mxl:relationType> element
  This attribute is only important if the field type 'relation' is selected. If a relationship field is created, this is the type. For possible values, see: Mavim Manager relationship types
- <mxl:relationCharac> element
  This attribute is only important if the field type 'relation' is selected. The value (ID) references the topic that must be installed in the XML file. This topic is then used as a characteristic when creating the relationship field.

Not all of the aforementioned properties are relevant for each field type <mxl:Type>. Certain properties can conflict with each other. If non-relevant (and therefore meaningless) combinations of properties and field types are imported, these will be ignored. For a complete overview of which properties are relevant for which types, see Field types and relevant properties.

A field definition contains the following elements:

- <mxl:id> element
  The internal ID of a field definition. The format of an ID is limited to that of an XML tag of type "ID". This means that an ID must always start with a letter. Also, the ID must be unique within the file. This ID is also used e.g. with hyperlinks or relationships to reference this topic.
- <mxl:sourceid> element
  Free string for the key/ID of the field definition as identified in the source.
- <mxl:targetid> element
  Free string for the key/ID of the field definition as identified in the target. If data has previously been imported from a certain application, the keys/IDs can be included. The target application can use this to update “its” topics.
- <mxl:name> element
  The topic name is a required element. The element contains the name of the field definition. This element has an attribute ‘mxl:lang’. There can be only one name element per language. For possible language values see: Mavim data languages.
- <mxl:fieldValues> element
  This element belongs to the description of the field definition of any type except the list type. A standard value can be specified here that is used for the field if nothing has been entered. Field values are a collection of elements of type <mxl:fieldValue>. For more information on field values see <mxl:fieldValues> - Field values.
- <mxl:listValues> element
  This element belongs to the description of list field definition. It defines the textual values in the list. The list values are a collection of <mxl:listValue> elements. For more information on the list values see <mxl:listValues> - List values.

15.1.2.3.1.1.1 <mxl:listValues> - List values

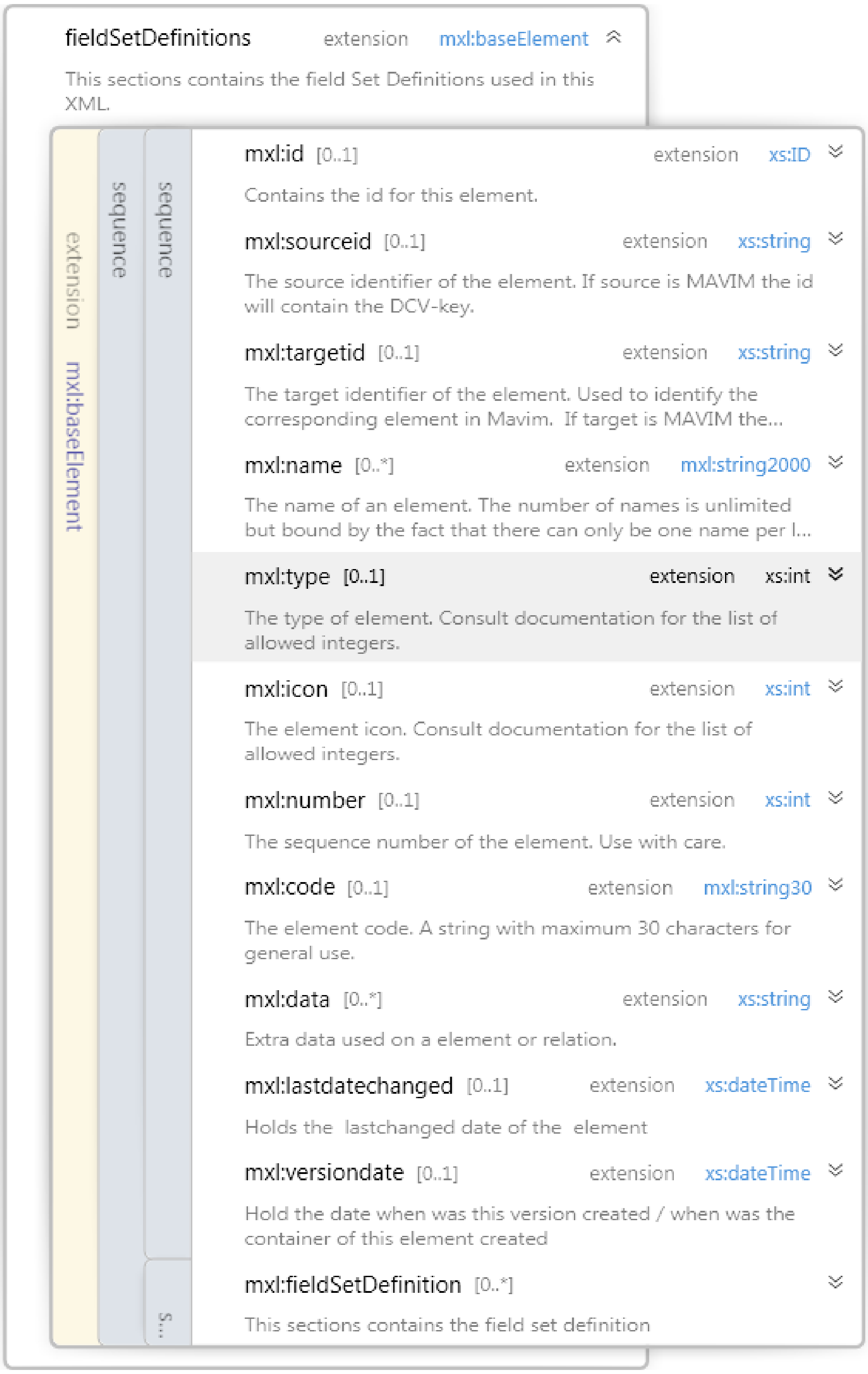

attributes
mxl:dataType
field definition data types supported by Mavim.
listValues
Container element that contains list value definitions.
mxl:listValue
0..∞
Element that contains a list value.
attributes
mxl:id
mxl:guid
mxl:lang
mxl:listIndex

Properties of list values are:

- <mxl:dataType> element
  The attribute value defines the data type of the list. The value must be 'text'.
- <mxl:listValue>
  The content of this element defines the textual content of a list value. The attributes:
  - <mxl:id> attribute. This attribute is mandatory. It defines the id of the list value. The actual value of a field of type list refers to this id.
  - <mxl:guid> attribute. This attribute is optional. The attribute value is a globally unique identifier.
  - <mxl:lang> attribute. This attribute is optional. It defines the language of the list value.
  - <mxl:listIndex> attribute. This attribute is mandatory. It defines the index of the list value in the list.

15.1.2.3.2 <mxl:fieldSetDefinitions> - Field set definitions

The element <mxl:fieldSetDefinitions> is the collection of elements of type <mxl:fieldSetDefinition>. There can be only one element of type <mxl:fieldSetDefinitions>.



fieldSetDefinitions extension mxl:baseElement

This sections contains the field Set Definitions used in this XML.

mxl:id [0..1] extension xs:ID

Contains the id for this element.

mxl:sourceid [0..1] extension xs:string

The source identifier of the element. If source is MAVIM the id will contain the DCV-key.

mxl:targetid [0..1] extension xs:string

The target identifier of the element. Used to identify the corresponding element in Mavim. If target is MAVIM the...

mxl:name [0..*] extension mxl:string2000

The name of an element. The number of names is unlimited but bound by the fact that there can only be one name per l...

mxl:type [0..1] extension xs:int

The type of element. Consult documentation for the list of allowed integers.

mxl:icon [0..1] extension xs:int

The element icon. Consult documentation for the list of allowed integers.

mxl:number [0..1] extension xs:int

The sequence number of the element. Use with care.

mxl:code [0..1] extension mxl:string30

The element code. A string with maximum 30 characters for general use.

mxl:data [0..*] extension xs:string

Extra data used on a element or relation.

mxl:lastdatechanged [0..1] extension xs:dateTime

Holds the lastchanged date of the element

mxl:versiondate [0..1] extension xs:dateTime

Hold the date when was this version created / when was the container of this element created

mxl:fieldSetDefinition [0..*]

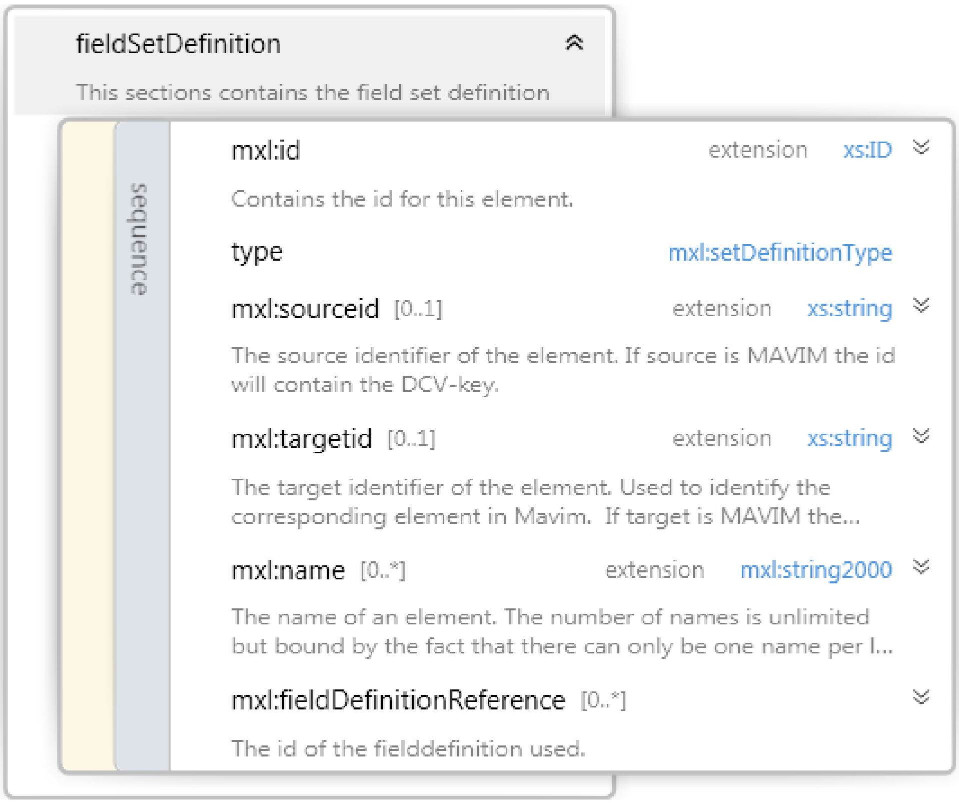
This sections contains the field set definition

15.1.2.3.2.1 <mxl:fieldSetDefinition> - Field set definition

The element <mxl:fieldSetDefinition>contains the following elements:

- <mxl:id> element
  The internal ID of a field set definition. The format of an ID is limited to that of an XML tag of type "ID". This means that an ID must always start with a letter. Also, the ID must be unique within the XML file. This ID is also used in hyperlinks and relationships to reference this topic.

- <mxl:Type> element
  The type of the field set definition. For possible type see Field types
- <mxl:sourceid> element
  Free string for the key/ID of the field set definition as identified in the source.
- <mxl:targetid> element
  Free string for the key/ID of the field set definition as identified in the target. If data were previously imported from a certain application, the keys/IDs can be included. This can help the target application to update “its” topics.
- <mxl:name> element
  This required element contains the name of the field set definition. This element has an attribute language 'mxl:lang'. There can be only on element per language. For possible values for language see: Mavim data languages.
- <mxl:fieldDefinitionReference> element
  Reference to a differently defined field <mxl:fieldDefinition>. This element has a required attribute <mxl:id> in which the actual reference to a field definition can be found. The collection of elements <mxl:fieldDefinitionReference> determines the composition of the collection of elements <mxl:fieldDefinitionReference> determines the composition of the field set.



fieldSetDefinition

This sections contains the field set definition

sequence

mxl:id extension xs:ID

Contains the id for this element.

type mxl:setDefinitionType

mxl:sourceid [0..1] extension xs:string

The source identifier of the element. If source is MAVIM the id will contain the DCV-key.

mxl:targetid [0..1] extension xs:string

The target identifier of the element. Used to identify the corresponding element in Mavim. If target is MAVIM the...

mxl:name [0..*] extension mxl:string2000

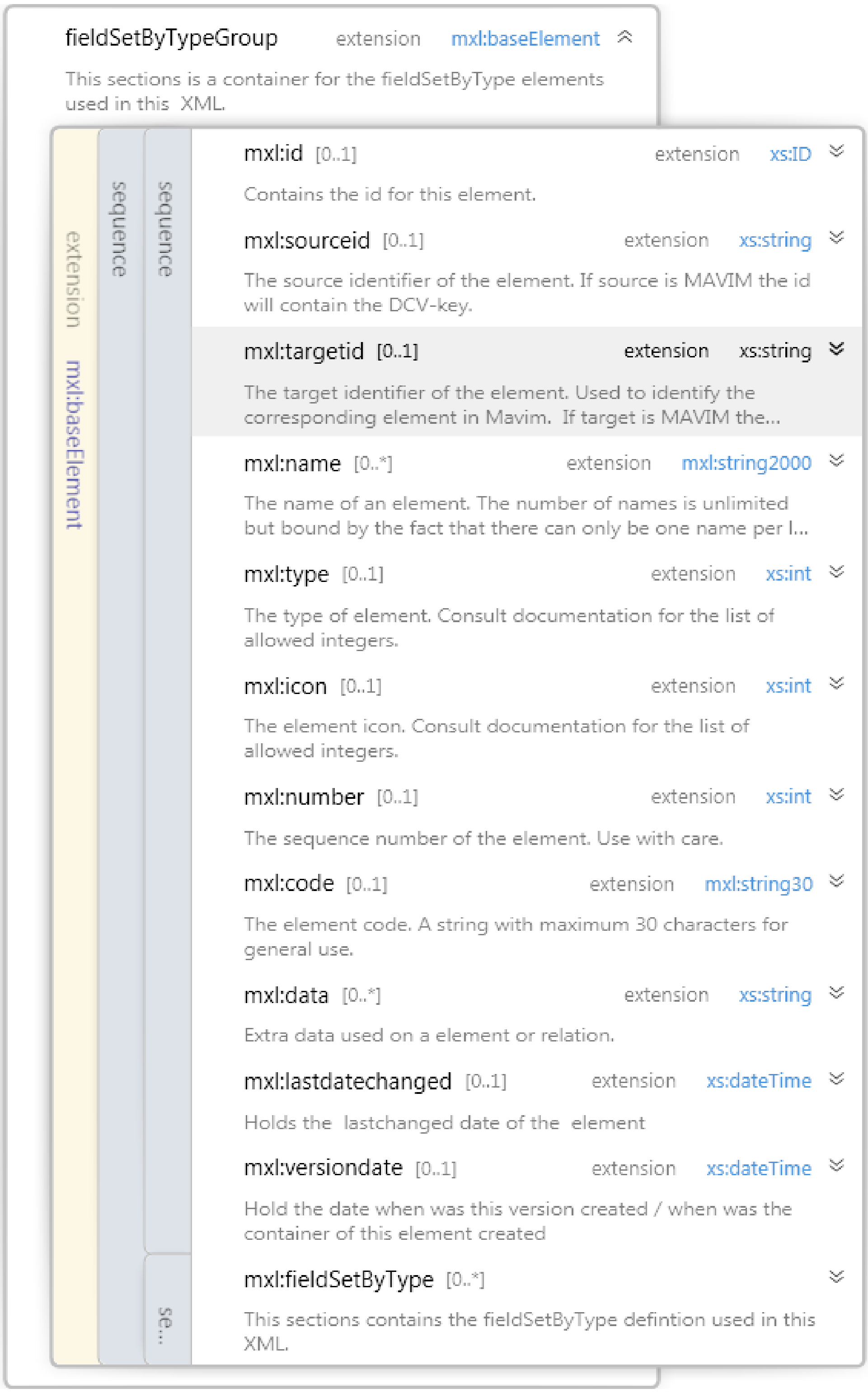
The name of an element. The number of names is unlimited but bound by the fact that there can only be one name per l...

mxl:fieldDefinitionReference [0..*]

The id of the fielddefinition used.

15.1.2.3.3 <mxl:fieldSetByTypeGroup> - Type definitions

The element <mxl:fieldSetByTypeGroup> is the collection of elements of type <mxl:fieldSetByType>. There can be only one element of type <xml:fieldSetByTypeGroup>.



fieldSetByTypeGroup extension mxl:baseElement

This sections is a container for the fieldSetByType elements used in this XML.

mxl:id [0..1] extension xs:ID

Contains the id for this element.

mxl:sourceid [0..1] extension xs:string

The source identifier of the element. If source is MAVIM the id will contain the DCV-key.

mxl:targetid [0..1] extension xs:string

The target identifier of the element. Used to identify the corresponding element in Mavim. If target is MAVIM the...

mxl:name [0..*] extension mxl:string2000

The name of an element. The number of names is unlimited but bound by the fact that there can only be one name per l...

mxl:type [0..1] extension xs:int

The type of element. Consult documentation for the list of allowed integers.

mxl:icon [0..1] extension xs:int

The element icon. Consult documentation for the list of allowed integers.

mxl:number [0..1] extension xs:int

The sequence number of the element. Use with care.

mxl:code [0..1] extension mxl:string30

The element code. A string with maximum 30 characters for general use.

mxl:data [0..*] extension xs:string

Extra data used on a element or relation.

mxl:lastdatechanged [0..1] extension xs:dateTime

Holds the lastchanged date of the element

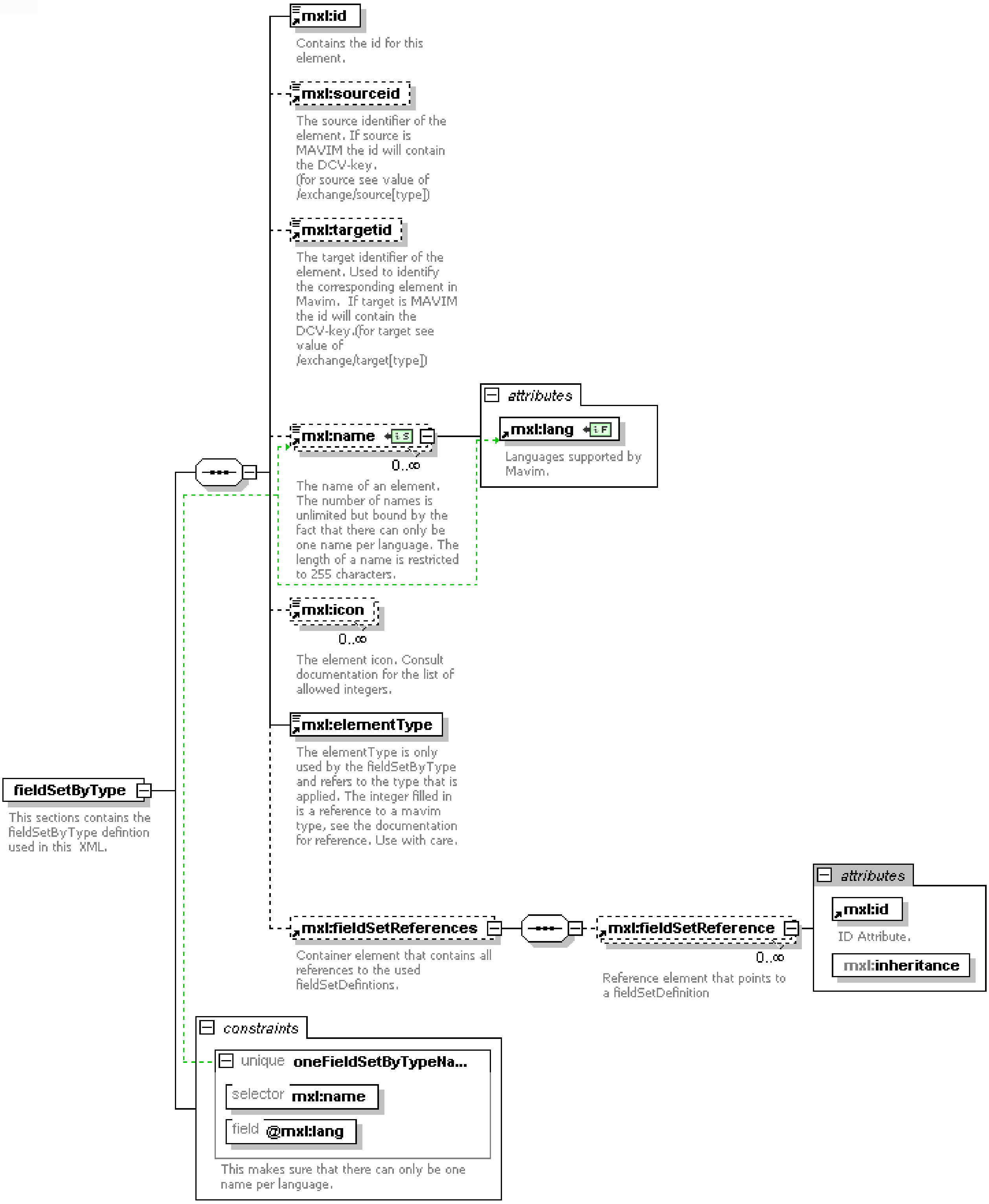
mxl:versiondate [0..1] extension xs:dateTime

Hold the date when was this version created / when was the container of this element created

mxl:fieldSetByType [0..*]

This sections contains the fieldSetByType defintion used in this XML.

15.1.2.3.3.1 *<mxl:fieldSetByType> - Topic type definition*


mxl:id
Contains the id for this element.
mxl:sourceid
The source identifier of the element. If source is MAVIM the id will contain the DCV-key. (for source see value of /exchange/source[type])
mxl:targetid
The target identifier of the element. Used to identify the corresponding element in Mavim. If target is MAVIM the id will contain the DCV-key.(for target see value of /exchange/target[type])
attributes
mxl:lang
Languages supported by Mavim.
mxl:name
0..∞
The name of an element. The number of names is unlimited but bound by the fact that there can only be one name per language. The length of a name is restricted to 255 characters.
mxl:icon
0..∞
The element icon. Consult documentation for the list of allowed integers.
mxl:elementType
The elementType is only used by the fieldSetByType and refers to the type that is applied. The integer filled in is a reference to a mavim type, see the documentation for reference. Use with care.
fieldSetByType
This sections contains the fieldSetByType defintion used in this XML.
attributes
mxl:id
ID Attribute.
mxl:inheritance
mxl:fieldSetReferences
Container element that contains all references to the used fieldSetDefintions.
mxl:fieldSetReference
0..∞
Reference element that points to a fieldSetDefinition
constraints
unique oneFieldSetByTypeNa...
selector mxl:name
field @mxl:lang
This makes sure that there can only be one name per language.

A topic type definition contains the following elements:

- <mxl:id> element
  The internal ID of a topic type definition. The format of an ID is limited to that of an XML tag of type "ID". This means that the ID must always start with a letter. Also, the ID must be unique within the file. This ID is also used in a hyperlink or relationship to reference this topic.
- <mxl:sourceid> element
  Free string for the key/ID of the topic type definition as identified in the source.
- <mxl:targetid> element

Free string for the key/ID of the topic type definition as identified in the target. If previously imported data exists from a certain application, keys/IDs can be included. This can help the target application with updating "its" topics.

- <mxl:name> element
  This element contains the name of the topic type definition. This element has a language attribute 'mxl:lang'. There can be only one name element per language. For possible values for language see: Mavim data languages.
- <mxl:elementType> element
  Topic type for which the collection of elements of type <mxl:fieldSetReference> is applicable. All topics of the corresponding type receive this collection. Possible values are not checked by the schema but by the import in Mavim Manager.

  The possible values are:

9	Primary topic
10	Topic
30	Standard topic
31	Start activity
32	End activity
33	Decision
34	Staff department
35	Detail process
36	Process
37	Module
38	Department
39	Activity
40	User Defined topic type
41	User Defined topic type
42	User Defined topic type
43	User Defined topic type
66	Group topic
510	Start event
511	Event
512	Stop event
513	Function
514	AND
515	OR
516	XOR
600	Active Directory group
601	Custom group
602	Active Directory user

- <mxl:fieldSetReferences> element
  This element contains the collection of elements of type <mxl:fieldSetReference>. See: <mxl:fieldSetReferences> - Field set references

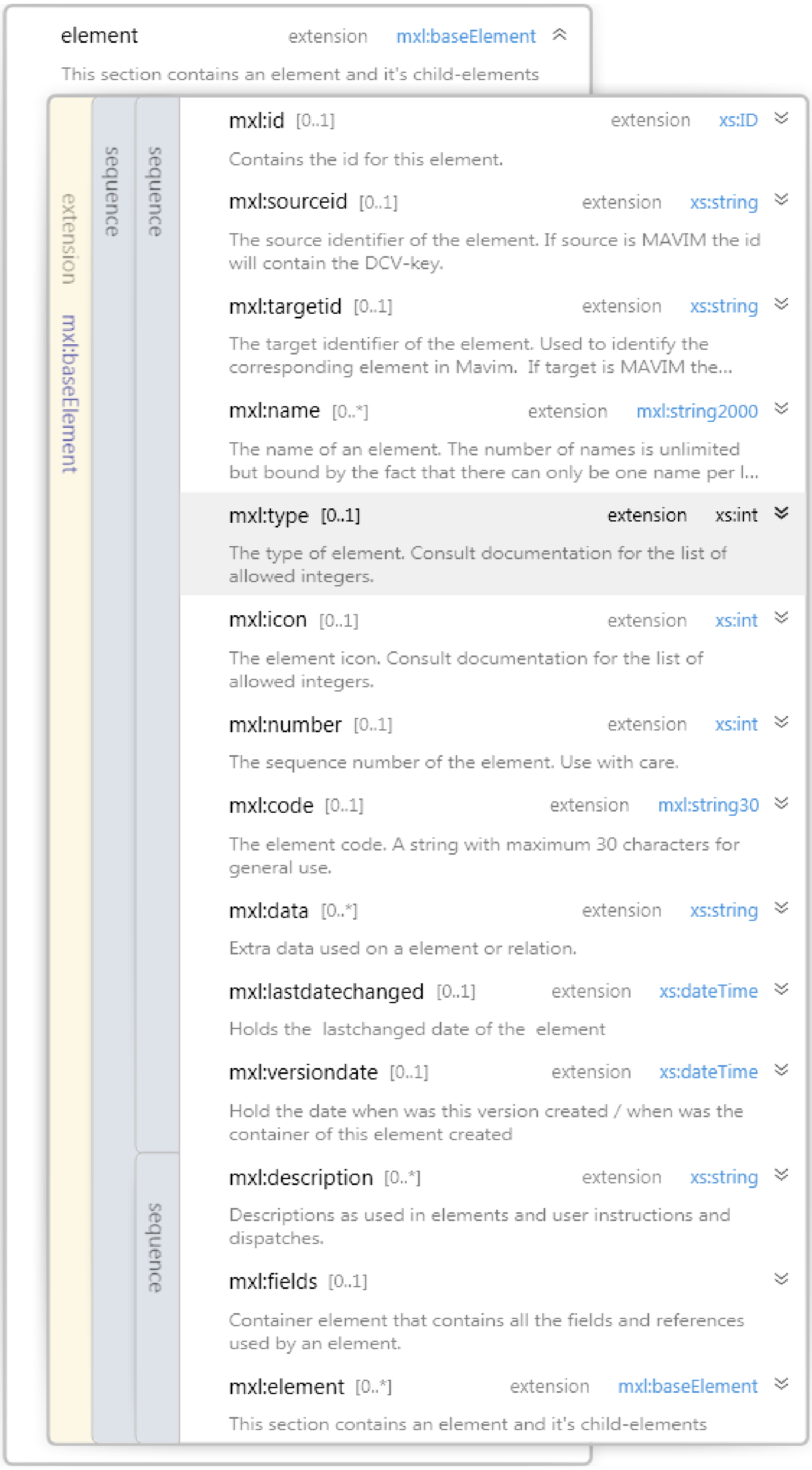
15.1.2.4 <mxl:element> - Mavim Manager topic

The Mavim Manager topic is written as element *<mxl:element>* and derived from the *mxl:baseElement* (see <mxl:baseElement> - General properties - Mavim Manager Topic). It contains all properties of this.

Also, the topic has a number of specific properties:

- <mxl:name> element
  The element is limited to one name per language and is checked using the required 'mxl:lang' xml attribute. The name can consist of a maximum of 2000 characters.
- <mxl:description> element
  Per language (mxl:lang) and per data type (mxl:type) a data block can be added to an element. For more information see: <mxl:description> - Description

- <mxl:fields> element
  Selection of data for the fields. See: <mxl:fields> - Fields
- <mxl:element> element
  Under a Mavim topic, multiple topics can be included as child topics (recursive). The hierarchical structure is described in this way.



element extension mxl:baseElement

This section contains an element and it's child-elements

mxl:id [0..1] extension xs:ID

Contains the id for this element.

mxl:sourceid [0..1] extension xs:string

The source identifier of the element. If source is MAVIM the id will contain the DCV-key.

mxl:targetid [0..1] extension xs:string

The target identifier of the element. Used to identify the corresponding element in Mavim. If target is MAVIM the...

mxl:name [0..*] extension mxl:string2000

The name of an element. The number of names is unlimited but bound by the fact that there can only be one name per l...

mxl:type [0..1] extension xs:int

The type of element. Consult documentation for the list of allowed integers.

mxl:icon [0..1] extension xs:int

The element icon. Consult documentation for the list of allowed integers.

mxl:number [0..1] extension xs:int

The sequence number of the element. Use with care.

mxl:code [0..1] extension mxl:string30

The element code. A string with maximum 30 characters for general use.

mxl:data [0..*] extension xs:string

Extra data used on a element or relation.

mxl:lastdatechanged [0..1] extension xs:dateTime

Holds the lastchanged date of the element

mxl:versiondate [0..1] extension xs:dateTime

Hold the date when was this version created / when was the container of this element created

mxl:description [0..*] extension xs:string

Descriptions as used in elements and user instructions and dispatches.

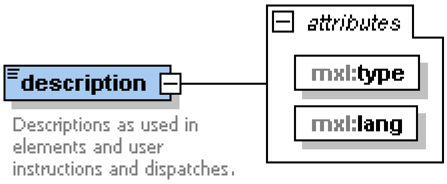
mxl:fields [0..1]

Container element that contains all the fields and references used by an element.

mxl:element [0..*] extension mxl:baseElement

This section contains an element and it's child-elements

15.1.2.4.1 <mxl:description> - Description


description
Descriptions as used in elements and user instructions and dispatches.
attributes
mxl:type
mxl:lang

The description element knows 2 required XML attributes:

- <mxl:lang> element. For the list of possible values see: Mavim data languages
- <mxl:type> element. The types are as follows:
  - ascii, description in ASCII-format
  - html, description in html-format
  - rtf, description in rich text-format
  - word, description as MSWord file. Base64 coded.

15.1.2.4.1.1 Hyperlinks in description

As in Rules, descriptions can contain hyperlinks to other topics. This applies to all supported formats. The hyperlinks always have the following format:

```
<#[mv_hyperlink ID=e1]>Hyp in Text</#[mv_hyperlink]>
```

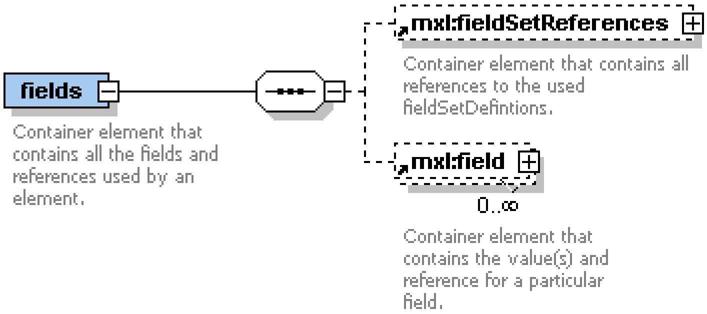
The code following ID= (in the example 'e1') is the ID of the topic (in the XML file) to which the hyperlink refers. The text between the two "tags" is the text that is shown in the description field of Rules as the XML file is imported.

Because data are in a CDATA block, the '[' and ']' must be escaped. In an ASCII format description, this will look as follows:

```
<#[[mv_hyperlink ID=e1]]]]><![CDATA[>Hyp in Text</#[[mv_hyperlink]]]]>
```

After e1 there are four square brackets, two of which close the hyperlink tag and two to close the CDATA block. After that a second CDATA block is opened. See example.

15.1.2.4.2 <mxl:fields> - Fields


fields
Container element that contains all the fields and references used by an element.
mxl:fieldSetReferences
Container element that contains all references to the used fieldSetDefintions.
mxl:field
0..∞
Container element that contains the value(s) and reference for a particular field.

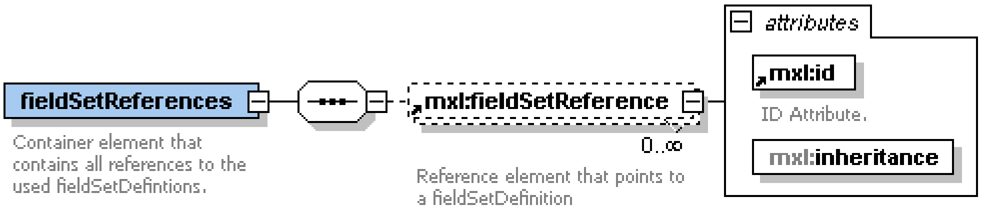
In the section <mxl:fields>, the field data are written. All fields have a definition that is described. See chapter Mavim data languages. This definition is referenced from a field.
Without a field definition, the XML file is not valid.
In a topic, the fields are placed under <mxl:fields>. There are two child elements:

- <mxl:fieldSetReferences> element
  Which contains a list of field set references.

- <mxl:field> element
  Which contains a list of values.

This is described in:
<mxl:fieldSetReferences> - Field set references
<mxl:field> - Fields for Mavim Rules topic

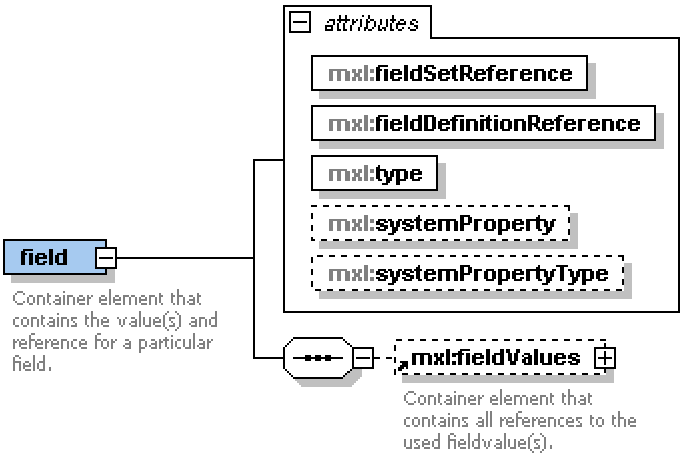
15.1.2.4.2.1 <mxl:fieldSetReferences> - Field set references


fieldSetReferences
Container element that contains all references to the used fieldSetDefintions.
mxl:fieldSetReference
0..∞
Reference element that points to a fieldSetDefinition
attributes
mxl:id
ID Attribute.
mxl:inheritance

Using a field set reference <mxl:fieldSetReference>, a field set definition is linked to the topic. A field set reference has the following properties:

- <mxl:id> element
  The topic(ID) which is referenced must be present in the XML file. This is always a field set definition. See <mxl:fieldSetDefinition> - Field set definition
- <mxl:inheritance> element
  This property specifies where the reference comes from and how it must be passed on to the subtopics. Possible values are:
  - topic_only -field set is only applied to this topic
  - topic_children -field set is applied to this topic and sub topics
  - children_only -field set is only applied to sub topics
  - branch -field set is applied to the entire branch
  - inherited_parent-field set is received from parent topic
  - inherited_branch-field set is received from parent topic and is valid for the entire branch
  - bytypedefinition-field set is received based on type. See <mxl:fieldSetByTypeGroup> - Type definitions

15.1.2.4.2.2 <mxl:field> - Fields for Mavim Manager topic


attributes
mxl:fieldSetReference
mxl:fieldDefinitionReference
mxl:type
mxl:systemProperty
mxl:systemPropertyType
field
Container element that contains the value(s) and reference for a particular field.
mxl:fieldValues
Container element that contains all references to the used fieldvalue(s).

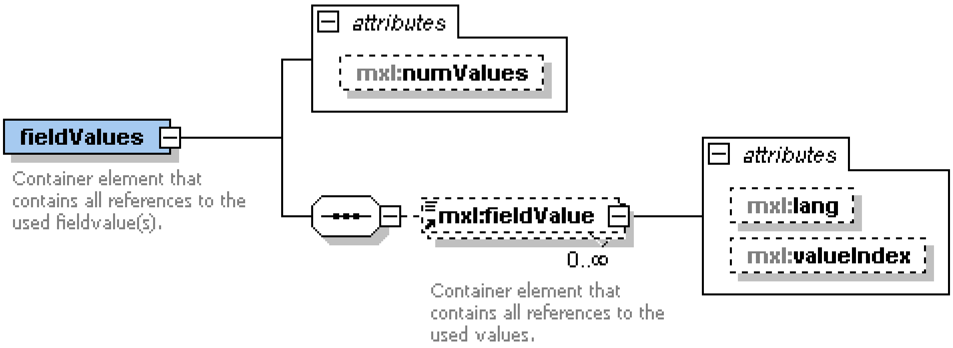
The actual data for fields is described in the <mxl:fields> element. This is the collection of elements of type <mxl:field>.

If the field definition is known for this topic (see <mxl:fieldSetReferences> - Field set references), then the actual data can be added to the topic.

Each field consists of the following properties:

- <mxl:fieldSetReference> element
  Together with the 'mxl:fieldDefinitionReference' attribute, this forms a unique key for the field. The topic ID always references a field set definition that must be present.
- <mxl:fieldDefinitionReference> element
  Together with the 'mxl:fieldSetReference' attribute, the attribute forms a unique key for the field. The topic ID always references a field definition that must be present.
- <mxl:type> element
  Indicates which data type (text, integer, boolean) the field contains. For possible values, see: Field types
- <mxl:systemProperty> element
  Possible values of the attribute are 'true' and 'false'
- <mxl:systemPropertyType> element
- <mxl:fieldValues> element
  Contains the collection of field data. See <mxl:fieldValues> - Field values

15.1.2.4.2.2.1 <mxl:fieldValues> - Field values


attributes
mxl:numValues
fieldValues
Container element that contains all references to the used fieldvalue(s).
mxl:fieldValue
0..∞
Container element that contains all references to the used values.
attributes
mxl:lang
mxl:valueIndex

Properties of field values are:

- <mxl:numValues> element
  If this attribute is present, it relates to a multiple value field. The value indicates how many values this involves.
- <mxl:fieldValue> element
  Within this element, the actual value of the field can be found. The element has two optional attributes that are self-explanatory:
  - <mxl:lang> element. If this attribute is present, it relates to a multi-language field. For possible values see Mavim data languages
  - <mxl:valueIndex> element. This value must be present for multiple value fields. (number of values attribute <mxl:numValues> present in the field values <mxl:fieldValues>). The index runs from 0 to numValues – 1.

15.1.2.5 <mxl:relation> - Mavim relationship

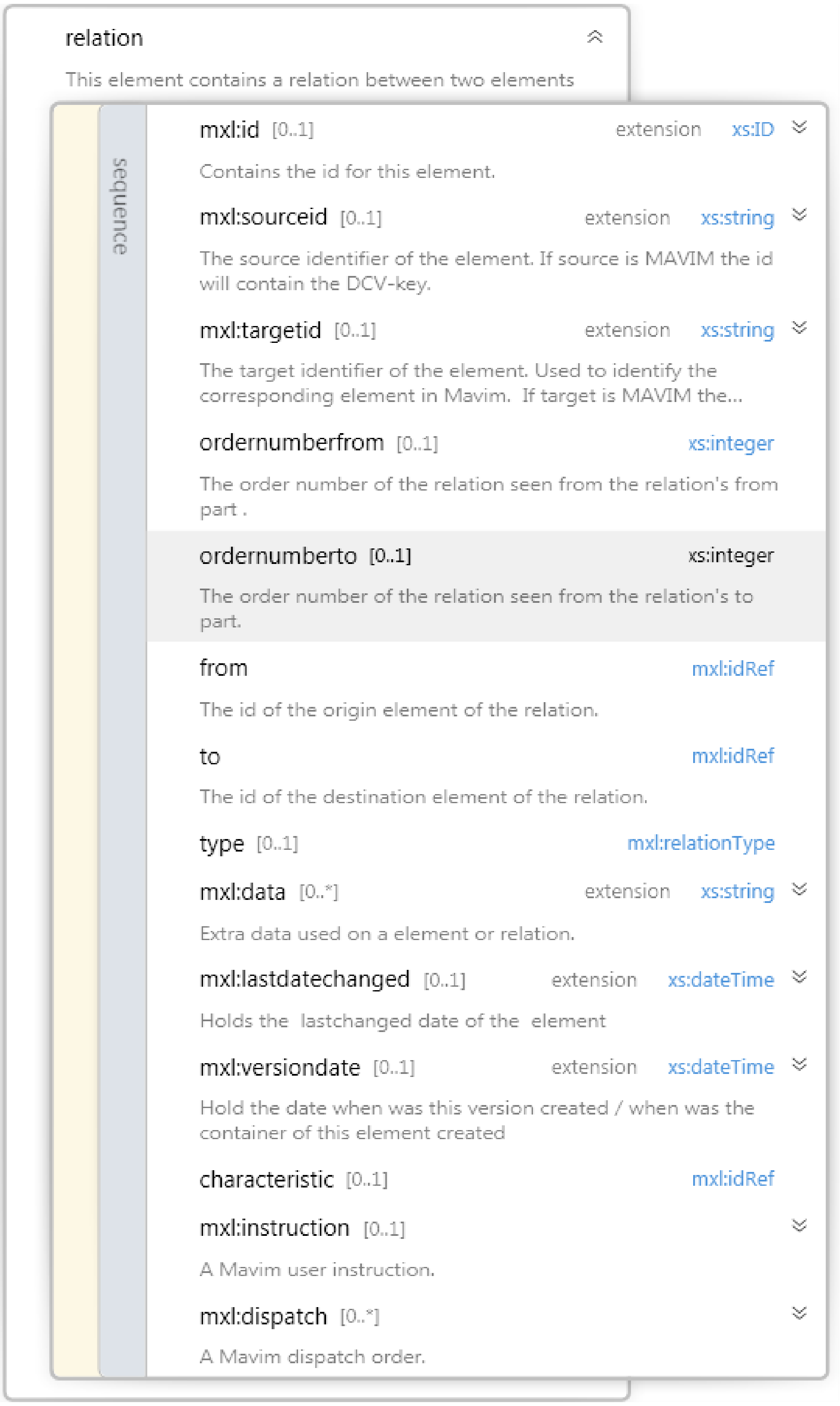
The Mavim relationship contains the following properties:

- <mxl:id> element
  The ID of the relationship. As with the topic, the ID is an XML tag of type "ID.
- <mxl:sourceid> element
  Free string for the key/ID of the relationship as identified in the source.
- <mxl:sourceid> element
  Free string for the key/ID of the relationship as identified in the target. If data were previously imported from a certain application, the keys/IDs can be included. This can help the target application to update "its" topics.

<mxl:from> element

The ID of the topic where the relationship comes from. The topic (ID) referenced must be present in the XML file

- <mxl:to> element
  The ID of the topic that the relationship references. The topic (ID) referenced must be present in the XML file.
- <mxl:ordernumberfrom> element
  The order number of the relationship seen from the from side of the relationship
- <mxl:ordernumberto> element
  The order number of the relationship seen from the to side of the relationship
- <mxl:type> element
  Determines the type of the relationship. For possible values, see: Mavim Manager relationship types.
- <mxl:characteristic> element
  The characteristic of the relationship. The topic (ID) referenced must be present in the XML file.
- <mxl:instruction> element
  For the description of the user instructions, see <mxl:instruction>
- <mxl:dispatch> element
  For the description of the dispatch instruction, see <mxl:dispatch>



relation

This element contains a relation between two elements

mxl:id [0..1] extension xs:ID

Contains the id for this element.

mxl:sourceid [0..1] extension xs:string

The source identifier of the element. If source is MAVIM the id will contain the DCV-key.

mxl:targetid [0..1] extension xs:string

The target identifier of the element. Used to identify the corresponding element in Mavim. If target is MAVIM the...

ordernumberfrom [0..1] xs:integer

The order number of the relation seen from the relation's from part .

ordernumberto [0..1] xs:integer

The order number of the relation seen from the relation's to part.

from mxl:idRef

The id of the origin element of the relation.

to mxl:idRef

The id of the destination element of the relation.

type [0..1] mxl:relationType

mxl:data [0..*] extension xs:string

Extra data used on a element or relation.

mxl:lastdatechanged [0..1] extension xs:dateTime

Holds the lastchanged date of the element

mxl:versiondate [0..1] extension xs:dateTime

Hold the date when was this version created / when was the container of this element created

characteristic [0..1] mxl:idRef

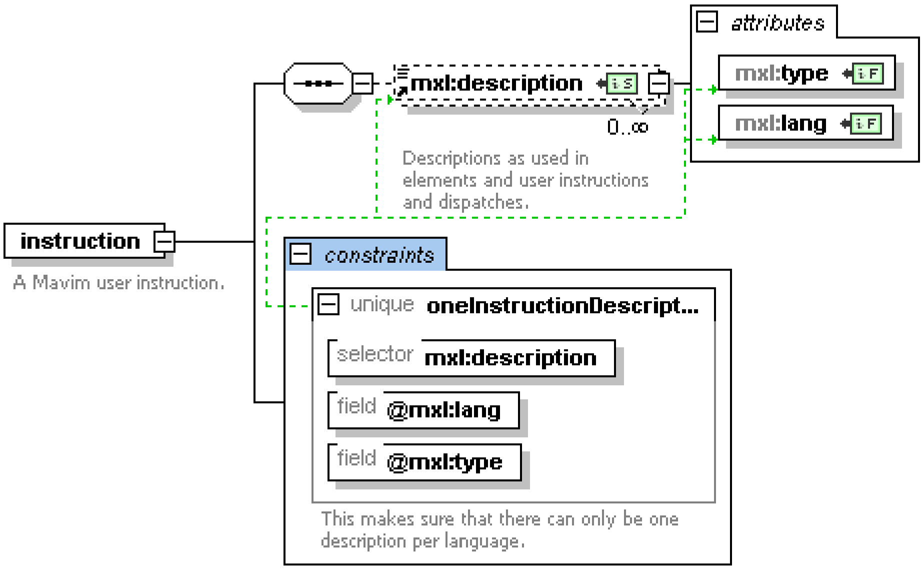
mxl:instruction [0..1]

A Mavim user instruction.

mxl:dispatch [0..*]

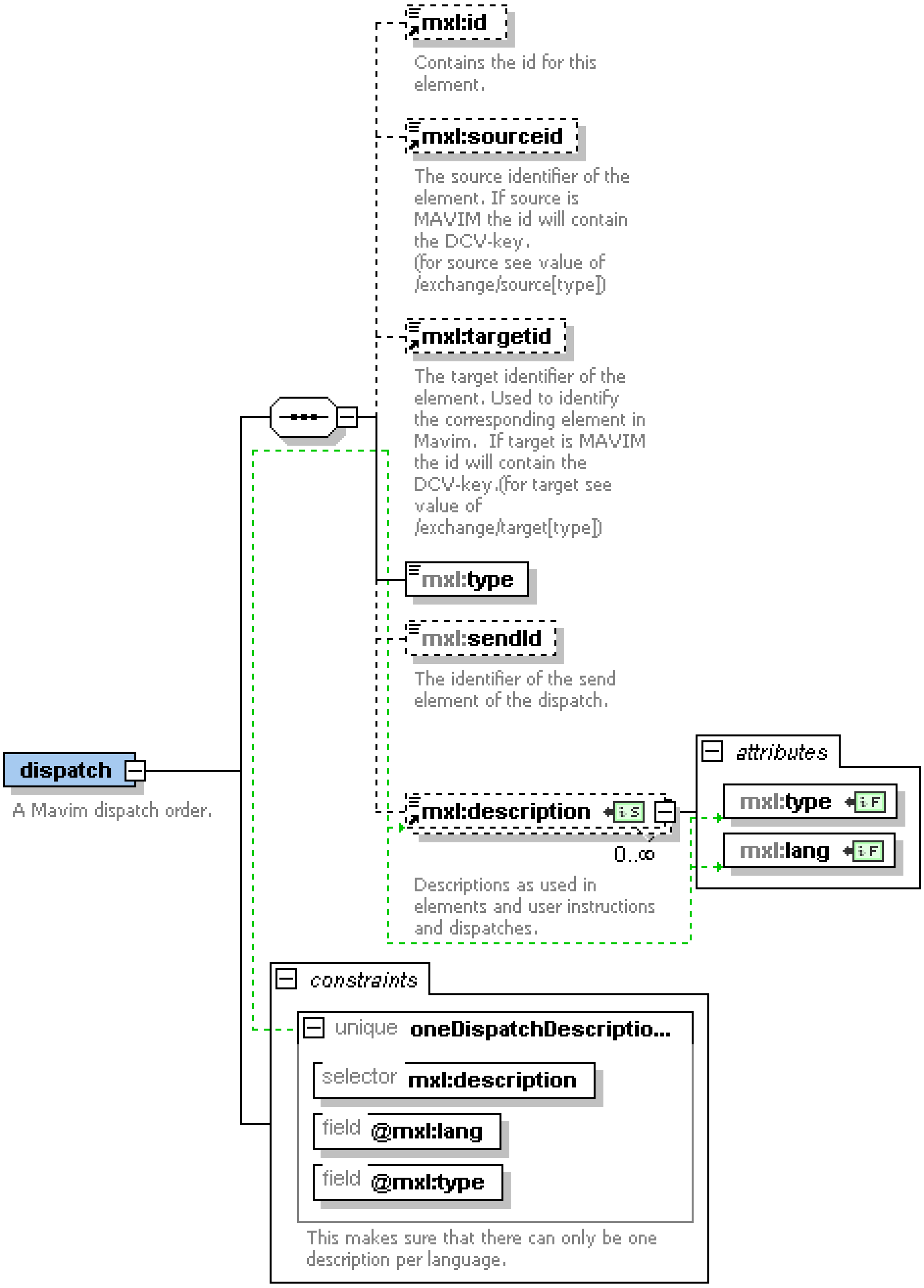
A Mavim dispatch order.

15.1.2.5.1 <mxl:instruction>


attributes
mxl:type
mxl:lang
mxl:description
0..∞
Descriptions as used in elements and user instructions and dispatches.
instruction
A Mavim user instruction.
constraints
unique oneInstructionDescript...
selector mxl:description
field @mxl:lang
field @mxl:type
This makes sure that there can only be one description per language.

A user instruction is a compiled topic and, like a Rules topic, can have descriptions in multiple languages and multiple formats. The valid restrictions are described in <mxl:description> - Description.

15.1.2.5.2 <mxl:dispatch>


mxl:id
Contains the id for this element.
mxl:sourceid
The source identifier of the element. If source is MAVIM the id will contain the DCV-key. (for source see value of /exchange/source[type])
mxl:targetid
The target identifier of the element. Used to identify the corresponding element in Mavim. If target is MAVIM the id will contain the DCV-key.(for target see value of /exchange/target[type])
mxl:type
mxl:sendId
The identifier of the send element of the dispatch.
dispatch
A Mavim dispatch order.
attributes
mxl:type
mxl:lang
mxl:description
0..∞
Descriptions as used in elements and user instructions and dispatches.
constraints
unique oneDispatchDescriptio...
selector mxl:description
field @mxl:lang
field @mxl:type
This makes sure that there can only be one description per language.

A dispatch instruction is optional and has no maximum limit. The dispatch instruction consists of the following elements:

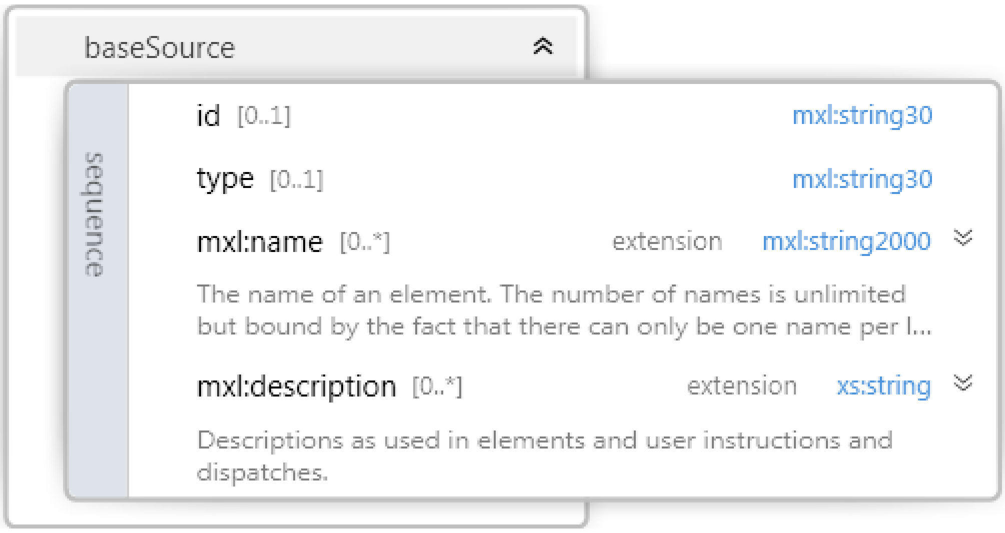
- <mxl:id> element
  The ID of the dispatch instruction is of XML type ID and is required. It must therefore fulfil the same conditions as the ID of a topic.
- <mxl:sourceid> element
  Free string for the key/ID of the dispatch instruction as identified in the source.
- <mxl:targetid> element
  Free string for the key/ID of the dispatch instruction as identified in the source. If data were previously imported from a certain application, the keys/IDs can be included. This can help the target application to update "its" topics.
- <mxl:type> element
  This element determines the type of the dispatch instruction. For possible values, see: Mavim Manager dispatch instruction types.

- <mxl:sendId> element
  This optional field describes the ID of the topic to which the dispatch instruction refers. The topic (ID) that is referenced must be present in the XML file.
- <mxl:description> element
  The description is optional and can contain multiple language formats. For this there is also a maximum of one description per combination of language and format.

15.1.3 Complex types

15.1.3.1 *<mxl:baseSource> - Source information*

This XML type described the source of the XML. De XML elements *mxl:source* en *mxl:target* zijn van dit type (zie <mxl:source> - source application information en
<mxl:target> - Target application information)



The name of an element. The number of names is unlimited but bound by the fact that there can only be one name per l...

Descriptions as used in elements and user instructions and dispatches.

15.1.3.2 *<mxl:baseElement> - General properties - Mavim Manager Topic*

The base element describes the properties for Mavim Manager topics. Topics are derived from this element and therefore contain the same properties. The base element itself from the schema is not used in the XML file.

The base element has the following properties:

- <mxl:id> element
  The internal ID of a topic. This only has meaning within one XML file. The format of the ID is limited to that of an XML tag of type "ID". That means e.g. that an ID must always start with a letter. Also, the ID must be unique within the file. This ID is also used with a hyperlink or a relationship to reference this topic.
- <mxl:sourceid> element
  Free string for the key/ID of the topic as identified in the source.
- <mxl:targetid> element
  Free string for the key/ID of the topic as identified in the source. If data were previously imported from a certain application the keys/IDs can be attached. This can help the target application to update "its" topics.
- <mxl:name> element
  The name of a topic. This has a required attribute for the language: <mxl:lang> For a list of possible values, see: Mavim data languages
  A name consists of a maximum of 2000 characters.

- <mxl:type> element
  The topic type is determined according to a numeric code. The possible topic types are covered in the Mavim Manager topic types. Some topic types relate to system elements. These can occur in the XML file as representatives but will never be created during an import into Rules.
- <mxl:icon> element
  The icon shown is determined according to the numeric code. The available icons are covered in Mavim Manager relationship types. In this list, you can see that there is a dependency between the topic type and the possible icons: some topic types have a fixed icon that cannot be changed.
- <mxl:number> element
  The topic sequence number is used to arrange topics under their parent topic.
- <mxl:code> element
  The code of a topic. A code can consist of a maximum of 30 characters.

<mxl:data> element

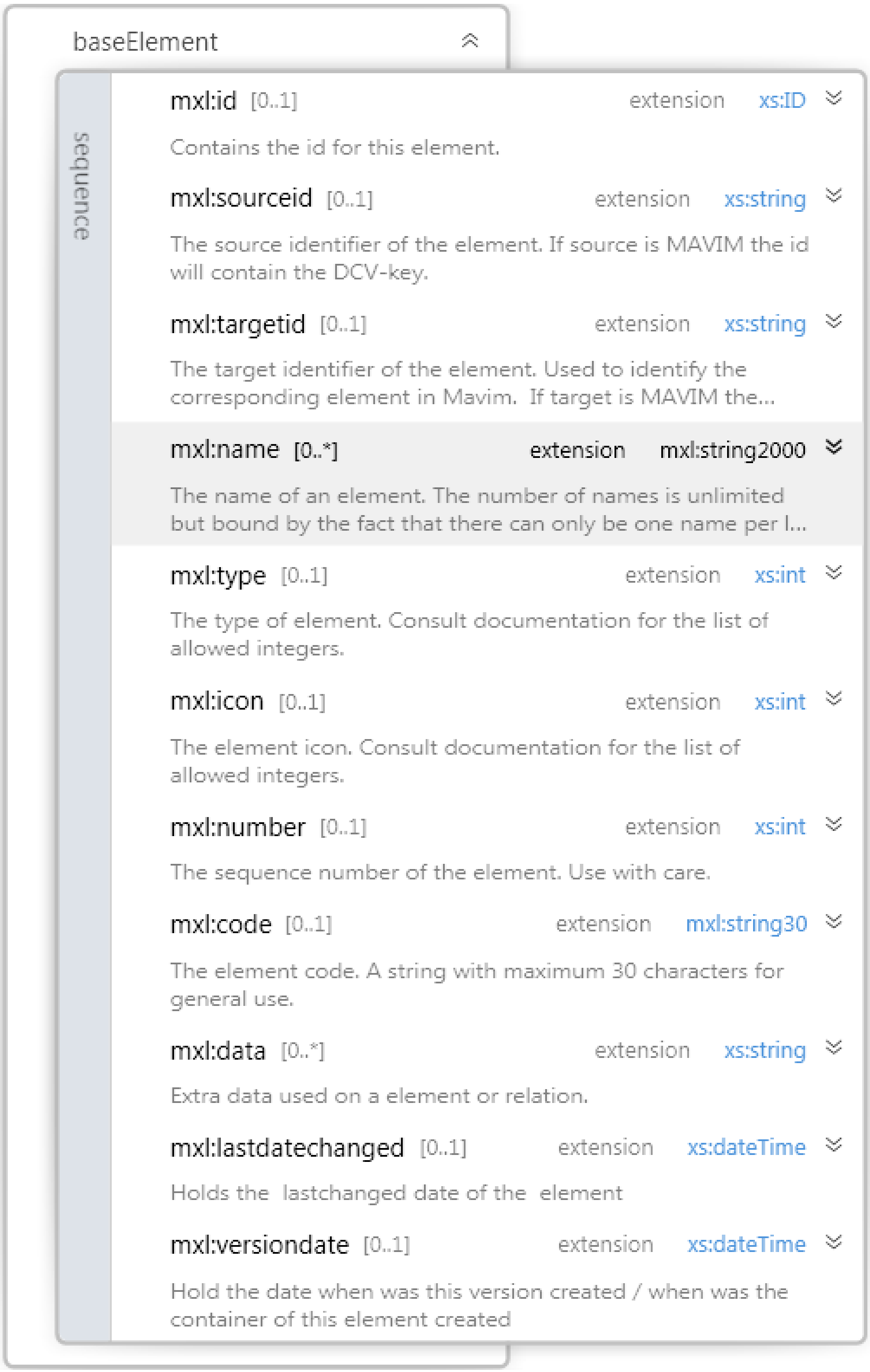
Contains corresponding data for a topic. This includes e.g. custom icons, internal and external files, Visio charts or matrices.
Only custom icons and internal and external files can be imported by Rules. This is not possible for Visio charts and matrices.

The element <mxl:type> is a required field and describes the type of data that are included. The types are:

- Icon
- Icon file
- Internal file
- External file
- Visio chart
- Matrix

The element <mxl:name> contains the string including the name of the data.

- <mxl:lastdatechanged> element
  Shows the date and time of the latest changes for this topic in the format:
  YYYY-MM-DD T H:MM:SS
- <mxl:versiondate> element
  Specific for versions: Shows the date and time for when the version was created. In the format
  YYYY-MM-DD T H:MM:SS



baseElement

sequence

mxl:id [0..1] extension xs:ID

Contains the id for this element.

mxl:sourceid [0..1] extension xs:string

The source identifier of the element. If source is MAVIM the id will contain the DCV-key.

mxl:targetid [0..1] extension xs:string

The target identifier of the element. Used to identify the corresponding element in Mavim. If target is MAVIM the...

mxl:name [0..*] extension mxl:string2000

The name of an element. The number of names is unlimited but bound by the fact that there can only be one name per l...

mxl:type [0..1] extension xs:int

The type of element. Consult documentation for the list of allowed integers.

mxl:icon [0..1] extension xs:int

The element icon. Consult documentation for the list of allowed integers.

mxl:number [0..1] extension xs:int

The sequence number of the element. Use with care.

mxl:code [0..1] extension mxl:string30

The element code. A string with maximum 30 characters for general use.

mxl:data [0..*] extension xs:string

Extra data used on a element or relation.

mxl:lastdatechanged [0..1] extension xs:dateTime

Holds the lastchanged date of the element

mxl:versiondate [0..1] extension xs:dateTime

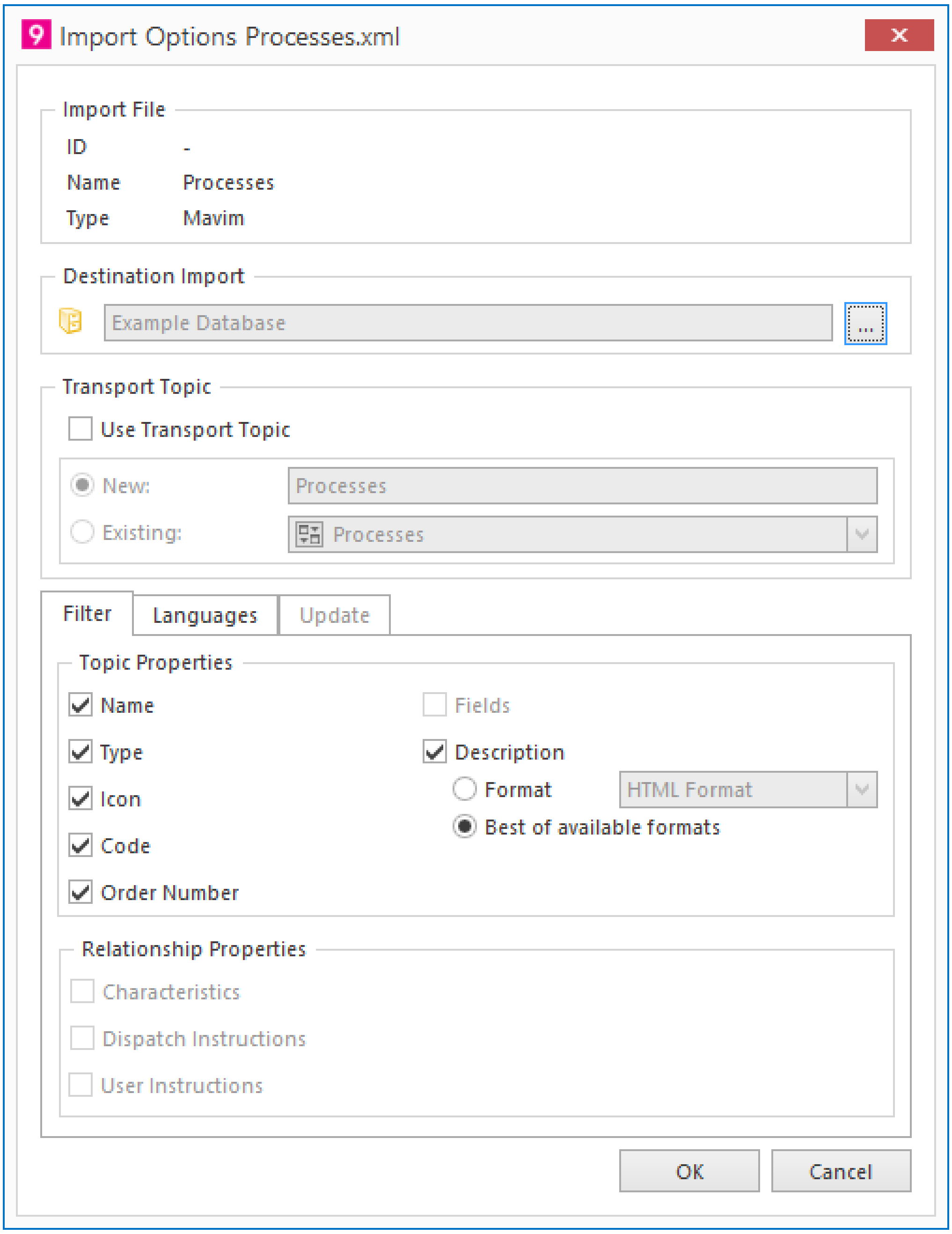
Hold the date when was this version created / when was the container of this element created

15.2 Import XML

Importing an XML document allows you to use data from another Mavim database or another source in your own database, independent of **Version Management**. To import an XML document:

1. Select the topic below which you want to import the XML information.
2. From the **File** tab select **XML Document** in the **Import** group.
3. Select the XML document at the correct file location and click **Open**.

4. Mavim then checks the XML for validity. If the document is valid, it is scanned for the available topic and/or relationship properties. For more information: see Scanning the XML Document.
5. The **Import Options** screen will appear with a number of sections at the top and below that the tabs **Filter**, **Languages** and **Update**:



Import Options Processes.xml

Import File

ID	-
Name	Processes
Type	Mavim

Destination Import

Example Database

Transport Topic

☐ Use Transport Topic

● New: Processes

○ Existing: Processes

Topic Properties

☑ Name

☑ Type

☑ Icon

☑ Code

☑ Order Number

☐ Fields

☑ Description

○ Format: HTML Format

● Best of available formats

Relationship Properties

☐ Characteristics

☐ Dispatch Instructions

☐ User Instructions

Based on the available topic and or relationship properties, the corresponding options are enabled and selected. If you are using an existing transport topic, the last used options are selected.

6. Adjust the following settings to your preferences:
Destination Import
Here you can select the topic below which you want to import the XML document. Not all topics can be imported, see List of importable topics.

Transport Topic
Indicate whether you want to use a transport topic, and if so, if you want to create a new transport topic map or if you want to use an existing one. You do not have to use a transport topic. If using an

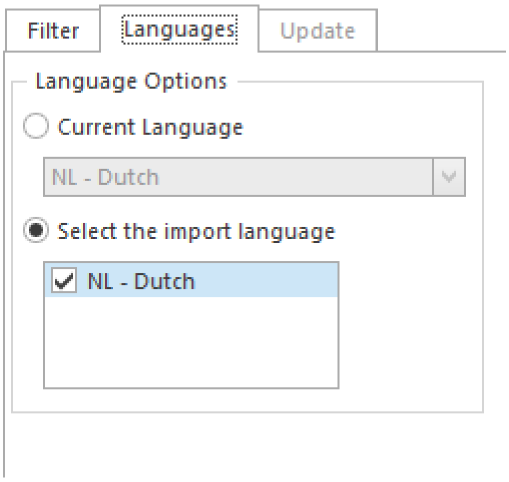
existing transport topic map, the last used **Filter** and **Languages** settings will be applied. See also: XML transport topic.

Filter

By default all topic and relationship properties are imported. Here you can indicate which properties you do or do not want to include in the import.
The description can be imported in four different formats. Importing in Word, RTF and HTML format fully retains the layout and the images in a description. ASCII format does not retain the images.

For relationships you can set whether you will import the characteristics, dispatch instructions, and/or user instructions.

Languages



Filter | Languages | Update

Language Options

○ Current Language

NL - Dutch

● Select the import language

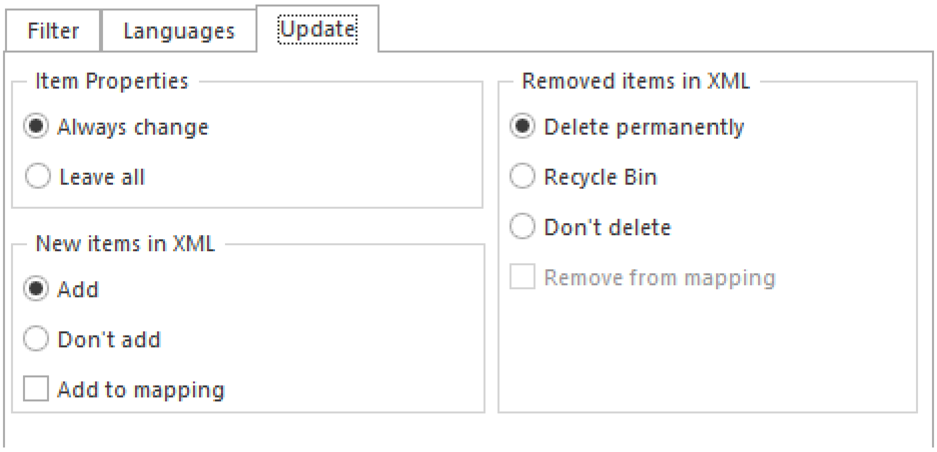
☑ NL - Dutch

In this tab you indicate how the language-dependent topics are processed in the XML document. The following two options are available:

- **Current language**: only those properties that are in the same language as the current data language in Mavim will be imported. The user can choose which of the source languages available in the XML file will be used. This makes it possible to move data from one language to another.
- **Select the import language**: you can specify which of the languages available in the XML file will be imported into the Mavim database. The default option is that all properties in all available languages are imported.

> If the XML file consists of languages other than the currently active language, it can also be imported. The topics in the active language will be given the default topic names.

Update



Item Properties

● Always change

○ Leave all

New items in XML

● Add

○ Don't add

□ Add to mapping

Removed items in XML

● Delete permanently

○ Recycle Bin

○ Don't delete

□ Remove from mapping

If you are using an existing transport topic map (repeated import), the Update tab becomes enabled. In this tab, you can arrange the following:

- How topic and/or relationship properties should be updated.
- Whether new topics and/or relationships added to the XML document since the last import should be imported into Mavim.
- Whether topics and/or relationships removed from the XML document since the last import should be removed from Mavim, and if so, how they should be removed.

Item Properties
This option allows you to determine how you want to update the properties of topics and/or relationships that were previously imported into Mavim and are included in the transport topic map.

- Always change: the properties of the topics and or relationships in the XML document are updated in Mavim.
- Leave all: the properties and/or relationships in the XML document are not updated in Mavim.

New Items in XML
This option allows you to specify whether you want to import topics and/or relationships that were added to the XML document since the last import, and if yes, how you want to import them.

- Add: the new topics and/or relationships will be imported into Mavim.
- Don't add: the new topics and/or relationships will not be imported into Mavim.
- Add to mapping: the new topics and/or relationships will be added to the transport topic map.

Deleted Items in XML
This option allows you to specify how you want to remove topics and/or relationships from Mavim that have been removed from the XML document since the last import.

- Delete permanently: the removed topics/and or relationships are permanently deleted from Mavim. They are also removed from the transport topic map.
- Recycle bin: the removed topics and/or relationships are moved to the recycle bin.
- Don't delete: the removed topics and/or relationships are not deleted.
- Remove from mapping: removed topics and/or relationships will be removed from the transport topic map. This option is only available when you have chosen Recycle bin or Don't delete.

7. Click OK to confirm import of the XML document with the selected settings.

For a further explanation of XML, see the Technical Documentation.

15.2.1 Scanning the XML Document

Mavim scans the document for the following reasons:

- Mavim checks whether the document is in XML format. If not, Mavim will show an error notification and cancel the import.
- Mavim checks whether the document conforms to the Mavim schema (that the file is valid).

- Mavim checks whether the XML document has been previously imported in order to select the corresponding transport topic map.
- Mavim checks whether there are topics and/or relationships in the document that have a unique ID. This is a prerequisite for being able to create or use a transport topic map for updating topics and relationships.
- Mavim checks which topic and relationship properties are available in the document.
- Mavim checks which language-dependent properties of topics and relationships (such as name, description, user instructions, dispatch instructions etc.) are available in the document and in which language.

15.2.2 List of importable topics

Type Name	Type ID
Imported Version	5
Primary Topic	9
Topic	10
Standard Topic	30
Start Activity	31
Stop Activity	32
Decision	33
Staff Department	34
Detailed Process	35
Process	36
Module	37
Department	38
Activity	39
User Defined Topic Type (1)	40
User Defined Topic Type (2)	41
User Defined Topic Type (3)	42
User Defined Topic Type (4)	43
Group Topic	66
External File	101
Internal File	102
User Defined Characteristics	110
Condition	112
Comments on current version	125
SAP Connections	450
SAP Connection	451
Connections	455
M3 Connections	463
M3 Connection	464
AX Connections	465
AX Connection	466
Icons	470
Icon	471
Start Event	510
Event	511
Stop Event	512
Function	513

AND	514
OR	515
XOR	516
Stop Function	517
AND	518
OR	519
XOR	520
Field Settings	550
Field Definitions	551
Field Sets	552
Field Set	553
Field	554
Field	555
Field	556
Field	557
Field	558
Field	559
Field	560
Cockpit Data	561
Minimum	562
Maximum	563
Target	564
Field	565
Field	566
Type Definition (Not Used)	573
SAP properties	590
SAP Title	591
SAP System	592
SAP TransactionID	593
SAP Client	594
SAP Language	595
Active Directory Group	600
Custom Group	601
Active Directory User	602
Entity	603
Entity Instance	604
Domains	610
Domain	611
Transactions	612
Transaction	613
Business Objects	614
Business Object	615
Sequence	617
Phase	618
Gateway	653
Event	654

Task	655
Sub-Process	656
Collaboration Chart	657
Boundary Event	658
Choreography Activity	659
Choreography Sub-Process	660
Participants	661
Participant	662
Data Objects	663
Data Object	664
Data Stores	665
Data Store	666
Global Tasks	667
Global Task	668
Global Choreography Tasks	669
Global Choreography Task	670
Global Conversations	671
Global Conversation	672
Resources	680
Resource	681
Orchestrations	682
Collaborations	683
Choreographies	684
Conversations	685
Orchestration Chart	686
Pool Process	687
Choreography Chart	688
Conversation Chart	689
Conversation	690
Call Activity	691
Call Choreography	692
Call Conversation	693
Sub-Conversation	694
Event Sub-Process	695
Catalogs	696
Model	697
SharePoint Connections	700
SharePoint Connection	701
View	702
Business Layer Folder	703
Application Layer Folder	704
Technology Layer Folder	705
Motivation Extension Folder	706
Implementation Migration Extension Folder	707
Business Actor	708
Business Actors	709

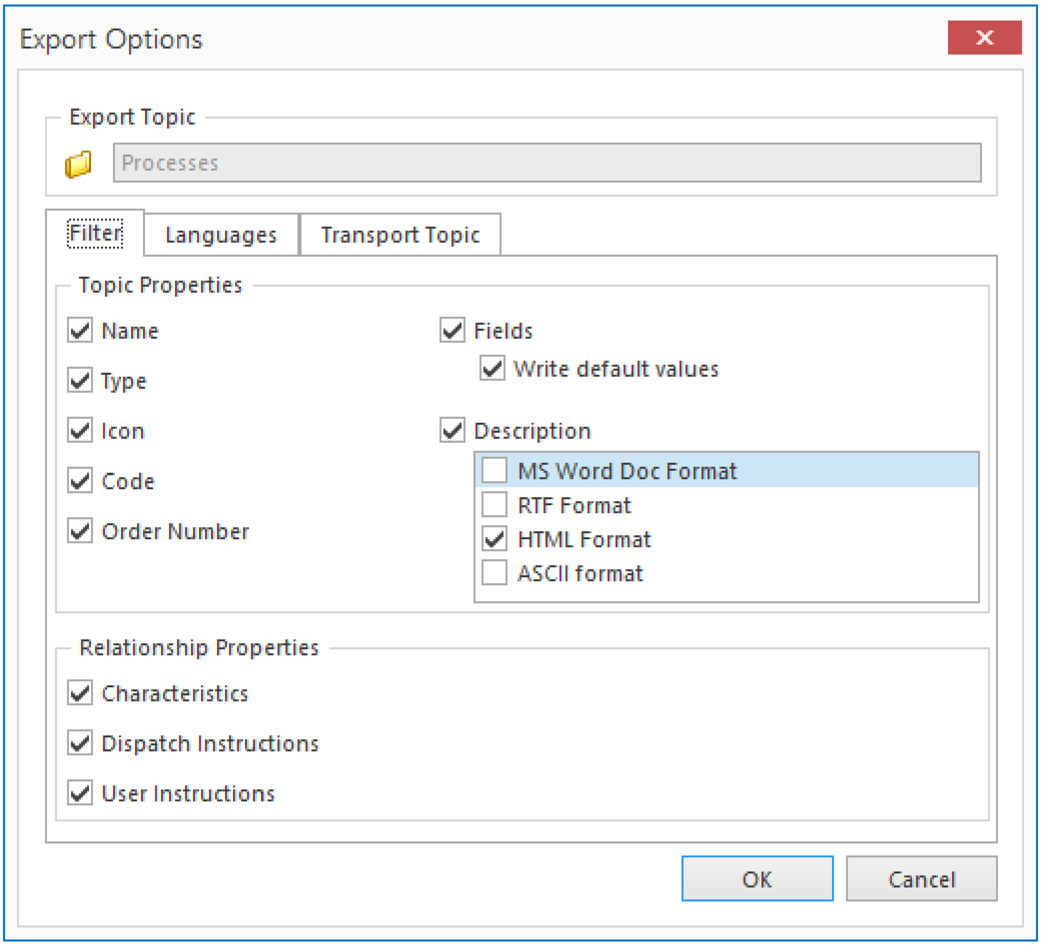
Business Role	710
Business Roles	711
Business Collaboration	712
Business Collaborations	713
Business Interface	714
Business Interfaces	715
Location	716
Locations	717
Business Process	718
Business Processes	719
Business Function	720
Business Functions	721
Business Interaction	722
Business Interactions	723
Business Event	724
Business Events	725
Business Service	726
Business Services	727
Business Object	728
Business Objects	729
Business Representation	730
Business Representations	731
Meaning	732
Meanings	733
Value	734
Values	735
Product	736
Products	737
Contract	738
Contracts	739
Application Component	740
Application Components	741
Application Collaboration	742
Application Collaborations	743
Application Interface	744
Application Interfaces	745
Application Function	746
Application Functions	747
Application Interaction	748
Application Interactions	749
Application Service	750
Application Services	751
Data Object	752
Data Objects	753
Node	754
Nodes	755

Device	756
Devices	757
Network	758
Networks	759
Communication Path	760
Communication Paths	761
Infrastructure Interface	762
Infrastructure Interfaces	763
System Software	764
System Softwares	765
Infrastructure Function	766
Infrastructure Functions	767
Infrastructure Service	768
Infrastructure Services	769
Artifact	770
Artifacts	771
Stakeholder	772
Stakeholders	773
Driver	774
Drivers	775
Assessment	776
Assessments	777
Goal	778
Goals	779
Principle	780
Principles	781
Requirement	782
Requirements	783
Constraint	784
Constraints	785
Work Package	786
Work Packages	787
Deliverable	788
Deliverables	789
Plateau	790
Plateaus	791
Gap	792
Gaps	793
Group	794
Junction	795
Views	796
Groups	797
Junctions	798
Generic Topic	799
Generic Topics	800
Generic Layer Folder	801

15.3 Export XML

Exporting an XML document is intended for allowing use of data from one Mavim database in other Mavim databases or other systems, independently of Version Management. To generate an XML document:

1. Choose the topic you want to export in the **Topics** screen.
2. From the **File** tab select **XML Document** in the **Export** group.
3. The **Export Options** screen will appear, with the topic you wish to export and below that the tabs **Filter**, **Languages** and **Transport Topic**.



Export Options

Export Topic

Processes

Topic Properties

☑ Name
☑ Type
☑ Icon
☑ Code
☑ Order Number
☑ Fields
☑ Write default values
☑ Description
□ MS Word Doc Format
□ RTF Format
☑ HTML Format
□ ASCII format

Relationship Properties

☑ Characteristics
☑ Dispatch Instructions
☑ User Instructions

4. Adjust the following settings to your preferences:

Export topic
This shows the topic and its underlying hierarchy that you want to export. Not all topics can be exported, see: List of exportable topics.

Filter
All properties of topics and relationships are exported by default. Here you can indicate which properties you want to include in the export.
The description can be exported in four different formats. Export in Word or RTF format fully retains the layout and the images in a description. In ASCII format the images are not included. In HTML format the images are placed in a separate folder called <xmlfilename>_files.

Only relationships between exported topics are included in the export. For these relationships you can indicate whether you want to export the characteristics, dispatch and/or user instructions. Characteristics do not need to be explicitly added to the export. If they are defined for a relationship, they are automatically included in the export.
Dispatch instructions are only included if the topic to which the dispatch instruction refers is part of the export.

Languages
By default names and descriptions of topics and instructions are exported in the active data language. Upon export it is possible to include or select other languages than the active language, as long as those languages are activated.
Upon import of the XML document in Mavim, you can explicitly set which languages will be imported. This allows you to deviate from the settings you have selected here.

Transport topic
Indicate whether or not you want to use a transport topic map. If this is the case you can select whether you want to use an existing transport topic map or create a new one. It is also possible to not use a transport topic map (this is the **Default** setting). If using an existing transport topic map, the last used **Filter** and **Languages** settings will be applied. See also: XML transport topic

5. Click **OK** to confirm creating the XML document with the selected settings.
6. Give a name and a path for the XML document and click **Save**.

For a further explanation of XML, see Technical Documentation.

15.3.1 List of exportable topics

Type Name	Type ID
Imported Version	5
Version	6
Comment on imported version	7
Comments on current version	8
Primary Topic	9
Topic	10
Relationship Categories	13
External References	14
Standard Topic	30
Start Activity	31
Stop Activity	32
Decision	33
Staff Department	34
Detailed Process	35
Process	36
Module	37
Department	38
Activity	39
User Defined Topic Type (1)	40
User Defined Topic Type (2)	41
User Defined Topic Type (3)	42
User Defined Topic Type (4)	43
External Files	53
Matrices	54
Charts	55

Internal Files	56
Characteristics	59
With What	60
Who	61
Where	62
When	63
Why	64
Group Topic	66
Matrix Range Definition	67
Conditions	68
Matrix Characteristic Definition	79
Matrix	100
External File	101
Internal File	102
User Defined Characteristics	110
Condition	112
Create	116
Read	117
Edit	118
Delete	119
Comments	124
Comments on current version	125
With What Characteristic	150
RoleType	151
Where Characteristic	152
When Characteristic	153
Why Characteristic	154
Whereto	156
Matrix Definitions	160
Matrix Definition	161
Comment on imported version	170
Flowchart	406
Hierarchical Chart	407
IDEF0 Chart	408
Network Chart	409
Generic Chart	411
Pronto Chart	413
Cross Function Flowchart	415
◆	430
EPC Chart	432
Cockpit Definition	440
BPMN 2.0	444
BPMN	445
MindManager Map	446
ArchiMate	447
SAP Connections	450

SAP Connection	451
Connections	455
M3 Connections	463
M3 Connection	464
AX Connections	465
AX Connection	466
Icons	470
Icon	471
Start Event	510
Event	511
Stop Event	512
Function	513
AND	514
OR	515
XOR	516
Stop Function	517
AND	518
OR	519
XOR	520
Field Settings	550
Field Definitions	551
Field Sets	552
Field Set	553
Field	554
Field	555
Field	556
Field	557
Field	558
Field	559
Field	560
Cockpit Data	561
Minimum	562
Maximum	563
Target	564
Field	565
Field	566
View	571
Type Definition (Not Used)	573
SAP properties	590
SAP Title	591
SAP System	592
SAP TransactionID	593
SAP Client	594
SAP Language	595
	599
Active Directory Group	600

Custom Group	601
Active Directory User	602
Entity	603
Entity Instance	604
	605
Domains	610
Domain	611
Transactions	612
Transaction	613
Business Objects	614
Business Object	615
Sequence	617
Phase	618
Web Form Definitions	620
Web Form definition	621
Report Definitions	630
Report Definition	631
Search Items	632
Search Item	633
Characteriscs	634
Icons	635
Fields	636
Field	637
Report View	639
	640
Templates	641
Template	642
Gateway	653
Event	654
Task	655
Sub-Process	656
Collaboration Chart	657
Boundary Event	658
Choreography Activity	659
Choreography Sub-Process	660
Participants	661
Participant	662
Data Objects	663
Data Object	664
Data Stores	665
Data Store	666
Global Tasks	667
Global Task	668
Global Choreography Tasks	669
Global Choreography Task	670
Global Conversations	671

Global Conversation	672
Resources	680
Resource	681
Orchestrations	682
Collaborations	683
Choreographies	684
Conversations	685
Orchestration Chart	686
Pool Process	687
Choreography Chart	688
Conversation Chart	689
Conversation	690
Call Activity	691
Call Choreography	692
Call Conversation	693
Sub-Conversation	694
Event Sub-Process	695
Catalogs	696
Model	697
SharePoint Connections	700
SharePoint Connection	701
View	702
Business Layer Folder	703
Application Layer Folder	704
Technology Layer Folder	705
Motivation Extension Folder	706
Implementation Migration Extension Folder	707
Business Actor	708
Business Actors	709
Business Role	710
Business Roles	711
Business Collaboration	712
Business Collaborations	713
Business Interface	714
Business Interfaces	715
Location	716
Locations	717
Business Process	718
Business Processes	719
Business Function	720
Business Functions	721
Business Interaction	722
Business Interactions	723
Business Event	724
Business Events	725
Business Service	726

Business Services	727
Business Object	728
Business Objects	729
Business Representation	730
Business Representations	731
Meaning	732
Meanings	733
Value	734
Values	735
Product	736
Products	737
Contract	738
Contracts	739
Application Component	740
Application Components	741
Application Collaboration	742
Application Collaborations	743
Application Interface	744
Application Interfaces	745
Application Function	746
Application Functions	747
Application Interaction	748
Application Interactions	749
Application Service	750
Application Services	751
Data Object	752
Data Objects	753
Node	754
Nodes	755
Device	756
Devices	757
Network	758
Networks	759
Communication Path	760
Communication Paths	761
Infrastructure Interface	762
Infrastructure Interfaces	763
System Software	764
System Softwares	765
Infrastructure Function	766
Infrastructure Functions	767
Infrastructure Service	768
Infrastructure Services	769
Artifact	770
Artifacts	771
Stakeholder	772

Stakeholders	773
Driver	774
Drivers	775
Assessment	776
Assessments	777
Goal	778
Goals	779
Principle	780
Principles	781
Requirement	782
Requirements	783
Constraint	784
Constraints	785
Work Package	786
Work Packages	787
Deliverable	788
Deliverables	789
Plateau	790
Plateaus	791
Gap	792
Gaps	793
Group	794
Junction	795
Views	796
Groups	797
Junctions	798
Generic Topic	799
Generic Topics	800
Generic Layer Folder	801

15.4 Repeated import and export

As mentioned previously, topic IDs play an important part in the recognition of topics during repeated import and/or export. This section explains how you can import data from external systems into Mavim Manager in such a way that they will be automatically updated. The reverse scenario, exporting data from Mavim Manager for use in external systems, is covered in the next section.

Repeated import of data from external systems into Rules

To import data into Mavim Manager (such as the output of employee data from an organisation system), you must convert the data to the Mavim Manager XML format. Your external system may supply the data in Mavim Manager XML format directly. You can also perform XML conversion using XSLT scripts (as explained in the following section) to convert from one XML format to the Mavim XML format. It important to connect a **source ID** to a data cluster (see figure below).

```
        <mxl:source>
                <mxl:id>D123456</mxl:id>
                <mxl:type>EXCEL</mxl:type>
                <mxl:name mxl:lang=”en”>Example</mxl:name>
```

```
            </mxl:source>
```

Example of a source ID

You will have to create this string yourself. It will ensure that Mavim Manager will recognise the cluster, and will update the data upon repeated import. Additionally, every element needs to be assigned a unique ID. Without IDs, Mavim Manager would always create new topics, and never update existing topics. Both forms of identification need to be retained for a second export to Mavim Manager. Therefore it is advisable to use a map similar to that created by Mavim Manager for the export of data. This is essential where data must be re-imported into the external systems after being modified in Mavim Manager.

Repeated export of data from Mavim Manager for use in external systems

Upon export of data from Mavim Manager, a source ID will be generated automatically, when using a transport topic at least. The source ID is not of major significance for exports, because Mavim Manager always generates IDs for topics and relationships independently of the use of a transport topic. It is only when re-importing into Mavim Manager that the source ID is important.

As with data imports, it will probably be necessary to create a map to allow the repeated importing of data into external systems. In many cases, the external system will have different requirements for the format of the IDs. Additionally, the properties of Mavim Manager topics (name, description, code etc.) will often be separated in external systems, or conversely, separate Mavim Manager topics might be combined into a data cluster (e.g. a record in a database). In both instances, an mpa will have to ensure the conversion of Mavim Manager topics to the data format used in external systems.

15.4.1 Conversion

The use of XML is excellent for converting to other file formats. Considering the extensive possibilities, it would be impractical to give a detailed description. A few aspects are detailed below:
Firstly, an XSLT file is needed to convert an XML file to the required file format. This file format can consist of another XML format or plain text. An application such as MS Visual Studio can perform the conversions. It is also possible to use code to perform the conversion in an application.

Below, an XSLT script is used to show how a Mavim Manager XML file can be converted into a CSV file which can then be opened in MS Excel, for example.

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0" xmlns:mxl="http://www.mavim.com/ns/mxl"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">

<xsl:param name="lang" select="'nl'"/>

<xsl:output method="text" omit-xml-declaration="yes" indent="no" encoding="iso-8859-1"/>

<xsl:strip-space elements="*"/>

<xsl:template match="text()"/>

<xsl:template match="mxl:element">
            <xsl:call-template name="remove-commas"><xsl:with-param name="string"
select="mxl:name[@mxl:lang=$lang]"/></xsl:call-template>,<xsl:call-template name="remove-
commas"><xsl:with-param name="string" select="mxl:code"/></xsl:call-template>,<xsl:call-template
name="remove-commas"><xsl:with-param name="string" select="mxl:number"/></xsl:call-
template>,<xsl:call-template name="remove-commas"><xsl:with-param name="string"
select="mxl:description[@mxl:type='ascii' and @mxl:lang=$lang]"/></xsl:call-
template><xsl:text>&#10;</xsl:text>
            <xsl:apply-templates/>
</xsl:template>
```

```
<xsl:template name="remove-commas">
            <xsl:param name="string"/>
            <xsl:value-of select="translate(normalize-space($string),',','')"/>
</xsl:template>
</xsl:stylesheet>
```

15.5 Supporting Data

15.5.1 Mavim data languages

Mavim data languages are indicated using the <mxl:lang> attribute. This is in the xs;language (iso) format. The values that can be used are:

- nl -Dutch
- en -English
- de -German
- cs -Czech
- da -Danish
- fi -Finnish
- fr -French
- hu -Hungarian
- it -Italian
- mi -Maori
- nb -Norwegian
- pl -Polish
- pt -Portuguese
- ru -Russian
- zh -Chinese
- es -Spanish
- sv -Swedish
- tr -Turkish
- el -Greek

15.5.2 Mavim Manager topic types

Type Name	Type ID
Primary Topic	9
Topic	10
Process	36
Detailed Process	35
Start Activity	31
Activity	39
Decision	33
Stop Activity	32
Start Event	510
Event	511
Stop Event	512
Function	513

AND	514
OR	515
XOR	516
Department	38
Staff Department	34
Module	37
Group Topic	66
Standard Topic	30
User Defined Topic Type (1)	40
User Defined Topic Type (2)	41
User Defined Topic Type (3)	42
User Defined Topic Type (4)	43
Active Directory User	602
Active Directory Group	600
Custom Group	601
Entity	603
Entity Instance	604
Domains	610
Domain	611
Transactions	612
Transaction	613
Business Objects	614
Business Object	615
Task	655
Sub-Process	656
Gateway	653
Event	654
Boundary Event	658
Pool Process	687
Choreography Activity	659
Choreography Sub-Process	660
Conversation	690
Orchestration Chart	686
Collaboration Chart	657
Choreography Chart	688
Conversation Chart	689
Participant	662
Data Object	664
Data Store	666
Global Task	668
Global Choreography Task	670
Global Conversation	672
Participants	661
Data Objects	663
Data Stores	665
Global Tasks	667

Global Choreography Tasks	669
Global Conversations	671
Orchestrations	682
Collaborations	683
Choreographies	684
Conversations	685
Catalogs	696
Business Layer Folder	703
Application Layer Folder	704
Technology Layer Folder	705
Motivation Extension Folder	706
Implementation Migration Extension Folder	707
Generic Layer Folder	801
Business Actor	708
Business Actors	709
Business Role	710
Business Roles	711
Business Collaboration	712
Business Collaborations	713
Business Interface	714
Business Interfaces	715
Location	716
Locations	717
Business Process	718
Business Processes	719
Business Function	720
Business Functions	721
Business Interaction	722
Business Interactions	723
Business Event	724
Business Events	725
Business Service	726
Business Services	727
Business Object	728
Business Objects	729
Business Representation	730
Business Representations	731
Meaning	732
Meanings	733
Value	734
Values	735
Product	736
Products	737
Contract	738
Contracts	739
Application Component	740

Application Components	741
Application Collaboration	742
Application Collaborations	743
Application Interface	744
Application Interfaces	745
Application Function	746
Application Functions	747
Application Interaction	748
Application Interactions	749
Application Service	750
Application Services	751
Data Object	752
Data Objects	753
Node	754
Nodes	755
Device	756
Devices	757
Network	758
Networks	759
Communication Path	760
Communication Paths	761
Infrastructure Interface	762
Infrastructure Interfaces	763
System Software	764
System Softwares	765
Infrastructure Function	766
Infrastructure Functions	767
Infrastructure Service	768
Infrastructure Services	769
Artifact	770
Artifacts	771
Stakeholder	772
Stakeholders	773
Driver	774
Drivers	775
Assessment	776
Assessments	777
Goal	778
Goals	779
Principle	780
Principles	781
Requirement	782
Requirements	783
Constraint	784
Constraints	785
Work Package	786

Work Packages	787
Deliverable	788
Deliverables	789
Plateau	790
Plateaus	791
Gap	792
Gaps	793
Group	794
Groups	797
Junction	795
Junctions	798
Generic Topic	799
Generic Topics	800
View	702
Views	796

15.5.3 Mavim Manager relationship types

	Relationship Type
With What characteristic	when
Role type	who
When characteristic	when
Where characteristic	where
Who characteristic	why
Where to	whereto
Custom icon relationship	icon
SAP – definition relationship	sapdef
M3 – definition relationship	m3def

15.5.4 Mavim Manager dispatch instruction types

None	none
Send (to…)	send
Save	save
Incoming data	input
Mechanism data	mechanism
Controls	controls

15.5.5 List of Icons

Icon name	Icon ID

Module	2
Topic & Standard Topic	3
Imported Version	5
Start Activity	6
Stop Activity	7
Decision	8
Matrix	9
Condition	13
External Files	14
Topic	15
Topic within Business Process	16
Diamond	17
Derivation	18
External References	19
Co-worker	20
Functionary	21
User Group	23
Detailed Process	24
Primary Topic Organisation (Organisation chart)	25
Primary Topic (Business Process)	26
Primary Topic (Systems)	27
Activity	28
Network	29
Building	30
Frequency	31
Time	32
Relationship Categories	33
With What	34
Who	35
Where	36
When	37
Why	38
Staff Department	39
Version	42
Whereto	44
Comment	47
Internal Files	48
Document	49
Filing Cabinet/Archive	50
Memo	51
Car	52
File	53
E-Mail	54
Letter	55
Red hand	56
Password	57

Green traffic light	58
Red traffic light	59
Yellow traffic light	60
Lamp	61
Light on	62
Manual	63
File	64
Display/Software Application	65
Spanner	66
Hammer	67
Comment	70
Characteristics	71
Process	72
Primary Topic General	78
City	79
Fax	80
Telephone	81
Meeting	82
Yellow hand	83
Green hand	84
Flowchart	114
Hierarchical Chart	115
IDEF0 Chart	116
Network Chart	117
Book	131
SAP Connection	132
SAP Connections	133
Matrices	134
Matrix Definitions	135
Matrix Definition	136
Matrix Characteristic Definition	137
Matrix Range Definition	138
Group Topic	139
Cross Function Flowchart	145
Icons	167
EPC Chart	170
Event	174
Start Event	175
Stop Event	176
Function	177
AND	178
OR	179
XOR	180
Field Settings	181
Field Sets	182
Field Set	183

Field Definitions	184
Field	185
Type Definition (Not Used)	187
View	189
Generic Chart	190
Pronto Chart	216
Connections	218
SharePoint Connection	220
BPMN	222
Sequence	224
Phase	225
Domains	226
Domain	227
Transactions	228
Transaction	229
Business Objects	230
Business Object	231
Report Definitions	232
Report Definition	233
Report View	235
Search Item	236
Templates	237
Template	238
Web Form Definitions	247
Web Form definition	248
MindManager Map	251
Gateway	252
Event	253
Task	254
Sub-Process	255
Collaboration Chart	256
Boundary Event	257
Choreography Activity	258
Choreography Sub-Process	259
Participants	260
Participant	261
Data Objects	262
Data Object	263
Data Stores	264
Data Store	265
Global Tasks	266
Global Task	267
Global Choreography Tasks	268
Global Choreography Task	269
Global Conversations	270
Global Conversation	271

Entity	274
Entity Instance	275
Orchestrations	278
Collaborations	279
Choreographies	280
Conversations	281
Orchestration Chart	282
Pool Process	283
Choreography Chart	284
Conversation Chart	285
Conversation	286
Catalogs	292
Model	293
BPMN 2.0	296
M3 Connection	298
M3 Connections	299
SharePoint Connections	300
ArchiMate	301
View	303
Business Layer Folder	304
Application Layer Folder	305
Technology Layer Folder	306
Motivation Extension Folder	307
Implementation Migration Extension Folder	308
Business Actor	309
Business Actors	310
Business Role	311
Business Roles	312
Business Collaboration	313
Business Collaborations	314
Business Interface	315
Business Interfaces	316
Location	317
Locations	318
Business Process	319
Business Processes	320
Business Function	321
Business Functions	322
Business Interaction	323
Business Interactions	324
Business Event	325
Business Events	326
Business Service	327
Business Services	328
Business Object	329
Business Objects	330

Business Representation	331
Business Representations	332
Meaning	333
Meanings	334
Value	335
Values	336
Product	337
Products	338
Contract	339
Contracts	340
Application Component	341
Application Components	342
Application Collaboration	343
Application Collaborations	344
Application Interface	345
Application Interfaces	346
Application Function	347
Application Functions	348
Application Interaction	349
Application Interactions	350
Application Service	351
Application Services	352
Data Object	353
Data Objects	354
Node	355
Nodes	356
Device	357
Devices	358
Network	359
Networks	360
Communication Path	361
Communication Paths	362
Infrastructure Interface	363
Infrastructure Interfaces	364
System Software	365
System Softwares	366
Infrastructure Function	367
Infrastructure Functions	368
Infrastructure Service	369
Infrastructure Services	370
Artifact	371
Artifacts	372
Stakeholder	373
Stakeholders	374
Driver	375
Drivers	376

Assessment	377
Assessments	378
Goal	379
Goals	380
Principle	381
Principles	382
Requirement	383
Requirements	384
Constraint	385
Constraints	386
Work Package	387
Work Packages	388
Deliverable	389
Deliverables	390
Plateau	391
Plateaus	392
Gap	393
Gaps	394
Group	395
Junction	396
Views	397
AX Connection	398
AX Connections	399
Generic Layer Folder	400
Groups	401
Junctions	402
Generic Topic	403
Generic Topics	404

15.5.6 Field types

Field type	Format
text	string
integer	whole number
float	decimal number
boolean	true/false
date	yyyy/mm/dd
list	(not used)
relation	xml-id to element
relationlist	xml-id to element
internalfile	xml-id to element

kpimin	whole number
kpimax	whole number
kpinorm	whole number
saptransactiontitle	string
saptransactionsystem	string
saptransactionid	string
saptransactionclient	string
saptransactionlanguage	string

15.5.7 Field types and relevant properties

Depending on the type of field that is selected in a field definition, certain properties are relevant. These are shown in the following table:

	read Only	visible	requi-red	multiple Lan-guages	multiple Values	relation Type	relation Charac	field Values
text	✓		✓	✓	✓			✓
integer	✓		✓		✓			✓
float	✓		✓		✓			✓
boolean	✓		✓					✓
date	✓		✓		✓			✓
list *								
relation	✓		✓		✓	✓	✓	
relationlist *								
kpimin	✓		✓		✓			✓
kpimax	✓		✓		✓			✓
kpinorm	✓		✓		✓			✓
saptransactiontitle	✓		✓		✓			✓
saptransactionsystem	✓		✓		✓			✓
saptransactionid	✓		✓		✓			✓
saptransactionclient	✓		✓		✓			✓
saptransactionlanguage	✓		✓		✓			✓

Note: the field types marked with an asterisk (*) are not in use.

15.6 Limitations

The most important limitation when using Mavim XML files is that not all topic types can be imported or exported with the current version. From the list of possible topic types (see Mavim Manager topic types), it would seem that versions cannot be exported to XML files. This is not entirely true as the export function is still available, however, the data cannot be imported into a Mavim database. The XML data can be used for other applications as a source for importing.

If an XML is compiled with a number of topics at the highest level, Mavim will place these topics under the designated target topic.

16 COMMAND LINE PARAMETERS MAVIM MANAGER

16.1 Introduction

General information about using the command line with Mavim Manager.

- Always use a space between options
- For the Mavim Manager EXE, using a folder name with one or more spaces requires, double quotes. For example "C:\Folder Name\".
  (**Note**: It must be possible to create this folder if it does not yet exist. i.e. access rights)
- For all parameters, make sure to use single quotes such as:
  /LOG:'C:\temp\batch.log'
- Always use a log file to discover any errors. For example:

  /LOG:'C:\temp\batch.log'

- Command line parameters or action values that are not recognized will make the command invalid or abort the action. An error will be written to the log file.
- The given database must be valid and the given user must have SUPER access rights in order to execute a batch configuration.

16.2 Creating the batch file

Generating reports using a batch file simply requires the correct command and parameters. Below is a list of the available parameters and a description of what they do:

Parameter	Description
/DB	Used to define the database to be used. This is basically the full file name of the database file (i.e. <database_name.MVD>)
/LOC	Used to define the location of the database file.
/USR	With this parameter you can configure a user account to log in to the database.
/PWS	This parameter is used to configure the appropriate password for the user account used.
/DATALANG	This parameter is used to set the data language for the reports that will be exported.
/DCV	This parameter is used to configure which reports will be executed in the batch. Multiple DCV codes can be added separated by a space. Not setting this parameter will cause all reports in the database to be exported. Entering an incorrect DCV code will cause the export to fail.
/PATH	This parameter defines the path where the exported Excel sheets will be placed.
/ACTION:PUBLISHREPORT	This is basically the command to invoke the creation and export of the configured reports.
/CREATEFOLDERS:TRUE/FALSE	If this parameter is TRUE Rules will create a new folder for the created reports in the folder specified in /PATH. If FALSE all reports will be placed in the folder specified in /PATH. The default is TRUE, even when not specified.

An example commandline could look like this:

"C:\Program Files\Mavim\Manager10\Mavim.Manager.Application.exe" /DB:<database name>.mvd /LOC:C:\Users\Public\Documents\Mavim\Manager10\Data /DBE:2 /USR:SUPER

```
/ACTION:PUBLISHREPORT /PATH:C:\TEMP /DCV:d12950883c1293v0 /DATALANG:nl
/CREATEFOLDERS:FALSE
```

In this case a DCV code has been provided. When a DCV code is not provided, all reports in the database will be exported.

16.3 Versions and Website Publications

For the configuring the batch operation for versions and website publications, please refer to the information found in the functional help. Once the configuration has been created, you can automate the creation of versions and website publications by creating a batch file and scheduling a Windows Scheduled Task to execute it.

1. Create a new batch file in a known location.

2. The contents of the batch file is to be filled with the necessary command. As an example:

   "C:\Program Files\Mavim\Manager10\Mavim.Manager.Application.exe" /DB:<database name>.mvd /LOC:D:\Databases /USR:SUPER /ACTION:BATCHPUBLISH

   The configuration of the batch operation is stored in the database and does not need to be supplied in this command.

3. Now create a new Windows Scheduled Task and select to execute the batch file you just created at the desired time and interval.

16.4 Starting Mavim Manager

The Mavim Manager startup parameters have the following format:

/{KEY}:{Value}

Using these parameters you can open a Mavim database directly without having to log in, e.g.:

/DB:<*database*> /LOC:O:\data /USR:SUPER

Or:

/DB:O:\data\test.mvd /USR:SUPER

To open the database (chart/user) called TEST automatically in the 'mavim' SQL server database on the 'mavimdevsql' server, the startup parameters would look something like this:

/DB:TEST /LOC:MAVIMDEVSQL:MAVIM /USR:SUPER

When working with path names, it is advisable to place the path string in quotation marks to ensure that the string can be read correctly. This is essential for generated strings.

"C:\Program Files\Mavim\Manager10\Mavim.Manager.Application.exe" /DB:<*database*> /USR:SUPER

There is also a /PWS parameter that can be used when a password must be supplied.

If you have entered the password incorrectly in the parameters, Mavim will simply display the login screen so that the user can log in correctly. Any other errors are shown on screen and the application will be closed.

It is also possible to make use of Windows Authentication. To do this, use the the following parameter:

/INTEGRATEDSECURITY

The database also needs to be configured correctly for this.

```
"C:\Program Files\Mavim\Manager10\Mavim.Manager.Application.exe" /DB:TEST
/LOC:MAVIMDEVSQL:MAVIM /INTEGRATEDSECURITY /DBE:3
```

The name of the database can also be read from the MVMDB.INI

/NAME:database

The name of the database is in the MVMDB.INI file. The application will retrieve the rest of the information from the file. It is still possible to provide a user name and password using:

/USR:SUPER /PWS:password

/SYSTEMLANG:EN

This parameter will start Mavim Manager in the specified Application Language (NL, EN, FR or DE)

/VIEW:BASIC or /VIEW:B

This parameter will start Mavim Manager with the specified View (B/Basic or A/Advanced)

16.5 Exporting a version using Batch Configuration

It is possible to start Mavim Manager and export the latest version of a version:

Parameter	Description
/NAME	Use the name of the database from the MVMDB.INI file with the corresponding settings.
/DB	Used to define the database that is to be used. This is the full file name of the database (<database_name.MVD>)
/LOC	Used to define the location of the database file.
/USR	Used to define the user to gain access to the database.
/PWS	Used to configure the correct password for the configured user.
/INTEGRATEDSECURITY	This parameter is used to define the use of Windows Authentication (Single Sign On).
/DATALANG	This is used to define which data language is used.
/PATH	Used to configure where the generated version will be stored.
/FILE	Used to configure where the version will be stored including the file the version must have.
/ACTION:EXPORTVERSION	This is the parameter that ensures that a version will be exported.
/VERSIONNR	This parameter is used to define for which version definition the export will be done.
/PWSDC	Optional. Configures the password for the exported version.
/LOG	Optional. Location and name of the log file.

Example of a command line that will export the latest version of version definition 3000 including a password on the exported version.

```
"C:\Program Files\Mavim\Manager10\Mavim.Manager.Application.exe" /NAME:Demo /USR:SUPER
/ACTION:EXPORTVERSION /VERSIONNR:3000 /PWSDC:mysecretpassword /PATH:c:\temp
/LOG:c:\temp\exportmtd.log
```

It is possible to always use the same file name when exporting. Make use of the following parameter:

/FILE:'c:\destination\exportfilename.mtd'

Example of a command line that exports the most recent version of version definition 3000 using the same file name.

```
"C:\Program Files\Mavim\Manager10\Mavim.Manager.Application.exe" /DB:<database name>.mvd
/LOC:C:\Users\Public\Documents\Mavim\Manager10\Data /USR:SUPER /ACTION:EXPORTVERSION
/VERSIONNR:3000 /FILE:c:\temp\exportedversion.mtd /LOG:c:\temp\exportmtd.log
```

16.6 Exporting a version

It is possible to start Mavim Manager and export the latest version of a version:

Parameter	Description
/NAME	Use the name of the database from the MVMDB.INI file with the corresponding settings.
/DB	Used to define the database that is to be used. This is the full file name of the database (<database_name.MVD>)
/LOC	Used to define the location of the database file.
/USR	Used to define the user to gain access to the database.
/PWS	Used to configure the correct password for the configured user.
/INTEGRATEDSECURITY	This parameter is used to define the use of Windows Authentication (Single Sign On).
/DATALANG	This is used to define which data language is used.
/PATH	Used to configure where the generated version will be stored.
/FILE	Used to configure where the version will be stored including the file the version must have.
/ACTION:EXPORTVERSION	This is the parameter that ensures that a version will be exported.
/VERSIONNR	This parameter is used to define for which version definition the export will be done.
/PWSDC	Optional. Configures the password for the exported version.
/LOG	Optional. Location and name of the log file.

Example of a command line that will export the latest version of version definition 3000 including a password on the exported version.

```
"C:\Program Files\Mavim\Manager10\Mavim.Manager.Application.exe" /NAME:Demo /USR:SUPER
/ACTION:EXPORTVERSION /VERSIONNR:3000 /PWSDC:mysecretpassword /PATH:c:\temp
/LOG:c:\temp\exportmtd.log
```

It is possible to always use the same file name when exporting. Make use of the following parameter:

/FILE:'c:\destination\exportfilename.mtd'

Example of a command line that exports the most recent version of version definition 3000 using the same file name.

```
"C:\Program Files\Mavim\Manager10\Mavim.Manager.Application.exe" /DB:<database name>.mvd
/LOC:C:\Users\Public\Documents\Mavim\Manager10\Data /USR:SUPER /ACTION:EXPORTVERSION
/VERSIONNR:3000 /FILE:'c:\temp\exportedversion.mtd' /LOG:'c:\temp\exportmtd.log'
```

16.7 Executing (a part of) the Batch Configuration

It is possible to start Mavim Manager and execute a (part of) the Batch Configuration.

Parameter	Description
/NAME	Use the name of the database from the MVMDB.INI file with the corresponding settings.
/DB	Used to define the database that is to be used. This is the full file name of the database (<database_name.MVD>)
/LOC	Used to define the location of the database file.
/USR	Used to define the user to gain access to the database.
/PWS	Used to configure the correct password for the configured user.
/INTEGRATEDSECURITY	This parameter is used to define the use of Windows Authentication (Single Sign On).
/DATALANG	This is used to define which data language is used.
/PATH	Used to configure where the generated version will be stored.
/FILE	Used to configure where the version will be stored including the file the version must have.
/ACTION:EXPORTVERSION	This is the parameter that ensures that a version will be exported.
/VERSIONNR	This parameter is used to define for which version definition the export will be done.
/PWSDC	Optional. Configures the password for the exported version.
/LOG	Optional. Location and name of the log file.

Example:

```
"C:\Program Files\Mavim\Manager10\Mavim.Manager.Application.exe" /DB:<database name>.mvd
/LOC:'C:\Users\Public\Documents\Mavim\Manager10\Data' /DBE:2 /USR:SUPER
/ACTION:BATCHPUBLISH  /LOG:'c:\temp\publishbatch.log'
```

To execute a single line from the Batch Configuration, use the /DCV parameter to select the version definition.

```
"C:\Program Files\Mavim\Manager10\Mavim.Manager.Application.exe" /NAME:Demo
/ACTION:BATCHPUBLISH /DCV:d12950883c1293v1003 /LOG:'c:\logfiles\publishbatch.log'
```

Example of exporting a part of the Batch Configuration including exporting a version with a fixed file name.

```
"C:\Program Files\Mavim\Manager10\Mavim.Manager.Application.exe" /NAME:Demo
/ACTION:BATCHPUBLISH /VERSIONNR:3000 /PASSWORDDC:mysecretpassword
/FILE:'c:\temp\export.mtd' /LOG:'C:\temp\publishbatch.log'
```

16.8 Importing (and updating) a version

It is possible to start Mavim Manager and import and/or update a version. All settings for checking / updating are available as command line parameters.

Parameter	Description
/NAME	Use the name of the database from the MVMDB.INI file with the corresponding settings.
/DB	Used to define the database that is to be used. This is the full file name of the database (<database_name.MVD>)
/LOC	Used to define the location of the database file.
/USR	Used to define the user to gain access to the database.
/PWS	Used to configure the correct password for the configured user.
/INTEGRATEDSECURITY	This parameter is used to define the use of Windows Authentication (Single Sign On).
/DATALANG	This is used to define which data language is used.
/PATH	Used to configure where the generated version will be stored.
/ACTION:IMPORTVERSION	This parameter ensures that the version is imported.
/CHECKUPDATE	This parameter ensures that the imported version is also updated using all default settings.
/DROPPEDTOPICS:	Optional setting for handling removed topics during an update: • DELETE • TORECYCLEBIN (default) • UNCOUPLE
/UNCOUPLEDTOPICS:	Optional setting for Topics uncoupled during update: • RESTORE (default) • LEAVE
/VISIOOPTIONSADDIMPORTEDRELATIONSHIPS:	Optional setting for updating: • True (default) • False
/VISIOOPTIONSREMOVEDROPPEDIMPORTEDRELATIONSHIPS:	Optional setting for updating: • True (default) • False
/VISIOOPTIONSKEEPLOCALLYADDEDRELATIONSHIPS:	Optional setting for updating: • True (default) • False
/VISIOOPTIONSADDLOCALLYREMOVEDRELATIONSHIPS:	Optional setting for updating: • True (default) • False
/LOG	Specifies the name and location of the log file.

Example for importing a version:

```
"C:\Program Files\Mavim\Manager10\Mavim.Manager.Application.exe" /DB:SQLC
/LOC:VM2016:MavimDB2 /USR:SUPER /ACTION:IMPORTVERSION /PATH:'C:\Temp\Controlling
5001.mtd' /LOG:'C:\Temp\publishbatch.log'
```

Example for importing a version and immediately updating with all update settings:

```
"C:\Program Files\Mavim\Manager10\Mavim.Manager.Application.exe" /DB:SQLC
/LOC:VM2016:MavimDB2 /ACTION:IMPORTVERSION
/PATH:'C:\Temp\Controlling 5001.mtd' /DROPPEDTOPICS:TORECYCLEBIN
/UNCOUPLEDTOPICS:RESTORE /VISIOOPTIONSADDIMPORTEDRELATIONSHIPS:TRUE
/VISIOOPTIONSREMOVEDROPPEDIMPORTEDRELATIONSHIPS:TRUE
/VISIOOPTIONSKEEPLOCALLYADDEDRELATIONSHIPS:TRUE
/VISIOOPTIONSADDLOCALLYREMOVEDRELATIONSHIPS:TRUE
/LOG:'C:\Temp\update.log'
```

Example for importing a version and updating with the default settings

```
"C:\Program Files\Mavim\Manager10\Mavim.Manager.Application.exe" /DB:SQLC
/LOC:VM2016:MavimDB2 /ACTION:IMPORTVERSION
/PATH:'C:\Temp\Controlling 5001.mtd' /CHECKUPDATE /LOG:'C:\Temp\update.log'
```

16.9 Creating a Database with the Commandline

Using the following commandline parameters, you can create a Mavim database:

/ACTION:CreateDB /LOC=<*SQL Server*>:<*instance name*> /USR=<*Valid Mavim User*> /DB=<*Database name in Mavim*> /LICENSE:<*database code*> /SQLPWS=<*SQL password*> /SQLUSR=<*SQL User*> /ENCRYPTCONNECTION /TRUSTSERVERCERTIFICATE

If a database already exists with this name, it will not be overwritten unless the /Overwrite parameter is used.

Below is a description of the parameters used in this example:

Parameter	Description
/ACTION:CREATEDB	This parameter is used to create the database.
/LOC	Used to define the location of the database. For this parameter you must include the server name and the SQL instance name. (<servername>:<instance name>)
/DB	Name of the database to be used.
/LICENSE	The database code to be used to create the database.
/NAME	The name of the schema in SQL.
/SQLPWS	The SQL password to be used to log in to the SQL server/instance.
/SQLUSR	The SQL User to be used to log in to the SQL Server/instance.
/USR	The Mavim user. Note that you MUST provide a valid Mavim User name in order to create an SQL database from the commandline.
/DBTEMPLATE	Use this parameter to create the database from a template and include the location of the .mvdx file.

/OVERWRITE	This parameter is mandatory if the database already exists and must be overwritten.
/ENCRYPTCONNECTION	Depending on your SQL server, this option could be necessary for a successful connection to the SQL database.
/TRUSTSERVERCERTIFICATE	Depending on your SQL server, this option could be necessary for a successful connection to the SQL database.

16.10 Restoring a Database with the Commandline

Using the following commandline parameters, you can restore a Mavim database from backup file.

/ACTION:RestoreDB /LOC=<*SQL Server*>:<*instance name*> /DB=<*Database name in Mavim*> /SQLPWS=<*SQL password*> /SQLUSR=<*SQL User*> /Logfile=<logfile location> /File:<location of backup file> /ENCRYPTCONNECTION /TRUSTSERVERCERTIFICATE

If a database already exists with this name, it will not be overwritten unless the /Overwrite parameter is used.

Below is a clear description of the parameters used in this example:

Parameter	Description
/ACTION:RestoreDB	This parameter is used to create the database.
/LOC	Used to define the location of the database. For this parameter you must include the server name and the SQL instance name. (<servername>:<instance name>)
/DB	Name of the database to be used.
/NAME	The name of the schema in SQL.
/SQLPWS	The SQL password to be used to log in to the SQL server/instance.
/SQLUSR	The SQL User to be used to log in to the SQL Server/instance.
/FILE	Location of backup file .mvdx
/OVERWRITE	This parameter is mandatory when the database already exists.
/PASSWORDBACKUPDB	This parameter can be used when the backup is protected with a password
/ENCRYPTCONNECTION	Depending on your SQL server, this option could be necessary for a successful connection to the SQL database.
/TRUSTSERVERCERTIFICATE	Depending on your SQL server, this option could be necessary for a successful connection to the SQL database.

16.11 Renumbering a Database

It is possible to change the database number for a database using the following commandline parameters:

/LOC=<*SQLSERVER*>:<*DATABASENAME*> /DB=<*SCHEMA*> /LICENSE=<*XXXXXXX*>
/ACTION=RENUMBERDB /LOGFILE='C:\TEMP\batcherrors.log

There are a number of criteria that must be met before this command can be executed successfully:

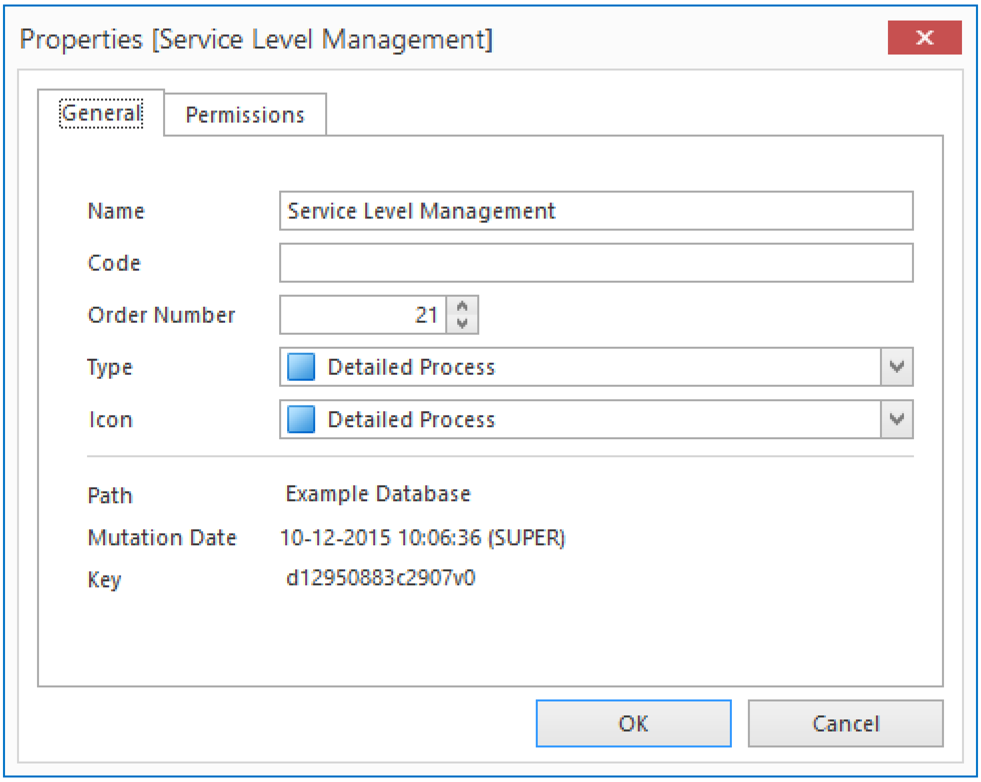
- A new valid license number must be included for the /LICENSE parameter.
- The database must be up to date. If it is currently a lower version than the executable used, the database will be updated automatically first.
- The correct SQL server and database name must be supplied

The RENUMBERDB parameter will update the database number.

17 CODE LINKING

The code field in the property window of a topic can be used to create direct links to topics in Mavim Rules websites. For example, if you want to create a direct link to the "Service Level Management" topic in a Mavim Rules website, you can follow these steps:

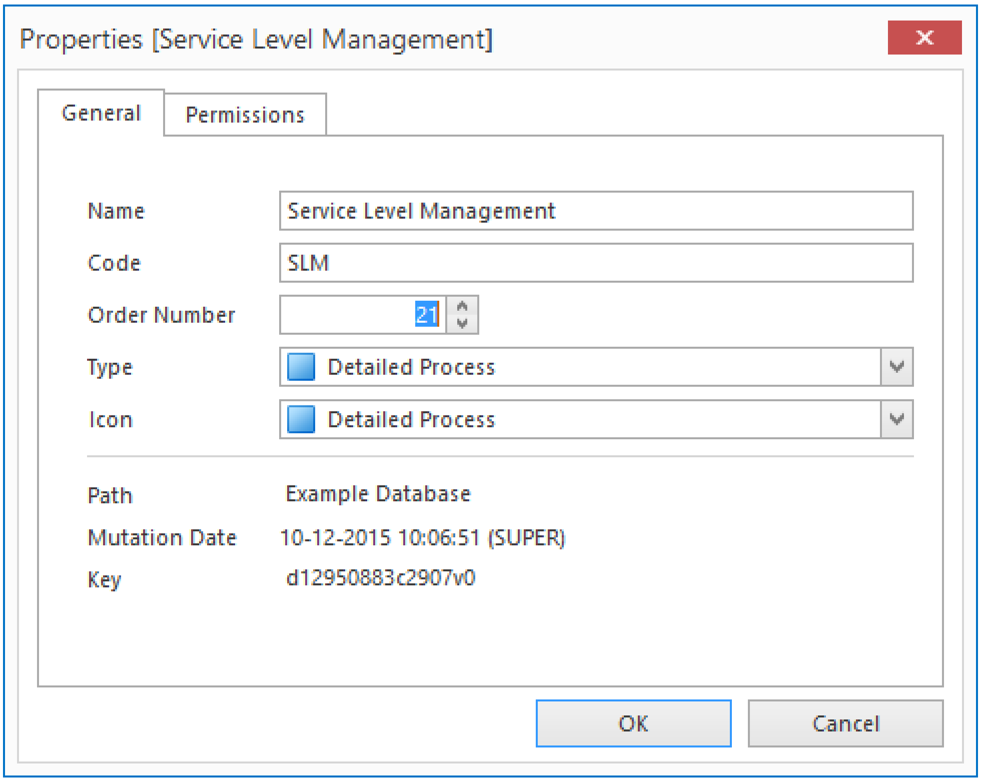
1. Select the topic in the topics screen in Mavim Manager.
2. Press F4 on your keyboard, or select right-click and select Properties.
3. The following dialog appears:



Properties [Service Level Management]

Name: Service Level Management
Code:
Order Number: 21
Type: Detailed Process
Icon: Detailed Process
Path: Example Database
Mutation Date: 10-12-2015 10:06:36 (SUPER)
Key: d12950883c2907v0

4. Enter a code in the Code field. It is important to use a code that is both recognisable and unique. If you plan to use code linking often, it is a good idea to create convention guidelines for the codes. In this example, the code "SLM" is used:



Properties [Service Level Management]

Name: Service Level Management

Code: SLM

Order Number: 21

Type: Detailed Process

Icon: Detailed Process

Path: Example Database

Mutation Date: 10-12-2015 10:06:51 (SUPER)

Key: d12950883c2907v0

5. Click **OK**.
6. Create a version and publish it on a Mavim Rules Web server. For more information about these procedures, see the Mavim Manager Help file in the **Version Management** and **Publication** topics (press F1 in Mavim Manager).
7. You can now find the direct URL of the topic "Service Level Management". You can either create a URL to the topic within a certain version, or to the newest version on the Web server based on that version definition:

   Within version:
   http://<Web server>/<publication folder>/<version name and number>/Theme/Html/Default.html?code=SLM&local=true

   Newest version:
   http://<Web server>/<publication folder>/<version name and number>/Theme/Html/Default.html?code=SLM&local=false

   The second option uses the Mavim Rules PHP scripts installed on the Web server.

> Code linking is case-sensitive. The words "code", "local", "true" and "false" in the URL must always be in lower case. The code itself must be the same as specified in the **Properties** dialog.