



Qlik Deployment Framework

Qlik Sense and Qlik Deployment Framework

June, 2016





Table of Contents

Version 1.6	3
Qlik Sense and Qlik Deployment Framework	3
Exercises	3
Manage and add containers	3
Qlik Sense folder connection	4
Single LIB mount	5
Separate LIB mounts	7
Initiating using InitLink	9
Initiating using 1.Init (legacy)	9
Qlik Sense Server	10
Optional settings	11
Additional Notes	11

Version 1.6

Qlik Sense and Qlik Deployment Framework

From QDF 1.4.1 QlikView and Qlik Sense are seamlessly integrates between each other. This means that container resources like sub functions and global variables are identical and available within both systems. When using Qlik Sense containers are created and maintained using [Qlik Deployment Framework Deploy Tool](#) that is available on Qlik Community. Read the *Qlik Deployment Framework-ReadMe.pdf* document that is attached with the Deploy tool for more info on how to use the deploy tool.

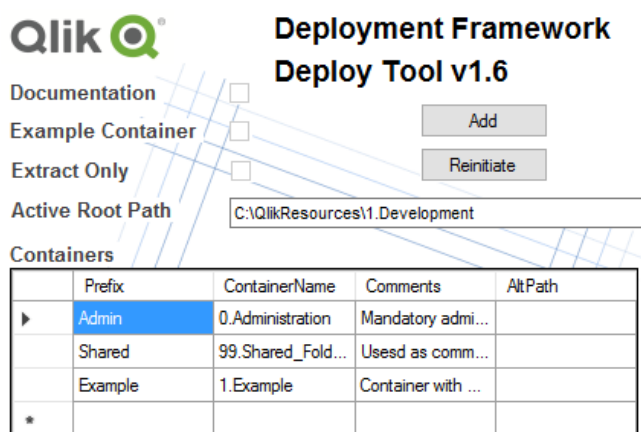
Exercises

There are several Qlik Sense QDF exercises available in the *Qlik Deployment Framework-Qlik Sense Exercises.pdf* document that will be installed using the *Deploy Tool*.

Manage and add containers

After the initial installation it's possible to enhance the framework (add containers) using the deploy tool. Active frameworks are identified automatically when adding a correct URL. Deploy tool will store last used URL to make it easy.

1. When active framework is found the Deploy path text will change to **Active Root Path**. Also *Documentation* and *Example Container* check boxed will be grayed out.



2. From the Containers grid new containers can be added

3. Container properties:
 - **Prefix** (Mandatory) this is the Container identifier, this is a **unique** short name for company department or project. This is used when mounting containers within the Qlik Scripts. Do not use long names or spaces

- **Container Name** (Mandatory) this is the physical container name shown on disk, could be company department or project. This could also include subfolders within the framework, examples:
 - **1.AcmeSales**
 - **100.DataSources\1.SQL-DB**

- **Comments** (optional) is used as descriptive meta-data.

- **AltPath** (optional) is used when creating a container in a different location. Example

C:\QlikTempContainers

4. To add the new containers press **Add**.

Qlik Sense folder connection

Qlik Sense has a new folder access method called *folder connection*. Adding a *folder connection* will create a **LIB** URL to the repository folder. The traditional QlikView way to accessing files (*c:\xx\yy* or *\\ServerName\yy*) is in Sense called *legacy mode* and disabled by default while accessing files using LIB's is called *native mode* and enabled by default. Qlik Deployment Framework works in native mode and utilizes the LIB url's.

Qlik Sense integration into QDF can be done in two different ways. First is to mount the whole container structure (*Root*) under on single LIB (*Singe LIB mounts*) and the second is to mount each container individually as its own LIB folder connection (*Separate LIB mounts*). Mode used is depending on the LIB setup and will be identified automatically when the framework initializes. The modes can be mixed from app to app in the same setup, for example depending on security level.

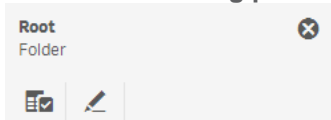
Example:

QlikView Global Variable	Qlik Sense Global Variable	
<code>vG.BasePath=\\Server\QDF\1.Example\</code>	<code>vG.BasePath=LIB://Example</code>	<i>Separate Mounts</i>
	<code>vG.BasePath=LIB://Root\1.Example</code>	<i>Single Mount</i>

Single LIB mount

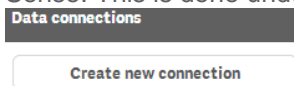
When using single mount, QDF locates containers in the same way as QlikView by use of the container map.

For single LIB mount add a Sense folder connection (LIB) named *Root* pointing to the framework starting point

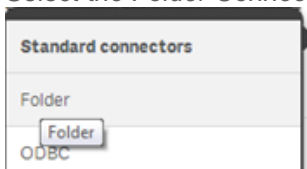


In this example a lib named Root pointing to QDF's root folder.

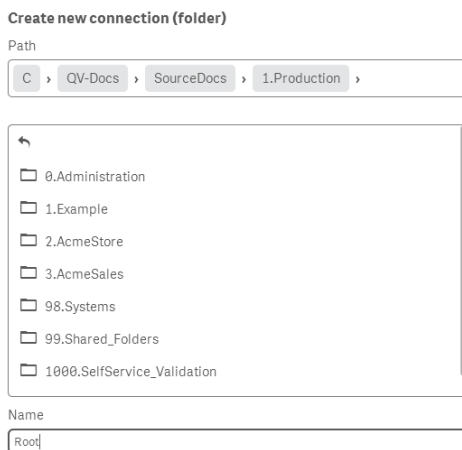
1. To create the **LIB://Root** described above we first need to create a new *Folder Connection* in Qlik Sense. This is done under *Data connections* selecting *Create new connections*.



2. Select the *Folder Connection* alternative.



3. Point to the root location of QDF container architecture and give it the name **Root**. Type in UNC or drive path here.



The *Folder Connection* name is case sensitive.

4. Last add the initiation script first in the load editor :
`$(Include=lib://Root\0.Administration\InitLink.qvs);`

5. Run the script, a similar text like below should be presented stating that single LIB mount have been identified.

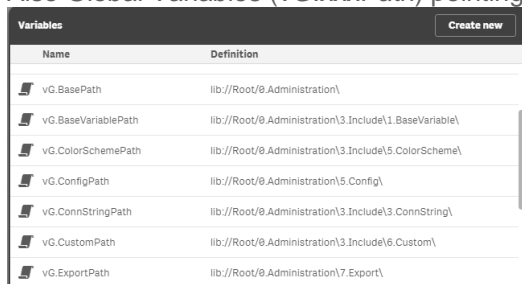
```
Started loading data
```

```
'### DF InitLink Started, trying to link to 1.Init.qvs  
script'
```

```
'### DF Info, identified Sense root path lib://Root/  
(single LIB mount)'
```

```
'### DF Info, identified Sense home container  
lib://Root/0.Administration\'
```

6. Also Global Variables (*vG.xxxPath*) pointing to *lib://root* should be available in the variable editor.



Name	Definition
vG.BasePath	lib://Root/0.Administration\
vG.BaseVariablePath	lib://Root/0.Administration\3.Include\1.BaseVariable\
vG.ColorSchemePath	lib://Root/0.Administration\3.Include\5.ColorScheme\
vG.ConfigPath	lib://Root/0.Administration\5.Config\
vG.ConnStringPath	lib://Root/0.Administration\3.Include\3.ConnString\
vG.CustomPath	lib://Root/0.Administration\3.Include\6.Custom\
vG.ExportPath	lib://Root/0.Administration\7.Export\

Optional LIB name

To change default **Root** folder connection (LIB) name add **SET** *vG.RootContainer='LIB name'* before QDF Initiation, the value should match the folder connection name pointing to **root**, see more under *Optional settings*.

Home container

Home container is the place where the generated global variables will be pointing to (ex. *vG.QVDPPath*) and the location where sub functions and global custom variables are read from. The default home container will be identified automatically (usually *0.Administration*) to change this behavior add **SET** *vG.HomeContainer='Physical container name'* in the script beginning, see more under *Optional settings*.

Notes

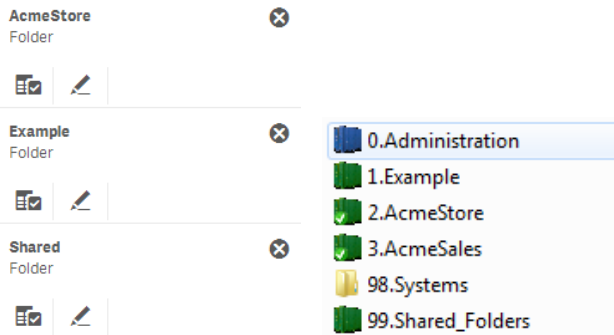
When using single mount and a container is not under the root (called *Alt Root Path* in QDF container Map) a Separate LIB mount need to be created. This mount should have the same name as the *Variable Prefix* in the Container Map.

Home container identification is only done automatically directly under the root LIB, to use specific container always use *vG.HomeContainer* variable before of 1.Init initiation.

Example **SET** *vG.HomeContainer= '98.Systems\1.Oracle'*

Separate LIB mounts

In this mode the container map in QDF is not read within Qlik Sense, instead each folder connection (LIB) is pointing to an accessible container. In other words add a folder connection for each container you want to use in your application, use same folder connections name (LIB) the *Variable Prefix* name in Container Map. A good thing of using separate mounts is that in Qlik Sense server version LIB's can be managed by the system administrator thereby restricting file access.

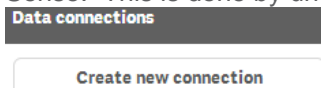


Example, add folder connection with the name *Shared* pointing to your shared container (99.Shared_Folders), this will generate global variables to the shared resources (`vG.SharedQVDPPath= LIB://Shared\2.QVDPPath`).

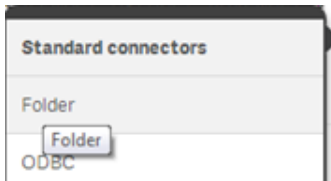
Home container

Home container is the place where the generated global path variables will be pointing to and the location where sub functions and global custom variables are read from (ex `vG.QVDPPath`). A Sense folder connection (LIB) need to be created. The lib name **Home** is default, but this is easy to change. Just add a `SET vG.HomeContainer='LIB-Name'` statement in the script beginning before framework initiation see Optional settings for more info.

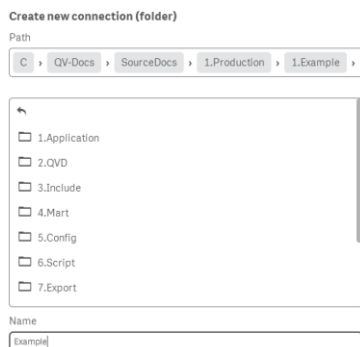
1. To create `LIB://Example` shown above we first need to create a new *Folder Connection* in Qlik Sense. This is done by under *Data connections* selecting *Create new connections*.



2. And select *Folder Connection*



3. Point to the location of 1.Example container path and give it the name Example (same name as used in the container map).



- When done you add initiation statement pointing to the Example in to the load editor, easiest is to use statement below.

```
SET vG.HomeContainer='lib://Example';
$(Include=$(vG.HomeContainer)\UnitLink.qvs);
```

- Run the script, a similar text like below should be presented stating that separate LIB mount have been identified.

Started loading data

```
'### DF InitLink Started, trying to link to 1.Init.qvs
script'
```

```
'### DF Info, identified Sense home container
lib://Example/ (Separate LIB mounts)'
```

```
'### DF 1.Init.qvs Started'
```

- Last validate that Global Variables are available (*vG.xxxPath*)

Variables Create new	
Name	Definition
vG.ApplicationPath	lib://ASales/1.Application\
vG.BasePath	lib://ASales/
vG.BaseVariablePath	lib://ASales/3.Include\1.BaseVariable\
vG.ColorSchemePath	lib://ASales/3.Include\5.ColorScheme\
vG.ConfigPath	lib://ASales/5.Config\
vG.ConnStringPath	lib://ASales/3.Include\3.ConnString\
vG.CustomPath	lib://ASales/3.Include\6.Custom\
vG.ExportPath	lib://ASales/7.Export\
vG.HomeContainer	lib://ASales
vG.ImportPath	lib://ASales/Import\
vG.IncludePath	lib://ASales/3.Include\
vG.LocalePath	lib://ASales/3.Include\2.Locale\

Note. When using Separate LIB mounts the global variable *vG.RootPath* (pointing at the framework root path) is removed as LIB's does not have any actual root path, different LIB's can point on different file systems.

★ vG.RootPath <NULL>

Initiating using InitLink

From QDF 1.5 Qlik Sense InitLink support has been added. This makes it easier to initialize the framework within Qlik Sense. Instead of a long url to *1.Init.qvs* the InitLink will “sling shot” to *1.Init*, example:

Example Single LIB Mount:

```
$(Include=lib://Root\0.Administration\InitLink.qvs);
```

Example Separate LIB Mounts:

```
$(Include=lib://Example\InitLink.qvs);
```

or

```
SET vG.HomeContainer='lib://Example';  
$(Include=$(vG.HomeContainer)\InitLink.qvs);
```

Initiating using 1.Init (legacy)

The initiation work in almost the same way as in QlikView, the same *1.Init.qvs* Include file (in the script beginning) initiates the framework. The only difference is that the url need a folder connection (LIB).

Example Single LIB Mount:

```
$(Include=lib://Root/1.Example/3.Include\1.BaseVariable\1.Init.qvs);
```

Example Separate LIB Mounts:

```
$(Include=lib://Example/3.Include\1.BaseVariable\1.Init.qvs);
```

When using the optional settings, like changing the default home container these settings should be places above the initiation statement, example:

```
SET vG.HomeContainer='Example'  
$(Include=lib://Example/3.Include\1.BaseVariable\1.Init.qvs);
```

Qlik Sense Server

When creating Lib's in Qlik Sense Server (during app development) the Lib name also contains concatenated *ComputerName* + *UID* this to make Lib's personal by default. When developing using QDF we do not want personal LIB's, instead the QDF folder LIB's should be common for all developers assigned by security rules. In the QMC *Data Connections* tab we can easily modify the name and apply one or more security rules depending on developer access rights.

Our Example Lib have (*ComputerName* + *UID*) added to the name

Example (sense1dot1_qlikservice) ✕

Folder



We need to change this in the QMC Data connections Tab

Data connections

Remove unwanted additional text and press apply

Data connection edit

Data connection edit

IDENTIFICATION

Name

Example (sense1dot1_qlikservice)

IDENTIFICATION

Name

Example

An *Associated Security rule* can also be added to the Data connection, so that only authorized developers have access to the container (In our case Example Container).

Create associated rule

Optional settings

Single LIB Mount

SET vG.HomeContainer='1.Example' Physical container name before 1.Init initiation script will use 1.Example container as Home container instead of the default 0.Administration container.

Remember, here you need to type the physical container folder name instead of LIB name.

SET vG.RootContainer='QDF' before 1.Init initiation script will use QDF LIB Folder as Root instead of the default Folder connection name (**Root**)

Separate LIB Mounts

SET vG.HomeContainer='Example' before 1.Init initiation script will use Example Folder connection as Home container instead of the default Folder connection name (**Home**).

SET vG.HomeContainer='lib://Example' before 1.Init initiation script will use Example Folder connection as Home container instead of the default Folder connection name (**Home**). This will also work with variables in the Include script as shown below:

```
SET vG.HomeContainer='lib://Example';  
$(Include=$(vG.HomeContainer)\UnitLink.qvs);
```

Additional Notes

This is an early release of QDF for Sense integrating, there are several tasks that cannot be done inside Sense, like:

- Management tool for global variables is still *VariableEditor* (QlikView), management of the variable files can be done using any editor (example notepad++) instead.
- Functions that create or delete files/folders from disk will by default not work in Qlik Sense as the *Execution* function is disabled, enable by using Legacy mode in the Engine. For example, the *CreateFolder* function uses execution to create folders.