

These exercises are designed for QlikView

## Extract Qlik Deployment framework Exercise

1. Unzip QlikDeploymentFramework\_Deploy\_Tool.zip
2. Run QlikDeploymentFramework.exe file



3. Modify Extract location if needed
4. Press Deploy
5. Double click on the extract box when done to see the files.
6. Close Tool with X.

## Exercises on 1.Init and load Global Variables

### Exercise number 1, basic 1.Init load

1. Open example application:  
*1.Example\1.Application\2.LoadIncludeExample\LoadIncludeExample.qvw*
2. Open the Editor, remark or remove everything.
3. Check in the Relative Path box
4. On the sheet top use menu, insert Include statement
5. Go to path *3.Include\1.BaseVariable\1.Init.qvs*
6. Include statement should look like this (copy/paste):  

```
$(Include=..\..\..\3.include\1.basevariable\1.init.qvs);  
$(Include=..\..\..\3.include\1.basevariable\1.init.qvs);  
$(Include=..\..\3.include\1.basevariable\1.init.qvs);  
$(Include=..\3.include\1.basevariable\1.init.qvs);
```
7. Run app and check *Variable Overview*, are the Global Variables there?
8. Using InitLink (a hidden link file in the container base) the initiation string is shorter:  

```
$(Include=..\..\..\InitLink.qvs);  
$(Include=..\..\InitLink.qvs);  
$(Include=..\InitLink.qvs);  
$(Include=.\InitLink.qvs);
```
9. Try this as well

### Exercise number2, try Calendar example

Try a working application in this exercise, using connection string and database inside 0.Administration container.  
No script modifications needed, the application use Environmental Global Variables populated by 1.Init.qvs

1. Open *1.Example\1.Application\6.Calendar-Example*
2. Check out the script 1.Init in the beginning of *Deployment Framework* tab
3. Run application, data and calendar will be populated
4. The calendar function itself is used later in the Calendar Generation Tab, Example:  
`CALL CalendarGen('DeliveryDate','DeliveryDate');`

## BONUS! Exercise number 3, advanced 1.Init load

1. Use *1.Example\1.Application\2.LoadIncludeExample\LoadIncludeExample.qvw*
2. Remove existing variables in Variable Overview
3. Open the Editor, remark or remove everything.
4. On the sheet top use menu, insert section below.
5. Run the script, check if Global Variables are in the QlikView application

```
// Advanced search for 1.Init.qvs only works with 0.95 or later
// Script to automatically identify and start Deployment Framework
// Based on InitLink.qvs stored in vG.BasePath
// Identifying Container based on InitLink.qvs
let vG.BasePath=;
SET vL.Path_tmp = ;
    for vL.x =1 to 10-1
        LET vL.Path='..\ '&'$(vL.Path_tmp)';
        LET vL.Path_tmp='$(vL.Path)';
        $(Include=$(vL.Path)InitLink.qvs);
        exit for when not '$(vG.BasePath)'= ''
    next
SET vL.Path = ;
SET vL.Path_tmp = ;
$(Include=$(vG.SubPath)\4.GenericContainerLoader.qvs);
```

This script will automatically try to find 1.init.qvs by searching for *InitLink.qvs* it's a hidden file stored in every container root.

1. Add this section below that will check if *vG.BasePath* contains a value this means that QDF is loaded. Exit script: `exit script when '$(vG.BasePath)'= '' ;`

## Variable Editor Exercises

You are a QlikView Administrator your assignment is to create a container structure that I aligned with your organization and source systems. There are two data sources that should have its own container (SAP,SQL) and there are two departments using QlikView (*AcmeStore* and *AcmeCorp*).

### Exercise number 1, Basic Create containers

1. Open Variable Editor by using *VariableEditor Shortcut* or open *0.Administration\6.Script\2.VariableEditor\VariableEditor.qvw*
2. Go to the Container Map section
3. Create a test container named *1.AcmeStore* with the prefix *Store*
4. Create a second test container called *2.AcmeCorp* with the prefix *Corp*
5. Create a system container called *1000.System\1.SAP* with the prefix *SAP*
6. Create a system container called *1000.System\2.SQL* with the prefix *SQL*
7. Press Update Container Map
8. Press Create New Containers
9. Check your container structure

### Exercise number 2, checkout Container map

1. Open the file *0.Administration\3.Include\1.BaseVariable\ContainerMap.csv*
2. This is the master container Map, all other maps in standard containers are copies of this one

### Exercise number 3, Link containers

As an administrator you want to collect data from the SAP container in to your Admin container.

1. Open the example application again:  
*1.Example\1.Application\2.LoadIncludeExample\LoadIncludeExample.qvw*
2. Open check *Variable Overview* and delete all Variables
3. In the script add ***call LCGV('SAP');*** after the 1.init section
4. Run the script
5. Check if Global Variables for SAP are loaded in, example *vG.SAPBasePath*
6. Also check that the path is correct
7. Delete all variables again
8. Now load only **one** Global Variable linked to *2.QVD* folder in the *SAP* container by using:  
***call LCGV('SAP','QVD');***
9. Try ***call LCGV('SAP','QVD;ConnString');***  
to load *QVD* and *ConnString* Global Variables links to *SAP* container

## Exercise number 4, Create Custom Global Variables

Custom Global Variables are used to get same variables/expressions across multiple applications.

Global Variables are created in Variable Editor and by default stored in *CustomVariable.csv*.


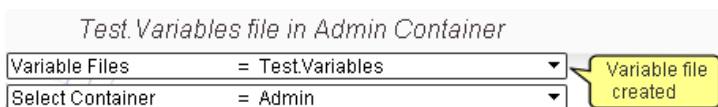

Custom Global Variables also uses the prefix (*vG.*) this to separate global variables against application specific variables.

1. Open Variable Editor by using *VariableEditor Shortcut* or open  
*1.Example\6.Script\2.VariableEditor\VariableEditor.qvw* now you are running Variable Editor outside  
0.Administration Container this user/developer cannot create containers.
2. Go to the Variable Editor section
3. Enter the Variable Name *LET vG.TestVariable1*
4. Enter Variable Value *10+10* (no space)
5. Enter second Variable Name *LET vG.TestVariable2*
6. Enter Variable Value *vG.TestVariable1+10* (no space)
7. Click Update Variables
8. Open and load *LoadIncludeExample.qvw*,  
*vG.TestVariable1* and *vG.TestVariable2* should be there, with the values 20 and 30
9. Add Variable Name *SET vG.TestVariable3* and Variable Value *10+10*
10. Open and load *LoadIncludeExample.qvw*, *vG.TestVariable3* should now have value 10+10
11. Try a more advance SET Expression:  
*SET vG.TestVariable4*  
*=TextBetween(Replace('\$(vG.BasePath)', '\$(vG.RootPath)', ','), ',' & '\_&Left(DocumentName(), len(DocumentName())-4)*
10. Open the example application again:  
*1.Example\1.Application\2.LoadIncludeExample\LoadIncludeExample.qvw*
11. Add a Text Object and put in the value = *\$(vL.TestVariable4)*
12. In the text box you should see the value *Example\_LoadIncludeExample*  
We have now created a concatenated container and application name global expression.

## Exercise number 5, checkout Custom Variables file (Custom Global Variables)

1. Open the file *1.Example\3.Include\1.BaseVariable\CustomVariables.csv*
2. This is the initial Custom Global Variables file it is easy to create additional variable files addressing different purposes.
3. Custom Global variables are created by 1.Init by using the function *LoadVariableCSV*  
call *LoadVariableCSV('\$(vG.BaseVariablePath)\CustomVariables.csv');*

## BONUS! Exercise number 6, Create a new Global Variables file

4. Create a new Variables file by typing the name *Test* in *add variable file box* and press enter  

5. After press the *Create Variable File* button a new Variable File named *Test.Variable.csv* have been created under *0.Administration* container.  

6. This new file can be modified by Variable Editor using *Variable Files* dropdown list and select *TestVariables*
7. To use this variable file in your scripts run the sub function call.  
call *LoadVariableCSV('\$(vG.SharedBaseVariablePath)\Test.Variables.csv');* // Execute the function using
8. To delete the file type *Del Test* in the *Add Variable File* box press enter.  


9. Press *Delete Variable File* button



## **BONUS! Exercise number 7, Copy Variables files and entries**

There are several ways of moving variables between containers the most basic is to copy / paste.

Also the possibility to use *Send to Excel* functionality and after paste the sheet into the other container.

Or just copy the *3.Include\1.BaseVariable\Custom.Variables.csv* across the containers.

## **BONUS! Exercise number 8, Universal Variables**

*Universal Variables (vU.)* is used to differentiate custom variables used by all applications within an environment (Shared Folders) from custom variables used within each container.

1. Change Select Container to *Shared*

A screenshot of a dropdown menu. The text inside the dropdown is 'Select Container = Shared'. A small downward-pointing arrow is visible on the right side of the dropdown box.

2. Create Variables in Shared container use *vU.* Prefix instead of *vG.* Example *vU.TestVariable*
3. Press button *Update Variables*
4. Open and reload *LoadIncludeExample.qvw* and check that the new shared *vU.* Variables exists.

## **System Variables**

System Variables is a function that uses Global Variables to store system settings. These are variables like log folders and others needed to monitor the platform, for example Governance Dashboard and Variable Editor can use these.

The System Variables are only loaded manually when needed by using the *3.SystemVariables.qvs* include:

*\$(Include=\$(vG.BaseVariablePath)\3.SystemVariables.qvs);*

1. Change Select Container to *Home*
2. Change Variable Files to *System.Variables*
3. The predefined System Variables appear, by default the setup will work on Windows 7/2008/2012 and single server installation
4. Check out the Variable Name *vL.DefaultServer* with value *ComputerName()*, this means that *vL.DefaultServer* (and *vG.QVSClusterNode1*) will have the same name as the computer running the script usually the Publisher server.
5. Open the example application again:  
*1.Example\1.Application\2.LoadIncludeExample\LoadIncludeExample.qvw*
6. Load System Variables by using *\$(Include=\$(vG.BaseVariablePath)\3.SystemVariables.qvs);*
7. Check that Variable *vG.QVSClusterNode1* has the same name as your computer.  
Once again we can observe LET variable executing inside the QDF script logic.

## Exercises on Load qvd files

### Exercise number 1, basic load

1. Open the Load QVD example:  
1.Example\1.Application\1.QVD-Generator-example\QVD-Generator-Example.qvw
2. Read through the script code to understand how the framework is integrated with the application.
3. Load the application
4. There should be qvd's in the 1.Example\2.QVD folder by using the global Variable *vG.QVDPath*

### Exercise number 2, Store QVD into SQL container

1. Add new Container Link function in Deployment Framework Tab:
2. `call LCGV('SQL','QVD');`
3. In Setup tab change `LET vL.QVDPath = '$(vG.QVDPath)';` to  
`LET vL.QVDPath = '$(vG.SQLQVDPath)';` That is the new Global variable created by the function
4. Reload the application
5. Check in *1000.System\2.SQL\2.QVD* that the QVD files exists

### Exercise number 3, move QVD loader

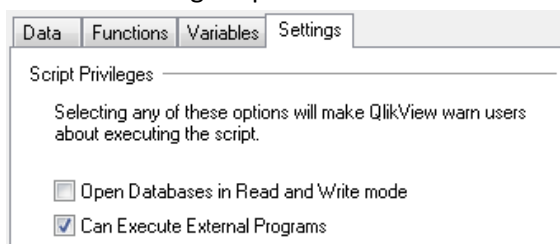
In this example the QVD loader will be moved to our SAP System folder *1000.System\1.SAP* and will distribute QVD's to our *1.AcmeStore* container.

1. Copy QVD-Generator-Example.qvw to *1000.System\1.SAP\1.Application* folder
2. We will now use the NorthWind Access database located in the Example container from the SAP container.  
Check out the connection include in the script  
`$(Include=$(vG.ExampleConnStringPath)\0.Example_Access_Northwind-ExampleConnString.qvs);`
3. Add connection to Acme Store container, in QDF tab (after 1.Init):  
`call LCGV('Store','QVD');`
4. In SETUP tab change the QVD path so that it points to Store by using *vG.StoreQVDPath*  
`LET vL.QVDPath='$(vG.StoreQVDPath)';`
5. Load the application, the qvd's should now be available for use by the Acme Store developers.

### Exercise number 4, create QVD folder by using CreateFolder function

There is a function to create folder and folder structures called *CreateFolder('folder path and name')* script editor in *QVD-Generator-Example.qvw*.

1. To use *CreateFolder* function it must first be loaded in by adding include sub in Deployment Framework tab:
2. After add function `call CreateFolder('$(vG.StoreQVDPath)1.NorthWind');`
3. In SETUP tab change the QVD path to  
`LET vL.QVDPath='$(vG.StoreQVDPath)1.NorthWind\';`
4. Before executing script check the box *Can Execute External Programs.*, else warning messages will pop up.



5. Execute the script and check that the new folder is created under store qvd.

## BONUS! Exercise number 5, create Universal Variables (as QVD path)

1. Open Variable Editor
2. In the Variable Editor tab change Variable Files to Custom Variable (default) and change Select Container to Shared Container *Shared*  
*Custom Variables file in Shared Container*  

Variable Files	= Custom.Variables	▼
Select Container	= Shared	▼
3. Add a new Universal Variable (vU.) in the Shared container  
`SET vU.NorthWindQVDPath = $(vG.StoreQVDPath)1.NorthWind\`
4. Write a comment explaining the variable use
5. Click Update Variables
6. Go back to `1000.System\1.SAP\1.Application\QVD-Generator-Example.qvw`
7. Open script and in SETUP tab change:  
`LET vL.QVDPath='$(vG.StoreQVDPath)1.NorthWind';` to  
`LET vL.QVDPath='$(vU.NorthWindQVDPath)';`
8. Reload `QVD-Generator-Example.qvw` There should now be refreshed QVD files in  
`1.AcmeStore\2.QVD\1.NorthWind\`

## Exercises on Binary load

### Exercise number 1, Binary load from a QlikView Mart

When binary loading the QDF logic is more difficult to use, binary load must be the first statement of a script. Best practice is thereby to use relative search path to the qvw mart in the binary section, instead of the framework global variables. Example:

`Binary [..\..\4.mart\0.example_northwind_mart\example_northwind_mart.qvw];`

1. Open `1.Example\4.Mart\0.Example_Northwind_Mart\Example_Northwind_Mart.qvw`
2. In QVD Mart Loader Tab change '`$(vG.QVDPath)`' to path where your QVD's are stored  
'`$(vG.StoreQVDPath)1.NorthWind`' or '`$(vU.NorthWindQVDPath)`'
1. Remember to use `call LCGV('xxxx');` in QDF tab if the QVD's are stored in another container.
2. Load application
3. We will test to load our QlikView mart from North Wind Dashboard example, Open  
`1.Example\1.Application\4.NorthWindExampleMartApp\NorthWindExampleMart.qvw`
4. Open the script and check out the Binary load
5. Add binary script  
`Binary [..\..\..\4.Mart\0.example_northwind_mart\example_northwind_mart.qvw];`
6. Binary load must be the first statement of a script. Best practice is thereby to use relative search path to the qvw mart in the binary section, instead of the framework global variables. The Deployment framework 1.Init include sections will follow the Binary load section.

## BONUS! Exercise number 2, reuse the Universal Variable

The point in creating a Universal Variables is to reuse these across containers.

This exercise will show this by loading a QlikView Mart based on the Universal Variable created earlier.

1. Open *1.Example\4.Mart\0.Example\_Northwind\_Mart\Example\_Northwind\_Mart.qvw*
2. Open the script and add *call LCGV('Store');* after *1.init.qvs*, we need this when connecting to the Store container.
3. Open the script and change to the QVD Mart Loader tab and replace *call QVDLoad ('\$(vG.QVDPath)');* with *call QVDLoad ('\$(vU.NorthWindQVDPath)');* (That we created earlier)
4. Run the load script
5. Check if the data model is there (ctrl/t), if not run the load script again and check if the model is there.
6. Save the application

The reason why we need to run the script two times is because the Universal Variable is executed before the *LCGV* function. This is not a big problem, only time the application needs to be reloaded twice is when moving between environments. A work around is to reload the Universal Variables again after *LCGV* function is run.

## Exercises on Settings and Templates page in Variable Editor

### Exercise number 1, Use of Templates to reset Variables

There are several templates that can be utilized as examples and as quick start.

1. First select Container *Home*
2. Create a Variable backup by pressing the same button again, press:  
*Reset values and Create a Backup*
3. After press *Settings and Templates* button
4. Select Apply Variable Template, chose Empty Template
5. Press the button *Apply updates to Custom.Variables*
6. This will retrieve a Template that is empty and go back to Variable Editor
7. The Variables are not removed until you have pushed the Update Variable Button
8. Push the Update Variable Button
9. If you change your mind use the *Retrieve Backup* button on the lower left