



Operations Guide for QlikView

Operations Guide

March, 2015





Table of Contents

Operations Guide V 1.5.1	3
QlikView Architecture overview	4
QlikView Platform Components	5
Deployment Framework Resource Containers	6
Environmental Global Variables	6
Global Variable links between Containers	6
Container Map	7
Container architecture	7
Moving application between containers	11
Deployment process and DTAP Environments	11
QlikView Management Console overview	12
Access to QlikView Management Console	12
Start using QlikView Management Console	13
QlikView Management Console Administration	13
QMC Container setup in QlikView Publisher Enterprise edition	18
Security	20
Container security	20
Container Sharing	22
Application Security Table (Section Access)	22
Section Access in Combination with Publisher	23
Database Connection String	23
QlikView Server without Publisher license	24
User Access Control (UAC)	25
QDF Tools	26
1.Create-prj (creates project folders)	26
Variable Editor	26
Container Map Editor	26
Variable Editor Tab	29
Variable Files, System Variables	30
Settings and templates	31
Upgrade Containers	33
Batch Mode	34

Governance Dashboard	36
QlikView System Monitor	36
Qlik Index Monitor	37
QlikView log files and locations	38
<hr/>	
Application logging	38
QlikView Server Logs	38

Operations Guide V 1.5.1

The deployment framework consists of several correlating documents. This document explains how to develop using Qlik Deployment Framework (QDF). Deployment Framework is initially created for QlikView but has been enhanced for Qlik Sense.

The Deployment Framework documents are:

- **Getting Started Guide** Get an overall understanding of the framework basics and how to start installing and develop.
- **Operations Guide** for Administrators maintaining the platform, security, tasks and containers.
- **Development Guide** for Developers how to work with DF in an efficient way, naming conventions, data modeling, optimization tricks/tools and other guide lines regarding development.
- **Deployment Guide** Project management guide on how to govern and manage Qlik deployment (DTAP) process, how to create Qlik projects and development teams and skill sets

QlikView Architecture overview

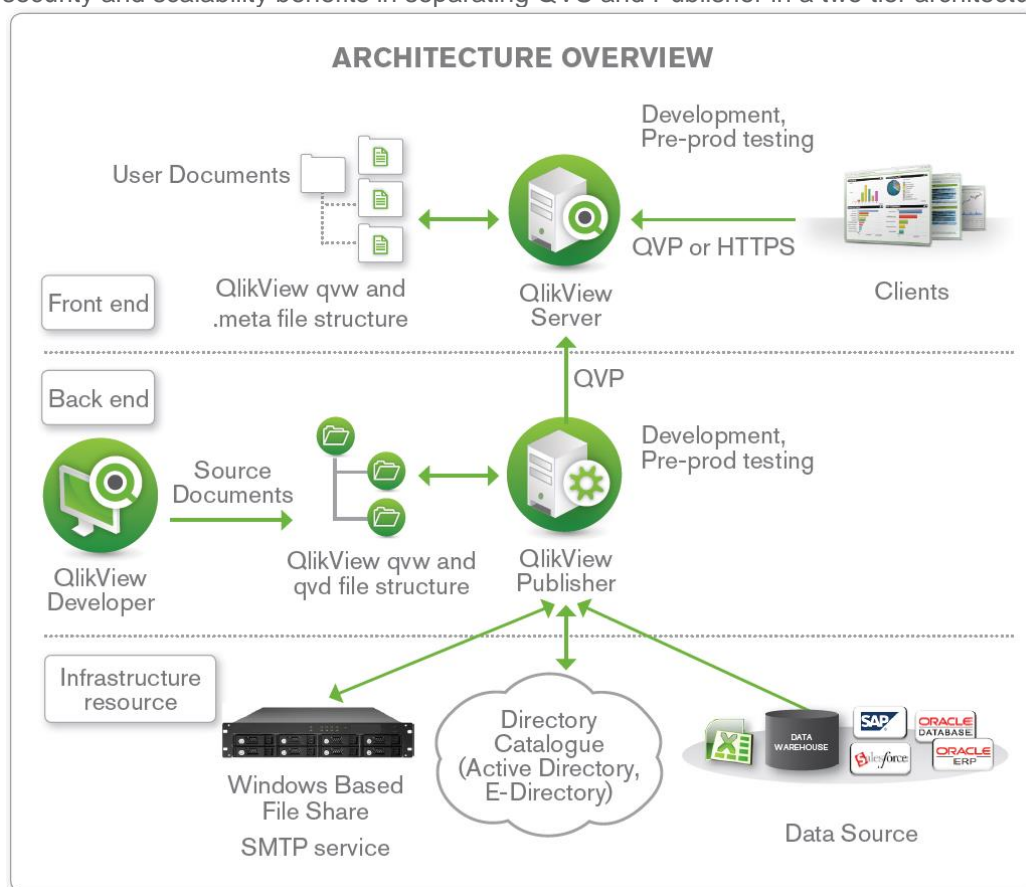
To administer the QlikView it's good to have an overall architecture understanding regarding the platform.

The QlikView Platform consists of several components. The main components are QlikView Server, QlikView Publisher and QlikView Developer.

- QlikView Server (QVS) handles the communication between clients and the QlikView documents. It loads QlikView documents into memory and calculates and presents user selections in real time. QlikView Server is a Front End system residing close to the business users.
- QlikView Publisher load data from different sources (oledb/odbc, xml, xls etc.), reduces and secures the QlikView source documents and distributes child documents to a QlikView Server. QlikView Publisher is a Back End system residing close to the data source.
- QlikView Developer is the development tool that creates the QlikView document (qvw) design and script logic that is run by the QlikView Publisher.
- **Qlik Deployment Framework** core is resided in the Back End inside the Source Documents folder.

QlikView Server and Publisher have different functionalities and handle CPU and memory very differently.

It is therefore best practice to separate these two functions on different servers. There are also security and scalability benefits in separating QVS and Publisher in a two tier architecture.

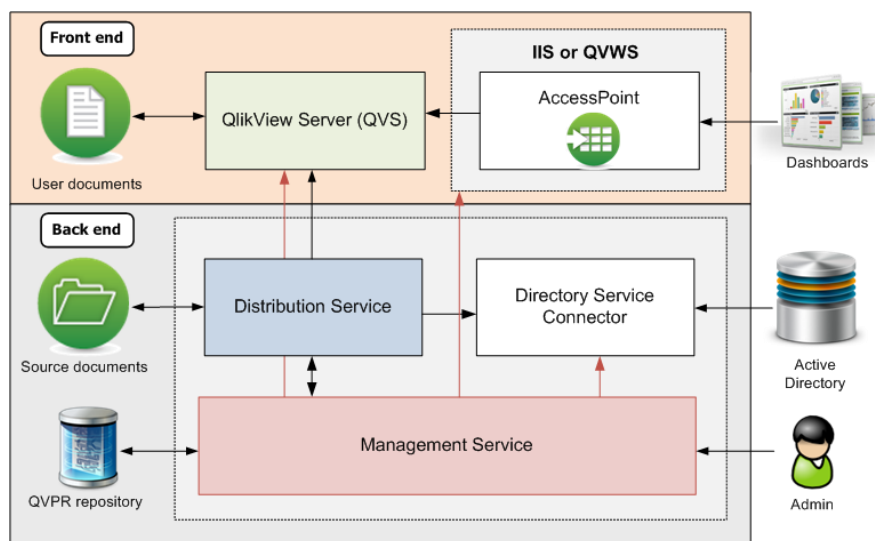


Example of a standard QlikView platform architecture design

Qlik Deployment Framework container structure and scripts are stored under *Source Document* in the Back End, using QlikView Publisher is recommended when using QDF this to separate Back End (Source Documents) against Front End (User Documents).

QlikView Platform Components

QlikView Platform is a Service Oriented Architecture (SOA) this means that components can be reorganizes depending on the customer needs.



Logical view of the QlikView components in a Front End/Back End configuration

Front End components

QlikView Server Service (QVS) is QlikView's associative database engine and main component in front end.

AccessPoint web portal (AP) is the presentation tier and AJAX engine. AccessPoint use our internal service QVWS or IIS. AP can be moved to a separate tier in front of QVS thereby creating a three tier architecture.

Back End components

QlikView Distribution Service (QDS) is the load and distribution engine.

Directory Service Connector (DSC) communicates with Directory Services usually Active Directory.

QlikView Management Console (QMS) is the administrative interface and could easily be moved out to a separate server. QMS is using the internal QVPR database in XML or MS SQL.

Deployment Framework Resource Containers

Qlik Deployment Framework is based on a **Resource Container Architecture**. Containers are identical but isolated folder structures placed side by side. A container can be moved and/or renamed without changing any QlikView script or logic inside it. Each container has identical file structures and includes the same base script functionality.

After Framework basic installation (using the Deploy Tool) pre-installed containers are:

- **Administration** container named *0.Administration*. This container contains admin stuff like Variable Editor, System Monitor. This container is used for administration of all the other containers.
- **Shared folder** container named *99.Shared_folder*. This container is used for company shared applications, kpi's and QlikView data files (QVD) like Active Directory data.
- **Example** container (Optional) (*1.Example*). Containing example applications to understand how QDF development works. When getting started browse through the example container, look at example scripts and read the information inside the folders.

Environmental Global Variables

QlikView applications stored inside the containers need to execute an initiation script during reload. The initiation will dynamically create variable called **Global Variables** they are associated to the container you are in.

When moving applications or container the initiation script will identify the new location and regenerate the Global Variables based on this new location so that applications works independent on location. This logic works with both UNC path and mapped drives.

Global Variable links between Containers

By using the function **LoadContainerGlobalVariables** it's possible to create Global Variable links (mounts) between containers, depending on security access. The short name **LCGV** can also be used as sub function call, for example: **LCGV ('SQL', 'QVD');**

Example: **call LoadContainerGlobalVariables ('AcmeTravel');** Will create all Global Variables linking to 2.AcmeTravel container. Variables created will have similar name as home container but with the additional *AcmeTravel* prefix, like *vG.AcmeTravelQVDPath* for QVD path to AcmeTravel container.

call LoadContainerGlobalVariables ('Oracle', 'QVD;Include'); Will create two Global Variable links to different resources in Oracle container, by using an additional switch and ';' separator creates Global Variables *vG.OracleQVDPath* and *vG.OracleIncludePath* (instead of linking all folders as in the first example).

More examples can be found in **Qlik Deployment Framework-Development Guide**.

Container Map

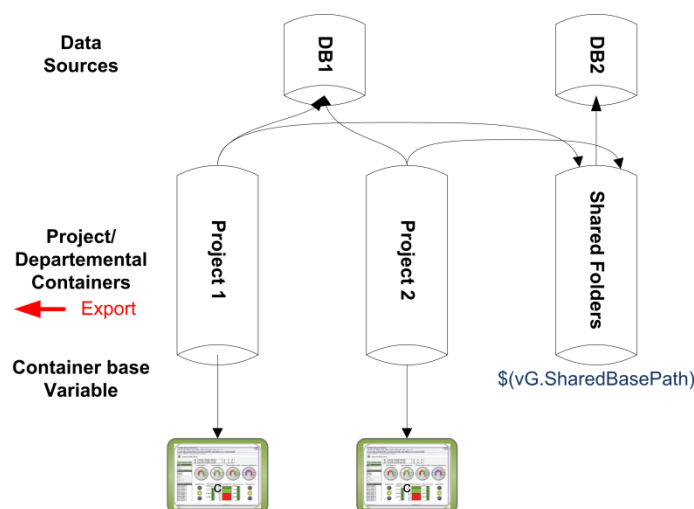
To locate and create links between containers a container map table is used. The table is stored in a meta-data file (*ContainerMap.csv*) the file is kept under *vG.BaseVariablePath* in all containers. But there is only one master Container Map this one is stored under *0.Administration* container, all other local container maps are read only or reduced versions of the master. When new containers are created all containers retrieves a read only map based of the master Container Map.

Container Map administration and container creation is done with a Variable Editor tool in 0.Administration Container.

Container architecture

In Variable Editor *Settings and Templates* menu there are different Container templates that can be used for demonstration.

There are several ways to organize containers and how the containers communicate between themselves. In the standard setup the containers are organized in one level like the example below. Most QlikView installations only need this basic setup, but in some circumstances more complex container architecture is needed. Down below are some different setups in combination with the Container Map schema.



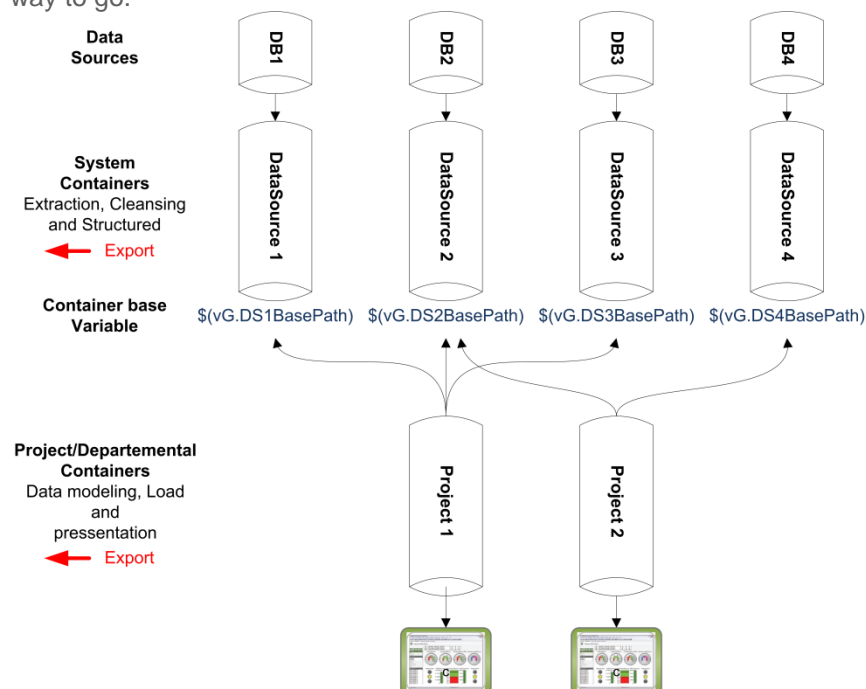
Basic Container setup, each container will load and store data from sources directly, common data is stored in the shared folders container and loaded into the project containers as shown in the picture above. Shared folders have its unique global path variable path that is common by the other containers.

Folder Name	Path Name	Comments	Container Base Path
0.Administration	Admin	Default Administration Container	vG.AdminBasePath
99.Shared_folders	Shared	Default Shared Container	vG.SharedBasePath
1.Company_Project	Proj1	First Project, Company or department	vG.Proj1BasePath
2.Company_Project	Proj2	Second Project, Company or department	vG.Proj2BasePath

Basic Container Map Example, called BasicContainerExample in Variable Editor

In a more complex environment having a separate container for each data source could be the right

way to go.

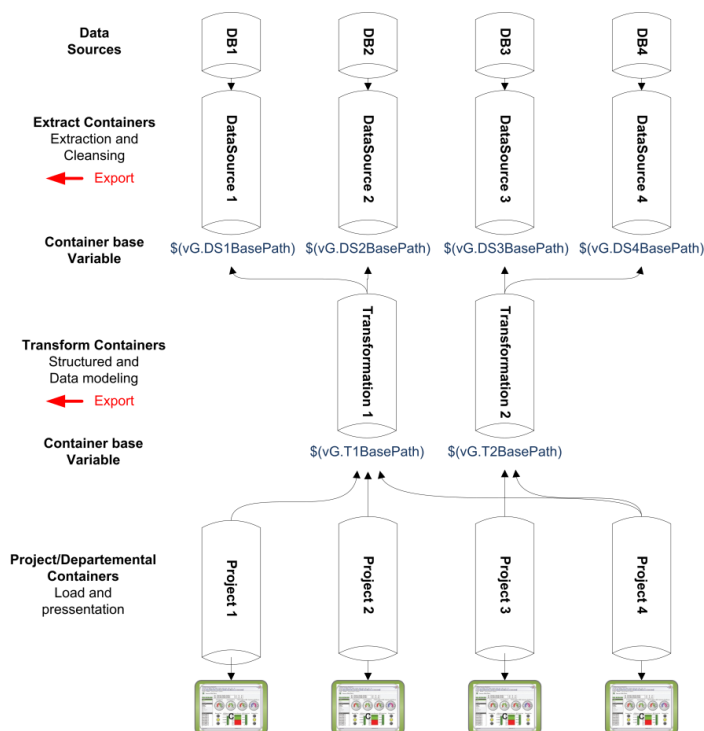


In This Example the project containers collect data from the System containers. All containers have unique global path variables that are utilized by QlikView Applications in the project containers. Export to other systems can be done in both levels. The system Containers are stored under 98.Systems sub folder to create separation between projects and source both visually and as a security boundary.

Folder Name	Path Name	Comments	Container Base Path
0.Administration	Admin	Default Administration Container	vG.AdminBasePath
99.Shared_folders	Shared	Default Shared Container	vG.SharedBasePath
1.AcmeStore	AcmeS	First Company or Department Container	vG.AcmeSBasePath
2.AcmeTravel	AcmeT	Second Company or Department Container	vG.AcmeTBasePath
98.Systems\1.Oracle	Oracle	First System Container containing Oracle Data	vG.OracleBasePath
98.Systems\2.SAP	SAP	Second System Container containing SAP Data	vG.SAPBasePath
98.Systems\3.SQL	SQL	Third System Container containing MS-SQL Data	vG.SQLBasePath

System Container Map Example called SystemContainerExample in Variable Editor.

It's also possible to align the containers in a traditional ETL process.

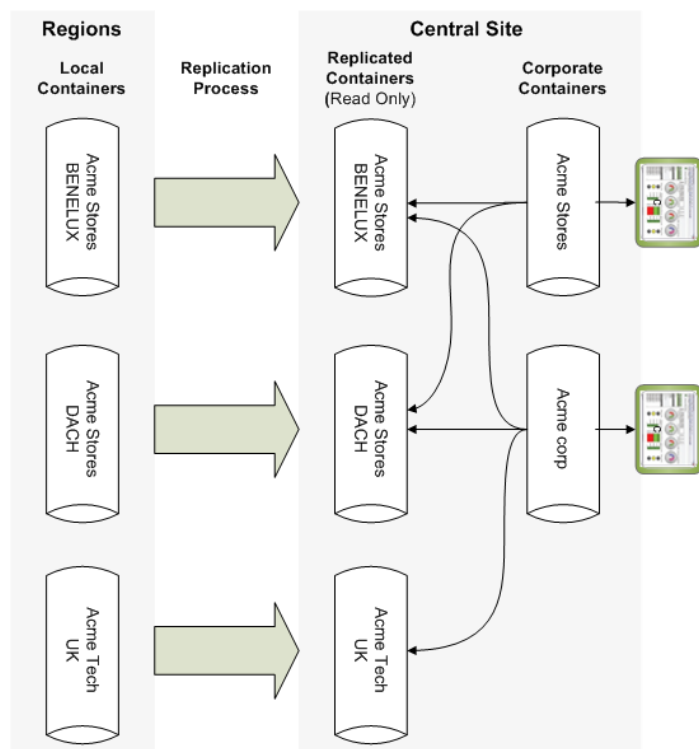


The Extract containers collect and clean data from the data sources, each extract container has a unique global path variable that is common to the transform containers. The Project containers load and present the transformed data in the transform containers. Export to other systems can be done in Extract or Transform levels. When designing the logical structures, align the container structures accordingly by using subfolders, in this example for the Extract and Transformation tier.

Folder Name	Path Name	Comments	Container Base Path
0.Administration	Admin	Default Administration Container	vG.AdminBasePath
99.Shared_folders	Shared	Default Shared Container	vG.SharedBasePath
1.AcmeStore	AcmeS	First Company or Department Container	vG.AcmeSBasePath
2.AcmeTravel	AcmeT	Second Company or Department Container	vG.AcmeTBasePath
98.Systems\1.Oracle	Oracle	First System Container containing Oracle Data	vG.OracleBasePath
98.Systems\2.SAP	SAP	Second System Container containing SAP Data	vG.SAPBasePath
98.Systems\3.SQL	SQL	Third System Container containing MS-SQL Data	vG.SQLBasePath
97.Transform\1.Transform	T1	Transformation Container 1	vG.T1BasePath
97.Transform\2.Transform	T2	Transformation Container 2	vG.T2BasePath

ETL Container Map Example called ETLContainerExample in Variable Editor.

This is an example of Distributed regional Containers that is replicated to the central corporate site. The Region containers will be master and the replicated containers will be read-only slaves. The replication process is usually done with Windows DFS-R or RoboCopy scripts.



In an enterprise environment this architecture creates several advantages:

- The regions have possibility to create independent analytics and development
- The corporate can easily analyse data from all the regions.
- All the data is stored at corporate site so the reload speed of data will be very fast.
- Local applications could easily be hosted and/or used in the corporate
- Replication of compressed QlikView data use less bandwidth than sending queries across the WAN.

Folder Name	Path Name	Comments	Container Base Path	Alt Root Path
0.Administration	Admin	Default Administration Container	vG.AdminBasePath	
99.Shared_folders	Shared	Default Shared Container	vG.SharedBasePath	
1.AcmeStore	AcmeS	First Company or Department Container	vG.AcmeSBasePath	
2.AcmeTravel	AcmeT	Second Company or Department Container	vG.AcmeTBasePath	
98.Regions\1.DACH	DACH	Read Only Container for DACH	vG.DACHBasePath	
98.Regions\2.APAC	APAC	Read Only Container for APAC	vG.APACBasePath	
98.Regions\3.Americas	AM	Read Only Container for Americas	vG.AMBasePath	
1.AcmeStoreDACH	ASDACH	AcmeStore container in DACH region	vG.ASDACHBasePath	\\DACH\QV
1.AcmeStoreAPAC	ASAPAC	AcmeStore Container in APAC region	vG.ASAPACBasePath	\\APAC\QV
2.AcmeTravelAm	ATAM	AcmeTravel Container in Americas Region	vG.ATAMBasePath	\\AM\QV

Regional distributed container example called RegionalDistributedContainer

Moving application between containers

Moving applications between containers are necessary during DTAP promotion or environment consolidation. When moving between containers there are some considerations that sometimes need to be addressed.

What in the framework are applications depended on?

All container base structures are identical and applications usually work flawless across containers but there could be dependencies in the containers that need to be addressed, like:

- Connection string, resided in *vG.ConnStringPath*
- QVD and other sub folders, example *\$(vG.QVDPPath)MetaData*
- Custom Global Variables (created by **Variable Editor**) need to move to the new container.
- If running a *Binary* load statement, (if needed) change relative path in the script.
- Connections to Shared Folders container could need scripts/data from the Shared Folders container in the other environment

Global Variable links (created by [LoadContainerGlobalVariables](#)) to other containers need to be added in the Container Map if promoting to a new environment.

Deployment process and DTAP Environments

It is recommended to always have test, pre-production and production environments combined with a deploy process. Always test new or modify QlikView source applications before moving over to production. It is during acceptance testing that applications creating unusual memory and CPU behavior should be caught.

Create the similar container architecture in all environments, this makes it easy to secure the environments but still be able to promote applications between the environments.

Read more on this subject in the **QlikView Deployment Guide**.

QlikView Management Console overview

To manage QlikView there are several tools. The primary one is QlikView Management Console (QMC) and is usually resided in Back End on the Publisher Server. Only when using a Publisher license the full functionality of QMC will unlock. QlikView Management console is connected to all the QlikView services and folder structures like the **Deployment Framework** containers.

The main Publisher function is to load data from sources, to put security on the data and to distribute the data to the QlikView Servers. This is done by creating and scheduling tasks in QMC. Tasks can be chained to each other creating a workflow activated from one single task either by schedule or from external input (called an EDX trigger).

Access to QlikView Management Console

To access QlikView Management Console the administrator need to have security access. There are different kinds of administrators with different levels of security clearance. These are:

- **QlikView “Super” Administrator** that have full control in the Management Console, this user or preferably global group must be a member of the local group called **QlikView Administrators**. The user installing the QlikView platform will automatically have **Super Administrator** security level.
- **Document Administrators** or “Container Administrators” have access to a given set off containers added in the QMC by the **Super Administrator**. The **Document Administrators** will access a restricted version of the QMC. The **Document Administrator** is only allowed to work with the source documents that this user has permissions on. Always use global security groups for the Container Administrators role.
- **QlikView Management API** is a local group for users that could access QMC via API’s. The **Super Administrator** global security group could also have that access. There are tools in Deployment Framework that works best when having this access level.
- **QlikView EDX** is a local group used for external systems triggering tasks inside QMC by using specific API’s.

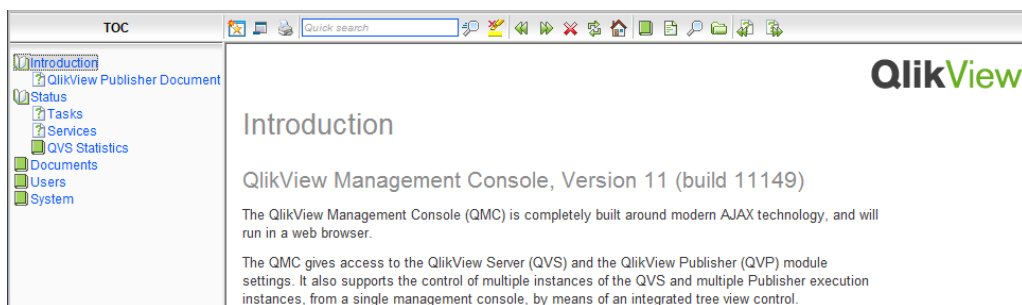
Read more regarding security related information in container security section.

Start using QlikView Management Console

Open and use port 4780 to access management Console service (*QlikviewManagementService*), example:

<http://localhost:4780/qmc/default.htm>

This Guide will not go in deep regarding all the functions and tweaks inside the QMC, there is always possible to get detailed help by clicking the Help Icon inside QMC.

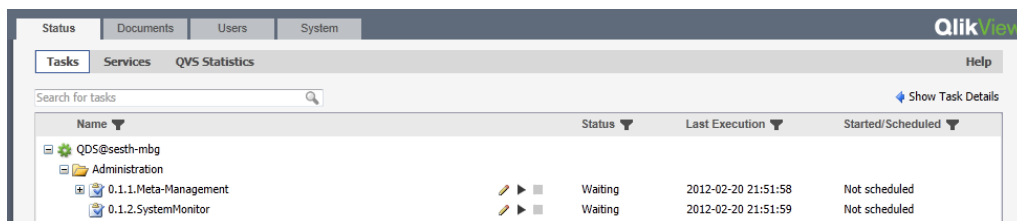


Before starting operative administrative tasks all the QlikView Services need to be connected during the QlikView Platform installation and QMC setup phase. Read more in **QlikView Server Reference Manual** and **Deployment Framework Getting Started** documentation.

QlikView Management Console Administration

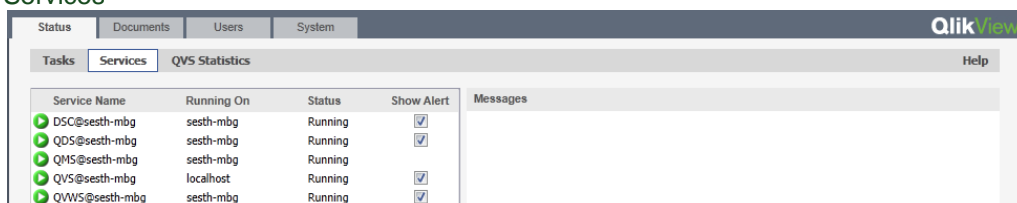
Task overview

The QMC start page is the Task overview page, this page is used to identify running, failed and successful tasks.



Only tasks related to your security clearance are shown and possible to access. To get details and logs relates to a task click on the **Show Task Details** icon. To run a Task manually, click the play button and to stop press stop button, to modify a task click on the pencil symbol.

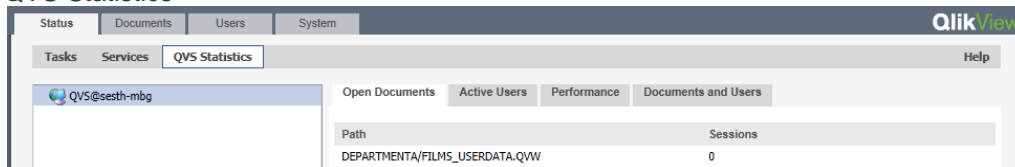
Services



On the Services page, an overview of the statuses of the various Windows services is presented.

The statuses of the services are automatically refreshed.

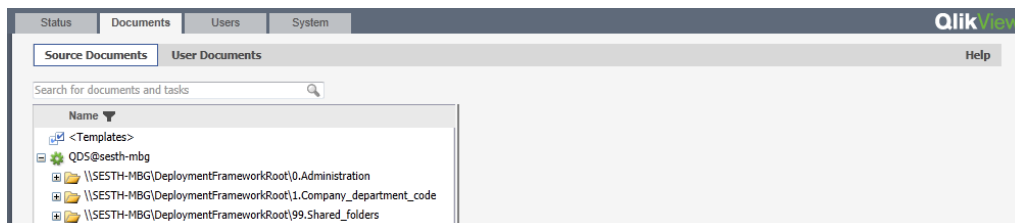
QVS Statistics



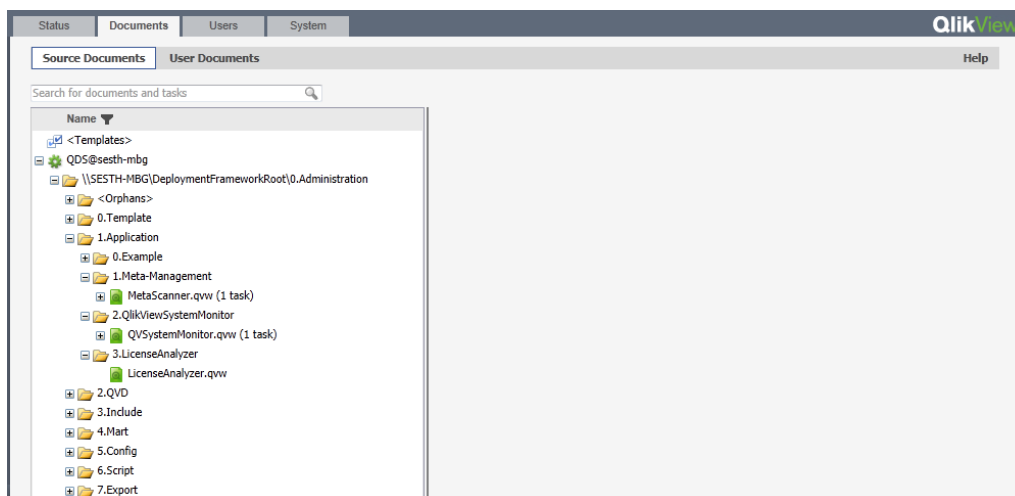
On the Statistics page, the live statistics on all of the QlikView Servers (Front End) that are managed by the QlikView Management Console (QMC) are presented.

Documents Tab and Source Documents


Here is where the source Qvw Documents in the Back End is shown. The different structures side by side is the Development Framework container structures.

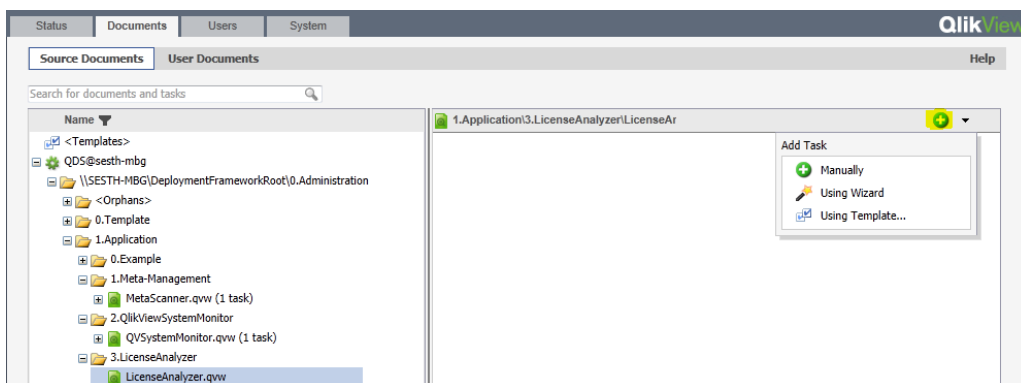


In this example the administrator have access to **0.Administration**, **1.Company_department_code** and **99.Shared_folders** containers.



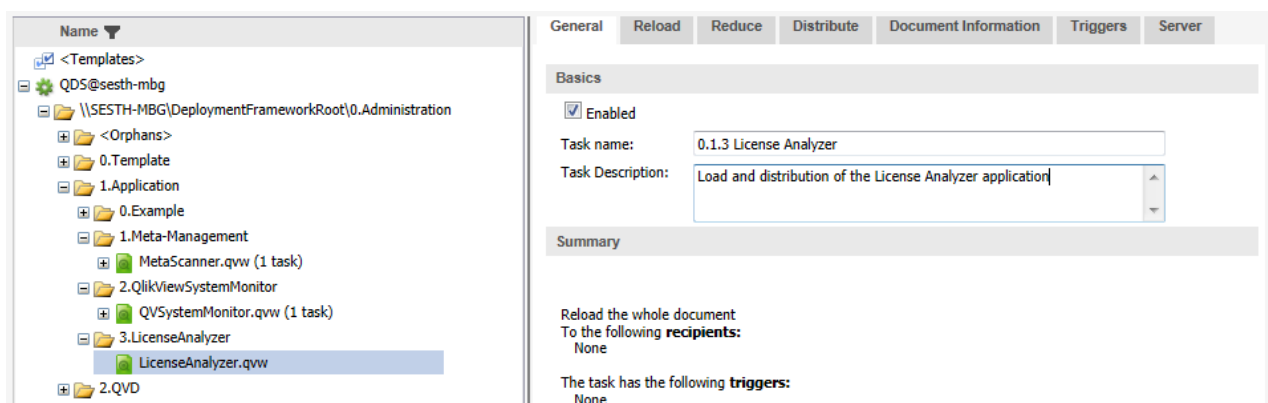
Inside the **0.Administration** container the *MetaScanner.qvw* and *QVSystemMonitor.qvw* are accessible and there is one task attached to each document.

Create a task on visible applications by clicking on the file and pressing the  Icon (Add Task button)



There are three ways of adding a task manually, using the Wizard or by using a predefined template.

Create a Task Manually



When adding a task manually a configuration box will appear. First we need to type the Task Name this name will show up in the Status tab. In Deployment Framework the best practice is to use the number series In front of the folders before the name, example: *0.1.3 License Analyzer* or *0.1.1.Meta-Management*

0 = 0.Administration Container name

1= 1.Application folder name

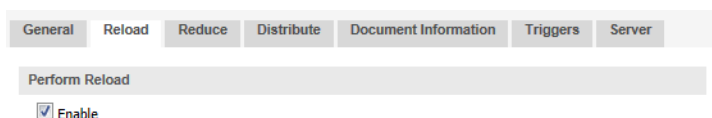
1= 1.Meta-Management subfolder name

Administration					
0.1.1.Meta-Management	Waiting	Never	Not scheduled		
0.1.2.SystemMonitor	Waiting	Never	Not scheduled		

This name standard will make it easier to understand in what container and where the QlikView file is placed.

Also finding tasks in chaining lists (like on event from another tasks) are easier when using numbers in the beginning.

Reload Tab




On the Reload tab, the current task is as default configured to load the document, uncheck Perform reload box to disable reloads. For advanced functionality read the online HelpReduce Tab

On the Reduce tab, a document can be configured how to be split into several reduced copies of the original. A reduced document contains only the reduced information. The Field Reduced Document Name is the user document name distributed to QlikView Server, change here to change child file name. Default name is same as the Source Document. For advanced functionality read the online Help

Distribute Tab

This tab contains configurations of source document distribution to becoming User Documents in QlikView Server.

Use Manual document distribution method and the authorization of recipients for easy distribution. To add a specific resource entry to distribute to, click on the **Add** icon  to the right in the pane, and configure the following fields:

- **Server** Select the desired QlikView server in the drop-down list. A **Document Administrator** will only have Access to his servers, , read more in security section.
- **Mount** Select the desired QlikView Server mount in the drop-down list. A **Document Administrator** will only have Access to his mounts, read more in security section.
- **User Type** The type of recipients. To manage the authorization of users and groups, select one of the following drop-down list options:
 - **All Users**, meaning that all users are authorized.
 - **All Authenticated Users**, meaning that any authenticated user is authorized.
 - **Named Users**, meaning that users that should be authorized are manually added, by searching for users and groups in a domain or on a computer

For advanced functionality read the online Help

Document Information Tab

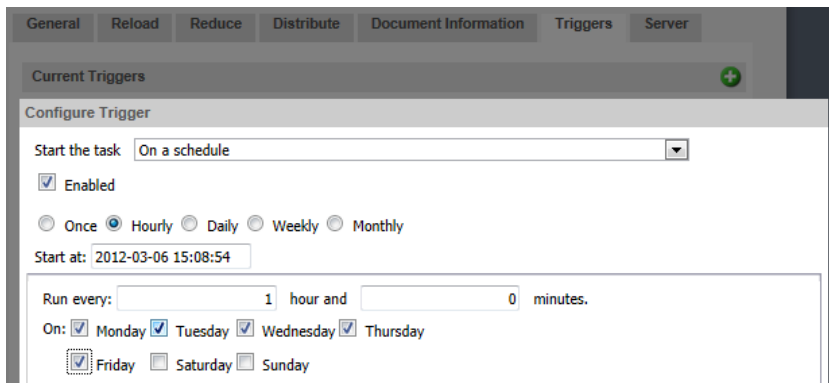
Create or apply a category for the current document. Categories are used to bundle documents in the status tab, to make categorization easier for the end-user. The categories are only visible to the end-user on the QlikView AccessPoint. Each document can only be part of one category.

Best practice in the Deployment Framework is to use application functional name as category, in this case Administration. Document description is displayed in Document Details on QlikView AccessPoint in Front End.

For advanced functionality read the online Help

Triggers Tab

On the Triggers tab, the current task can be configured to be started by triggers. A task can have multiple triggers, creating a workflow of tasks

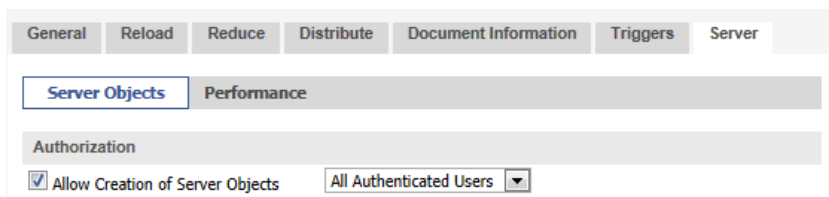


The screenshot shows the 'Triggers' tab in a software interface. At the top, there are tabs: General, Reload, Reduce, Distribute, Document Information, Triggers (selected), and Server. Below the tabs, there is a 'Current Triggers' section with a green plus icon. The main area is titled 'Configure Trigger'. It includes a dropdown menu for 'Start the task' set to 'On a schedule'. Below this is a checked 'Enabled' checkbox. There are radio buttons for frequency: 'Once' (unselected), 'Hourly' (selected), 'Daily' (unselected), 'Weekly' (unselected), and 'Monthly' (unselected). A 'Start at' field shows the date and time '2012-03-06 15:08:54'. A 'Run every' section has input fields for '1' hour and '0' minutes. Below that, there are checkboxes for days of the week: 'Monday' (checked), 'Tuesday' (checked), 'Wednesday' (checked), 'Thursday' (checked), 'Friday' (checked), 'Saturday' (unchecked), and 'Sunday' (unchecked).

In this example we are creating a schedule trigger. For advanced functionality read the online Help

Server Tab

In Server Tab the authorization of server objects and access control of the **current document** can be managed.



The screenshot shows the 'Server' tab in the same software interface. At the top, the tabs are: General, Reload, Reduce, Distribute, Document Information, Triggers, and Server (selected). Below the tabs, there are two sub-tabs: 'Server Objects' (selected) and 'Performance'. The main area is titled 'Authorization'. It includes a checked 'Allow Creation of Server Objects' checkbox and a dropdown menu set to 'All Authenticated Users'.

For advanced functionality read the online Help

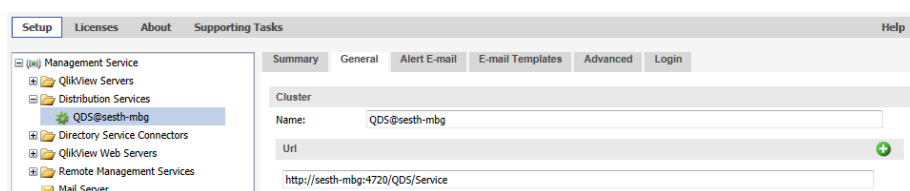
QMC Container setup in QlikView Publisher Enterprise edition


When using the Deployment Framework adding additional containers when needed is important, else the environment could become messy and difficult to administer. Creating the container folder structure is easiest done with the *2.VariableEditor\VariableEditor.qvw* application in the *0.Administration* container.

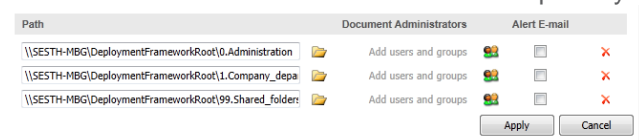
Go to **Script tools in Administration container** section for detailed info regarding creating a new container.

Setting up Containers in QMC

The new container also needs to be added in the QMC System/Setup Tab, and drill down to the Distribution Service folder. Super Admin security access is needed for this.



Highlight the QDS service inside the folder, to the right information regarding the service will show. In General tab scroll down to the bottom. In the Path list all containers should appear. Add additional containers into Publisher Distribution Service path by click on the **Add** icon  to the right in the pane.



Best practice is to use UNC path to the root share one level above the containers. Adding Document Administrator for access to a specific container read more under Container Security.

Setting up QlikView Server Mounted Folders

QlikView Server (QVS) in front end has the possibility to use mounted folders as distribution point. Best practice is to create Mounted Folders for each container distributing to the QVS. When using **Document Administrators** at least one Mounted Folder for each role is mandatory, else the **Document Administrators** will have no access to the QVS folders and not be able to create distribution tasks. Read more in the security section.

To create a QlikView Server Mounted Folder you must have Super Administrator security rights. Remember that Mounted Folders is accessed by the QVS that probably is on another server. In QMC System/Setup Tab expand QlikView Server icon and chose the QlikView Server that should have the new Mounted Folder.

Click on the Folders Tab.

Name	Path	Browsable
1.DepartmentA	C:\QlikViewResource\UserData\1.DepartmentA	<input checked="" type="checkbox"/>

Press the  icon to create a new Mounted Folder in the selected QVS.

Name	Path	Browsable
1.DepartmentA	C:\QlikViewResource\UserData\1.DepartmentA	<input checked="" type="checkbox"/>
2.DepartmentB	C:\QlikViewResource\UserData\2.DepartmentA	<input checked="" type="checkbox"/>

Type the new Mounted Folder **Name**, best practice is to use the same name as the Container added in the Distribution Service (read *Setting up Containers in QMC* section). Add the new Mounted Folder path. **Important** before applying, the new folders also need to be created on the QlikView Server file system, in this case *C:\QlikViewResource\UserData\2.DepartmentB*.

Security

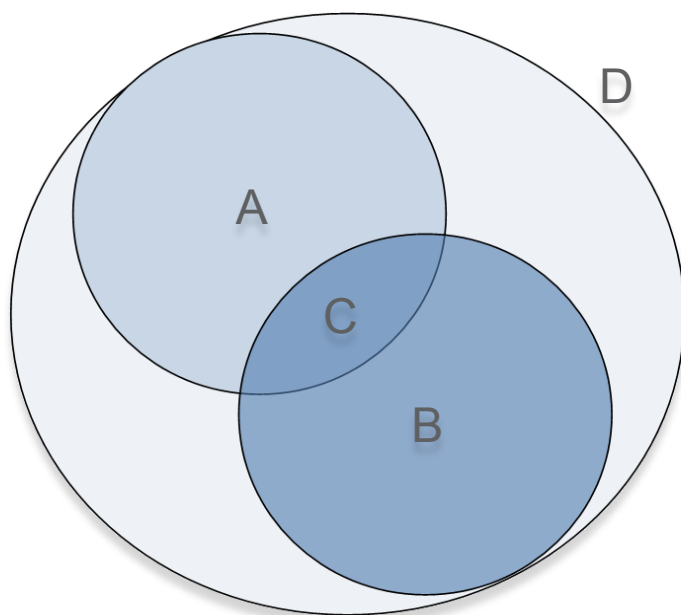
Container security

Containers could have different security settings. This is useful when several administrators and developers are working in different projects or departments but still use a common environment.

All containers are independent from each other and could be in separate Active Directory administrative security groups. Important is that a global security group surrounds all the containers, super admin and QlikView service accounts must reside in this group.

Example:

- A) Security group for Container A
- B) Security group for Container B
- C) Shared Folder security group where both Container A and Container B have access
- D) Super Admin security group, this group has full access to the root folder above the containers. The QlikView service accounts must be member in this group. The Super Admin security group should be a member of the local QlikView Administrators group.



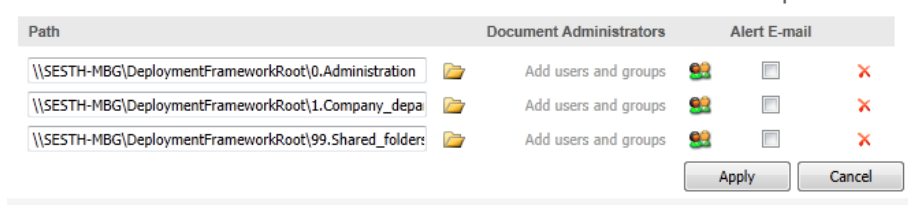
Container Security Considerations

As mentioned earlier, each container could have a separate security boundary. In addition (special cases) different security levels could be applied inside a container.

For example: To prevent connection string access, remove the departmental groups (A or B) on the `\3.Include\3.ConnString` folder. Only Super Admin group will then have access to the connection string folder in this container.

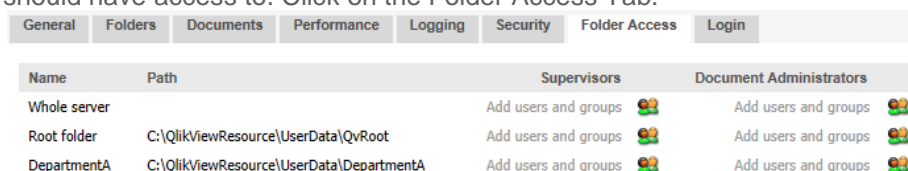
Document Administrator Security Settings

The security groups created in Active Directory correlates to the role based security functionality in QlikView Publisher. Add all the containers separately into Publisher Distribution Service path and put the security groups as **Document Administrator** on the different containers by using the 🧑 icon. Document Administrator could be a local administrator or a developer.



Document Administrator security in QlikView Server

Document Administrators also need access to a QlikView Server, the **Whole Server** or a mounted folder to be able to distribute documents to front end. In QMC System/Setup Tab expand QlikView Server icon and chose the QlikView Server that owns the mount the **Document Administrator** should have access to. Click on the Folder Access Tab.



Add the security groups for **Document Administrators** on the selected container by using the 🧑 icon. Use the **Supervisor** role to give full read access to QlikView Documents in the front end, and could be set for the **Whole Server** or for each mount.

Container Sharing

It's important to secure access to the container shares as well as the folder structures. When sharing a container or inside the container always have the container name in the share name, else it will be difficult to find among the shares. For extra security when for example export folder is shared to a system, use \$ to hide the share.

Folders and container that could be shared are:

<i>Root folder (all containers)</i>	Security group for the Super Admin	<i>Full Control</i>
<i>Container Base Folder</i>	Same as container security group for Admin/developers Security group for the Super Admin	<i>Change</i>
<i>4.Mart\Subfolder</i>	For advanced business discovery users to do local analytics with data and logic from a QlikView Mart. Use a Security group.	<i>Read</i>
<i>5.Config\Subfolder</i>	Share when settings/values need to be changed manually by key personal. Use a Security group.	<i>Change</i>
<i>6.Export\SubFolder</i>	Share for export to other systems. Best practice is to use hidden \$ share	<i>Read</i>

Application Security Table (Section Access)

In QlikView the security settings of the qvw-file is set in the script. Access rights and User Levels are defined in the Section Access part of the script. Section Access can be used to set access restrictions to data, sheets and sheet objects. All access control is managed via files, SQL databases or inline clauses in the same way as QlikView normally handles data. If an access section is defined in the script it must be followed by the statement Section Application in order to load normal data.

Section Access system fields

Access levels are assigned to users in one or several tables loaded within the Section Access. These tables can contain several different user-specific system fields, typically NTNAME and the field defining the access level, ACCESS. The full set of section access system fields are described below. Other fields like GROUP or ORGANISATION may be added to facilitate the administration, but QlikView does not treat these fields in any special way. None, all, or any combination of the security fields may be loaded in the access section. However, if the ACCESS field is not loaded, all the users will have ADMIN access to the document and the section access will not be meaningful.

Section access system fields are:

- **ACCESS** (mandatory) A field that defines what access the user should have.
- **NTNAME** A field that contains a string corresponding to a Windows NT Domain user name or group name.
- **USERID** A field that contains a user ID that has the privilege specified in the field ACCESS.
- **PASSWORD** A field that contains an accepted password.
- **SERIAL** A field that contains a number corresponding to the QlikView serial number.
- **NTDOMAINSID** A field that contains a string corresponding to a Windows NT Domain SID
- **NTSID** A field that contains a Windows NT SID.

Section Access in Combination with Publisher

While QlikView Publisher can use its “loop and reduce” functionality to reduce a QVW by rows by user or group as it is being reloaded, you can also accomplish this in Section Access dynamically as the document is opened, either method will work and both have benefits. The Loop and reduce from Publisher will help you to reduce the memory footprint of the QVWs on your server(s), while the Section Access method is portable with the document. Another reason to use Section Access is the application of authentication in the QVW, through SSO or user ID and password. This is especially important if the QVW is going to be enabled for download from the AccessPoint or otherwise distributed to users.

Database Connection String

Connection strings to data sources are security credential and should always reside in the architecture Back end, in the data tier. Well protected from unauthorized access. Remember that architecture front-end (QlikView Server and QlikView Web Server) does not have any open ports the back-end, these servers could be in different network zones and security boundaries. When distributing QlikView applications via the Publisher in Back end to the application tier, scripts and connection strings will automatically be removed.

Security considerations

By separating the connection string from the script reusability and higher security will be achieved. There are two ways to separate and reuse the connection strings:

Include File

Best practice is to keep the connection strings in a separate Include file. This behavior is supported by Deployment Framework. Use the Global Variable *vG.ConnStringPath* to connect inside your container, example:

```
// Connection string to Northwind Access data source
$(Muct_Include=$(vG.ConnStringPath)\0.example_access_northwind.qvs);
```

If the connection string is in another container, for example the Shared folders use the Global Variable *vG.SharedConnStringPath* to connect, example:

```
// Connection string to Northwind Access data source
$(Muct_Include=$(vG.SharedConnStringPath)\0.example_access_northwind.qvs);
```

Custom Global Variable

By using the Global Variable Editor there is the possibility to create a Global Variable representing the connection string. This method is not as secure as using an include file. Include files can be secured by different security groups this is not possible when using Global Variables that will be used by all the applications within a Container. When a container is secured and dedicated for a single source system (example Oracle container) connection strings as global variables could be used.

QlikView Server without Publisher license







QlikView infrastructure without a Publisher license does not have a Back End. The Source documents are loaded with a simple reload engine that is bundled with QlikView Server. This solution works fine with smaller installations, and in development and test environments. In a production environment QlikTech recommends the Publisher and Back End/ Front End separation.

Setting up Containers in QMC

Without a Publisher there is no Source Document structure in QlikView. Instead the Containers need to be mounted directly in the QlikView Servers User Document structure.

When not using a Publisher file system security on the containers will reflect into AccessPoint and QlikView Server, this is not a good behavior. Therefore it's important to add two mounted folders for every container:

- One mounted folder that points to the Container root, used for document reloading. This mount should have **Browsable** disable, denying end users access to this container.
Example: **Name** 0.Administration **Path** c:\QV-Docs\userdocs_test\0.Administration **Browsable** disable
- One mounted folder pointing to 1.Application inside the same Container. This mount should have **Browsable** enable, allowing end users access to the application folder.
Example: **Name** 0.Administration_1.Application **Path** c:\QV-Docs\userdocs_test\0.Administration\1.Application **Browsable** enable.

Mounted Folders				
Name	Path		Browsable	
0.Administration	c:\QV-Docs\userdocs_test\0.Administration	 	<input type="checkbox"/>	
0.Administration_1.Application	c:\QV-Docs\userdocs_test\0.Administration\1.Application	 	<input checked="" type="checkbox"/>	

A container that only is used for reloading and not for end user presentation do not need the second mount (to the 1.Application folder) it's very important that **Browsable** is *disabled*.

Do not uncheck the setting **Respect browsable flag on mount** in QlikView Web Servers settings. This will disable the Browsable functionality in QlikView Server Mount Settings.

QMC task scheduling

Creating a reload task with no publisher is quite straight forward. Remember that without a Publisher license only scheduled reloads is possible.

Start QlikView Management Console (QMC) and click on the Documents Tab. Select the (User Document) container and QlikView Document that needs to be reloaded and add schedule in the right pane.

User Access Control (UAC)

UAC in combination with c:\ProgramData\QlikTech\Documents as QDF repository will disable QlikView's write capability making Variable Editor to crash. The work around is to first execute QlikView Developer with elevated privileges (Run as Admin) and after open Variable Editor. This scenario can also occur in other folders if Windows is locked down.

Variable Editor Shortcut when using UAC

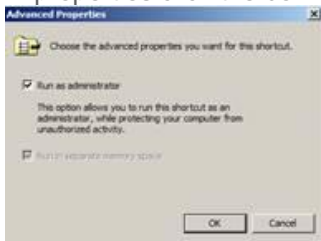
The default Variable Editor Shortcut will not work when UAC is activated and folders are secured. Workaround is to modify the Variable Editor shortcut for your file system (not relative as the default shortcut) change target to:

`"%ProgramFiles%\QlikView\Qv.exe"`

`"C:\ProgramData\QlikTech\Documents\0.Administration\6.Script\2.VariableEditor\VariableEditor.qvw"`



In properties click the box Advanced activate Run as Administrator



UAC Command line script example

Here is another and more flexible example how to get around UAC, in this case create a script and place it in 0.Administration container base path (next to the Variable Editor shortcut). Right click on the cmd file and select Run As Administrator. This one works with UNC path as well

```
@echo off
:: Change VEPATH if Deployment Framework structure is modified
SET VEPATH=\\6.Script\2.VariableEditor
:: Mount folder in VEPATH
pushd "%~dp0%VEPATH%"
:: Execute VariableEditor.qvw
VariableEditor.qvw
:: Unmount folder
popd
```

QDF Tools

The Administrative container is installed when deploying QDF. The container name is *0.Administration*, this container contains admin stuff like Variable Editor and Meta-management. The container is very important for administrative duties and to maintain the QlikView platform.

There are also good to have tools in Administration container. The tools reside only in the Administration container in the *6.Script* folder but will work in all containers. Copy the scripts into *6.Script* folder in other containers if needed. Tools available are:

1.Create-prj (creates project folders)

Create-prj.cmd creates *xxx-prj* folders beside the *xxx.qvw* files in the container that executes Create-prj.

folder structure. The *-prj* folders are used as object and script repository and is usually used for version control of QlikView Files. No configuration needed to use *Create-prj.cmd* just execute, works with both physical and UNC path.

The *Create-prj* folder and script will be copied to new containers when running *VariableEditor.qvw*


More documentation is available in the *1.Create-prj* folder

Variable Editor

Variable Editor is a QlikView application that graphically controls Deployment Framework. *System* and *Custom Global Variables* can be added and edit within Variable Editor and all containers are plotted in a Container Map (master is stored in Administration container) this map can also be edited with Variable Editor.

Variable Editor is found under *6.Script\2.VariableEditor\VariableEditor.qvw* in the *0Administration* container.

But can also be executed by using the Variable Editor Shortcut.

 **VariableEditor Shortcut**

Do not execute Variable Editor on a remote computer with high latency network access to the container folder structures, Variable Editor is not optimized to run on slow networks.

Help 

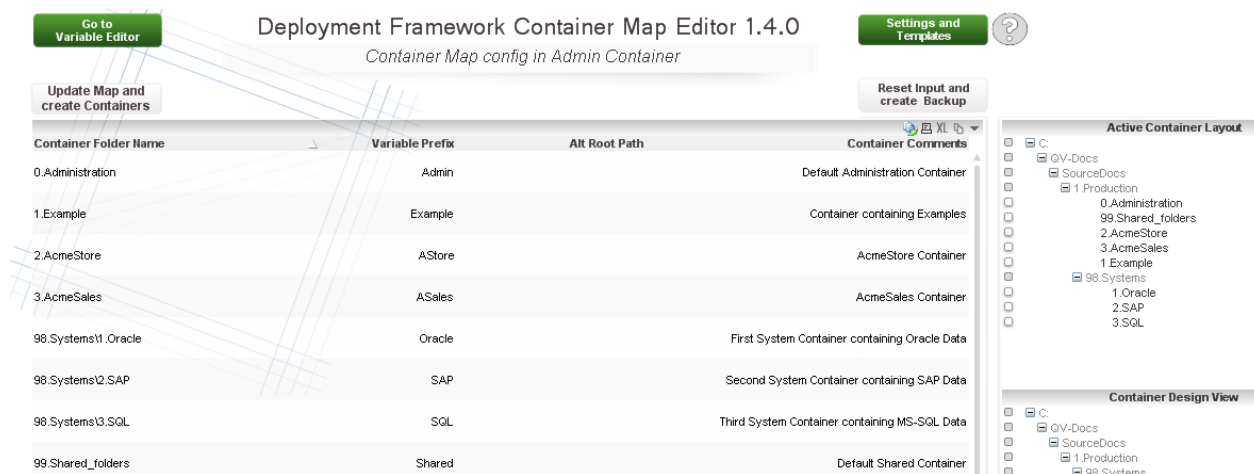
There is a help button in the VariableEditor available when needed.

Container Map Editor

Is used to administrate and populate containers within the framework. Press **Go to Container Map** to change to container view. Container Map is used by Deployment framework to find the containers and to create container connection variables. These variables are created when using the

LoadContainerGlobalVariables function, example:

call LoadContainerGlobalVariables ('HR'); Will connect to 1.AcmeHR container (if exist).



Edit or modify container map in the table, remember that it's only the container Map that is changing not the physical container structure.

Container Input Fields

- *ContainerFolderName* contains the Container folder Name. To create or add in a sub container structure type *folder name\container name*. Example 1: *1.Oracle* to create a container in the same level as 0.Administration
Example 2: *98.System\1.Oracle* to create a container under a system folder. To add a container in another file system.
- *ContainerPathName* , enter prefix share variable names in *ContainerPathName* field, example *Oracle*.
- *Alt root path*, Edit an optional container path in *alt root path* field. A container could also be copied in a subfolder structure the subfolder name will be created automatically.
- *Container Comments* Is descriptive Meta Data regarding the containers, very good to use for documenting the solution.

Reset Input and Create Backup

Will reset (revert) all inputs and also create a backup of the Container Map.
The backup name will be *ContainerMap_Backup.csv*.

Retrieve Backup

Use Retrieve Container Map Backup to get back to the backup stage.

Update Container Map

Use this button to apply the new Container map after adding and/or modifying the container layout.



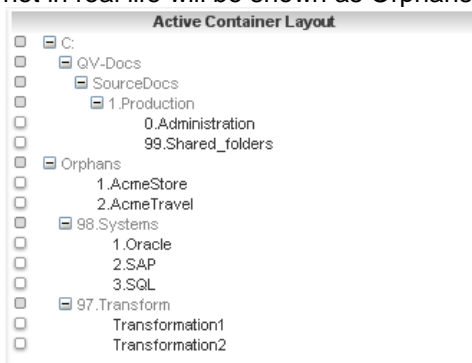
Create New Containers option

Create New Containers will create containers based on the current container Map. This button is only shown after Update Container Map is applied and accepted. New Containers can only be created from the 0.Administration container this means that the selected and applied container either is *vG.BasePath* or *vG.AdminBasePath*.



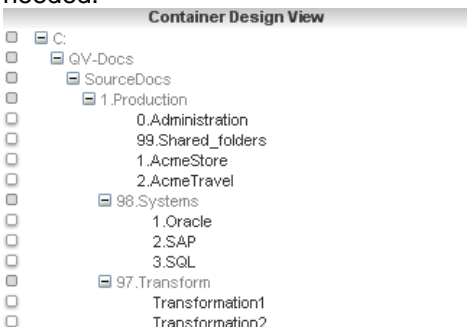
Active Container Layout

Shows physical containers that exist within the Container Map Container that exists in the Map and not in real life will be shown as Orphans as shown in the example below:



Container Design View

Shows the architecture while designing, in this view no Orphans is shown and no reload/refresh is needed.



Variable Editor Tab

Deployment Framework Global Variable Editor 1.3

HR_KPIVariables file in 0.Administration Container

Variable Files = HR_KPIVariables

Select Container = \$(vG.BasePath)

Update Variables

Refresh and Create a Backup

Add Variable File =

Variable Name	Variable Value	Variable Comments	Search Tag

Retrieve Backup

Search

Help

Variable Name
Type SET or LET in front of your variable.
Use vG. as global variable prefix.
Example SET vG.statistics

VariableValue
Do not combine numbers and letters
when using LET function use the
SET function instead

Variable Files
Variable files other than CustomVariables
will not be loaded by into the applications by default.
Add Sub Function below into the application script:
call LoadVariableCSV("\${vG.BasePath}\HR_KPIVariables.csv",
[Specific variable Optional])

Global Variables

The Global variables are modified by the Variable Editor and I stored in `$(BaseVariablePath)\CustomVariables.csv` files in each container. Global variables (with the prefix `vG.`) are loaded by default into QlikView during the framework initiation process in the beginning of the script (read more in using Deployment Framework Containers). Global variables should only be used when a variable is shared by several applications in a Container.

Universal Variables

By using Universal Variables that are stored in `$(SharedBaseVariablePath)\CustomVariables.csv` files in the Shared Folders Container, we get “single point of truth” across all containers. Universal Variables are by default loaded during the framework initiation process, have the prefix `vU` and is also modified by the Variable Editor application.

System Variables

System Variables are actually also Global Variables that start with (`vG.`), the difference is that System Variables are predefined variables used to store system settings like QlikView Server log path. System Variables are also not preloaded, `3.SystemVariables.qvs` include script needs to be run to load in the System Variables into QlikView.

System Variables are modified by the Variable Editor and I stored in `$(BaseVariablePath)\SystemVariables.csv`. There is usually only need for one System Variable version, the main is stored in 0.Administration container and is by default replicated out to the other containers.

Variable Input Fields

- **VariableName** Type *SET* or *LET* in front of your variable name. Use `vG.` or `vU.` as Global or Universal Variable prefix. Example1 *SET vG.statistics*. Example2 *SET vU.statistics*.
- **VariableValue** Type value or text, when entering text do not use brackets (") this is done automatically.
Do not combine numbers and letters when using LET function, use the SET function instead for this.
- **Comments** Used for comments like author and creation date
- **Search Tag** Used only for easy search

Variable Files, Custom Global Variables

Custom Global Variables will automatically be loaded into QlikView applications when using Deployment Framework. Each Container has its own Custom Global Variable file that the applications use.

For Global Variables that need to be used across containers modify Shared Custom Variable file with Variable editor.

Refresh Create a Backup

Will refresh the view without updating Variable files and at the same time create a backup.

Retrieve Backup

Use Retrieve Backup to get back to the backup stage created by *Change Variable File and Create a Backup* button.

Update Variables

Use this button to apply the new variables after adding and/or modifying.

Add and Remove Variable Files

Variable Editor has the possibility to add variable files into the selected container in addition to the default *Custom Global Variables*. Type the variable filename into the *Add Variable File* input box and press enter like example below:

Add Variable File = HR_KPI

The *Refresh and Create a Backup* box will now change to a Create Variable File box

Create Variable File

When pressing apply the new csv file (empty) will be created as *HR_KPIVariables.csv* and stored under selected container *3.Include\1.BaseVariable*.

To remove a Variable File add the command **del** before the filename and run the script like example below:

Add Variable File = del HR_KPI

The box will change to *Delete Variable File*.

Delete Variable
File

Variable files other than Custom Variables will not be loaded by *1.Init.qvs* into the applications by default.

Add Sub Function below into the application script instead:

```
$(Include=$(vG.SubPath)\2.LoadVariableCSV.qvs);
```

call LoadVariableCSV('\$(vG.BaseVariablePath)\HR_KPIVariables.csv', '[Specific variable Optional]')

Variable Files, System Variables

System Variables setting are hardware and system folder settings, example log locations needed to monitor the platform. QlikView System monitor uses these settings.

To execute System Variables inside a QlikView application include:

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

These are the default System Variables, change so that these settings represent your QlikView environment.

- `vG.ServerLogPath` QlikView Server logs path.
Default to `C:\ProgramData\QlikTech\QlikViewServer\`
- `vG.UserDocumentPath1` is QlikView Sever User Document path. If having more User Document folders use `vG.UserDocumentPath2`, `vG.UserDocumentPath3`...
Default to `C:\ProgramData\QlikTech\QlikViewServer\QlikView\`
- `vG.QMSPPath` QlikView Management Service ProgramData folder path.
Default to `C:\ProgramData\QlikTech\ManagementService\`
- `vG.QVPRPath` Publisher QVPR data base path (usually the same as `$(vG.QMSPPath)\QVPR\`).
Default to `$(vG.QMSPath)QVPR\`
- `vG.QDSPPath` Qlikview Publisher Distribution Service ProgramData folder path.
Default to `C:\ProgramData\QlikTech\DistributionService\`
- `vG.DSCPath` Directory Service Connector ProgramData path
Default to `C:\ProgramData\QlikTech\DirectoryServiceConnector\`
- `vG.QVWSPPath` Path to QlikView Web Service ProgramData
Default to `C:\ProgramData\QlikTech\WebServer\`
- `vG.SAPPath` SAP Connector ProgramData path.
Default to `C:\ProgramData\QlikTech\Custom Data\QvSAPConnector\`
- `vG.SFPPath` Sales Force Connector ProgramData path.
Default to `C:\ProgramData\QlikTech\Custom Data\SalesForce\`

Settings and templates

The Settings and templates page contains all the settings for the editor. It also includes Templates that is used as a starting point when creating different container designs and/or for different Server Publisher setups. The page is divided into two themes (Container and Variable) and will look different depending on from where the *Settings and Templates* button is pressed, this is done to show only relevant settings.

Settings page from the Container Map Tab

Deployment Framework Settings 1.4.0

Admin Container is selected

Return

Clear selections

Upgrades available

ContainerMap Template	= ETL Container Example
Copy Container Map to Shared Folder	= No
Copy Container Map to All Containers	= Yes
Copy to Alt Root Path Containers	= No
Copy Variable Editor to new Containers	= Yes
Batch Mode (No containers created in Editor)	= No

Container Map Preview

Folder Name	Variable Prefix	Alt Root Path	Container Comments
0.Administration	Admin		Default Administration Container

Settings page from the Variable Editor Tab

Deployment Framework Settings 1.4.0

Admin Container is selected

Select Container = Admin

Variable File Templates = Pub and QVS Separated

Variable Files = CustomVariables

Add new Variable File =

Return

Clear selections

Upgrades available

Variable Name	Variable Value	Variable Preview	Variable Comments
---------------	----------------	------------------	-------------------

Select Container

Use this dropdown to select witch container that we should view add or modify. Default containers are *vG.BasePath* (your home container) all the other containers are found based on the container map and if the container physically exists. This setting is activated from *Variable Editor* Tab and when *Copy Container Map to All Containers* is set to No

Clear selections

This will clear earlier template selections. When applied a backup of the Container map (*1.BaseVariable\ContainerMap_Backup.csv*) will also be created.

Copy System Variables to All Containers

This setting will copy the System Variables file stored in *0.Administration* (Master) to all containers in the Master Container Map file when pressing the *Update Variables* button. This setting is by default set to yes. Turn off when having different system settings in different containers. Turn off when having different System Variables in different containers and no “single point of truth” should exist.

Copy Container Map to Shared Folders

Creates the possibility for other container users to have the Shared container as a “single point of truth” but with individual Container Maps. This setting will copy the Container map file stored in *0.Administration* (Master) to the Shared Folders container when pressing the *Update Variables* button. This setting is by default set to No because *Copy Container Map to All Containers* will override this setting.

Copy Container Map to All Containers

This setting will copy the Container map file stored in *0.Administration* (Master) to the all containers in the master container map file when pressing the *Update Variables* button. This setting is by default set to yes. Turn off when having different container layouts in different containers. Turn off when having different container layouts in different containers and no “single point of truth” should exist.

Return

Will leave the settings sheet and return to Variable or Map sheet.

Upgrade Containers

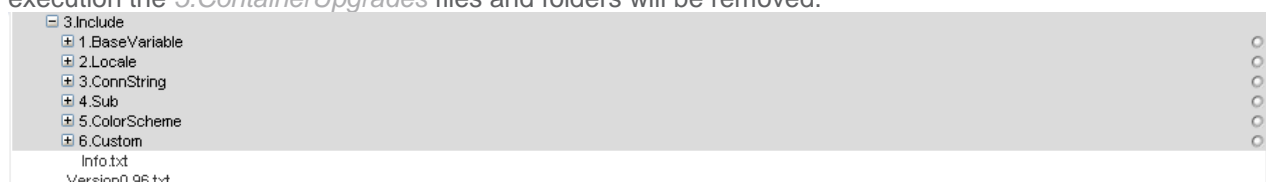
The Settings and templates sheet also contains Deployment Framework upgrade functionality and is disabled as default this function will upgrade all active containers. The QlikView Deploy Tool uses this function when updating the framework.

For manually upgrade put the files including subfolders into

0.Administration\0.Template\5.ContainerUpgrades folder. To activate the upgrade function a version text file must also be placed directly in the *5.ContainerUpgrades* folder, example *5.ContainerUpgrades\Version1.4.txt*. When this is done, reload the app and the upgrades available box will appear.



Pressing this box will lead you in to the upgrade page where all the available updates are shown an Upgrade button also exist, remember to check the upgrade File list before pressing the button. After execution the *5.ContainerUpgrades* files and folders will be removed.



Batch Mode

Running VariableEditor in Batch mode is useful when security restrictions hinder container creation via command line (cmd) in the user context. When editing and creating containers in batch mode nothing executes,

all containers end up as Orphans. VariableEditor only exports the modified ContainerMap into *0.Administration\3.Include\1.BaseVariable\ContainerMap.csv*.

The Container Creation is afterwards done by Publisher or Windows scheduler in the background using a context with higher permissions.

Limitations

Batch mode only works within the Administration container also QlikView needs to be allowed to use the *Export* Action, this to export the updated *ContainerMap.csv* into the administration container.

Using Batch Mode

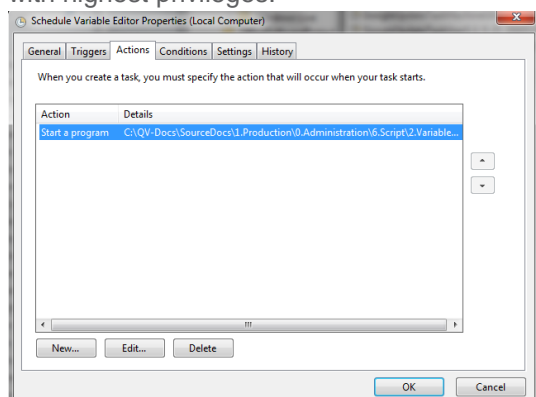
The command line switch *vL.RunBatchMode=YES* will execute VariableEditor in batch mode, Example below.

There is a Batch_Run.cmd script example in 2.VariableEditor folder that incorporates these commands.

```
"c:\Program Files\QlikView\Qv.exe" /r /vL.RunBatchMode=YES
```

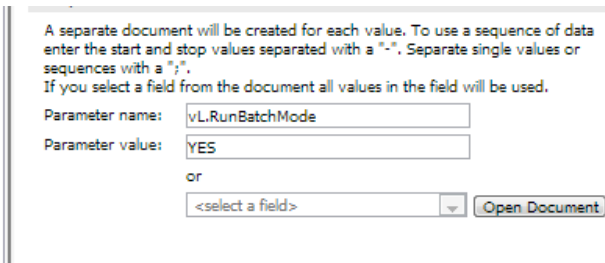
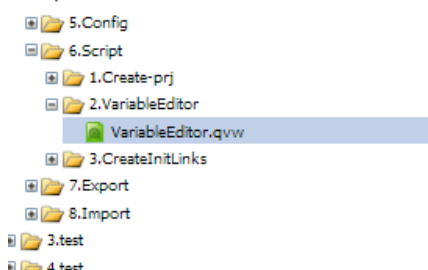
```
"\\Server\share\1.Production\0.Administration\6.Script\2.VariableEditor\VariableEditor.qvw"
```

Batch_Run.cmd script can alternatively be executed in Windows Task Scheduler, remember to run with highest privileges.



☒ Run with highest privileges

Batch Mode can also run via Publisher, create a reload task and add *vL.RunBatchMode=YES* as Script Parameter.

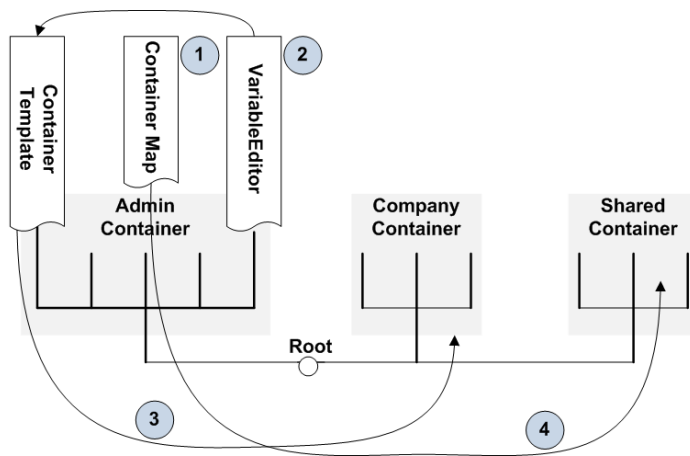


After using *vL.RunBatchMode* Variable Editor will activate the *Batch mode* setting automatically. When activated Variable Editor will not execute command line scripts, this is done purely via batch mode.

Batch mode (No container creation by Editor) = YES

To disable Batch mode open *Settings* tab and change Batch mode to NO

How does Container Map Editor work?

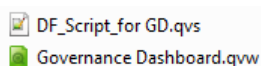


1. Master *Container Map* csv file will be updated when applying *Update Container Map*.
2. Executing *Create New container* button in Variable Editor a container template stored under *0.Template\1.ContainerTemplate* folder will be copied as a new container. Modified this template to change default Container behavior. When doing an upgrade the template will be updated together with active containers.
3. Updated container Map (read only) will be copied to all Containers
4. Updated container Map (read only) will be copied to Shared Container, this behavior can be modified in the Variable Editor Settings tab.

Governance Dashboard

Governance Dashboard needs to be installed via a separate installation package downloaded from Qlik Market and Installed under the *0.Administration\2.GovernanceDashboard* folder.

After install copy/paste the *0.Administration\2.GovernanceDashboard\DF_Script_for_GD.qvs* script snippet beside *2.GovernanceDashboard\profiles\default\GovernanceDashboard.qvw*, as shows below.



In the Governance Dashboard configuration page activate *Use Config Variable Script* and add *DF_Script_for_GD.qvs*.

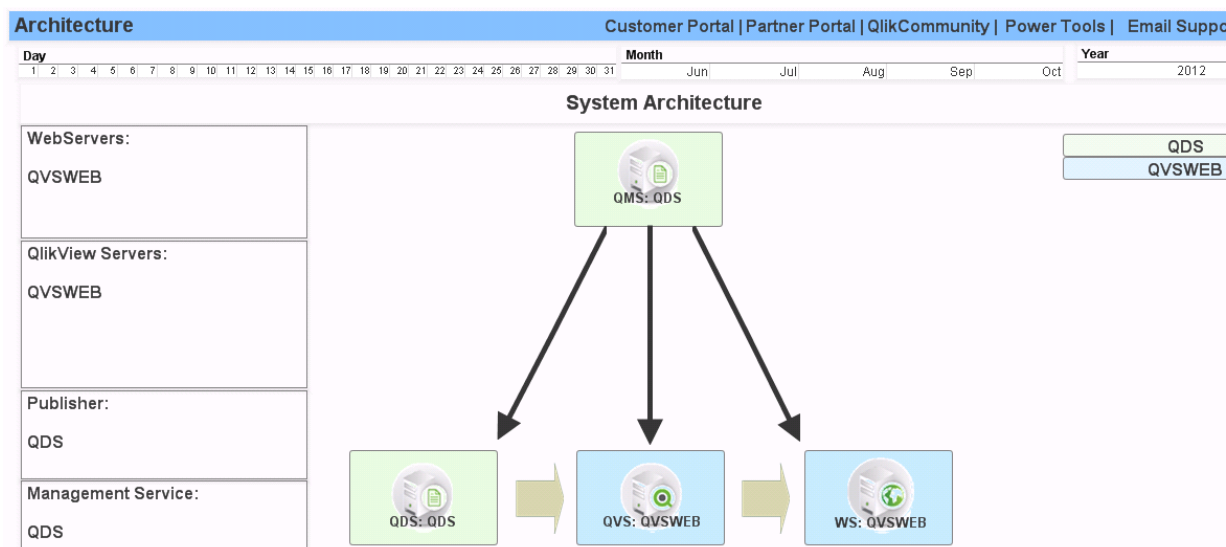


Remove all default values they will be automatically populated by Deployment Framework System Variables when executing the application. QlikView Server and Publisher log directories could be wrong, if so change global settings in Variable Editor, System Variables (read more under Variable Editor section).

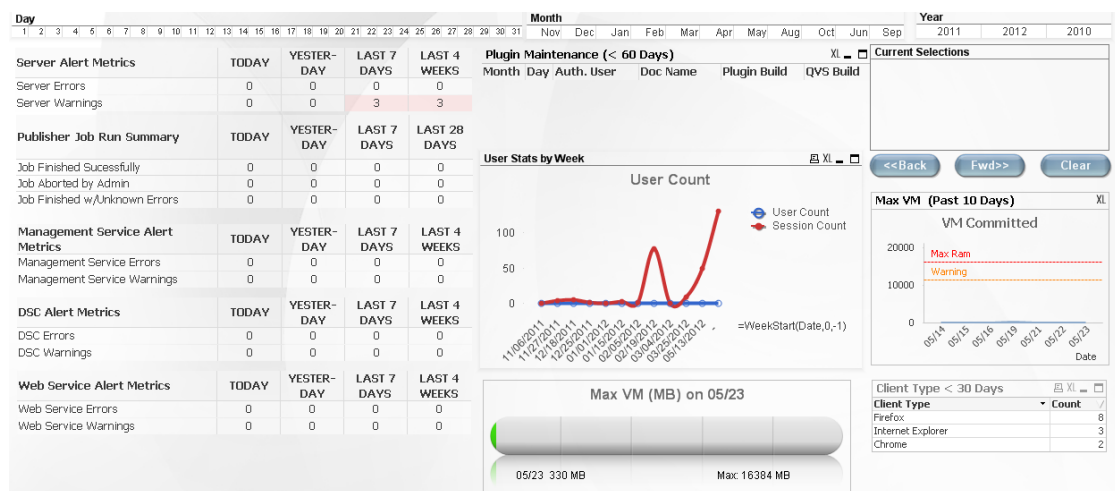
The script will also create exclusion rules for *Governance Dashboard* folder so that GD won't scan himself.

QlikView System Monitor

QlikView System Monitor combines the QV9 Operations Monitor, the QV10 Audit Log, the QV9 Server Monitoring Performance QVWs, and also now your SAP, Salesforce, DSC, Web Service, Publisher, and Management Service logs in order to work on the QV 10 platform. Many of the charts and tables contained in this app have been previously designed in other publicly issued documents.



Auto populated System architecture view



Resides in *0.Administration\1.Application\2.QlikViewSystemMonitor\QVSystemMonitor.qvw*

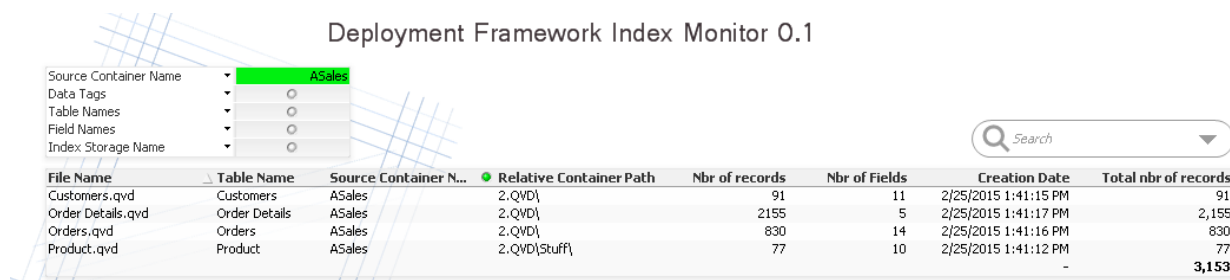
Best practice is to run System Monitor every night.

System log file path that the *QVSystemMonitor.qvw* uses is in the *3.Include\1.BaseVariable\3.SystemVariables.qvs* file under the *0.Administration* container. *3.SystemVariables.qvs* need to be configured if the QlikView components is installed on separate servers. Read more in [Deployment Framework Core](#) documentation

Qlik Index Monitor

As of QDF v1.5 QVD Index functionality has been added. An Index (by using the function IndexAdd) will be created that keeps meta information on QVD files. QVD files can be loaded by meta-data criteria's by using the function IndexLoad.

A simple Index monitor application (*0.Administration/3.IndexMonitor*) is available to monitor current Indexes available, and thereby also be able to monitor available QVD's in the environment (as long as index been added for the QVD's) Read more regarding index function in Development Guide.

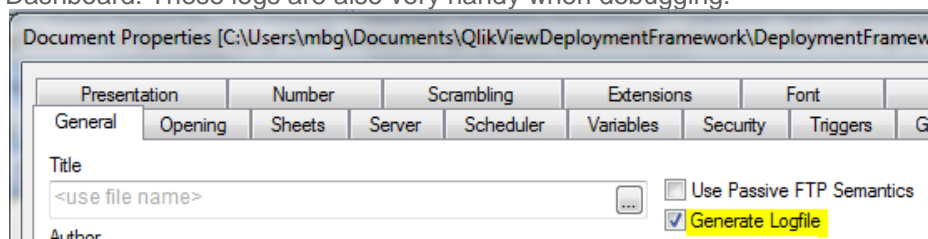


QlikView log files and locations

There are several log files generated by the QlikView platform. The best way to analyze QlikView logs is by using the QlikView System Monitor application included in the framework.

Application logging

It is best practice to turn Document logging on under Document Properties and General Tab in the QlikView Application. These logs can be used to monitor the system by use of the Governance Dashboard. These logs are also very handy when debugging.



If a script error not generated in the Deployment Framework scripts a good idea is to comment the initiation scripts and thereby using old Global Variables. The advantages of this is that the application log and debug sequence is shorter. Remember to activate DF initiation after the debugging.

QlikView Server Logs

These are the most important log files in QlikView that logs utilization license usage and memory/CPU usage. QVS logs are stored in a directory specified in QMC on the *System>Setup>Logging* tab. The default location is *%ProgramData%\QlikTech\QlikViewServer*.

If using several QlikView Servers change the location and collect the logs in a central UNC path

In the deployment framework the global variable is *vG.ServerLogPath*