



Reference Manual

Version 10.0 for Microsoft Windows®

First Edition, Lund, Sweden, October 2010

Authored by QlikTech International AB JNN/LAA/CEN

Copyright © 1994-2010 Qlik®Tech International AB, Sweden.

Under international copyright laws, neither the documentation nor the software may be copied, photocopied, reproduced, translated or reduced to any electronic medium or machine-readable form, in whole or in part, without the prior written permission of QlikTech International AB, except in the manner described in the software agreement.

Qlik®Tech and Qlik®View are registered trademarks of QlikTech International AB.

Microsoft, MS-DOS, Windows, Windows NT, Windows 2000, Windows 2003, Windows XP, Windows Vista, SQL Server, FoxPro, Excel, Access and MS Query are trademarks of Microsoft Corporation.

IBM, AS/400 and PowerPC are trademarks of International Business Machines Corporation.

Borland, Paradox, Delphi and dBASE are trademarks of Borland International.

ORACLE and SQL*Net are trademarks of Oracle Corporation.

MacOS is a trademark of Apple Corporation.

CONTENT

PART I: USING THE QLIKVIEW OCX CONTROL

1	INTRODUCTION	7
1.1	About this documentation	7
1.2	Disclaimers	7
2	VERSION HISTORY	9
3	INSTALLING AND USING THE QLIKVIEW OCX CONTROL	11
3.1	About the QlikView OCX control	11
3.2	Contents of the QlikView OCX SDK Package	11
3.3	Installing QlikView OCX for use	12
3.4	Installing required library files	13
3.5	Installing additional Microsoft components	14
3.6	Copying and registering the QlikView OCX component	14
4	INCLUDING QLIKVIEW OCX IN VB, VC++ OR DELPHI PROJECTS ...	17
4.1	Using the QlikView OCX component from Visual Basic	17
4.2	Using the QlikView OCX component from Visual C++	19
4.3	Using the QlikView OCX component from Delphi	26
4.4	Run-time licensing of the QlikView OCX component	34
4.5	Registration free deployment of the QlikView OCX component	35
5	USING AUTOMATION WITH QLIKVIEW	37
5.1	The QlikView/QlikView OCX Automation Interface	37
5.2	How Automation can control QlikView/QlikView OCX	38
6	COMMENTS TO MEMBER DESCRIPTIONS IN PART II	41
6.1	Description of method and event members	41
6.2	Description of property members	42

PART II: QLIKVIEW OCX SPECIFIC AUTOMATION MEMBERS

7	QLIKVIEW OCX SPECIFIC METHODS	45
----------	--	-----------

7.1 Method summary	45
7.2 Method descriptions	45
8 QLIKVIEW OCX SPECIFIC PROPERTIES	49
8.1 Properties summary	49
8.2 Property descriptions	50
9 QLIKVIEW OCX SPECIFIC EVENTS	57
9.1 Events summary	57
9.2 Event descriptions	58

PART I: USING THE QLIKVIEW OCX CONTROL

1 INTRODUCTION

1.1 About this documentation

General

This documentation provides a systematic description of the QlikView OCX control.

Programming languages

The host application in which the QlikView OCX component is embedded can be written e.g. in Visual Basic or Visual C++. This document is however not a manual for VB or VC++. The reader is supposed to possess a basic knowledge of VB or VC++ programming.

Style coding

In this documentation all menu commands and dialog options are shown in Arial bold. All file names and paths are shown in Courier NewBold. Sample VB and VC++ code is shown in Courier New and Courier New Bold.

1.2 Disclaimers

- Specifications are subject to change during the continued development of QlikView products.
- As changes are made to the QlikView/QlikView OCX type library in future releases, programs using typed object calls will have to be recompiled with the new type library in order to function with the new release.
- The QlikView Automation interface lets you manipulate QlikView objects on a very deep internal level. Incorrect use of functionality may cause program errors.

2 VERSION HISTORY

What's new in version 7.0 - 7.2

No OCX specific major features were added in these versions.

What's new in version 7.5

The following OCX specific major features have been added with release version 7.5.

- Starting from release 7.5 the QlikView OCX control has been unified into one single file named **QlikOCX.ocx**. This file serves as all three levels of QlikView OCX. Which level you will run is determined by the license serial number you receive with your QlikView OCX development package. In order to determine license level, the developer OCX will contact the Qlik-Tech LEF server when first activated.

What's new in version 7.51

The following OCX specific major features have been added with release version 7.51.

- The QlikView Analyzer for IE plug-in client and the QlikView OCX embeddable component have been unified into one single file.
- QlikX objects (single sheet objects shown in an ActiveX container) are now available also with the QlikView OCX embeddable component.
- The QlikView OCX component is now also available in a 64-bit version (x64).

What's new in version 7.52

The following OCX specific major features have been added with release version 7.52.

- Support for firing events to the hosting page or application has been added. The OnDataChanged event fires whenever data or selections changes in a QlikView document. The “**OnQvEvent**” event can be fired by means of macros in the QlikView document.

What's new in version 8.0

There is no OCX specific major new features added with release version 8.0 but several new general features for all QlikView product types. For details, see the QlikView Reference manual.

What's new in version 8.2

The following OCX specific major features have been added with release version 8.20.

- Support for customizing Help by means of the **OnHtmlHelp** event. This event is similar in structure to the **OnMenuCommand** event.
- Simplification of deployment by introduction of a “ProductLevel” attribute that allows an Developer OCX Development license to generate all types of run-time licenses.

What's new in version 8.5

The following OCX specific major features have been added with release 8.5:

- Drag and drop from QlikView to Microsoft Office documents.
- Print support for OCX embedded in Microsoft Office documents.
- Registration free deployment of OCX.
- **LockPaint** and **UnlockPaint** methods exposed on OCX to allow more direct screen updates.
- **OnCommand** event (similar to **OnMacro**) to allow host application to restrict access to functionality that should not be exposed.

What's New in Version 9.0

No OCX specific major features were added in this version. Developers should consider the implications of QlikView Personal Edition on programs developed with QlikView OCX without the Developer license. For example, the `Application.GetRunlevel` method is now obsolete and demo applications intended for off-line access may require an embedded license when the end-user license level is Personal Edition.

What's New in Version 10

No OCX specific major features were added in this version.

3 INSTALLING AND USING THE QLIKVIEW OCX CONTROL

3.1 About the QlikView OCX control

What is QlikView OCX?

QlikView OCX or the QlikView OCX control is an ocx component containing the QlikView program for embedding into host application programs developed by software manufacturers on an OEM basis.

General properties of the QlikView OCX control

The QlikView OCX controls show the sheet area of QlikView, i.e. the sheet and tab row of a QlikView document.

Menu bars, tool bars and the status bar are not provided by the QlikView OCX control. These must, if desired be programmed in the host application.

All UI functionality normally found within the QlikView sheet area is available as normal in the OCX control. This includes context menus, although the OCX interface contains possibilities to disable, modify and replace them. Starting in version 8.20, help can also be disabled and modified.

Basically all QlikView functionality can be controlled programmatically by the host application by means of the general QlikView Automation interface and the specific OCX Automation interface described in part II of this document.

3.2 Contents of the QlikView OCX SDK Package

The QlikView OCX SDK Package has the following content:

- QlikView OCX Reference manual.pdf is a PDF version of this document. By default installed in the “**Program files\QlikView\documentation**” folder.
- QlikView Reference manual is available for each language that QlikView is available. By default installed in the “**Program files\QlikView\documentation**” folder.

-
- **QlikOCX.ocx** is the ocx component. By default installed in the “**Program files\QlikView\QlikOcx**” folder, along with the language dlls and compiled html help files.
 - Runtime library is a directory containing all library files and Microsoft installation programs necessary for developing a complete the QlikView OCX application.
 - **APIGuide.qvw** is the detailed QlikView Automation interface documentation. By default installed in the “**Program Files\qlikview\documentation**” folder.
 - QlikView Automation Reference.pdf is the general QlikView Automation interface documentation. By default installed in the “**Program files\QlikView\documentation**” folder.

3.3 Installing QlikView OCX for use

Before the QlikView OCX control can be used on a computer the following steps must be taken:

- 1 A number of runtime library files must be available the target computer. See below for details on how to do this.
- 2 In some cases certain Microsoft installation programs must be executed. See below for details.
- 3 The QlikView OCX file must be copied to the target computer and registered. See below for details on how to do this.
- 4 If you are designing a program embedding the QlikView OCX control you will have to register your license serial number for the OCX when you first add it to your project. This will also create a run-time license in the host program, which makes the QlikView OCX component run on end-user computers as part of that program without further license registration.
- 5 A host application must be designed or installed as an EXE on the target computer. See below for details on how to access the QlikView OCX component from Visual Basic or Visual C++.
- 6 In most cases a QlikView document must be copied to or be made available from the target computer. The QlikView document is normally developed using the standard QlikView program.

3.4 Installing required library files

Below you find three short descriptions of three different ways of getting all necessary runtime files onto the target computer. All three methods are based on the Microsoft Windows Installer standard, which is the recommended deployment technique.

Using the QlikView OCX Merge Modules

(QlikOCX.msm for 32 bit systems and QlikOCXx64.msm for x64 systems)

To get the QlikOcx component properly installed and registered with your application msi installation package you can use the merge module provided in the Runtime library folder and merge it into your package. It is recommended, not required, that you use the QvpDll.msm merge module along with the QlikOcx merge module to enable the possibility to use server side QlikView documents in your application.

For more information about merge modules and how to use them, see Microsoft's homepage and your msi development tool provider's homepage.

From version 7.52 there are no longer any automatic dependencies to other merge modules.

Using the QlikView OCX Runtime library

(QvOCXRTL.exe, QvOCXRTL.msi, QvX64OCXRTL.exe or QvX64OCXRTL.msi)

This is a complete installation of the QlikView OCX, but without documentation, complementary SDK files and with less dialogs. Use any of these installation packages if you want to deploy the QlikOcx component separately from your own application.

The exe file is intended for computers that do not have the latest version of the Windows Installer Service. The msi can only be used on computers with Windows Installer Service 3.0 or higher installed.

Creating a custom msi package

If you want to create an msi installation package for your application, the following merge modules and runtime file has to be included in the package additional to the **QlikOcx.ocx**, language resource dll files and compiled html files (*.chm):

Module/File name	Type	Description	Version
------------------	------	-------------	---------

MSXML	Merge module	MSXML 4.0	4.00.9004.0
GDIplus	Dll file	Microsoft GDI+	dll*5.50.4134.600

The merge modules and runtime files can be found in the **Runtime library**-folder, installed by the QlikView OCX SDK Package. For detailed information please see the **ReadMe.txt** located in the same folder.

There are also a number of registry files (*.reg) available in the **Runtime library**-folder, these are examples on how to specify the correct interface for your QlikOcx component.

*The GDIplus dll is available as a cabinet self-extracting executable, to make sure you use the latest, please consult Microsoft's homepage. Also make sure you follow Microsoft's recommendations and requirements on how to install this file.

3.5 Installing additional Microsoft components

OLE DB support

If OLE DB database connections are to be used it is recommended that the proper Microsoft Data Access Components, the MDAC, are used. To obtain the proper version, please go to Microsoft's homepage. The Microsoft Data Access Components are normally already installed on computers with a reasonably modern version of Windows and Microsoft Office.

3.6 Copying and registering the QlikView OCX component

Copying the QlikView OCX control to the target computer

The QlikView OCX component (**QlikOCX.ocx**) must first be copied to the target computer. The OCX file may be put in any location on the disk.

The OCX file must then be registered by Windows before it can be used. Registration of OCX and DLL files is done by the program **Regsvr32.exe**, which can be found in the **windows\system** or **windows\system32** directory.

After registration the OCX file may not be moved without re-registering it.

OCX registration for development purposes

If you are unfamiliar with registering OCXs follow the steps below:

- 1 Double-click on the OCX file.
- 2 Unless files with OCX extension are already associated with **Regsvr32.exe** on your computer the Windows dialog Open with will appear.
- 3 Click Other and browse your way to the **windows\system** or **windows\system32** directory.
- 4 Select **Regsvr32.exe** and click **Open**.
- 5 After a few seconds a message will be shown indicating that the registration succeeded.

OCX registration for commercial purposes

If the QlikOcx has not been properly registered by the installation package, you might have to do it manually. Registration can be made from a command-line statement like:

```
regsvr32 /s "C:\Mypath\qlikocx.ocx"
```

where **C:\MyPath** is replaced by the relevant path the **QlikOcx.ocx** file on your target computer. This operation can also be performed by any commercial installation utility software. The **/s** modifier makes the registration run in silent mode, i.e. no messages are displayed.

4 INCLUDING QLIKVIEW OCX IN VB, VC++ OR DELPHI PROJECTS

4.1 Using the QlikView OCX component from Visual Basic

Including the control in a VB.NET project

The QlikView OCX control must first be included in the Visual Basic project. This is done as follows:

- 1 Right-click on the toolbox in Visual Basic.
- 2 Select **Components** on the **Context** menu.
- 3 Scroll down the list until you find QlikView OCX ActiveX control module. If it does not appear in the list, the QlikView OCX control has not been properly registered (see previous section).
- 4 Select that component by checking the check box to the left of it.
- 5 Click **OK**. A QlikView icon should now appear in the toolbox.

Placing a QlikView OCX control on a VB form

You may now include the QlikView OCX control on a form in your VB project.

- 1 Click the QlikView icon in the toolbox.
- 2 Use the mouse to mark the size and position of the QlikView OCX control on the form.

Note At the moment only one QlikView OCX control may be used in one project.

Accessing the QlikView OCX control from VB code

Once the component has been included and placed on a form it may be referenced as an object by the name **QlikOCX1**.

The active QlikView document can be referenced as **QlikOCX1.ActiveDocument**.

In addition to the QlikView OCX specific Automation members listed in Part II below, almost all members of the standard QlikView Automation

interface may be used for control of the OCX component. See the QlikView Automation Reference manual for details.

Note that in cases where member names are shared between the general QlikView Automation interface and the QlikView OCX specific Automation interface, the latter will have preference. The actual difference is however negligible.

Examples :

```
Private Sub Command1_Click()  
    QlikOCX1.OpenDocument "C:\MyDoc", "", ""  
End Sub  
  
Private Sub Command2_Click()  
    QlikOCX1.ActiveDocument.Reload  
End Sub
```

Useful VB support modules

If you need to use QlikView internal constant definitions, include the module file `qvmenu.bas` to your project. The file which can be found in the `\Run-time library\Resources\Vb` directory installed by the QlikView OCX SDK Package contains QlikView internal menu constant definitions. This file will be updated together with the OCX control.

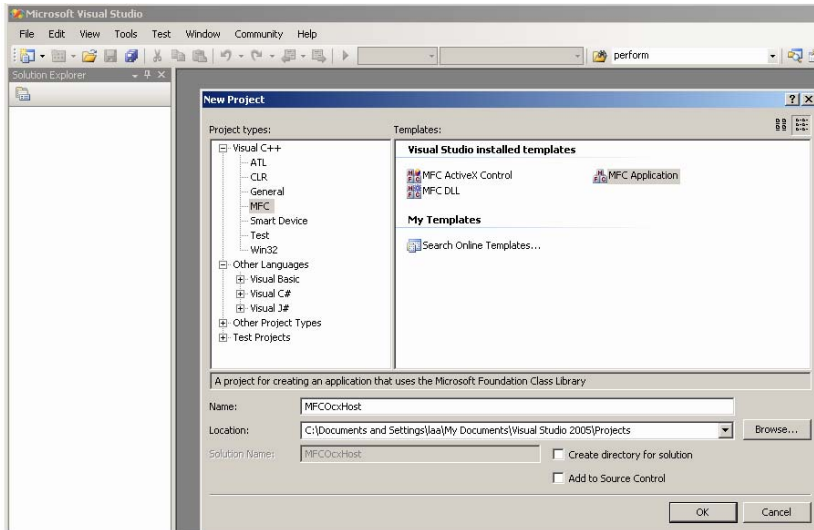
If you plan to manipulate the QlikView object menus the module file `MenuDef.bas` may be quite useful. It contains declarations of system calls for manipulating menus.

```
Rem MenuDef.bas  
Public Const MF_SEPARATOR = 2048  
Public Const MF_STRING = 0  
Declare Sub AppendMenu Lib "user32" Alias  
    "AppendMenuA" (ByVal _  
        hMenu As Long, ByVal uFlags As Long, ByVal uIDNewItem  
        As Long, _  
        ByVal lpNewItem As String)  
Public Const MF_BYCOMMAND = 0  
Public Const MF_BYPOSITION = 1024  
Declare Sub DeleteMenu Lib "user32" (ByVal hMenu As  
    Long, _  
    ByVal uPosition As Long, ByVal uFlags As Long)  
Declare Function GetMenuItemCount Lib "user32" _  
    (ByVal hMenu As Long) As Integer
```

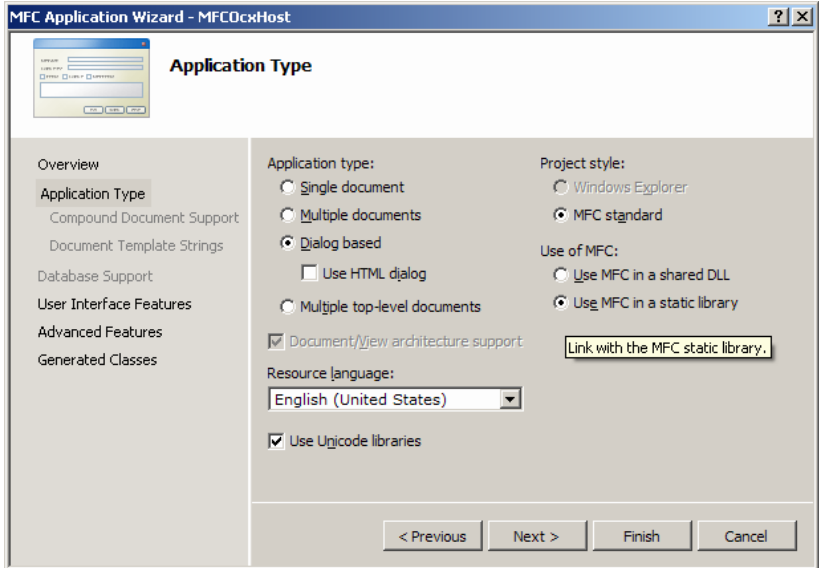
4.2 Using the QlikView OCX component from Visual C++

Including the control in a VC++ project

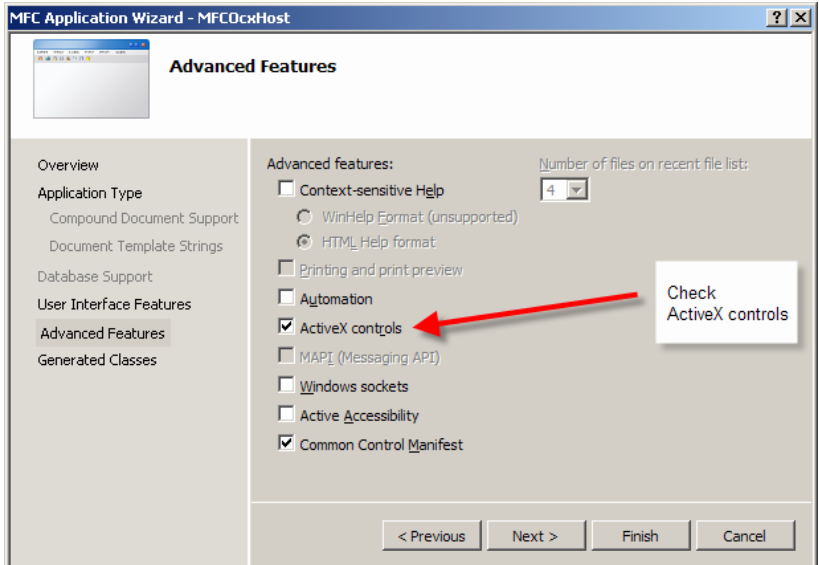
This step-by-step instruction is based on adding the Qlikview OCX to a Visual C++ dialog application. It is intended as a complement to the fuller tutorials targeting C# and Office integration, so it only describes the special steps necessary to make a working solution in Visual C++.



Select **New Project** and **MFC Application**.

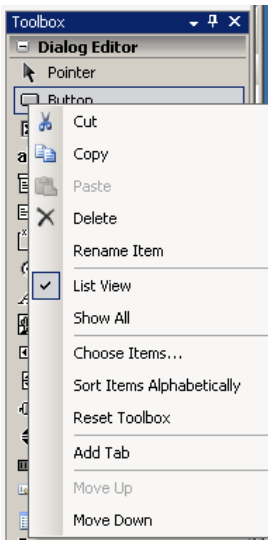


Make a **Dialog based** application.

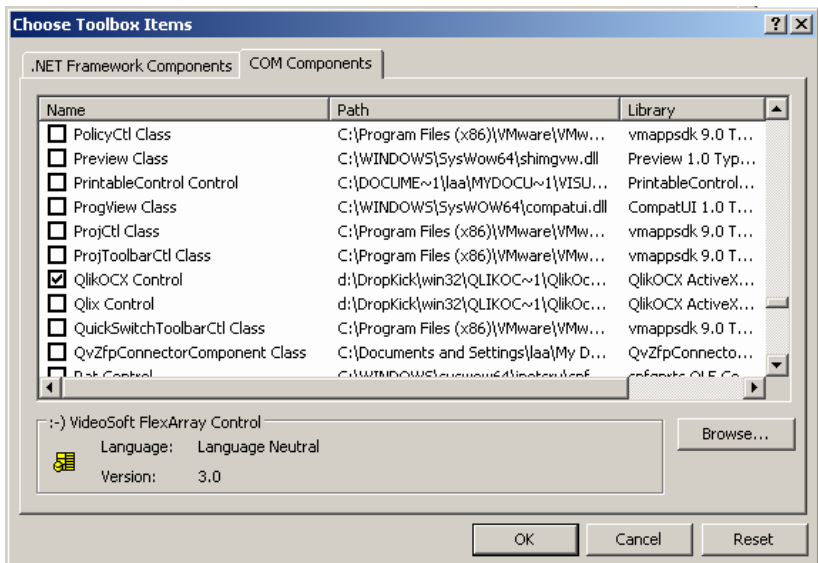


Make sure you include support for **Active X controls**.

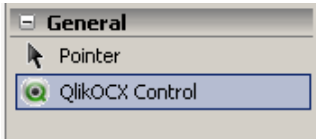
Open the application dialog in C++ resource view and select **QlikOCX control** into the toolbox by right clicking and selecting **Choose Items...**



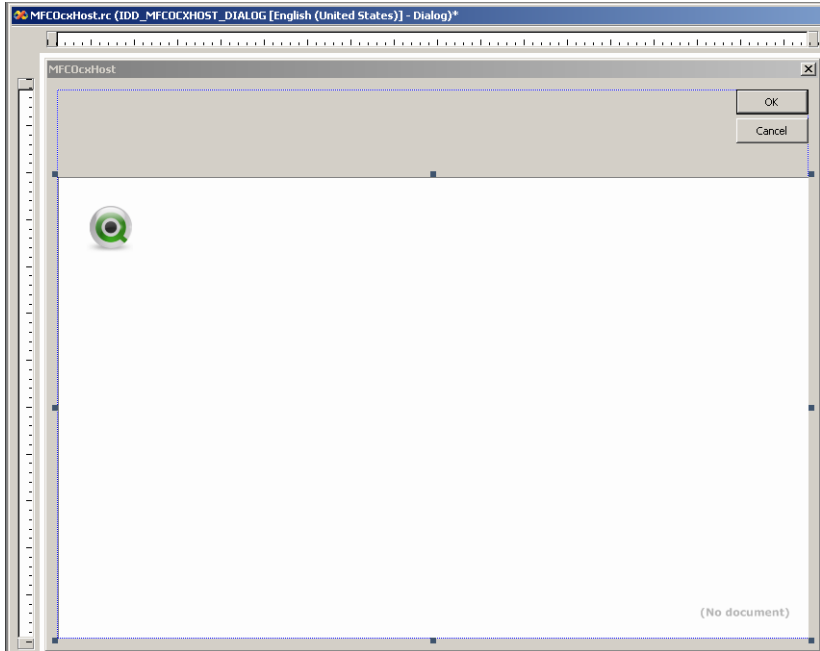
Select **QlikOCX Control**.



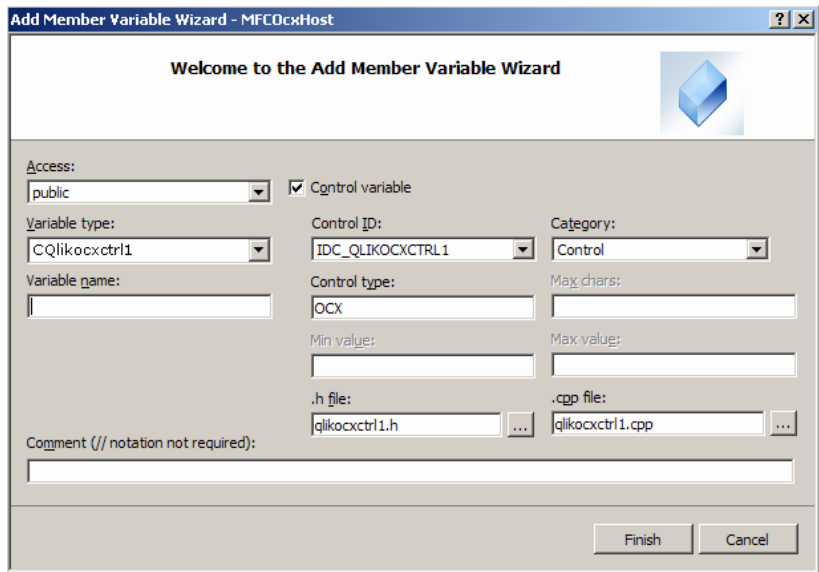
Press **Ok** and then select the **QlikOCX control** in the toolbox.



Drag it to the dialog and size it appropriately.



Compile the application and then ctrl-double click on the OCX control in the resource view of the dialog.



Name the variable (e.g. m_QlikOCX1) and press **Finish**.

Try to compile. You will get many compilation errors due to the following lines in the Wizard-generated code that was added.

```
Global * GetApplication()
{
    Global * result;
    GetProperty(0x1, VT_DISPATCH, (void*)&result);
    return result;
}
void SetApplication(Global * propVal)
{
    SetProperty(0x1, VT_DISPATCH, propVal);
}
Doc * GetActiveDocument()
{
    Doc * result;
```

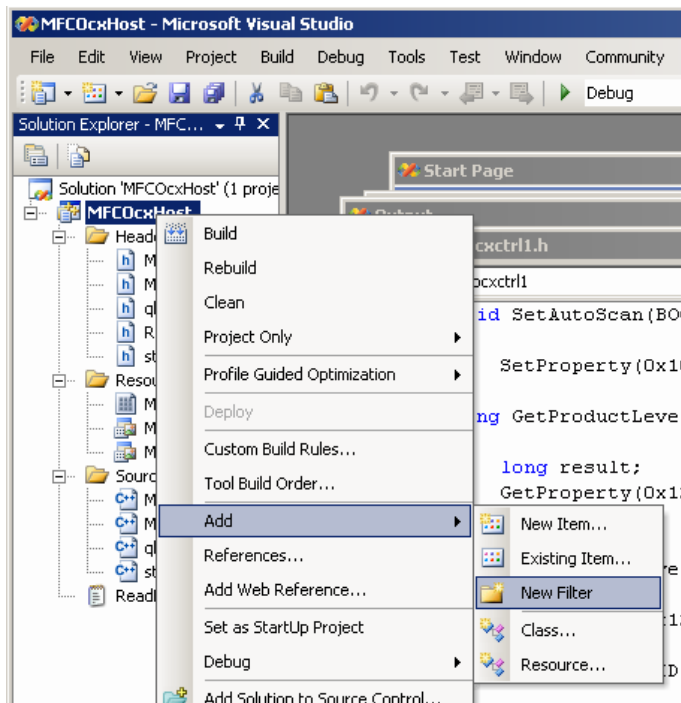
```

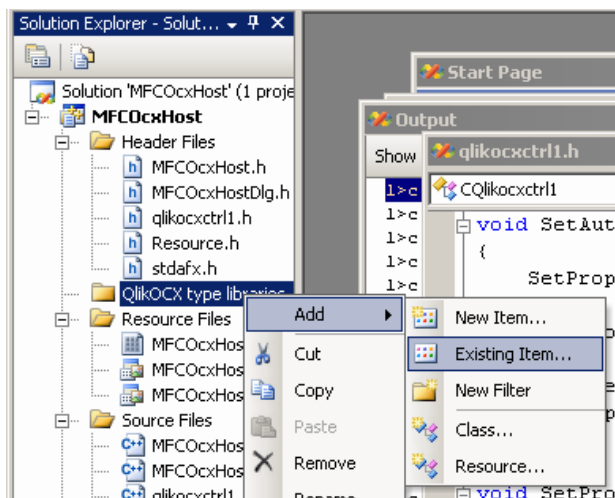
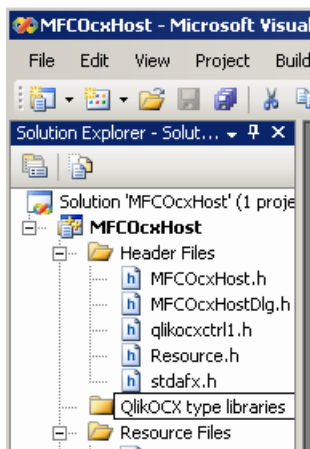
        GetProperty(0x2, VT_DISPATCH, (void*)&result);
        return result;
    }

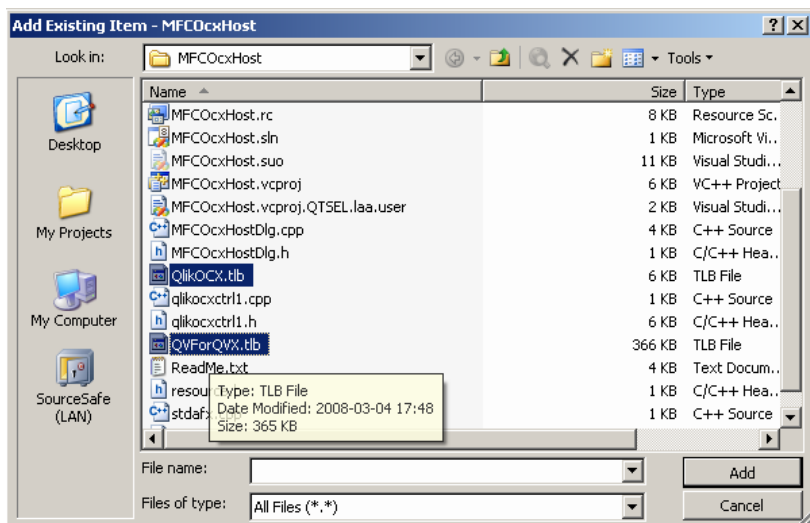
    void SetActiveDocument(Doc * propVal)
    {
        SetProperty(0x2, VT_DISPATCH, propVal);
    }

```

The reason for the compilation errors is that **Global** and **Doc** are definitions that belong to the QlikView API. These definitions reside in a different type library and Visual C++ does not handle the situation very well. In order to avoid the problem and also to be able to use the API more efficiently, we recommend that you **#import** the type libraries into the C++ project. Copy the appropriate ones to your project, add a filter to hold them and add them to the project using the following sequence of actions.







When you have added the type libraries, edit `stdafx.h` and add the following lines at the end:

```
#import "QvForQVX.tlb" no_namespace rename("GetLocaleInfo", "QvGetLocaleInfo")

#import "QlikOCX.tlb" no_namespace
```

The second import is not necessary, but being able to access code in both type libraries in the same way is convenient. The rename directive for the first import removes a warning.

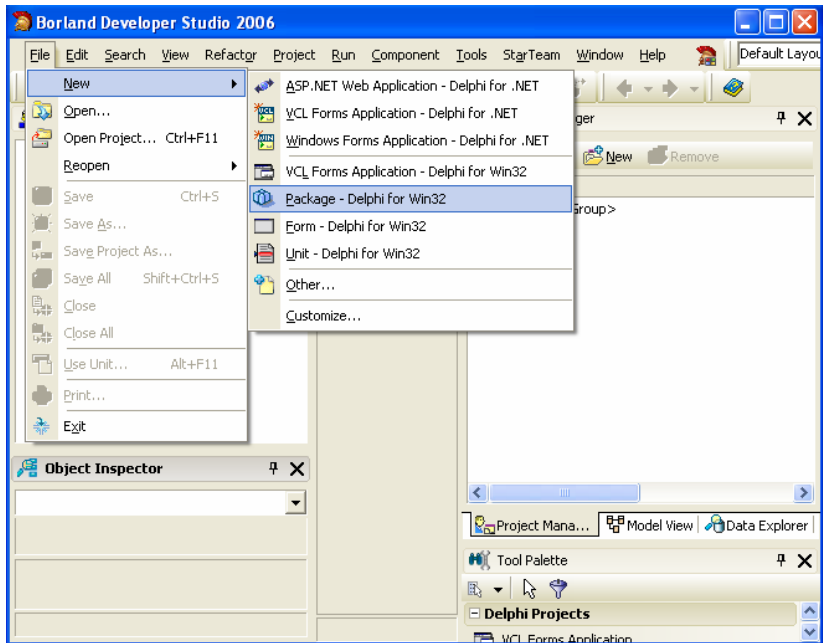
The compilation should now succeed.

4.3 Using the QlikView OCX component from Delphi

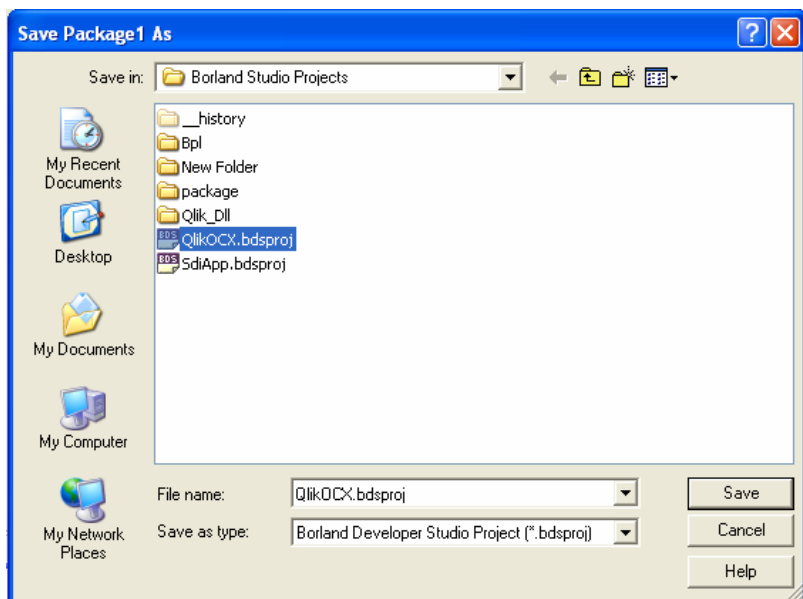
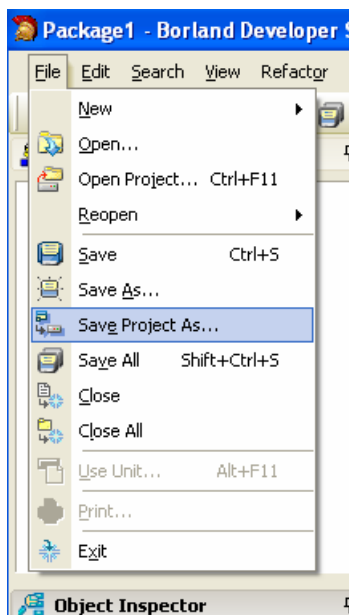
With Borland Delphi you need to perform some steps to be able to use the QlikView OCX (QlikOCX) in the applications you develop. To complicate things a bit more we have found that the runtime license for the OCX is not correctly handled in the Delphi development system, so this document describes the necessary steps to get a QlikOCX that you can drag into your projects and at the same time getting the unit files for the OCX to include the correct runtime license.

Step 1: Create a new package for the component

From the menu, **new**, select **File/Package – Delphi for Win32**.

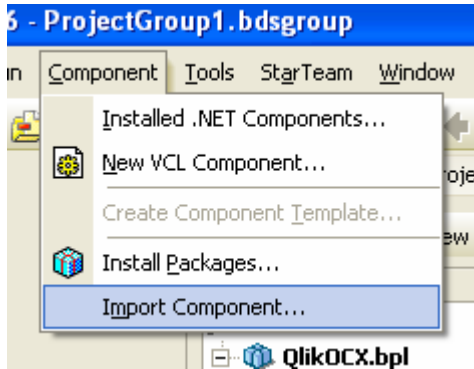


Rename and save the package using **File/Save Project As**.

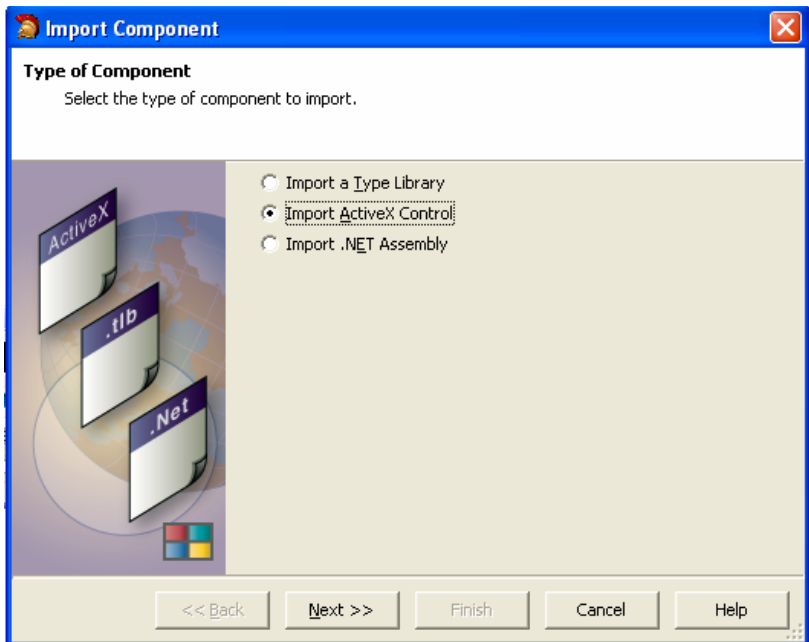


Step 2: Import the component

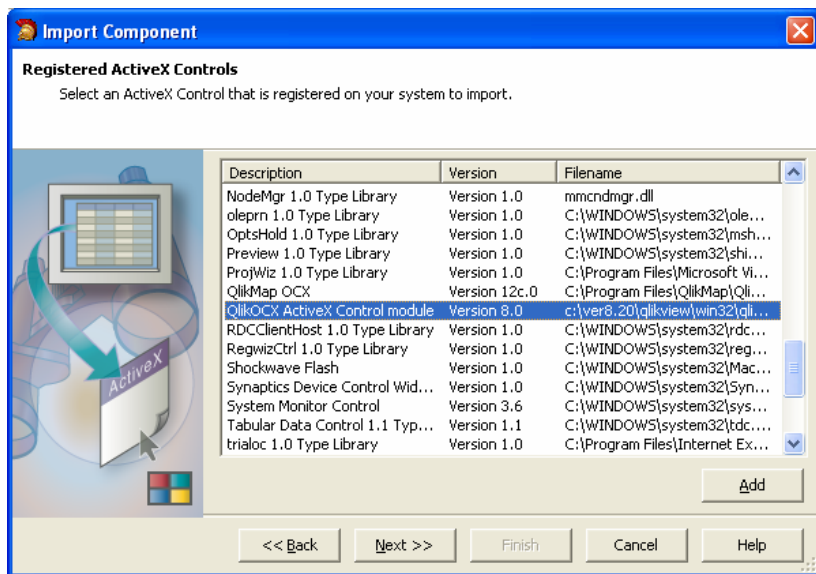
From the menu, select **Component/Import Component**.



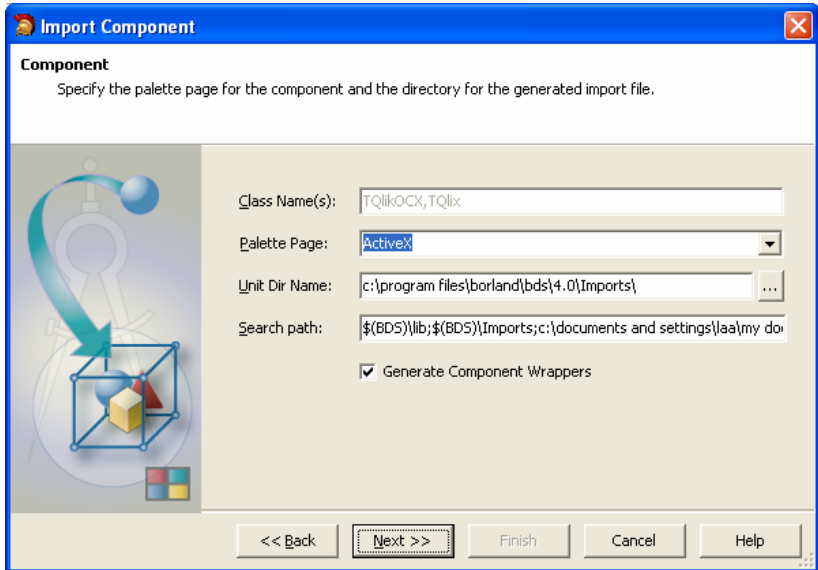
Choose the option **Import ActiveX Control** and click **Next**.



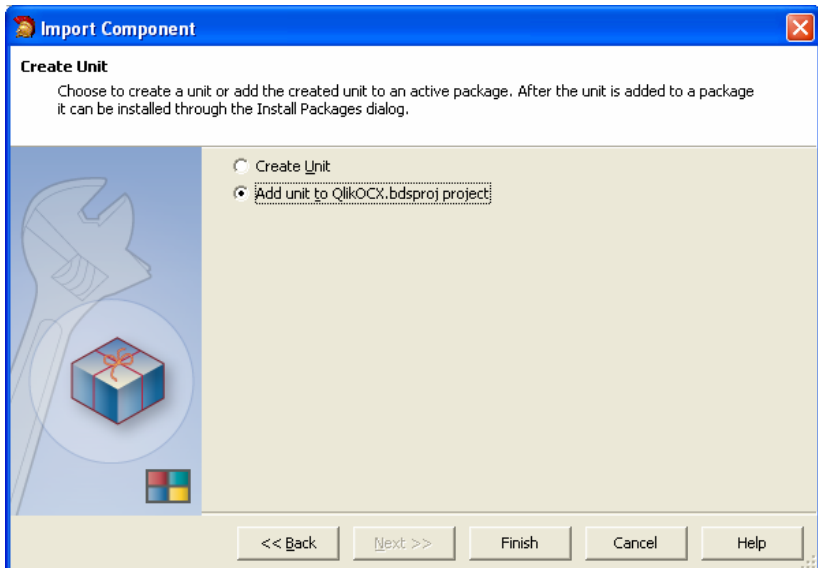
A list of registered ActiveX controls is now displayed. Select the required control and click **Next**.



Click **Next** again. The palette page can be changed if required, which will determine the **Category** in the **Tool Palette** that is used, but it is recommended to keep the default Category of ActiveX.

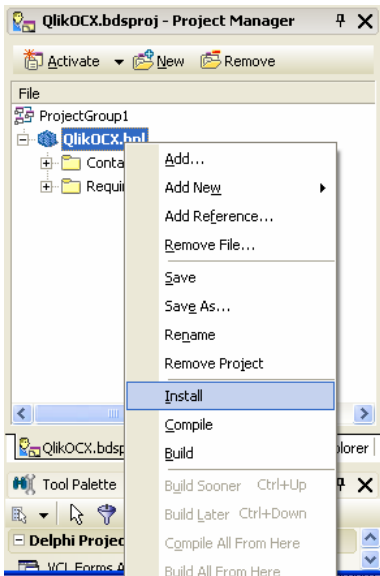


Select **Add unit to [package name] project** and click **Finish**.



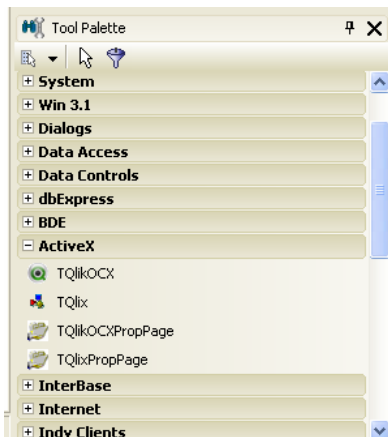
Step 3: Install the package

In the **Project Manager** pane, right-click on the package name and select **Install**. Delphi should now display a message confirming that the package is installed. Click **OK**. Save the package using **File/Save**.



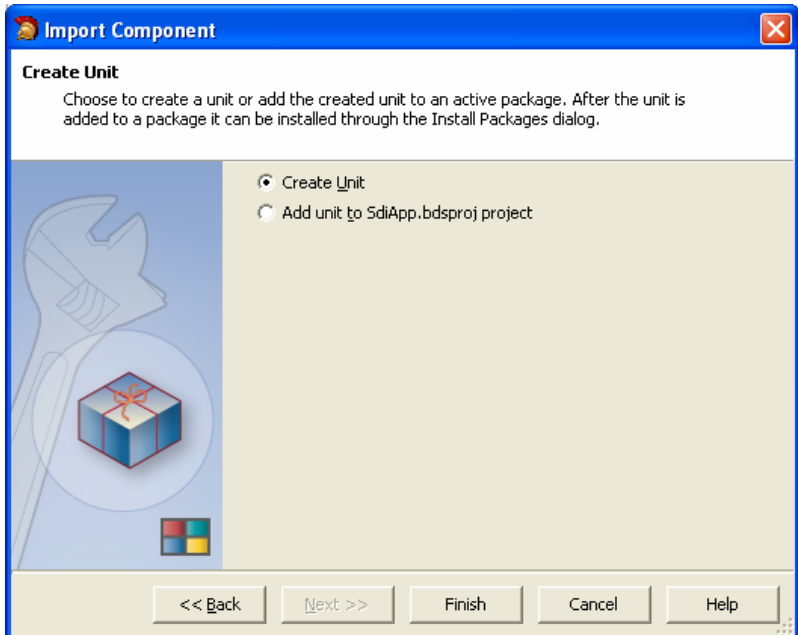
Step 4: Check that the ActiveX control is installed

Open an existing application, or start a new one, so that the Tool Palette is displayed. The newly installed control should be available for use in the **ActiveX** Category.



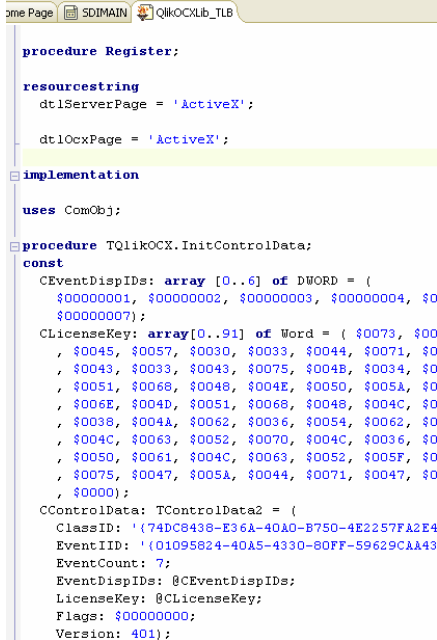
Step 5: Re-import the component to get a correct runtime-license embedded

Perform all the operations under step 2 except for the final one and select **Creat unit** instead.



Verify that the LicensKey is extracted correctly by searching for **InitControlData** in **QlikOCXLib_TLB**.

The data should look something like this



```
procedure Register;

resourcestring
  dtlServerPage = 'ActiveX';
  dtlOcxPage = 'ActiveX';

implementation

uses ComObj;

procedure TQlikOCX.InitControlData;
const
  CEventDispIDs: array [0..6] of DWORD = (
    $00000001, $00000002, $00000003, $00000004, $0
    $00000007);
  CLicenseKey: array[0..91] of Word = ( $0073, $00
    , $0045, $0057, $0030, $0033, $0044, $0071, $0
    , $0043, $0033, $0043, $0075, $004B, $0034, $0
    , $0051, $0068, $0048, $004E, $0050, $005A, $0
    , $006E, $004D, $0051, $0068, $0048, $004C, $0
    , $0038, $004A, $0062, $0036, $0054, $0062, $0
    , $004C, $0063, $0052, $0070, $004C, $0036, $0
    , $0050, $0061, $004C, $0063, $0052, $005F, $0
    , $0075, $0047, $005A, $0044, $0071, $0047, $0
    , $0000);
  CControlData: TControlData2 = (
    ClassID: '{74DC8438-E36A-40A0-B750-4E2257FA2E4
    EventIID: '{01095824-40A5-4330-80FF-59629CAA43
    EventCount: 7;
    EventDispIDs: @CEventDispIDs;
    LicenseKey: @CLicenseKey;
    Flags: $00000000;
    Version: 401);
```

Congratulations! You now have an installed QlikView OCX in Delphi.

4.4 Run-time licensing of the QlikView OCX component

When designing a host program embedding the QlikView OCX control you will have to register your development license serial number for the OCX when you first add it to your project.

A run-time license based on the same serial number will then be embedded in the host program, which makes the QlikView OCX component run on end-user computers as part of that program without further license registration.

Should the end-user attempt to use the QlikView OCX component for other use than with the purchased host program, the OCX control will prompt for a license serial number.

The registration free deployment option that was introduced in version 8.5 works particularly well in combination with run-time licensing. This option allows the host program to load the same version of the QlikOCX component independently of other versions registered on the machine.

4.5 Registration free deployment of the QlikView OCX component

Registration free deployment of the QlikView OCX is accomplished by the host program calling the DLL entry point **StartRegFreeOperation**. Calling this entry point will allow the OCX to work with no additional registration. In C# it looks like this:

```

using System;
    using System.Collections.Generic;
    using System.Windows.Forms;

    using System.Runtime.InteropServices;

    public class QlikOcxFunctions
    {
        [DllImport("QlikOcx.ocx", EntryPoint =
"StartRegFreeOperation")]
        public static extern void StartRegFreeOperation();
    }

    namespace DropKickDemo3
    {
        static class Program
        {
            /// <summary>
            /// The main entry point for the application.
            /// </summary>
            [STAThread]
            static void Main()
            {
                if (MessageBox.Show ("Do you want to run
registration free?", "DropKickDemo3",
MessageBoxButtons.YesNo) == DialogResult.Yes) {

QlikOcxFunctions.StartRegFreeOperation();
                }
                Application.EnableVisualStyles();

                Application.SetCompatibleTextRenderingDefault (false);
                Application.Run (new Form1());
            }
        }
    }

```

5 USING AUTOMATION WITH QLIKVIEW

5.1 The QlikView/QlikView OCX Automation Interface

About Automation

QlikView and QlikView OCX is equipped with an Automation interface (Automation was previously known as OLE Automation). This interface allows an external program or internal macro to access and control the QlikView application / QlikView OCX control.

Description of the general QlikView Automation interface

The general QlikView Automation interface is described in detail in the QlikView API guide (**APIguide.qvw**), found on the Documentation folder installed from the QlikView OCX SDK Package. This interface may be use in full when working with QlikView OCX. In addition to these Automation members, specific QlikView OCX members are described in part II of this document.

Invocation of QlikView/QlikView OCX Automation

The Automation interface is an integral part of QlikView/QlikView OCX and you do not have to perform any special tasks to activate it. The QlikView/QlikView OCX type library is contained in the QlikView OCX control.

Multiple Automation interface versions

The QlikView OCX type library and the QlikView type library are registered as separate interfaces in the Windows registry. This means that a version of full QlikView which differs from the installed QlikView OCX can still be used with full Automation functionality.

Only one version of the QlikView OCX Automation interface can be installed on a computer.

5.2 How Automation can control QlikView/ QlikView OCX

External control of QlikView/QlikView OCX

QlikView objects are accessible by means of Automation from external programs,

E.g. programs written in Visual Basic or C++ supporting Automation.

Such code can be used to control QlikView from other applications or from stand-alone programs.

Stand-alone executable files can be invoked from a QlikView document by means of launch buttons.

QlikView events

A number of predefined events may be used to trigger a macro in the QlikView/QlikView OCX internal VBScript module. All these events also trigger the OnMacro event in QlikView OCX, which can be picked up and acted upon by the host application.

The events are:

Document events (Qvw documents):

- 1 On opening a QlikView document.
- 2 On script re-execution.
- 3 On the Reduce Data command.
- 4 On a selection in any field in the document.

Sheet events

- 5 On sheet activation.
- 6 On sheet deactivation.

Sheet object events

- 7 On sheet object activation.
- 8 On sheet object deactivation.

Button events

- 9 On pressing a macro button in the layout.

Field events

- 10 On selection in a specified field.

- 11 On selection in any field logically associated with a specified field.
- 12 On locking selections in a specified field.
- 13 On unlocking selections in a specified field.

Variable events

- 14 When new data is entered into a variable.
- 15 When the value of a variable changes.

In addition, QlikView OCX 7.52 and later versions have the general OnDataChanged event, which is a general notification event signaling that data or selections have changed in the loaded QlikView document.

6 COMMENTS TO MEMBER DESCRIPTIONS IN PART II

6.1 Description of method and event members

Full name

Name of class followed by a period and the name the member.

ParameterNo

Number of parameter or an indicator that lines describes the return value.

Parameter name

Name of parameter

Type

The data type of the parameter. The type may be a VB type or a QlikView class name.

Direction

Describes the parameter

- in normal (in-parameter, must always be given).
- in optional (in-parameter, may be omitted).
- out (return values from event members).

OpenDocument

Opens a QlikView document in the control.

ParameterNo	Parameter	Type	Direction	Comment
return value	-	Document	-	QlikView document returned
1	DocName	String	in	QV document file name (qvw or qva) with path
2	UserName	String	in	QlikView document UserID
3	Password	String	in	QlikView document Password

Example of use (VB) :

```
Private Sub Command3_Click()
    QlikOCX1.OpenDocument "C:\MyDoc", "", ""
End Sub
```

Comment

Short description of return value and of each parameter

Examples

One or two basic examples showing the member used in a context in VB.

- The member is always underlined.
- VB reserved words and comments are in bold.

6.2 Description of property members

Full name

Name of class followed by a period and the name of the member.

Type

The data type of the property. The type may be a VB data type or a QlikView class name

Direction

Indicates if the property is

- read only (cannot be changed via Automation).
- read write (can be changed via Automation).
- objectref (a reference to a QlikView Automation object)

DocName

Type	Direction	Comment
Object	read-write	Name of QV document to be pre-opened in control.

Example of use (VB) :

```
Private Sub SetDocName()  
    QlikOCX1.DocName = "C:\MyDoc.qvw"  
End Sub
```

Comment

Short description of the property

Examples

One or two basic examples showing the member used in a context in VB.

- The member is always underlined.
- VB reserved words and comments are in bold.

Example of use (VB) :

```
Private Sub QlikOCX1_OnQvEvent()  
End Sub  
Private Sub QlikOCX1_OnQvEvent(ByVal EventCategory As  
String, ByVal EventSource As String)  
    MsgBox (EventCategory + " - " + EventSource)  
End Sub
```

PART II: QLIKVIEW OCX SPECIFIC AUTOMATION MEMBERS

7 QLIKVIEW OCX SPECIFIC METHODS

7.1 Method summary

Member	Comment
AboutBox	Displays the QlikView OCX about box.
Drag	Not implemented in the QlikView OCX control.
HasOpenDocument	Returns true if the control currently has an open QV document.
Move	Not implemented in the QlikView OCX control.
OpenDocument	Opens a QlikView document in the control.
SetFocus	Moves the focus to the specified object.
ShowWhatsThis	Not implemented in the QlikView OCX control.
ZOrder	Places a specified object at the front or back of the z-order within its graphical level.

7.2 Method descriptions

AboutBox

Displays an **About QlikOCX** box:

ParameterNo	Parameter	Type	Direction	Comment
return value	-	-	-	No return value

Example of use (VB.NET) :

```
Private Sub Button1_Click (ByVal sender As
System.Object, ByVal e As System.EventArgs) Handles
Button1.Click
    AxQlikOCX1.AboutBox()
End Sub
```

Example of use (C#):

```
private void button1_Click(object sender, EventArgs e)
{
    axQlikOCX1.AboutBox();
}
```

Example of use (Delphi):

```
Procedure TSDIAppForm.Button1Click(Sender: TObject);
begin
    QlikOcx1.AboutBox;
end;
```

HasOpenDocument

Returns true if the control currently has an open QV document.

ParameterNo	Parameter	Type	Direction	Comment
return value	-	Boolean	-	True if control has currently open QlikView document

Example of use (VB.NET) :

```
Private Sub Button2_Click(ByVal sender As
System.Object, ByVal e As System.EventArgs) Handles
Button2.Click
    If AxQlikOCX1.HasOpenDocument Then
        AxQlikOCX1.ActiveDocument.CloseDoc()
    End If
End Sub
```

Example of use (C#):

```
private void button2_Click(object sender, EventArgs e)
{
    if (axQlikOCX1.HasOpenDocument())
    {
        axQlikOCX1.ActiveDocument.CloseDoc();
    }
}
```

Example of use (Delphi):

```
procedure TSDIAppForm.Button2Click(Sender: TObject);
begin
    if QlikOCX1.HasOpenDocument then begin
        QlikOCX1.ActiveDocument.CloseDoc
    end;
end;
```

OpenDocument

Opens a QlikView document in the control.

ParameterNo	Parameter	Type	Direction	Comment
return value	-	Document	-	QlikView document returned

ParameterNo	Parameter	Type	Direction	Comment
1	DocName	String	in	QV document file name (qvw or qva) with path
2	UserName	String	in	QlikView document UserID
3	Password	String	in	QlikView document Password

Example of use (VB.NET) :

```
Private Sub Button3_Click(ByVal sender As
System.Object, ByVal e As System.EventArgs) Handles
Button3.Click
    AxQlikOCX1.OpenDocument("c:\OcxDemo.qvw", "", "")
End Sub
```

Example of use (C#):

```
private void button3_Click(object sender, EventArgs e)
{
    axQlikOCX1.OpenDocument("c:\OcxDemo.qvw", "",
    "");
}
```

Example of use (Delphi) :

```
procedure TSDIAppForm.Button3Click(Sender: TObject);
begin
    QlikOCX1.OpenDocument('c:\OcxDemo.qvw', '', '');
end;
```

OpenDocumentEx

Opens a QlikView document in the control with extended functionality. Please refer to the QlikView API documentation for detailed information.

SetFocus

Moves the focus to the specified object.

ParameterNo	Parameter	Type	Direction	Comment
return value	-	-	-	No return value

Example of use (VB) :

```
Private Sub Command3_Click()
    QlikOCX1.SetFocus
End Sub
```

ZOrder

Places a specified object at the front or back of the z-order within its graphical level.

	Parameter	Type	Direction	Comment
return value	-	-	-	No return value
Position	1	Integer	in, optional	0 or omitted = front 1 = back

Example of use (VB) :

```
Private Sub Example()  
    QlikOCX1.ZOrder  
End Sub
```


8 QLIKVIEW OCX SPECIFIC PROPERTIES

8.1 Properties summary

Member	Comment
ActiveDocument	Returns a reference to the currently active QlikView document in the control.
CausesValidation	Not implemented in the QlikView OCX control.
Container	Returns a reference to the container object of the control.
DocName	Name of QV document to be pre-opened in control.
DragIcon	Not implemented in the QlikView OCX control.
DragMode	Not implemented in the QlikView OCX control.
Height	Returns or sets the height of the control.
HelpContextID	Not implemented in the QlikView OCX control.
Index	Not implemented in the QlikView OCX control.
Left	Returns or sets the distance in between the internal left edge of the control and the left edge of its container
Name	Returns the name used in code to identify the control.
By default QlikOCX1.	
Object	Not implemented in the QlikView OCX control.
ObjectID	Qualified ObjectID if a single object is displayed in the QlikView OCX control.
Parent	Returns a reference to the parent object of the control.
Password	Returns or sets a password to be used when opening a QlikView document.
ProductLevel	Returns or sets the level of functionality for the OCX
Src	Reserved for plug-in client! Must not be used!
TabIndex	Returns or sets the tab order of the control within its parent form.
TabStop	Returns or sets a value indicating whether a user can use the TAB key to give the focus to an object.
Tag	Not used in the QlikView OCX control.
ToolTipText	Returns or sets the tooltip text that is shown when the cursor is passed over the control.
Top	Returns or sets the distance between the internal top edge of the control and the top edge of its container

Member	Comment
UserName	Returns or sets a user ID to be used when opening a QlikView document.
WhatsThisHelpID	Not implemented in the QlikView OCX control.
Width	Returns or sets the width of the control.
Visible	Returns or sets a variable controlling whether the control should be visible. True if control should be visible, false if control should be hidden.

8.2 Property descriptions

ActiveDocument

Type	Direction	Comment
QlikView.Document	read-only	Returns a reference to the currently active QlikView document in the control.

Example of use (VB.NET) :

```

Private Sub Button2_Click(ByVal sender As
System.Object, ByVal e As System.EventArgs) Handles
Button2.Click
    If AxQlikOCX1.HasOpenDocument Then
        AxQlikOCX1.ActiveDocument.CloseDoc()
    End If
End Sub

```

Example of use (C#) :

```

private void button2_Click(object sender, EventArgs e)
{
    if (axQlikOCX1.HasOpenDocument())
    {
        axQlikOCX1.ActiveDocument.CloseDoc();
    }
}

```

Example of use (Delphi) :

```

procedure TSDIAppForm.Button2Click(Sender: TObject);
begin
    if QlikOCX1.HasOpenDocument then begin
        QlikOCX1.ActiveDocument.CloseDoc
    end;
end;

```

Container

Type	Direction	Comment
Object	read-only	Returns a reference to the container object of the control.

Example of use (VB) :

```
Private Sub Example()  
    Set cont = QlikOCX1.Container  
End Sub
```

DocName

Type	Direction	Comment
Object	read-write	Name of QV document to be pre-opened in control.

Example of use (VB) :

```
Private Sub SetDocName()  
    QlikOCX1.DocName = "C:\MyDoc.qvw"  
End Sub
```

Height

Type	Direction	Comment
Single	read-write	Returns or sets the height of the control.

Example of use (VB) :

```
Private Sub MoveAndSize()  
    QlikOCX1.Top = QlikOCX1.Top + 100  
    QlikOCX1.Left = QlikOCX1.Left + 100  
    QlikOCX1.Width = QlikOCX1.Width * 0.95  
    QlikOCX1.Height = QlikOCX1.Height * 0.95  
End Sub
```

Left

Type	Direction	Comment
Single	read-write	Returns or sets the distance in between the internal left edge of the control and the left edge of its container.

Example of use (VB) :

```
Private Sub MoveAndSize()  
    QlikOCX1.Top = QlikOCX1.Top + 100  
    QlikOCX1.Left = QlikOCX1.Left + 100  
    QlikOCX1.Width = QlikOCX1.Width * 0.95  
    QlikOCX1.Height = QlikOCX1.Height * 0.95  
End Sub
```

Name

Type	Direction	Comment
String	read-only	Returns the name used in code to identify the control. By default QlikOCX1

Example of use (VB) :

```
Private Sub Example()  
    n = QlikOCX1.Name  
End Sub
```

ObjectID

Type	Direction	Comment
String	read-write	Qualified ObjectID if a single object is displayed in the QlikView OCX control.

Example of use (VB) :

```
Private Sub Example()  
    n = QlikOCX1.ObjectID = "Document\LB01"  
End Sub
```

Parent

Type	Direction	Comment
Object	read-only	Returns a reference to the parent object of the control.

Example of use (VB) :

```
Private Sub Example()  
    Set p = QlikOCX1.Parent  
End Sub
```

Password

Type	Direction	Comment
String	read-write	Returns or sets a password to be used when opening a QlikView document.

Example of use (VB) :

```
Private Sub SetPwd()  
    QlikOCX1.Password = "abc123"  
End Sub
```

ProductLevel

Type	Direction	Comment
Integer	read-write	Returns or sets the level of functionality for the OCX

0 – Analyzer (requires server)

1 – Analyzer+

2 – Professional

3 – Developer

The actual ProductLevel can never exceed the maximum level specified by the license.

Src This property is reserved for the QVA for IE plug-in client. Must not be used with OCX control!

TabIndex

Type	Direction	Comment
Integer	read-write	Returns or sets the tab order of the control within its parent form.

Example of use (VB) :

```
Private Sub Example()  
    QlikOCX1.TabIndex = 1  
End Sub
```

TabStop

Type	Direction	Comment
Boolean	read-write	Returns or sets a value indicating whether a user can use the TAB key to give the focus to an object.

Example of use (VB) :

```
Private Sub DisableTab()  
    QlikOCX1.TabStop = false  
End Sub
```

ToolTipText

Type	Direction	Comment
Integer	read-write	Returns or sets the tooltip text that is shown when the cursor is passed over the control.

Example of use (VB) :

```
Private Sub SetText()  
    QlikOCX1.ToolTipText = "This a QlikView OCX  
control"  
End Sub
```

Top

Type	Direction	Comment
Single	read-write	Returns or sets the distance in between the internal top edge of the control and the top edge of its container.

Example of use (VB) :

```
Private Sub MoveAndSize()  
    QlikOCX1.Top = QlikOCX1.Top + 100  
    QlikOCX1.Left = QlikOCX1.Left + 100  
    QlikOCX1.Width = QlikOCX1.Width * 0.95  
    QlikOCX1.Height = QlikOCX1.Height * 0.95  
End Sub
```

UserName

Type	Direction	Comment
String	read-write	Returns or sets a user ID to be used when opening a QlikView document.

Example of use (VB) :

```
Private Sub SetUID()  
    QlikOCX1.UserName = "jnn"  
End Sub
```

Width

Type	Direction	Comment
Single	read-write	Returns or sets the width of the control.

Example of use (VB) :

```
Private Sub MoveAndSize()  
    QlikOCX1.Top = QlikOCX1.Top + 100  
    QlikOCX1.Left = QlikOCX1.Left + 100  
    QlikOCX1.Width = QlikOCX1.Width * 0.95  
    QlikOCX1.Height = QlikOCX1.Height * 0.95  
End Sub
```

Visible

Type	Direction	Comment
Boolean	read-write	Returns or sets a variable controlling whether the control should be visible. True if control should be visible, false if control should be hidden.

Example of use (VB) :

```
Private Sub HideQlikView OCX()  
    QlikOCX1.Visible = false  
End Sub
```


9 QLIKVIEW OCX SPECIFIC EVENTS

9.1 Events summary

Member	Comment
DragDrop	Not implemented in the QlikView OCX control.
DragOver	Not implemented in the QlikView OCX control.
GotFocus	Not implemented in the QlikView OCX control.
LostFocus	Not implemented in the QlikView OCX control.
OnContextMenu	Occurs when a context menu is invoked by the user right-clicking in the QlikView OCX control.
OnContextMenuCommand	Occurs when a context menu command is invoked in the QlikView OCX control.
OnDataChanged	Occurs whenever the data or logical state changes in a QlikView document, ie on selections, variable changes, after reload etc.
OnHtmlHelp	Occurs when help is requested by pressing F1 or pushing the Help button in a dialog in the QlikView OCX control
OnMacro	Occurs when a macro trigger goes off in a QlikView document running in the QlikView OCX control.
OnMouseOver	Occurs once when mouse cursor is placed over OCX control area on screen.
OnQvEvent	Occurs as a result of a macro in a QV document executing FireQvEvent or FireQvUserEvent methods
Validate	Not implemented in the QlikView OCX control.

9.2 Event descriptions

OnContextMenu

Occurs when a context menu is invoked by the user right-clicking in the QlikView OCX control.

ParameterNo	Parameter	Type	Direction	Comment
1	ContextObject	String	in	Unique ID of sheet object where context menu was requested.
2	ContextMenu	Long	in out	Reference to context menu
3	OKToContinue	Integer	out	To be set to true if the built-in context menu is to be shown.

Example of use (VB) :

```
Rem MenuDef.bas module:
Public Const MF_SEPARATOR = 2048
Public Const MF_STRING = 0
Declare Sub AppendMenu Lib "user32" Alias
"AppendMenuA" _
    (ByVal hMenu _As Long, ByVal uFlags As
Long, _
    ByVal uIDNewItem As Long, ByVal lpNewItem
As String)
Public Const MF_BYCOMMAND = 0
Public Const MF_BYPOSITION = 1024
Declare Sub DeleteMenu Lib "user32" (ByVal hMenu As
Long, _
    ByVal uPosition As Long, ByVal uFlags As
Long)
Declare Function GetMenuItemCount Lib "user32" (ByVal
hMenu As Long) _
    As Integer
```

```

Rem Form code:
Rem Module MenuDef.bas as above assumed available
Rem Module QVMenu.bas assumed available
Private Sub QlikOCX1_OnContextMenu (ByVal
ContextObject As String, _
ByVal ContextMenu As Long, OkToContinue As
Integer)
    If ContextObject = "LB11" Then
        Rem ** delete default menu command by
command **
        Dim i, j
        j = GetMenuItemCount (ContextMenu) - 1
        For i = 0 To j
            DeleteMenu ContextMenu, 0, MF_BYPOSITION
        Next
    Else
        AppendMenu ContextMenu, MF_SEPARATOR, 0, 0
    End If
    Rem ** add new custom command to menu **
    AppendMenu ContextMenu, MF_STRING, 45000,
"Confirmed Clear"
End Sub

```

Example of use (VC++) :

```

void CTestDialog::OnOnContextMenuQlikocxctrl1(LPCTSTR
ContextObject, long ContextMenu, short FAR*
OkToContinue)
{
    HMENU hMenu = reinterpret_cast<HMENU>
(ContextMenu);
    ::AppendMenu (hMenu, MF_SEPARATOR, 0, NULL);
    ::AppendMenu (hMenu, MF_STRING, 42, _T ("Confirmed
Clear"));
}

```

OnContextMenuCommand

Occurs when a context menu command is invoked in the QlikView OCX control.

ParameterNo	Parameter	Type	Direction	Comment
1	ContextObject	String	in	Unique ID of sheet object where context menu was requested.
2	MenuCom-mand	Long	in, out	Context menu command ID.

ParameterNo	Parameter	Type	Direction	Comment
3	OKToContinue	Integer	out	To be set to true if Menu-Command is to be executed by QV.

Example of use (VB) :

```

Rem Form code:
Rem Module QVMenu.bas assumed available (QV constant
definitions)
Private Sub QlikOCX1_OnContextMenuCommand(ByVal
ContextObject As String,
MenuCommand As Long, OkToContinue As
Integer)
    Dim MsgBoxResult
    If MenuCommand = 45000 Then
        MsgBoxResult = MsgBox("Do you really want to
clear?", vbYesNo)
        If MsgBoxResult = vbYes Then
            MenuCommand = ID_POPUP_CLEAR
            OkToContinue = True
        Else
            OkToContinue = False
        End If
    End If
End Sub

```

Example of use (VC++) :

```

#include "QVResource.h" //QlikView constant
definitions
void
CTestDialog::OnOnContextMenuCommandQlikocxctrl1(LPCTSTR
ContextObject, long FAR* MenuCommand, short FAR*
OkToContinue)
{
    if (*MenuCommand == 42) {
        int r = AfxMessageBox (_T ("Do you really want
to clear?"),
MB_YESNO);
        if (r == IDYES) {
            *MenuCommand = ID_POPUP_CLEAR;
        } else {
            *OkToContinue = VARIANT_FALSE;
        }
    }
}

```

OnDataChanged

Occurs whenever the data or logical state changes in a QlikView document, ie on selections, variable changes, after reload etc.

ParameterNo	Parameter	Type	Direction	Comment
-	-	-	-	No parameters

Example of use (VB) :

```
Private Sub QlikOCX1_OnDataChanged()  
    MsgBox("Hello!")  
End Sub
```

OnHtmlHelp

Occurs when help is requested by pressing F1 or pushing the Help button in a dialog in the QlikView OCX control.

ParameterNo	Parameter	Type	Direction	Comment
1	HelpData	Long	in	Identifies which item help is asked for
2	HelpCommand	Integer	in	No parameters
3	OKToContinue	Integer	out	To be set to true if Help is to be executed by QV.

Example of use (VB) :

```
Private Sub QlikOCX1_OnHtmlHelp(ByVal HelpData As  
Long, ByVal HelpCommand As Integer, OkToContinue As  
Integer)  
    Dim MsgBoxResult  
    MsgBoxResult = MsgBox("Do you really want to show  
help for item " & HelpData & "?", vbYesNo)  
    If MsgBoxResult = vbYes Then  
        OkToContinue = True  
    Else  
        OkToContinue = False  
    End If  
End Sub
```

OnMacro

Occurs when a macro trigger goes off in a QlikView document running in the QlikView OCX control.

ParameterNo	Parameter	Type	Direction	Comment
1	MacroName	String	in	Name of macro invoked by trigger in QV document.

ParameterNo	Parameter	Type	Direction	Comment
2	OKToContinue	Integer	out	To be set to true if VBScript macro in QV document is to be executed.

Example of use (VB) :

```

Private Sub QlikOCX1_OnMacro(ByVal MacroName As
String, OkToCont As Integer)
    Dim MsgBoxResult
    MsgBoxResult = MsgBox("Want to run macro '" &
MacroName & "'", vbYesNo)
    If MsgBoxResult = vbNo Then
        OkToCont = False
    End If
End Sub

```

Example of use (VC++) :

```

void CTestDialog::OnOnMacroQlikocxctrl1(LPCTSTR
MacroName, short FAR*
    OkToContinue)
{
    CString s = _T ("Do you really want to run the ");
    s += MacroName;
    s += _T (" macro?");
    int r = AfxMessageBox (s, MB_YESNO);
    if (r == IDNO) {
        *OkToContinue = VARIANT_FALSE;
    }
}

```

OnMouseOver

Occurs once when mouse cursor is placed over OCX control area on screen.

ParameterNo	Parameter	Type	Direction	Comment
-	-	-	-	No parameters

Example of use (VB) :

```

Private Sub QlikOCX1_OnMouseOver()
    MsgBox("Hello!")
End Sub

```

OnQvEvent

Occurs as a result of a macro in a QV document executing FireQvEvent or FireQvUserEvent methods.

ParameterNo	Parameter	Type	Direction	Comment
1	EventCategory	String	-	The EventCategory parameter of the Automation method FireQvEvent or the string '\$user' after using the FireQvUserEvent.
2	EventSource	String	-	The EventSource parameter of the Automation methods FireQvEvent or FireQvUserEvent.

Example of use (VB) :

```

Private Sub QlikOCX1_OnQvEvent ()
End Sub

Private Sub QlikOCX1_OnQvEvent (ByVal EventCategory As String, ByVal EventSource As String)
    MsgBox (EventCategory + " - " + EventSource)
End Sub

```
