QE Visual Component Library

Introduction

The Visual Component Library (VCL) is a Qt plugin library that provides standard widgets that represent common components found on beam-lines such as valves, shutters, slits.
These widgets are designed to be used on assembly level UI forms called up by the Engineering GUIs.

By providing these as widgets, this allows UI form designers to just drag-and-drop a component onto their form, as opposed to constructing the the component themselves from the the available more basic widgets.
The other advantages are a more common look-and-feel amongst the assembly level ui forms, and being a widget, the opportunity to add behind the scenes smarts tailored to the particular component.

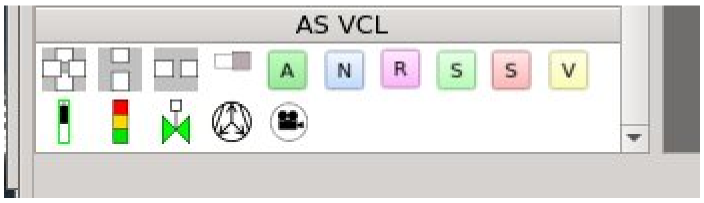
The VCL is located in perforce at: //ASP/tec/gui/qevcl/trunk

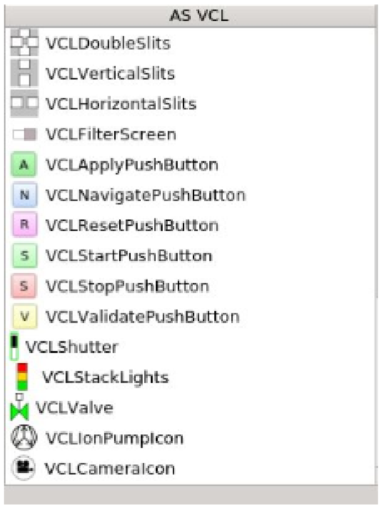
The latest released version of the VCL (as of 26 Mar 2023) is **qevcl/release_1-2-16** and this is/was/will be made available in **bundle0039**.

Using the VCL Widgets

General

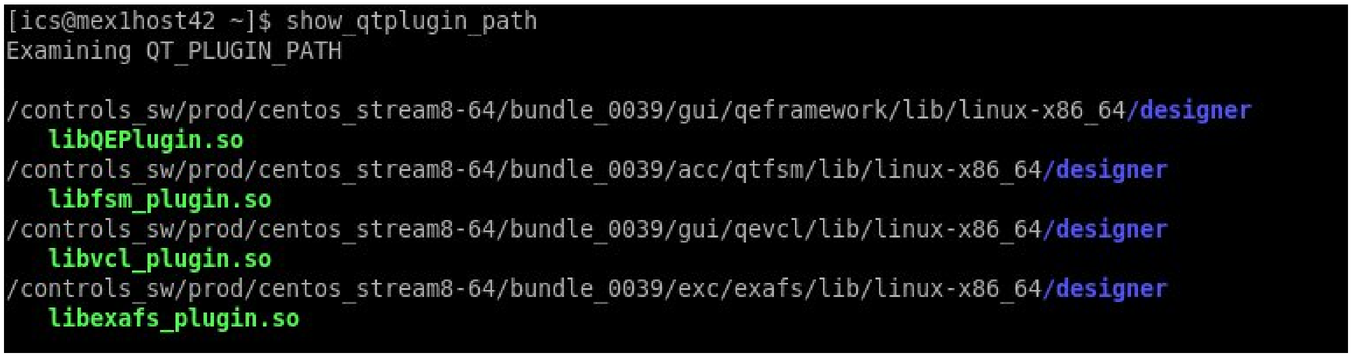
The VCL is a plugin library, and as as such, in order to use the VCL widgets, the QT_PLUGIN_PATH must be modified to include the VCL, both at design time when using the Qt designer tool, and at run time when using qegui (or any other Qt display manager).   Within designer, the VCL widgets appear in the "AS VCL" section (collapsed and expanded view):





Bundle Users

For bundle users (since bundle 0023), the QT_PLUGIN_PATH now automatically includes a reference to the VCL plugin library. This can be verified by running the show_qtplugin_path script, as illustrated below.



```
[ics@mex1host42 ~]$ show_qtplugin_path
Examining QT_PLUGIN_PATH

/controls_sw/prod/centos_stream8-64/bundle_0039/gui/qeframework/lib/linux-x86_64/designer
   libQEPlugin.so
/controls_sw/prod/centos_stream8-64/bundle_0039/acc/qtfsm/lib/linux-x86_64/designer
   libfsm_plugin.so
/controls_sw/prod/centos_stream8-64/bundle_0039/gui/qevcl/lib/linux-x86_64/designer
   libvcl_plugin.so
/controls_sw/prod/centos_stream8-64/bundle_0039/exc/exafs/lib/linux-x86_64/designer
   libexafs_plugin.so
```

Non-Bundle Users

Not for the faint hearted: You will have to sync the VCL component to your local work space, modify the configure/RELEASE file to suit your EPICS and EPICS Qt environment and run make. When updating the QT_PLUGIN_PATH environment variable, remember **do not** add the final designer directory to the path - Qt implicitly looks for and searches for a designer sub-directory.

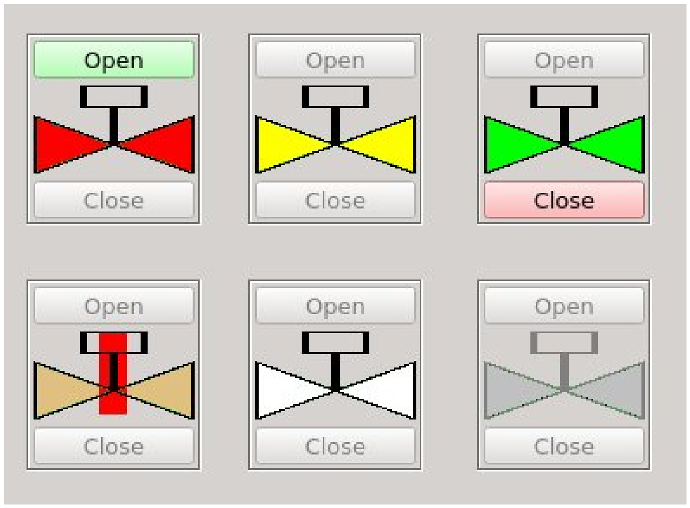
Available VCL Widgets

At the time of last update, 31 Mar 2023, the following VCL widgets have been defined.

VCL Valve Widget

This widget may be used to control/monitor a valve. The widget presents a valve status icon together with an open and a close control button.

From left to right, top to bottom, the following figure represent the possible states: closed (red), moving (yellow), open (green), invalid (orange plus vertical red bar), unknown (white) and disconnected (grayed-out).



The Open and Close control buttons are enabled only when the valve status is Closed and Open respectively (and of course the command PV must also be connected).

The widget's PV names are specified by means of a single **deviceName** property, e.g. MEX1HU02VLV01 (***note***: the valve TLA has not been allocated in the CSBS yet (i.e. as of 16 Sep 2020), VLV is currently just a best guess for this example).

The underlying EPICS database template should use, or be similar to, the following developed for communication with the Siemens S7-1500 series PLCs being deployed for the BRIGHT beam-lines.

```
//ASP/tec/eps/opcuaStdComps/trunk/OpcUaStandardComponentsApp/Db/opcua_gate_value_shutter.template
```

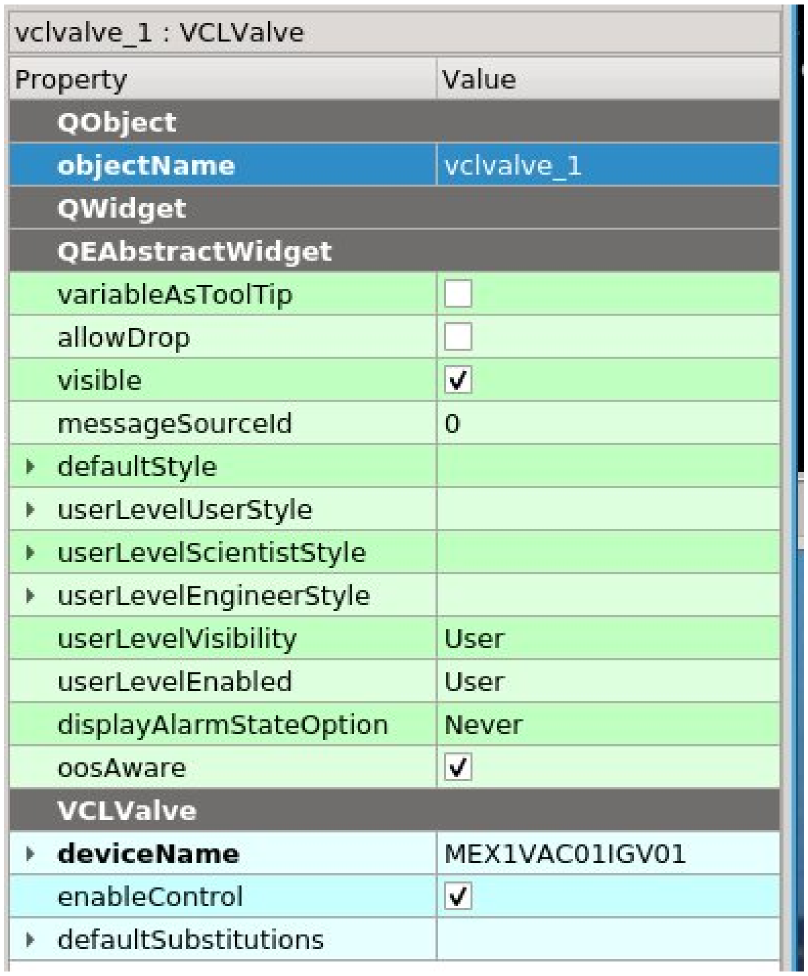
The template provides the following records used by the widget (there are others not used by the widget):

```
record (mbbo, "$(COMP):OPEN_CLOSE_CMD")  { ...  }
record (mbbi, "$(COMP):OPEN_CLOSE_STATUS")  { ...  }
```

For the instantiation of the EPICS database template, COMP should be the valve component name allocated by the CSBS.

Properties

In Qt designer, the widget's properties look like this:



vclvalve_1 : VCLValve

Property	Value
QObject	
objectName	vclvalve_1
QWidget	
QEAbstractWidget	
variableAsToolTip	☐
allowDrop	☐
visible	✓
messageSourceId	0
defaultStyle	
userLevelUserStyle	
userLevelScientistStyle	
userLevelEngineerStyle	
userLevelVisibility	User
userLevelEnabled	User
displayAlarmStateOption	Never
oosAware	✓
VCLValve	
deviceName	MEX1VAC01IGV01
enableControl	✓
defaultSubstitutions	

The VCLValve widget inherits directly from QEAbstractWidget, so is a proper QE Framework widget with all the standard EPICS Qt properties. The component name is defined by the **deviceName** property.

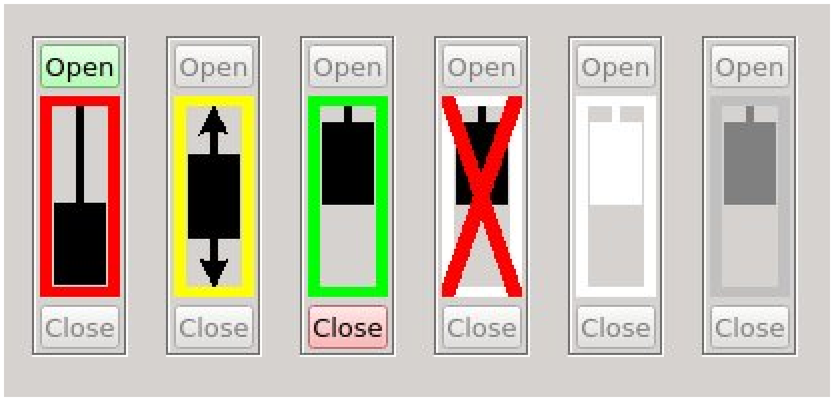
The normal macro substitution mechanism is available if required, e.g. **deviceName** may be set to "MEX1$(ASSEM)VLV01" and e.g. we can then define ASSEM=HU02 as a run time macro, or a priority macro in a QEForm or a QEPushButton.

The **enableControl** property removes the Open and Close buttons when deselected. This is intended for a status only display, either when control is not desired or not available.

The **defaultSubstitutions** are the equivalent of a regular QE widgets **variableSubstitutions** property. This is really only useful when designing to get a complete look and feel if/when a substitution is used.

VCL Shutter Widget

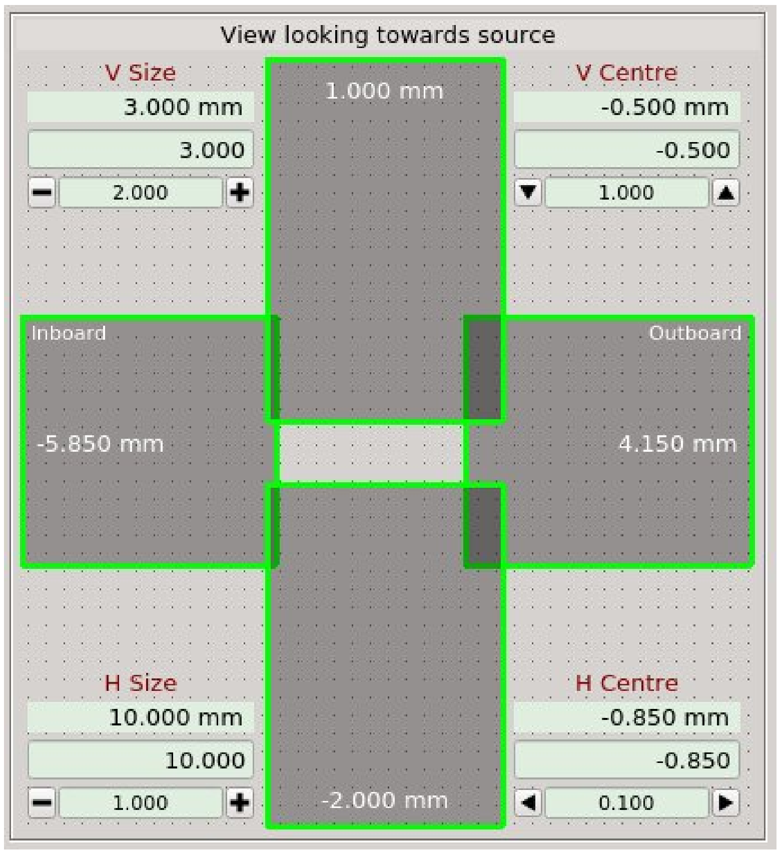
This widget may be used to control/monitor a shutter. It is isomorphically identical to the VCLValve widget , it uses the same widget properties and expects the same underlying PV name structure. The main difference is the icons used to represent each state. From left to right, the following figure represent the possible states: closed (red), moving (yellow), open (green), invalid (white plus red diagonal cross), unknown (white) and disconnected (grayed-out).



VCL Double Slits Widget

This widget is used to represent, control and monitor the horizontal and virtual axis (size and centre) associated with a pair of double slits. It also presents the associate blade read-back positions.

The following figure illustrates this widget monitoring a real MEX1 double slit.



Note: The double slit widget provides an up stream view (i.e. towards the source) schematic of the slits, as such the outboard blade is in the right hand side of the widget.

The boarder of each blade reflects the alarm severity associated with the motor readback value.

The widget assume the horizontal centre motor follows the standard axis convention, i.e.increasing centre value are in the outwards direction, and hence right-wards om the widget.

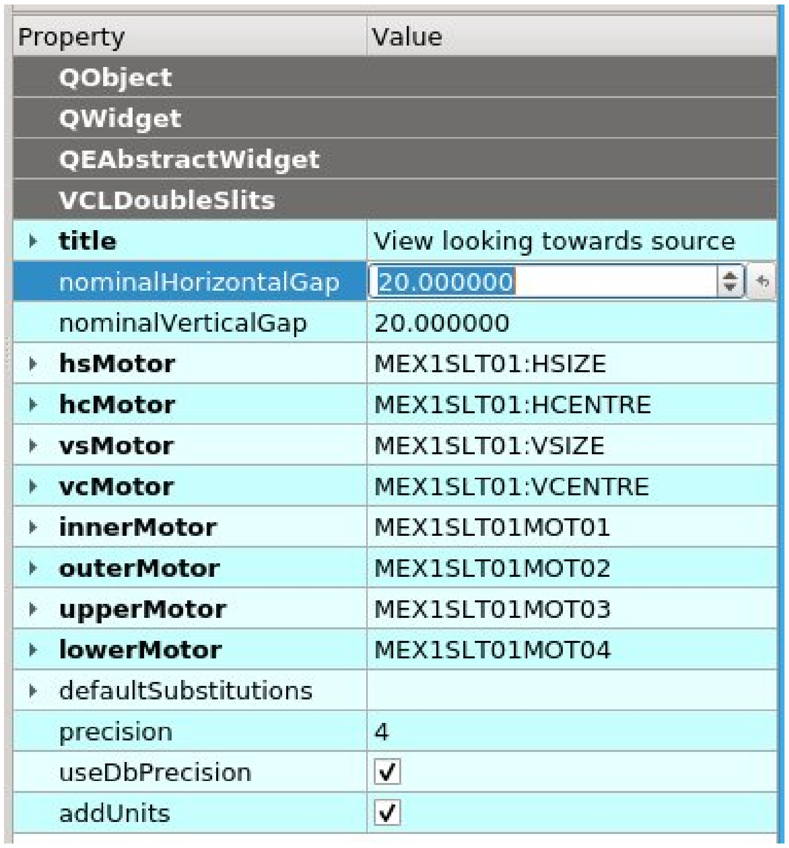
For each centre and size axis, the widget provides the means to monitor the axis position, set the position, set the position tweak value and tweak/nudge the position in the forward or in reverse direction. The blades are drawn to reflect the actual blade positions (based on centre and size as opposed to the blades physical position).

The widget's PV names are specified by 8 motor record names. The widget appends the ".VAL", ".RBV", ".TWV", ".TWR" and ".TWF" field names as appropriate.

This widget might be rather large for an assembly level UI form. It would be sensible to have a QEPushButton on the form that opens the double slit widget in a new/separate form.

Properties

In Qt designer, the widget's properties look like this:



Property	Value
QObject	
QWidget	
QEAbstractWidget	
VCLDoubleSlits	
title	View looking towards source
nominalHorizontalGap	20.000000
nominalVerticalGap	20.000000
hsMotor	MEX1SLT01:HSIZE
hcMotor	MEX1SLT01:HCENTRE
vsMotor	MEX1SLT01:VSIZE
vcMotor	MEX1SLT01:VCENTRE
innerMotor	MEX1SLT01MOT01
outerMotor	MEX1SLT01MOT02
upperMotor	MEX1SLT01MOT03
lowerMotor	MEX1SLT01MOT04
defaultSubstitutions	
precision	4
useDbPrecision	✓
addUnits	✓

title - the text of this property appears at the top of the widget. Macros may be used. In future, this may be specifiable by means of a string PV.

nominalHorizontalGap - this specifies the nominal horizontal gap, and helps the widget draw the horizontal blade positions roughly to scale. The default value is 20.

nominalVerticalGap - this specifies the nominal vertical gap, and helps the widget draw the vertical blade positions roughly to scale. The default value is 20.

hsMotor - this defines the name of the (virtual) motor record that control the horizontal size.

hcMotor - this defines the name of the (virtual) motor record that control the horizontal centre.

vsMotor - this defines the name of the (virtual) motor record that control the vertical size.

vcMotor - this defines the name of the (virtual) motor record that control the vertical centre.

innerMotor - this defines the name of the motor record that control the inner blade.

outerMotor - this defines the name of the motor record that control the outer blade.

upperMotor - this defines the name of the motor record that control the upper blade.

lowerMotor - this defines the name of the motor record that control the lower blade.

defaultSubstitutions - the equivalent of a regular QE widgets **variableSubstitutions** property. This is really only useful when UI form designing to get a complete look and feel if/when substitutions are used.

precision - when **useDbPrecision** set to false, this will override the precision used to display values (just like most other EPICS Qt widgets). The default value is 4.

useDbPrecision - then true, the **precision** property value determines precision used by the widget. The default is false.

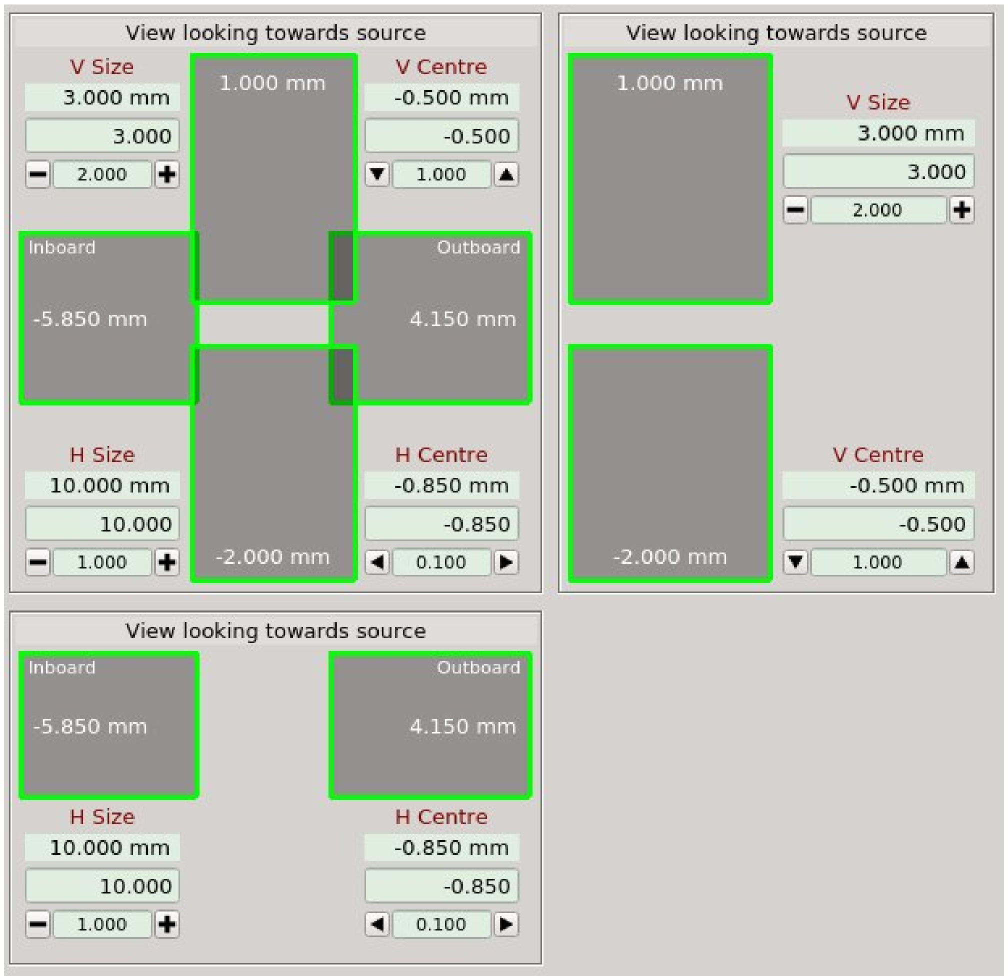
addUnits - controls whether the units (from the EGU field, e.g. mm) are appended to displayed values. The default is true.

The normal macro substitution mechanism is available if required, e.g. **hsMotor** may be set to "$(DLS):HSIZE_VRT" and e.g. we can then define DLS=SR 12ID01SLT21 as a run time macro, or a priority macro definition in a QEForm or a QEPushButton.

VCL Horizontal Slits Widget and Vertical Slits Widget

Each of these widgets are a subset of double slits widget and is used to represent, control and monitor the vertical or horizontal slits. It also presents the associate blade read-back positions.

The following figure illustrates these widgets monitoring real MEX1 vertical and horizontal slits.



Note: the vertical and horizontal widgets provides a down stream view schematic of the slits which are consistent with the one of the double slits widget.

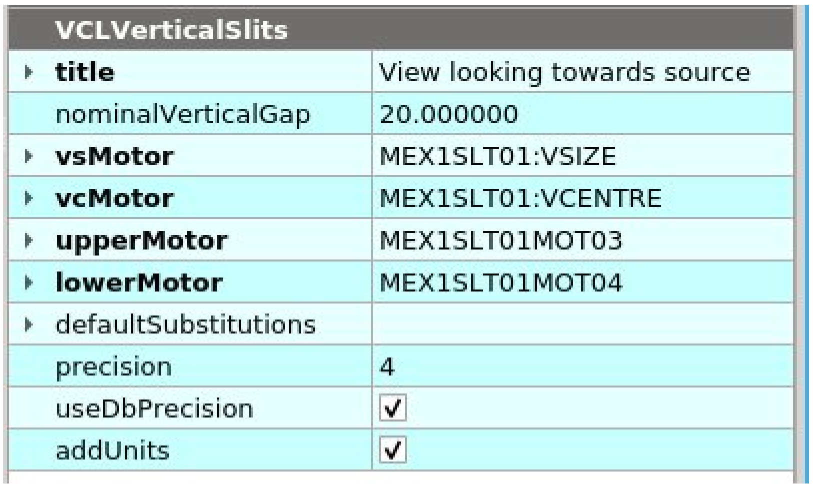
As same as the double slits widget, for each centre and size axis, the widget provides the means to monitor the axis position, set the position, set the position tweak value and tweak/nudge the position in the forward or in reverse direction. The blades are drawn to reflect the actual blade positions (based on centre and size as opposed to the blades physical position).

The widget's PV names are specified by 4 motor record names. The widget appends the ".VAL", ".RBV", ".TWV", ".TWR" and ".TWF" field names as appropriate.

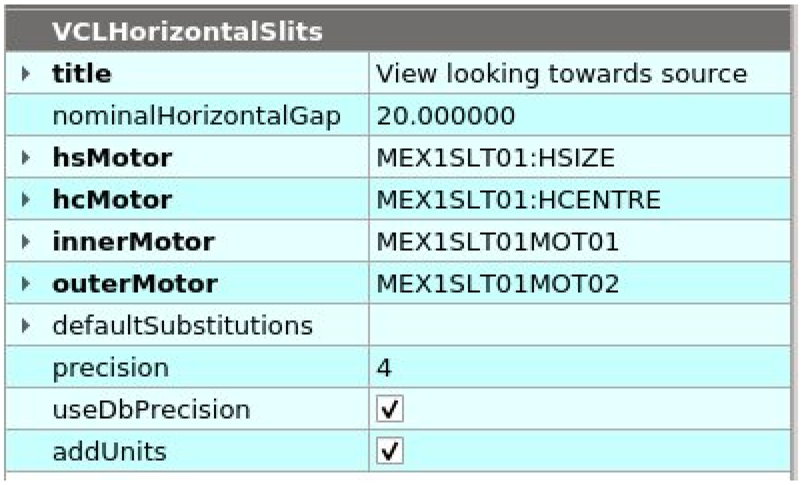
These widgets might be rather large for an assembly level UI form. It would be sensible to have a QEPushButton on the form that opens the double slit widget in a new/separate form.

Properties

In Qt designer, the widget's properties look like below:



VCLVerticalSlits	
title	View looking towards source
nominalVerticalGap	20.000000
vsMotor	MEX1SLT01:VSIZE
vcMotor	MEX1SLT01:VCENTRE
upperMotor	MEX1SLT01MOT03
lowerMotor	MEX1SLT01MOT04
defaultSubstitutions	
precision	4
useDbPrecision	✓
addUnits	✓

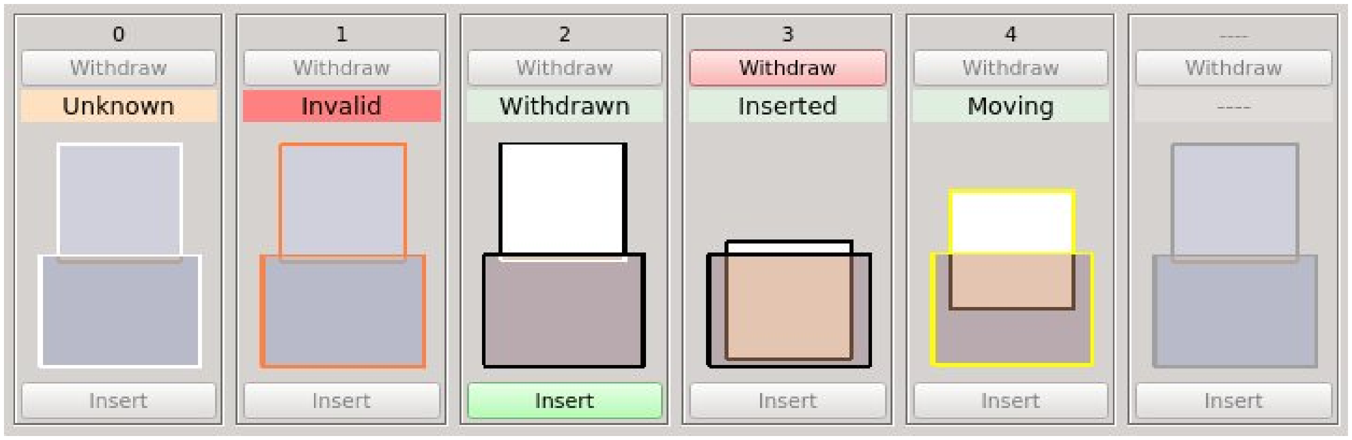


VCLHorizontalSlits	
title	View looking towards source
nominalHorizontalGap	20.000000
hsMotor	MEX1SLT01:HSIZE
hcMotor	MEX1SLT01:HCENTRE
innerMotor	MEX1SLT01MOT01
outerMotor	MEX1SLT01MOT02
defaultSubstitutions	
precision	4
useDbPrecision	✓
addUnits	✓

Refer to the double slits widget section, their properties are a subset of the double slits widget's properties.

VCL Filter/Screen Widget

This widget is used to show the state of a filter or screen. The following shows the widget in various states. From left to right, the following figure represent the possible states:
unknown, invalid, inserted, withdrawn, moving and disconnected (grayed-out).



The widget should be used in conjunction with the filter/screen template, located at (in perforce)

//ASP/tec/eps/opcuaEPSComps/trunk/OpcUaEPSComponentsApp/Db/opcua_pneumatic_actuator.template

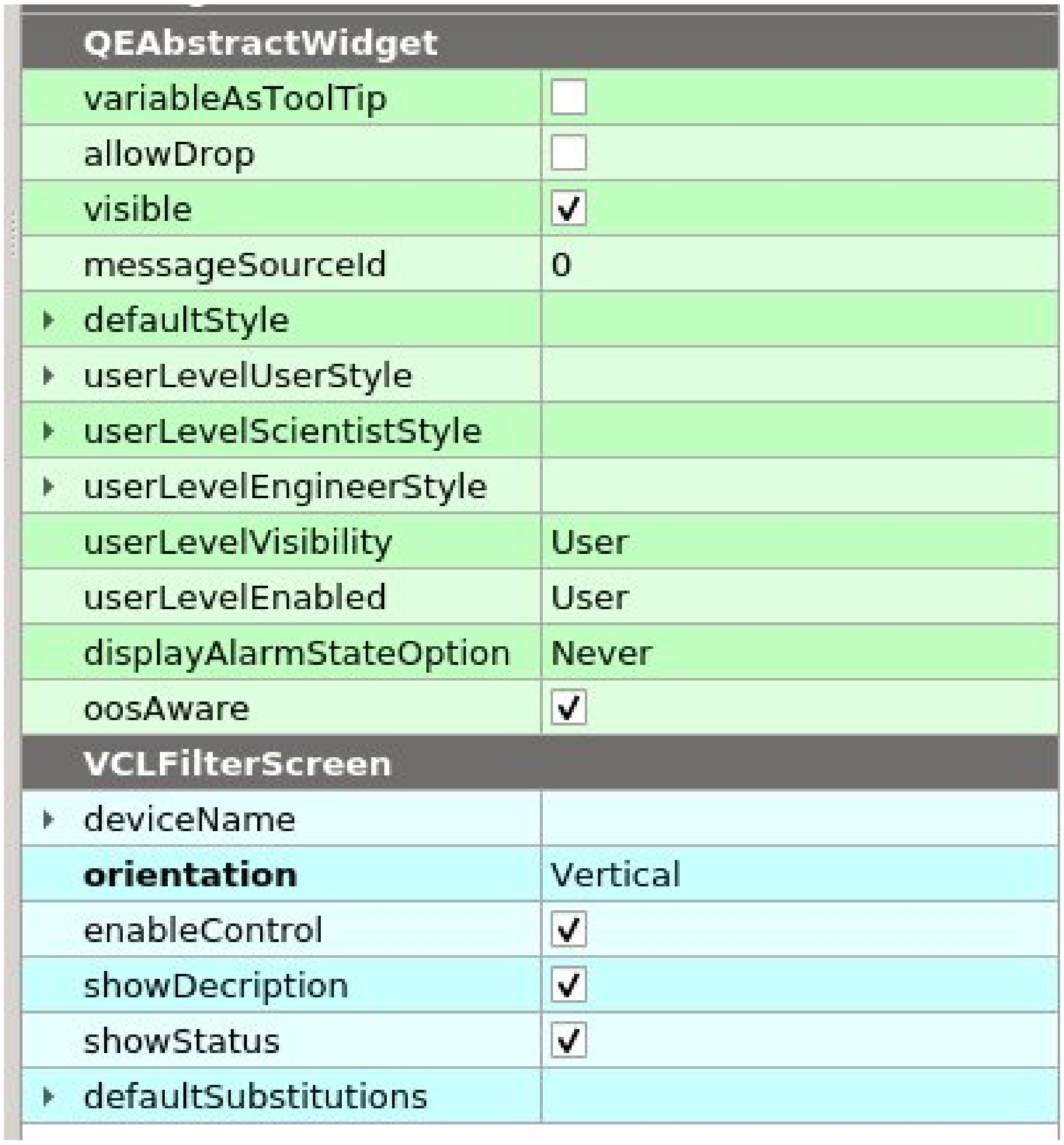
The template provides the following records used by the widget (there are others not used by the widget):

```
record (stringin, "$(COMP):NAME")  { ...  }
record (bo, "$(COMP):INSERT_CMD")  { ...  }
record (bo, "$(COMP):WITHDRAW_CMD")  { ...  }
record (mbbi, "$(COMP):STATUS")  { ...  }
```

For the instantiation of the EPICS database template, COMP should be the filter/screen component/device name allocated by the CSBS.

Properties

In Qt designer, the widget's properties look like below:



QEAbstractWidget	
variableAsToolTip	☐
allowDrop	☐
visible	✓
messageSourceId	0
defaultStyle	
userLevelUserStyle	
userLevelScientistStyle	
userLevelEngineerStyle	
userLevelVisibility	User
userLevelEnabled	User
displayAlarmStateOption	Never
oosAware	✓
VCLFilterScreen	
deviceName	
orientation	Vertical
enableControl	✓
showDecription	✓
showStatus	✓
defaultSubstitutions	

The **deviceName** property defines the component name.

The **orientation** property controls whether the widget is displayed vertically or horizontally.

The **enableControl** property removes the Insert and Withdraw buttons when deselected. This is intended for a status only display, either when control is not desired or not available.

The **showDescription** property controls wheather the PV defined component name is displayed.

The **showStatus** property controls wheather the estual inserted/withdrawn status is doispalyed. The iocon status is always displayed.

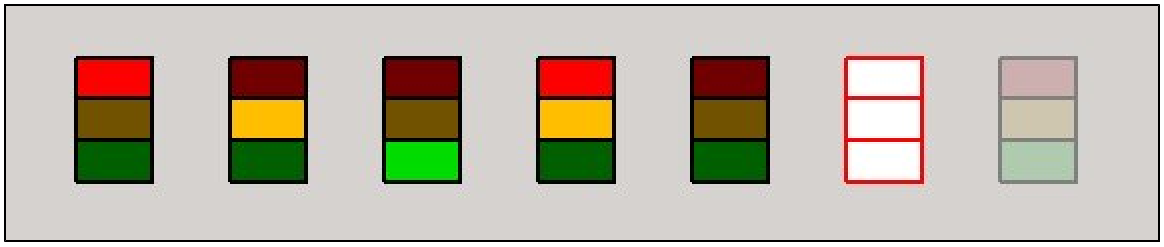
The **defaultSubstitutions** property allows for the provision for default substitution values.

VCL Stack Lights Widget

This widget is used to represent standard 3-lamp stack light assembly as used on the new beamlines. The following illustrates the Stack Light widget in various states.
From left-to-right, these are:

- red lamp on - beam in the hutch
- amber/yellow lamp on - no access
- green lamp on - open access
- red and amber/yellow lamps on - beam in hutch and no access
- no lamps on - faulted or in mode change or main panel keys not engaged.

- white with orange boarder - underlying PV is invalid
- greyed out - underlying PV is disconnected.



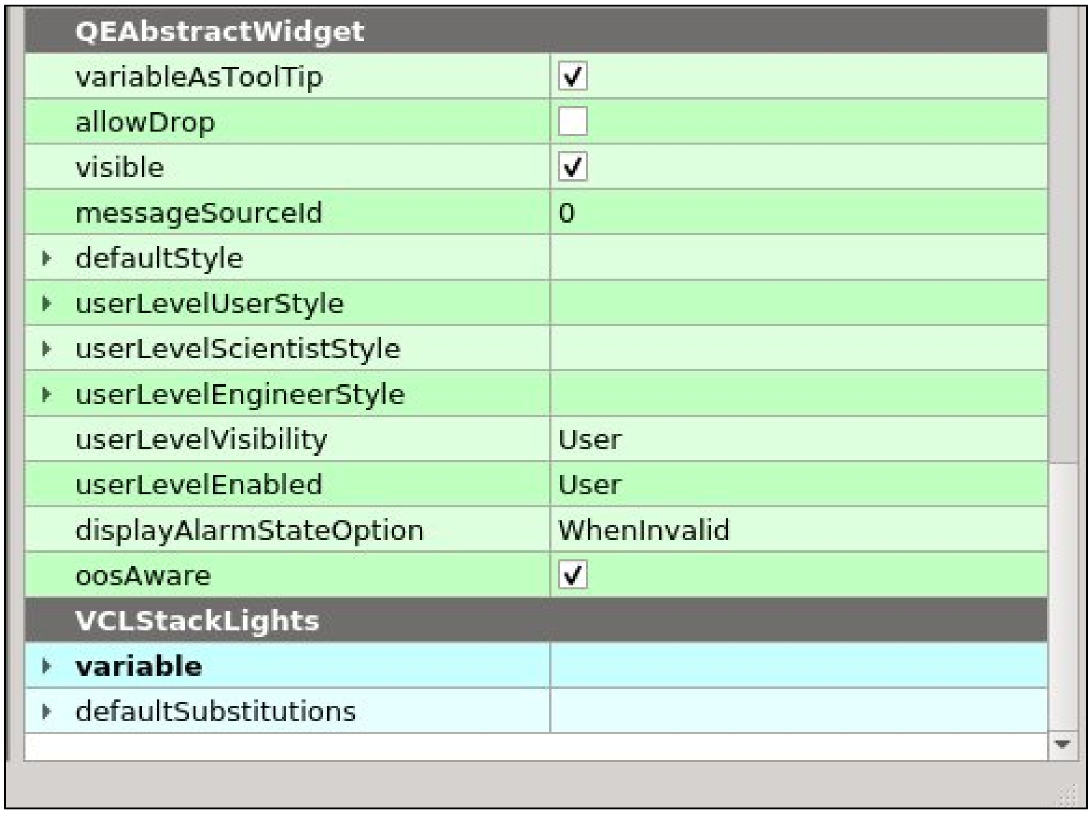
Unlike (most) other VCL widgets, this widget is a singular widget that takes a PV variable name (as opposed to a container holding multiple widgets, and requiring a component name).

The underlying PV is used to provide 3 bits that control the on/off display state of each lamp. Bit 0 is used to control the top/red lamp, bit 1 the middle /amber/yellow lamp and bit 2 the bottom/green lamp, and all other bits are ignored.

Note: the widget makes no attempt to validate these bit patterns, and no attempt to to flag illegal/unexpected lamp combination.

Properties

In Qt designer, the widget's properties look like below:



QEAbstractWidget	
variableAsToolTip	✓
allowDrop	☐
visible	✓
messageSourceId	0
defaultStyle	
userLevelUserStyle	
userLevelScientistStyle	
userLevelEngineerStyle	
userLevelVisibility	User
userLevelEnabled	User
displayAlarmStateOption	WhenInvalid
oosAware	✓
VCLStackLights	
variable	
defaultSubstitutions	

The widget requires a CA (or PVA) variable. For completeness, a defaultSubstitutions property is also defined.
Note: inherited style related properties are ignored.

VCL Push Buttons

These six widgets inherited directly from QEPushButton, and are identical to QEPushButton in all respects, except for the default values of two properties.

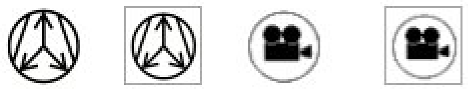
These are the button **text** property and the **defaultStyle** (i.e. colour) property. See Jira issue GUI-251.

Note: the shorter default text value also means the default width is less, i.e. 80 pixels, as opposed to 101 pixels.

class	default text	default style colour	usage
VCLApplyButtom	Apply	green	Apply/confirmation
VCLNavigateButton	Navigate	pale-blue	Open a new ui file
VlCResetButton	Reset	pale-purple	Reset
VCLStartButton	Start	pale-green	Open/Go/Approve/Write/Acknowledge/....
VCLStopButton	Stop	pale-red	Stop/Cancel/Close...
VCLValidateButton	Validate	pale-yellow	Check/Test/Validate/...

VCL Icon Widgets

There are currently two icon widgets available, namely VCLIonPumpIcon and VCLCameraIcon. As the names suggests, these are just icons, and provide no active functionality other than being decorative, and are neither interactive with the user nor with EPICS. The figure below show each widget, with and without a boarder.



The icon widgets inherit directly from VCLAbstactIcon which in turn inherits QLabel. The icon widgets provide no extra designer properties, and indeed hides/removes QLabel's text property.

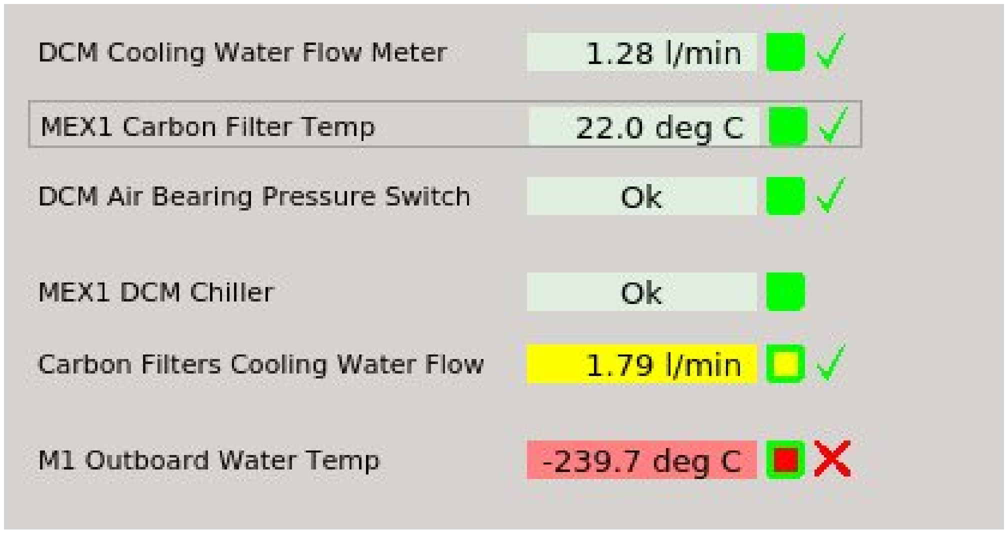
The embedded icon image is set scaled by default. The icon widgets may be resized/stretched like any other widget. Because QLabel inherits from QFrame, the frameShape and frameShadow properties are available if desired.

VLC EPS Component

This widget inherited directly from QEFrame, which inturn inherits from QFrame.
It provides EPS statues of devices that are connected to the EPS PLC, and intended for use with the OPC/UA EPS component EPICS database tempates.

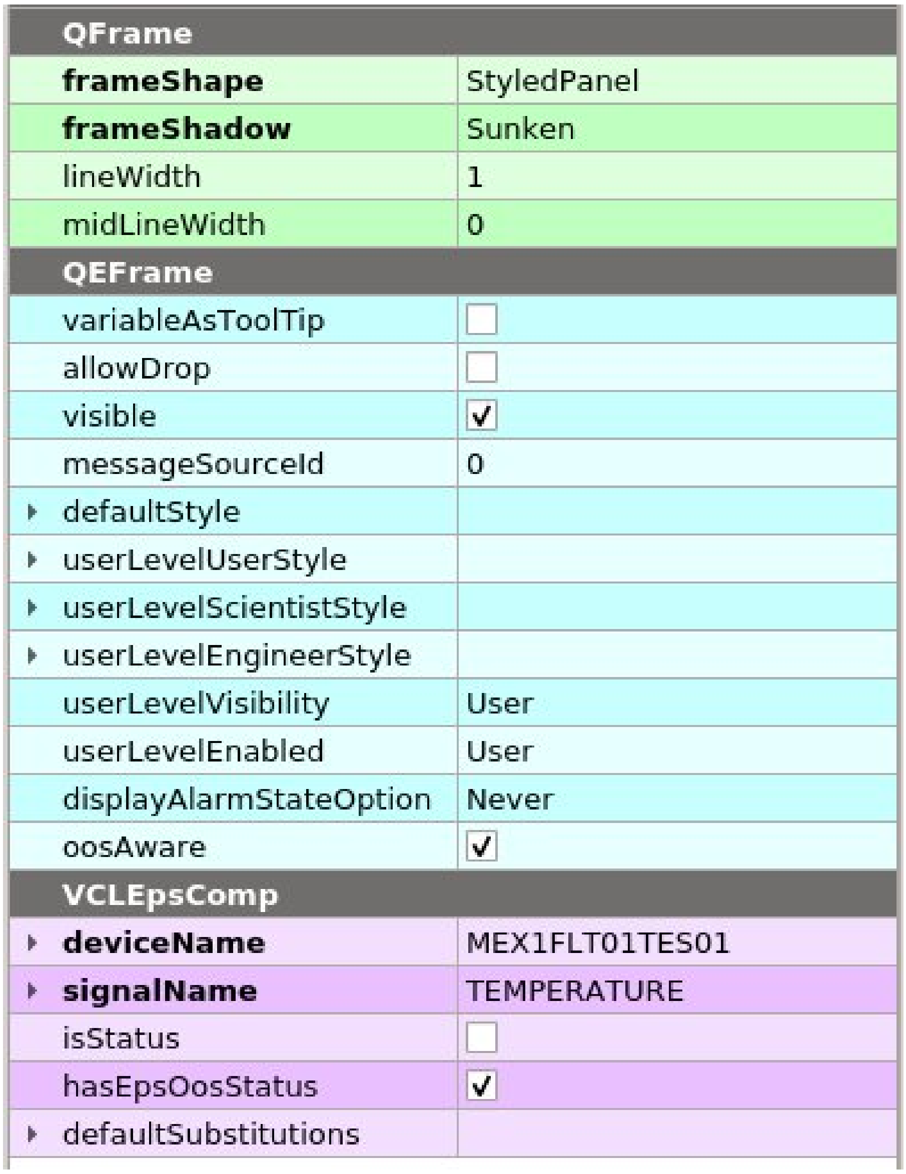
The status comprises (from left to right) the componet name, current value, a combined current status (inner part) and latch status (shown on the edge), and an optional EPS Out of Service (OOS) status (green tick for in service, red cross when out of service).

Examples of this widget are show below. From top to bottom, the 2nd item has the border turn on, the 4th item has no EPS OOS status, the 6th item is out of service (and hence not latched).



DCM Cooling Water Flow Meter	1.28 l/min	🟩	✓
MEX1 Carbon Filter Temp	22.0 deg C	🟩	✓
DCM Air Bearing Pressure Switch	Ok	🟩	✓
MEX1 DCM Chiller	Ok	🟩	
Carbon Filters Cooling Water Flow	1.79 l/min	🟨	✓
M1 Outboard Water Temp	-239.7 deg C	🟥	✗

In Qt designer, the properties properties look like this"



QFrame	
frameShape	StyledPanel
frameShadow	Sunken
lineWidth	1
midLineWidth	0
QEFrame	
variableAsToolTip	☐
allowDrop	☐
visible	✓
messageSourceId	0
defaultStyle	
userLevelUserStyle	
userLevelScientistStyle	
userLevelEngineerStyle	
userLevelVisibility	User
userLevelEnabled	User
displayAlarmStateOption	Never
oosAware	✓
VCLEpsComp	
deviceName	MEX1FLT01TES01
signalName	TEMPERATURE
isStatus	☐
hasEpsOosStatus	✓
defaultSubstitutions	

The **deviceName** property defines the component name, The descriptive PV name is **deviceName**:NAME.

The **signalName** property specifies the signal name.

The **isStatus** property controls whether the value PV name is either **deviceName**:**signalName**_MONITOR or **deviceName**:**signalName**_STATUS.

THe **hasEpsOosStatus** property controls whether the widget attempt to connect to and display the EPS OOS status.

The **defaultSubstitutions** property allows for the provision for default substitution values.

Release Notes

qevcl/release_1-2-16

Added the hasEpsOosStatus property to the EPS Coponent widget, and emphasized the in service status with a green tick (as opposed to a greyed out cross).

qevcl/release_1-2-15

Added the VCL EPS component. Also tweaked the colors on the stack light widget.

qevcl/release_1-2-14

Modified the filterScreen widget to allow the control buttons (insert/withdraw) be be select-able using the new bool property **enableControl**, and brings it more inline with the shutter and valve widgets.

The shutter and valve widgets also modified to properly honor ther **enableControl** property status and stop user level login modifying visibility.

qevcl/release_1-2-13

Modified the filter/screen widget to add moving state and distinguish (new icon images) disconnected, unknown and invalid states.

Also added new properties to show/hide name label and/or status label.

qevcl/release_1-2-12

Modified the filter/screen widget to bring channel names into alignment with the.opcua_pneumatic_actuator.template

qevcl/release_1-2-11

Modified the double and horizontal slits to reverse the viewing orientation, i.e. from downstream ***towards*** upstream.

qevcl/release_1-2-10

Modified the slit blades (for double, horizontal and vertical slits widgets) to be partially see through, and re-badged from Inner/Outer to Inboard/Outboard.

qevcl/release_1-2-9

Modified the Valve and Shutter widgets to hide the Open and Close control buttons when control is disabled, i.e. wghen the enableControl property is set False.

qevcl/release_1-2-8

Updated to include the icon widgets.
Updated the double and horizontal slits such that the Inner/Outer labsls to not mask/hide the context menu and tool tips of the blade widgets.
Updated the filter-screen widget to honor the sines and minimum sizes set up within designer.

qevcl/release_1-2-7

Modified the shutter and valve widgets to provide a enableControl property (default true) to allow this widget to sensibly display shutters and valves from other operational areas, such as the front end.

qevcl/release_1-2-6

Modified the filter screen widget to be displayed horizontally (default) or vertically.

qevcl/release_1-2-5

Added a number of pre-formatted (colour and text) push button widgets. These are otherwise functionally identical to the parent QEPushButton.

qevcl/release_1-2-4

Added the Stack Lights widget to the VCL.

qevcl/release_1-2-3

Added the filter screen widget; and created widget specific ,png files for the horizontal slits and vertical slits widgets.

qevcl/release_1-2-2

Added the horizontal slits and vertical slits widgets.

qevcl/release_1-2-1

Initial delivery of the 1-2 series of releases. Please review perforce for earlier release information.