

# EPICS QT Framework

## 2.2

Generated by Doxygen 1.7.4

Wed Jan 23 2013 15:43:31



# Contents

<b>1</b>	<b>QE framework - EPICS aware Qt Widgets and data access classes</b>	<b>1</b>
1.1	Documentation . . . . .	1
1.2	License . . . . .	2
1.3	Platforms . . . . .	2
1.4	Screenshots . . . . .	2
1.5	Downloads . . . . .	2
1.6	Installation . . . . .	2
1.7	Support . . . . .	3
1.8	Related Projects . . . . .	3
1.9	Credits: . . . . .	3
<b>2</b>	<b>GNU General Public License</b>	<b>5</b>
<b>3</b>	<b>ASgui screen shots</b>	<b>7</b>
<b>4</b>	<b>other applications using epicsqt widgets</b>	<b>13</b>
<b>5</b>	<b>Qt Designer</b>	<b>15</b>
<b>6</b>	<b>Qt Creator</b>	<b>17</b>
<b>7</b>	<b>Class Index</b>	<b>19</b>
7.1	Class Hierarchy . . . . .	19
<b>8</b>	<b>Class Index</b>	<b>23</b>
8.1	Class List . . . . .	23
<b>9</b>	<b>Class Documentation</b>	<b>27</b>
9.1	_Field Class Reference . . . . .	27

9.2	<a href="#">_Item Class Reference</a>	28
9.3	<a href="#">_QDialogItem Class Reference</a>	28
9.4	<a href="#">_QDialogLogin Class Reference</a>	28
9.5	<a href="#">_QPushButtonGroup Class Reference</a>	29
9.6	<a href="#">_QTableWidgetFileBrowser Class Reference</a>	29
9.7	<a href="#">_QTableWidgetLog Class Reference</a>	30
9.8	<a href="#">_QTableWidgetScript Class Reference</a>	30
9.9	<a href="#">QEAnalogIndicator::Band Struct Reference</a>	30
9.10	<a href="#">QEAnalogIndicator::BandList Class Reference</a>	31
9.11	<a href="#">qcastatemachine::ConnectionQCaStateMachine Class Reference</a>	31
9.12	<a href="#">ContainerProfile Class Reference</a>	31
9.13	<a href="#">contextMenu Class Reference</a>	33
9.14	<a href="#">contextMenuObject Class Reference</a>	35
9.15	<a href="#">QEPeriodic::elementInfoStruct Struct Reference</a>	35
9.16	<a href="#">flipRotateMenu Class Reference</a>	36
9.17	<a href="#">imageContextMenu Class Reference</a>	36
9.18	<a href="#">imageMarkup Class Reference</a>	37
9.19	<a href="#">localEnumerationItem Class Reference</a>	38
9.20	<a href="#">managePixmap Class Reference</a>	39
9.21	<a href="#">markupBeam Class Reference</a>	39
9.22	<a href="#">markupHLine Class Reference</a>	40
9.23	<a href="#">markupItem Class Reference</a>	41
9.24	<a href="#">markupLine Class Reference</a>	42
9.25	<a href="#">markupRegion Class Reference</a>	43
9.26	<a href="#">markupTarget Class Reference</a>	44
9.27	<a href="#">markupText Class Reference</a>	44
9.28	<a href="#">markupVLine Class Reference</a>	45
9.29	<a href="#">PeriodicDialog Class Reference</a>	46
9.30	<a href="#">PeriodicElementSetupForm Class Reference</a>	46
9.31	<a href="#">PeriodicSetupDialog Class Reference</a>	46
9.32	<a href="#">QEStripChart::PrivateData Class Reference</a>	47
9.33	<a href="#">QEStripChartItem::PrivateData Class Reference</a>	47
9.34	<a href="#">profilePlot Class Reference</a>	48
9.35	<a href="#">PushButtonSpecifications Struct Reference</a>	48

9.36	QBitStatus Class Reference . . . . .	48
9.37	QCaAlarmInfo Class Reference . . . . .	50
9.38	QCaConnectionInfo Class Reference . . . . .	51
9.39	QCaDataPoint Struct Reference . . . . .	51
9.40	QCaDataPointList Class Reference . . . . .	51
9.41	QCaDateTime Class Reference . . . . .	51
9.42	QCaEventFilter Class Reference . . . . .	52
9.43	QCaEventItem Class Reference . . . . .	52
9.44	QCaEventUpdate Class Reference . . . . .	52
9.45	QCaInstalledFiltersListItem Class Reference . . . . .	53
9.46	qcaobject::QCaObject Class Reference . . . . .	53
9.47	qcastatemachine::QCaStateMachine Class Reference . . . . .	55
9.48	QCaVariableNamePropertyManager Class Reference . . . . .	55
9.49	QEAnalogIndicator Class Reference . . . . .	56
9.49.1	Detailed Description . . . . .	59
9.49.2	Member Enumeration Documentation . . . . .	59
9.49.2.1	Modes . . . . .	59
9.49.2.2	Orientations . . . . .	59
9.49.3	Property Documentation . . . . .	60
9.49.3.1	backgroundColour . . . . .	60
9.49.3.2	borderColour . . . . .	60
9.49.3.3	centreAngle . . . . .	60
9.49.3.4	fontColour . . . . .	60
9.49.3.5	foregroundColour . . . . .	60
9.49.3.6	logScale . . . . .	60
9.49.3.7	logScaleInterval . . . . .	60
9.49.3.8	majorInterval . . . . .	60
9.49.3.9	maximum . . . . .	60
9.49.3.10	minimum . . . . .	60
9.49.3.11	minorInterval . . . . .	61
9.49.3.12	mode . . . . .	61
9.49.3.13	orientation . . . . .	61
9.49.3.14	showScale . . . . .	61
9.49.3.15	showText . . . . .	61

9.49.3.16	spanAngle	61
9.49.3.17	value	61
9.50	QEAAnalogProgressBar Class Reference	61
9.50.1	Member Enumeration Documentation	64
9.50.1.1	ArrayActions	64
9.50.1.2	Formats	65
9.50.1.3	Notations	65
9.50.1.4	UserLevels	65
9.50.2	Constructor & Destructor Documentation	65
9.50.2.1	QEAAnalogProgressBar	65
9.50.2.2	QEAAnalogProgressBar	66
9.50.3	Member Function Documentation	66
9.50.3.1	dbValueChanged	66
9.50.3.2	requestEnabled	66
9.50.4	Property Documentation	66
9.50.4.1	addUnits	66
9.50.4.2	alarmSeverityDisplayMode	66
9.50.4.3	allowDrop	66
9.50.4.4	arrayAction	66
9.50.4.5	enabled	67
9.50.4.6	format	67
9.50.4.7	int	67
9.50.4.8	leadingZero	67
9.50.4.9	localEnumeration	67
9.50.4.10	notation	68
9.50.4.11	precision	68
9.50.4.12	trailingZeros	68
9.50.4.13	useDbDisplayLimits	68
9.50.4.14	useDbPrecision	69
9.50.4.15	userLevelEnabled	69
9.50.4.16	userLevelEngineerStyle	69
9.50.4.17	userLevelScientistStyle	69
9.50.4.18	userLevelUserStyle	69
9.50.4.19	userLevelVisibility	69

9.50.4.20 variable . . . . .	70
9.50.4.21 variableAsToolTip . . . . .	70
9.50.4.22 variableSubstitutions . . . . .	70
9.50.4.23 visible . . . . .	70
9.51 QEBitStatus Class Reference . . . . .	70
9.51.1 Member Enumeration Documentation . . . . .	72
9.51.1.1 UserLevels . . . . .	72
9.51.2 Member Function Documentation . . . . .	72
9.51.2.1 dbValueChanged . . . . .	72
9.51.2.2 requestEnabled . . . . .	72
9.51.2.3 setVariableNameAndSubstitutions . . . . .	73
9.51.3 Property Documentation . . . . .	73
9.51.3.1 allowDrop . . . . .	73
9.51.3.2 enabled . . . . .	73
9.51.3.3 int . . . . .	73
9.51.3.4 userLevelEnabled . . . . .	73
9.51.3.5 userLevelEngineerStyle . . . . .	73
9.51.3.6 userLevelScientistStyle . . . . .	74
9.51.3.7 userLevelUserStyle . . . . .	74
9.51.3.8 userLevelVisibility . . . . .	74
9.51.3.9 variable . . . . .	74
9.51.3.10 variableAsToolTip . . . . .	74
9.51.3.11 variableSubstitutions . . . . .	75
9.51.3.12 visible . . . . .	75
9.52 QEByteArray Class Reference . . . . .	75
9.53 QEComboBox Class Reference . . . . .	76
9.53.1 Member Enumeration Documentation . . . . .	78
9.53.1.1 UserLevels . . . . .	78
9.53.2 Member Function Documentation . . . . .	78
9.53.2.1 dbValueChanged . . . . .	78
9.53.2.2 requestEnabled . . . . .	78
9.53.3 Member Data Documentation . . . . .	78
9.53.3.1 writeOnChange . . . . .	78
9.53.4 Property Documentation . . . . .	78

9.53.4.1	allowDrop	79
9.53.4.2	enabled	79
9.53.4.3	int	79
9.53.4.4	subscribe	79
9.53.4.5	userLevelEnabled	79
9.53.4.6	userLevelEngineerStyle	79
9.53.4.7	userLevelScientistStyle	80
9.53.4.8	userLevelUserStyle	80
9.53.4.9	userLevelVisibility	80
9.53.4.10	variable	80
9.53.4.11	variableAsToolTip	80
9.53.4.12	variableSubstitutions	80
9.53.4.13	visible	81
9.54	QEConfiguredLayout Class Reference	81
9.55	QEConfiguredLayoutManager Class Reference	83
9.56	QEDragDrop Class Reference	83
9.57	QEFFileBrowser Class Reference	85
9.58	QEFloating Class Reference	86
9.59	QEFloatingFormatting Class Reference	87
9.60	QEForm Class Reference	88
9.60.1	Member Function Documentation	89
9.60.1.1	setVariableNameAndSubstitutions	89
9.61	QEFrame Class Reference	89
9.61.1	Member Enumeration Documentation	91
9.61.1.1	UserLevels	91
9.61.2	Member Function Documentation	91
9.61.2.1	requestEnabled	91
9.61.3	Property Documentation	91
9.61.3.1	allowDrop	91
9.61.3.2	enabled	91
9.61.3.3	int	91
9.61.3.4	userLevelEnabled	92
9.61.3.5	userLevelEngineerStyle	92
9.61.3.6	userLevelScientistStyle	92



9.61.3.7	<a href="#">userLevelUserStyle</a>	92
9.61.3.8	<a href="#">userLevelVisibility</a>	92
9.61.3.9	<a href="#">variableAsToolTip</a>	93
9.61.3.10	<a href="#">visible</a>	93
9.62	<a href="#">QEGenericButton Class Reference</a>	93
9.63	<a href="#">QEGroupBox Class Reference</a>	95
9.63.1	<a href="#">Member Enumeration Documentation</a>	96
9.63.1.1	<a href="#">UserLevels</a>	96
9.63.2	<a href="#">Member Function Documentation</a>	96
9.63.2.1	<a href="#">requestEnabled</a>	96
9.63.3	<a href="#">Property Documentation</a>	96
9.63.3.1	<a href="#">allowDrop</a>	96
9.63.3.2	<a href="#">enabled</a>	97
9.63.3.3	<a href="#">int</a>	97
9.63.3.4	<a href="#">userLevelEnabled</a>	97
9.63.3.5	<a href="#">userLevelEngineerStyle</a>	97
9.63.3.6	<a href="#">userLevelScientistStyle</a>	97
9.63.3.7	<a href="#">userLevelUserStyle</a>	98
9.63.3.8	<a href="#">userLevelVisibility</a>	98
9.63.3.9	<a href="#">variableAsToolTip</a>	98
9.63.3.10	<a href="#">visible</a>	98
9.64	<a href="#">QEImage Class Reference</a>	98
9.64.1	<a href="#">Member Enumeration Documentation</a>	106
9.64.1.1	<a href="#">formatOptions</a>	106
9.64.1.2	<a href="#">FormatOptions</a>	106
9.64.1.3	<a href="#">ResizeOptions</a>	106
9.64.1.4	<a href="#">resizeOptions</a>	107
9.64.1.5	<a href="#">rotationOptions</a>	107
9.64.1.6	<a href="#">RotationOptions</a>	107
9.64.1.7	<a href="#">selectOptions</a>	107
9.64.1.8	<a href="#">UserLevels</a>	108
9.64.2	<a href="#">Constructor &amp; Destructor Documentation</a>	108
9.64.2.1	<a href="#">QEImage</a>	108
9.64.2.2	<a href="#">QEImage</a>	108

9.64.3	Member Function Documentation	108
9.64.3.1	dbValueChanged	108
9.64.3.2	requestEnabled	108
9.64.4	Member Data Documentation	109
9.64.4.1	displayButtonBar	109
9.64.4.2	initialVertScrollPos	109
9.64.5	Property Documentation	109
9.64.5.1	allowDrop	109
9.64.5.2	areaColor	109
9.64.5.3	beamColor	109
9.64.5.4	beamXVariable	109
9.64.5.5	beamYVariable	109
9.64.5.6	clippingHighVariable	109
9.64.5.7	clippingLowVariable	110
9.64.5.8	clippingOnOffVariable	110
9.64.5.9	enabled	110
9.64.5.10	enableHozSliceSelection	110
9.64.5.11	enableVertSliceSelection	110
9.64.5.12	formatOption	110
9.64.5.13	heightVariable	110
9.64.5.14	horizontalFlip	110
9.64.5.15	hozSliceColor	111
9.64.5.16	imageVariable	111
9.64.5.17	initialHosScrollPos	111
9.64.5.18	int	111
9.64.5.19	profileColor	111
9.64.5.20	regionOfInterest1HVariable	111
9.64.5.21	regionOfInterest1WVariable	111
9.64.5.22	regionOfInterest1XVariable	111
9.64.5.23	regionOfInterest1YVariable	111
9.64.5.24	regionOfInterest2HVariable	112
9.64.5.25	regionOfInterest2WVariable	112
9.64.5.26	regionOfInterest2XVariable	112
9.64.5.27	regionOfInterest2YVariable	112

9.64.5.28	<a href="#">regionOfInterest3HVariable</a>	112
9.64.5.29	<a href="#">regionOfInterest3WVariable</a>	112
9.64.5.30	<a href="#">regionOfInterest3XVariable</a>	112
9.64.5.31	<a href="#">regionOfInterest3YVariable</a>	112
9.64.5.32	<a href="#">regionOfInterest4HVariable</a>	112
9.64.5.33	<a href="#">regionOfInterest4WVariable</a>	113
9.64.5.34	<a href="#">regionOfInterest4XVariable</a>	113
9.64.5.35	<a href="#">regionOfInterest4YVariable</a>	113
9.64.5.36	<a href="#">resizeOption</a>	113
9.64.5.37	<a href="#">rotation</a>	113
9.64.5.38	<a href="#">showTime</a>	113
9.64.5.39	<a href="#">targetColor</a>	113
9.64.5.40	<a href="#">targetTriggerVariable</a>	113
9.64.5.41	<a href="#">targetXVariable</a>	113
9.64.5.42	<a href="#">targetYVariable</a>	114
9.64.5.43	<a href="#">timeColor</a>	114
9.64.5.44	<a href="#">userLevelEnabled</a>	114
9.64.5.45	<a href="#">userLevelEngineerStyle</a>	114
9.64.5.46	<a href="#">userLevelScientistStyle</a>	114
9.64.5.47	<a href="#">userLevelUserStyle</a>	114
9.64.5.48	<a href="#">userLevelVisibility</a>	115
9.64.5.49	<a href="#">variableAsToolTip</a>	115
9.64.5.50	<a href="#">variableSubstitutions</a>	115
9.64.5.51	<a href="#">verticalFlip</a>	115
9.64.5.52	<a href="#">vertSliceColor</a>	115
9.64.5.53	<a href="#">visible</a>	115
9.64.5.54	<a href="#">widthVariable</a>	115
9.65	<a href="#">QEInteger Class Reference</a>	116
9.66	<a href="#">QEIntegerFormatting Class Reference</a>	116
9.66.1	<a href="#">Detailed Description</a>	117
9.66.2	<a href="#">Member Function Documentation</a>	117
9.66.2.1	<a href="#">formatInteger</a>	117
9.66.2.2	<a href="#">formatIntegerArray</a>	117
9.66.2.3	<a href="#">formatValue</a>	118

9.67 QLabel Class Reference . . . . .	118
9.67.1 Detailed Description . . . . .	121
9.67.2 Member Enumeration Documentation . . . . .	122
9.67.2.1 ArrayActions . . . . .	122
9.67.2.2 Formats . . . . .	122
9.67.2.3 Notations . . . . .	122
9.67.2.4 UpdateOptions . . . . .	122
9.67.2.5 updateOptions . . . . .	123
9.67.2.6 UserLevels . . . . .	123
9.67.3 Constructor & Destructor Documentation . . . . .	123
9.67.3.1 QLabel . . . . .	123
9.67.3.2 QLabel . . . . .	123
9.67.4 Member Function Documentation . . . . .	123
9.67.4.1 dbValueChanged . . . . .	123
9.67.4.2 requestEnabled . . . . .	124
9.67.5 Property Documentation . . . . .	124
9.67.5.1 addUnits . . . . .	124
9.67.5.2 allowDrop . . . . .	124
9.67.5.3 arrayAction . . . . .	124
9.67.5.4 enabled . . . . .	124
9.67.5.5 format . . . . .	125
9.67.5.6 int . . . . .	125
9.67.5.7 leadingZero . . . . .	125
9.67.5.8 localEnumeration . . . . .	125
9.67.5.9 notation . . . . .	126
9.67.5.10 pixmap0 . . . . .	126
9.67.5.11 pixmap1 . . . . .	126
9.67.5.12 pixmap2 . . . . .	126
9.67.5.13 pixmap3 . . . . .	126
9.67.5.14 pixmap4 . . . . .	126
9.67.5.15 pixmap5 . . . . .	126
9.67.5.16 pixmap6 . . . . .	126
9.67.5.17 pixmap7 . . . . .	127
9.67.5.18 precision . . . . .	127

9.67.5.19 trailingZeros . . . . .	127
9.67.5.20 updateOption . . . . .	127
9.67.5.21 useDbPrecision . . . . .	127
9.67.5.22 userLevelEnabled . . . . .	127
9.67.5.23 userLevelEngineerStyle . . . . .	127
9.67.5.24 userLevelScientistStyle . . . . .	128
9.67.5.25 userLevelUserStyle . . . . .	128
9.67.5.26 userLevelVisibility . . . . .	128
9.67.5.27 variable . . . . .	128
9.67.5.28 variableAsToolTip . . . . .	128
9.67.5.29 variableSubstitutions . . . . .	128
9.67.5.30 visible . . . . .	129
9.68 QELineEdit Class Reference . . . . .	129
9.68.1 Member Enumeration Documentation . . . . .	131
9.68.1.1 ArrayActions . . . . .	131
9.68.1.2 Formats . . . . .	132
9.68.1.3 Notations . . . . .	132
9.68.1.4 UserLevels . . . . .	132
9.68.2 Constructor & Destructor Documentation . . . . .	132
9.68.2.1 QELineEdit . . . . .	132
9.68.2.2 QELineEdit . . . . .	133
9.68.3 Member Function Documentation . . . . .	133
9.68.3.1 dbValueChanged . . . . .	133
9.68.3.2 getConfirmWrite . . . . .	133
9.68.3.3 getSubscribe . . . . .	133
9.68.3.4 getWriteOnEnter . . . . .	133
9.68.3.5 getWriteOnFinish . . . . .	133
9.68.3.6 getWriteOnLoseFocus . . . . .	133
9.68.3.7 requestEnabled . . . . .	133
9.68.3.8 setConfirmWrite . . . . .	134
9.68.3.9 setSubscribe . . . . .	134
9.68.3.10 setWriteOnEnter . . . . .	134
9.68.3.11 setWriteOnFinish . . . . .	134
9.68.3.12 setWriteOnLoseFocus . . . . .	134

9.68.4	Property Documentation	134
9.68.4.1	addUnits	134
9.68.4.2	allowDrop	134
9.68.4.3	arrayAction	134
9.68.4.4	confirmWrite	135
9.68.4.5	enabled	135
9.68.4.6	format	135
9.68.4.7	int	135
9.68.4.8	leadingZero	135
9.68.4.9	localEnumeration	136
9.68.4.10	notation	136
9.68.4.11	precision	136
9.68.4.12	subscribe	136
9.68.4.13	trailingZeros	137
9.68.4.14	useDbPrecision	137
9.68.4.15	userLevelEnabled	137
9.68.4.16	userLevelEngineerStyle	137
9.68.4.17	userLevelScientistStyle	137
9.68.4.18	userLevelUserStyle	137
9.68.4.19	userLevelVisibility	138
9.68.4.20	variable	138
9.68.4.21	variableAsToolTip	138
9.68.4.22	variableSubstitutions	138
9.68.4.23	visible	138
9.68.4.24	writeOnEnter	138
9.68.4.25	writeOnFinish	138
9.68.4.26	writeOnLoseFocus	139
9.69	QELineEditManager Class Reference	139
9.70	QELink Class Reference	139
9.71	QELog Class Reference	141
9.72	QELogin Class Reference	143
9.73	QEPeriodic Class Reference	145
9.73.1	Member Enumeration Documentation	148
9.73.1.1	UserLevels	148

9.73.2	Member Function Documentation	148
9.73.2.1	dbElementChanged	148
9.73.2.2	dbValueChanged	148
9.73.2.3	requestEnabled	148
9.73.3	Member Data Documentation	149
9.73.3.1	allowDrop	149
9.73.4	Property Documentation	149
9.73.4.1	enabled	149
9.73.4.2	int	149
9.73.4.3	readbackLabelVariable1	149
9.73.4.4	readbackLabelVariable2	149
9.73.4.5	subscribe	149
9.73.4.6	userLevelEnabled	150
9.73.4.7	userLevelEngineerStyle	150
9.73.4.8	userLevelScientistStyle	150
9.73.4.9	userLevelUserStyle	150
9.73.4.10	userLevelVisibility	150
9.73.4.11	variableAsToolTip	151
9.73.4.12	variableSubstitutions	151
9.73.4.13	visible	151
9.73.4.14	writeButtonVariable1	151
9.73.4.15	writeButtonVariable2	151
9.74	QEPeiodicComponentData Class Reference	151
9.75	QEPeiodicTaskMenu Class Reference	152
9.76	QEPeiodicTaskMenuFactory Class Reference	152
9.77	QEpicsPV Class Reference	152
9.78	QEPlot Class Reference	153
9.78.1	Member Enumeration Documentation	157
9.78.1.1	UserLevels	157
9.78.2	Member Function Documentation	157
9.78.2.1	dbValueChanged	157
9.78.2.2	dbValueChangEd	157
9.78.2.3	requestEnabled	158
9.78.3	Member Data Documentation	158

9.78.3.1	allowDrop	158
9.78.4	Property Documentation	158
9.78.4.1	enabled	158
9.78.4.2	int	158
9.78.4.3	userLevelEnabled	158
9.78.4.4	userLevelEngineerStyle	158
9.78.4.5	userLevelScientistStyle	159
9.78.4.6	userLevelUserStyle	159
9.78.4.7	userLevelVisibility	159
9.78.4.8	variable1	159
9.78.4.9	variable2	159
9.78.4.10	variable3	160
9.78.4.11	variable4	160
9.78.4.12	variableAsToolTip	160
9.78.4.13	variableSubstitutions	160
9.78.4.14	visible	160
9.79	QEPushButton Class Reference	160
9.79.1	Member Enumeration Documentation	163
9.79.1.1	ArrayActions	163
9.79.1.2	CreationOptionNames	164
9.79.1.3	Formats	164
9.79.1.4	Notations	164
9.79.1.5	UpdateOptions	165
9.79.1.6	UserLevels	165
9.79.2	Constructor & Destructor Documentation	165
9.79.2.1	QEPushButton	165
9.79.2.2	QEPushButton	165
9.79.3	Member Function Documentation	165
9.79.3.1	clicked	165
9.79.3.2	dbValueChanged	166
9.79.3.3	launchGui	166
9.79.3.4	pressed	166
9.79.3.5	released	166
9.79.3.6	requestEnabled	166



9.79.4	Property Documentation	166
9.79.4.1	addUnits	166
9.79.4.2	alignment	167
9.79.4.3	allowDrop	167
9.79.4.4	altReadbackVariable	167
9.79.4.5	arguments	167
9.79.4.6	arrayAction	167
9.79.4.7	clickCheckedText	167
9.79.4.8	clickText	168
9.79.4.9	confirmAction	168
9.79.4.10	creationOption	168
9.79.4.11	enabled	168
9.79.4.12	format	168
9.79.4.13	guiFile	169
9.79.4.14	int	169
9.79.4.15	labelText	169
9.79.4.16	leadingZero	169
9.79.4.17	localEnumeration	169
9.79.4.18	notation	170
9.79.4.19	password	170
9.79.4.20	pixmap0	170
9.79.4.21	pixmap1	170
9.79.4.22	pixmap2	170
9.79.4.23	pixmap3	171
9.79.4.24	pixmap4	171
9.79.4.25	pixmap5	171
9.79.4.26	pixmap6	171
9.79.4.27	pixmap7	171
9.79.4.28	precision	171
9.79.4.29	pressText	171
9.79.4.30	program	171
9.79.4.31	releaseText	172
9.79.4.32	subscribe	172
9.79.4.33	trailingZeros	172

9.79.4.34	updateOption	172
9.79.4.35	useDbPrecision	172
9.79.4.36	userLevelEnabled	172
9.79.4.37	userLevelEngineerStyle	172
9.79.4.38	userLevelScientistStyle	173
9.79.4.39	userLevelUserStyle	173
9.79.4.40	userLevelVisibility	173
9.79.4.41	variable	173
9.79.4.42	variableAsToolTip	173
9.79.4.43	variableSubstitutions	173
9.79.4.44	visible	174
9.79.4.45	writeOnClick	174
9.79.4.46	writeOnPress	174
9.79.4.47	writeOnRelease	174
9.80	QEPvProperties Class Reference	174
9.80.1	Member Enumeration Documentation	176
9.80.1.1	UserLevels	176
9.80.2	Member Function Documentation	176
9.80.2.1	requestEnabled	176
9.80.3	Property Documentation	177
9.80.3.1	allowDrop	177
9.80.3.2	enabled	177
9.80.3.3	int	177
9.80.3.4	userLevelEnabled	177
9.80.3.5	userLevelEngineerStyle	177
9.80.3.6	userLevelScientistStyle	178
9.80.3.7	userLevelUserStyle	178
9.80.3.8	userLevelVisibility	178
9.80.3.9	variable	178
9.80.3.10	variableAsToolTip	178
9.80.3.11	variableSubstitutions	178
9.80.3.12	visible	179
9.81	QEPvPropertiesManager Class Reference	179
9.82	QERadioButton Class Reference	179

9.82.1	Member Enumeration Documentation	182
9.82.1.1	ArrayActions	182
9.82.1.2	CreationOptionNames	183
9.82.1.3	Formats	183
9.82.1.4	Notations	183
9.82.1.5	UpdateOptions	183
9.82.1.6	UserLevels	184
9.82.2	Constructor & Destructor Documentation	184
9.82.2.1	QERadioButton	184
9.82.2.2	QERadioButton	184
9.82.3	Member Function Documentation	184
9.82.3.1	clicked	184
9.82.3.2	dbValueChanged	184
9.82.3.3	launchGui	184
9.82.3.4	pressed	185
9.82.3.5	released	185
9.82.3.6	requestEnabled	185
9.82.4	Property Documentation	185
9.82.4.1	addUnits	185
9.82.4.2	alignment	185
9.82.4.3	allowDrop	185
9.82.4.4	arguments	186
9.82.4.5	arrayAction	186
9.82.4.6	clickCheckedText	186
9.82.4.7	clickText	186
9.82.4.8	confirmAction	186
9.82.4.9	creationOption	187
9.82.4.10	enabled	187
9.82.4.11	format	187
9.82.4.12	guiFile	187
9.82.4.13	int	187
9.82.4.14	labelText	188
9.82.4.15	leadingZero	188
9.82.4.16	localEnumeration	188

9.82.4.17 notation . . . . .	189
9.82.4.18 password . . . . .	189
9.82.4.19 pixmap0 . . . . .	189
9.82.4.20 pixmap1 . . . . .	189
9.82.4.21 pixmap2 . . . . .	189
9.82.4.22 pixmap3 . . . . .	189
9.82.4.23 pixmap4 . . . . .	189
9.82.4.24 pixmap5 . . . . .	189
9.82.4.25 pixmap6 . . . . .	189
9.82.4.26 pixmap7 . . . . .	190
9.82.4.27 precision . . . . .	190
9.82.4.28 pressText . . . . .	190
9.82.4.29 program . . . . .	190
9.82.4.30 releaseText . . . . .	190
9.82.4.31 subscribe . . . . .	190
9.82.4.32 trailingZeros . . . . .	190
9.82.4.33 updateOption . . . . .	190
9.82.4.34 useDbPrecision . . . . .	191
9.82.4.35 userLevelEnabled . . . . .	191
9.82.4.36 userLevelEngineerStyle . . . . .	191
9.82.4.37 userLevelScientistStyle . . . . .	191
9.82.4.38 userLevelUserStyle . . . . .	191
9.82.4.39 userLevelVisibility . . . . .	191
9.82.4.40 variable . . . . .	192
9.82.4.41 variableAsToolTip . . . . .	192
9.82.4.42 variableSubstitutions . . . . .	192
9.82.4.43 visible . . . . .	192
9.82.4.44 writeOnClick . . . . .	192
9.82.4.45 writeOnPress . . . . .	192
9.82.4.46 writeOnRelease . . . . .	192
9.83 QERecipe Class Reference . . . . .	193
9.84 QEScript Class Reference . . . . .	195
9.85 QEShape Class Reference . . . . .	197
9.85.1 Detailed Description . . . . .	201

9.85.2	Member Enumeration Documentation	201
9.85.2.1	animationOptions	201
9.85.2.2	shapeOptions	201
9.85.2.3	UserLevels	201
9.85.3	Constructor & Destructor Documentation	202
9.85.3.1	QEShape	202
9.85.3.2	QEShape	202
9.85.4	Member Function Documentation	202
9.85.4.1	dbValueChanged1	202
9.85.4.2	dbValueChanged2	202
9.85.4.3	dbValueChanged3	202
9.85.4.4	dbValueChanged4	202
9.85.4.5	dbValueChanged5	202
9.85.4.6	dbValueChanged6	203
9.85.4.7	requestEnabled	203
9.85.5	Property Documentation	203
9.85.5.1	allowDrop	203
9.85.5.2	animation1	203
9.85.5.3	animation2	203
9.85.5.4	animation3	203
9.85.5.5	animation4	203
9.85.5.6	animation5	203
9.85.5.7	animation6	204
9.85.5.8	color1	204
9.85.5.9	color10	204
9.85.5.10	color2	204
9.85.5.11	color3	204
9.85.5.12	color4	204
9.85.5.13	color5	204
9.85.5.14	color6	204
9.85.5.15	color7	204
9.85.5.16	color8	205
9.85.5.17	color9	205
9.85.5.18	enabled	205

9.85.5.19 int	205
9.85.5.20 offset1	205
9.85.5.21 offset2	205
9.85.5.22 offset3	205
9.85.5.23 offset4	206
9.85.5.24 offset5	206
9.85.5.25 offset6	206
9.85.5.26 point1	206
9.85.5.27 point10	206
9.85.5.28 point2	206
9.85.5.29 point3	206
9.85.5.30 point4	206
9.85.5.31 point5	206
9.85.5.32 point6	207
9.85.5.33 point7	207
9.85.5.34 point8	207
9.85.5.35 point9	207
9.85.5.36 scale2	207
9.85.5.37 scale3	207
9.85.5.38 scale4	207
9.85.5.39 scale5	207
9.85.5.40 scale6	207
9.85.5.41 userLevelEnabled	208
9.85.5.42 userLevelEngineerStyle	208
9.85.5.43 userLevelScientistStyle	208
9.85.5.44 userLevelUserStyle	208
9.85.5.45 userLevelVisibility	208
9.85.5.46 variable1	209
9.85.5.47 variable2	209
9.85.5.48 variable3	209
9.85.5.49 variable4	209
9.85.5.50 variable5	209
9.85.5.51 variable6	209
9.85.5.52 variableAsToolTip	209

9.85.5.53 variableSubstitutions . . . . .	210
9.85.5.54 visible . . . . .	210
9.86 QESlider Class Reference . . . . .	210
9.86.1 Member Enumeration Documentation . . . . .	212
9.86.1.1 UserLevels . . . . .	212
9.86.2 Member Function Documentation . . . . .	212
9.86.2.1 dbValueChanged . . . . .	212
9.86.2.2 requestEnabled . . . . .	212
9.86.3 Member Data Documentation . . . . .	213
9.86.3.1 writeOnChange . . . . .	213
9.86.4 Property Documentation . . . . .	213
9.86.4.1 allowDrop . . . . .	213
9.86.4.2 enabled . . . . .	213
9.86.4.3 int . . . . .	213
9.86.4.4 subscribe . . . . .	213
9.86.4.5 userLevelEnabled . . . . .	213
9.86.4.6 userLevelEngineerStyle . . . . .	214
9.86.4.7 userLevelScientistStyle . . . . .	214
9.86.4.8 userLevelUserStyle . . . . .	214
9.86.4.9 userLevelVisibility . . . . .	214
9.86.4.10 variable . . . . .	214
9.86.4.11 variableAsToolTip . . . . .	215
9.86.4.12 variableSubstitutions . . . . .	215
9.86.4.13 visible . . . . .	215
9.87 QESpinBox Class Reference . . . . .	215
9.87.1 Member Enumeration Documentation . . . . .	217
9.87.1.1 UserLevels . . . . .	217
9.87.2 Member Function Documentation . . . . .	217
9.87.2.1 dbValueChanged . . . . .	217
9.87.2.2 requestEnabled . . . . .	218
9.87.3 Property Documentation . . . . .	218
9.87.3.1 allowDrop . . . . .	218
9.87.3.2 enabled . . . . .	218
9.87.3.3 int . . . . .	218

9.87.3.4	<a href="#">subscribe</a>	218
9.87.3.5	<a href="#">userLevelEnabled</a>	218
9.87.3.6	<a href="#">userLevelEngineerStyle</a>	219
9.87.3.7	<a href="#">userLevelScientistStyle</a>	219
9.87.3.8	<a href="#">userLevelUserStyle</a>	219
9.87.3.9	<a href="#">userLevelVisibility</a>	219
9.87.3.10	<a href="#">variable</a>	219
9.87.3.11	<a href="#">variableAsToolTip</a>	219
9.87.3.12	<a href="#">variableSubstitutions</a>	220
9.87.3.13	<a href="#">visible</a>	220
9.88	<a href="#">QString Class Reference</a>	220
9.89	<a href="#">QStringFormatting Class Reference</a>	221
9.89.1	<a href="#">Member Enumeration Documentation</a>	222
9.89.1.1	<a href="#">arrayActions</a>	222
9.89.1.2	<a href="#">formats</a>	222
9.89.1.3	<a href="#">notations</a>	222
9.90	<a href="#">QStringFormattingMethods Class Reference</a>	223
9.91	<a href="#">QStripChart Class Reference</a>	224
9.92	<a href="#">QStripChartItem Class Reference</a>	226
9.93	<a href="#">QStripChartItemDialog Class Reference</a>	226
9.94	<a href="#">QStripChartTimeDialog Class Reference</a>	227
9.95	<a href="#">QSubstitutedLabel Class Reference</a>	227
9.95.1	<a href="#">Member Data Documentation</a>	228
9.95.1.1	<a href="#">labelText</a>	228
9.95.2	<a href="#">Property Documentation</a>	228
9.95.2.1	<a href="#">textSubstitutions</a>	228
9.96	<a href="#">QEToolTip Class Reference</a>	228
9.97	<a href="#">QEWidget Class Reference</a>	230
9.97.1	<a href="#">Detailed Description</a>	231
9.97.2	<a href="#">Member Function Documentation</a>	232
9.97.2.1	<a href="#">activate</a>	232
9.97.2.2	<a href="#">defaultFileLocation</a>	232
9.97.2.3	<a href="#">getColor</a>	232
9.97.2.4	<a href="#">getMessageSourceId</a>	233



9.97.2.5	<a href="#">getQcaltem</a>	233
9.97.2.6	<a href="#">openQEFile</a>	233
9.97.2.7	<a href="#">readNow</a>	233
9.97.2.8	<a href="#">setMessageSourceId</a>	233
9.97.2.9	<a href="#">setupContextMenu</a>	233
9.97.2.10	<a href="#">setVariableNameAndSubstitutions</a>	233
9.97.2.11	<a href="#">writeNow</a>	233
9.98	<a href="#">QEWidgets Class Reference</a>	234
9.99	<a href="#">qcastatemachine::ReadQCaStateMachine Class Reference</a>	234
9.100	<a href="#">RecordSpec Class Reference</a>	234
9.101	<a href="#">RecordSpecList Class Reference</a>	235
9.102	<a href="#">selectMenu Class Reference</a>	235
9.103	<a href="#">standardProperties Class Reference</a>	235
9.104	<a href="#">StateMachineTemplate Class Reference</a>	237
9.105	<a href="#">qcastatemachine::SubscriptionQCaStateMachine Class Reference</a>	237
9.106	<a href="#">trace Class Reference</a>	238
9.107	<a href="#">TrackRange Class Reference</a>	238
9.108	<a href="#">userInfoStruct Class Reference</a>	239
9.109	<a href="#">QEPeriodic::userInfoStructArray Struct Reference</a>	239
9.110	<a href="#">userLevelSignal Class Reference</a>	239
9.111	<a href="#">userLevelSlot Class Reference</a>	240
9.112	<a href="#">UserMessage Class Reference</a>	240
9.112.1	<a href="#">Detailed Description</a>	242
9.113	<a href="#">UserMessageSignal Class Reference</a>	243
9.113.1	<a href="#">Detailed Description</a>	244
9.114	<a href="#">UserMessageSlot Class Reference</a>	244
9.114.1	<a href="#">Detailed Description</a>	244
9.115	<a href="#">VideoWidget Class Reference</a>	245
9.116	<a href="#">WidgetRef Class Reference</a>	246
9.117	<a href="#">qcastatemachine::WriteQCaStateMachine Class Reference</a>	246
9.118	<a href="#">zoomMenu Class Reference</a>	246



## Chapter 1

# QE framework - EPICS aware Qt Widgets and data access classes

- QE is a layered software framework for accessing EPICS data using Channel Access on a range of platforms.
- The QE framework provides object oriented C++ access to control systems using EPICS (Experimental Physics and Industrial Control System). It is based on Qt, a widely used cross-platform application development framework.
- GUI or console based applications can be written that use QE at several levels. QE includes Qt plugin libraries, EPICS aware widgets, data formatting classes, and classes for accessing raw EPICS data in a Qt friendly way.
- QE also includes an application - QEgui - for displaying forms produced by the Qt development tool 'Designer'. Using this application a complete EPICS GUI system can be generated without writing any code. A GUI system produced in this way can interact with existing EPICS display tools such as EDM.
- QE handles much of the complexities of Channel Access including initiating and managing a channel. Applications using QE can interact with Channel Access using Qt based classes and data types. Channel Access updates are delivered using Qt's signals and slots mechanism.

### 1.1 Documentation

Support documents can be found in the [documentation](#) section of the epicsqt sourceforge project. The framework download (available on the epicsqt sourceforge [homepage](#)) also includes this documentation as well as full Doxygen generated documentation of all the epicsqt classes and widgets.

## 1.2 License

epicsqt is distributed under the terms of the [GNU General Public License](#).

## 1.3 Platforms

epicsqt might be usable in all environments where you find [Qt](#). It is compatible with Qt  $\geq 4.4$ .

## 1.4 Screenshots

- [ASgui screen shots](#)
- [other applications using epicsqt widgets](#)
- [Qt Designer](#)
- [Qt Creator](#)

Screenshots are only available in the HTML docs.

## 1.5 Downloads

Stable releases and development snapshots are available at the epicsqt [project page](#).

For getting a development snapshot from the SVN repository:

```
svn svn co https://epicsqt.svn.sourceforge.net/svnroot/epicsqt epicsqt
```

Alternatively, get a packaged file (epicsqt.tar.gz) from the [epicsqt repository site](#).

## 1.6 Installation

Read [QE\\_GettingStarted.pdf](#) in the documentation for setting up an environment for building or using the epicsqt framework.

To build the framework, open epicsqt.pro in QtCreator, ensure shadow build is turned off, and hit build.

The resultant library libQEPlugin.so will need to be installed or referenced up according to how it is to be used - see [QE\\_GettingStarted.pdf](#) for details.

Any Qt specific queries? start at [the Qt Project](#)

## 1.7 Support

Visit the sourceforge epicsqt [support page](#) for assistance.

## 1.8 Related Projects

[Qwt](#), The core of a Channel Access aware plotting widget.

## 1.9 Credits:

### Authors:

Andrew Rhyder, Anthony Owen, Glenn Jackson

### Project admin:

Andrew Rhyder <[andrew.rhyder@synchrotron.org.au](mailto:andrew.rhyder@synchrotron.org.au)>



## Chapter 2

# GNU General Public License

The EPICS QT Framework is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

The EPICS QT Framework is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with the EPICS QT Framework.

If not, see "<http://www.gnu.org/licenses/>





## Chapter 3

### ASgui screen shots

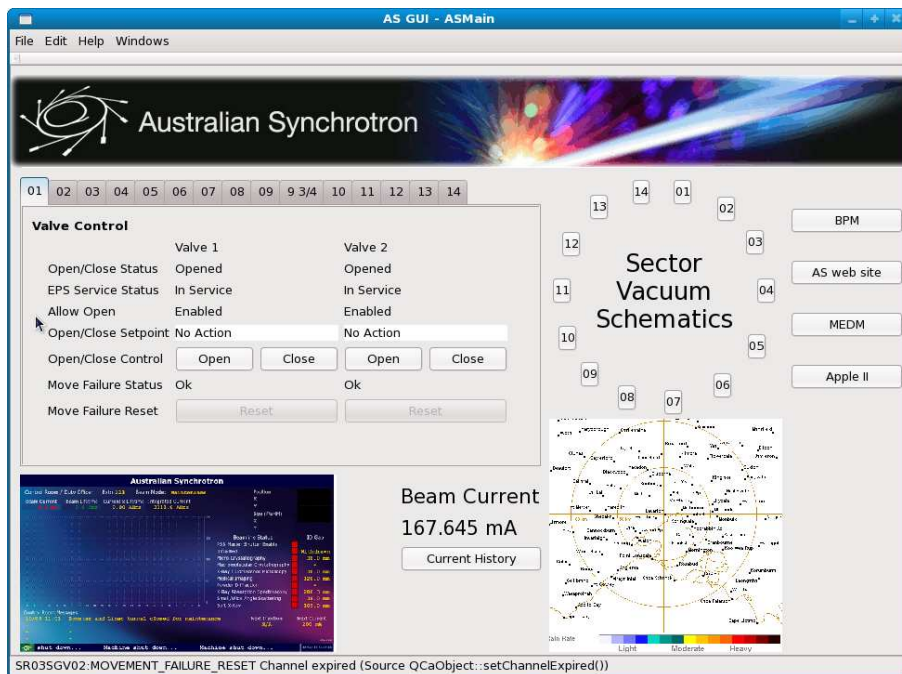


Figure 3.1: Australian Synchrotron mock up

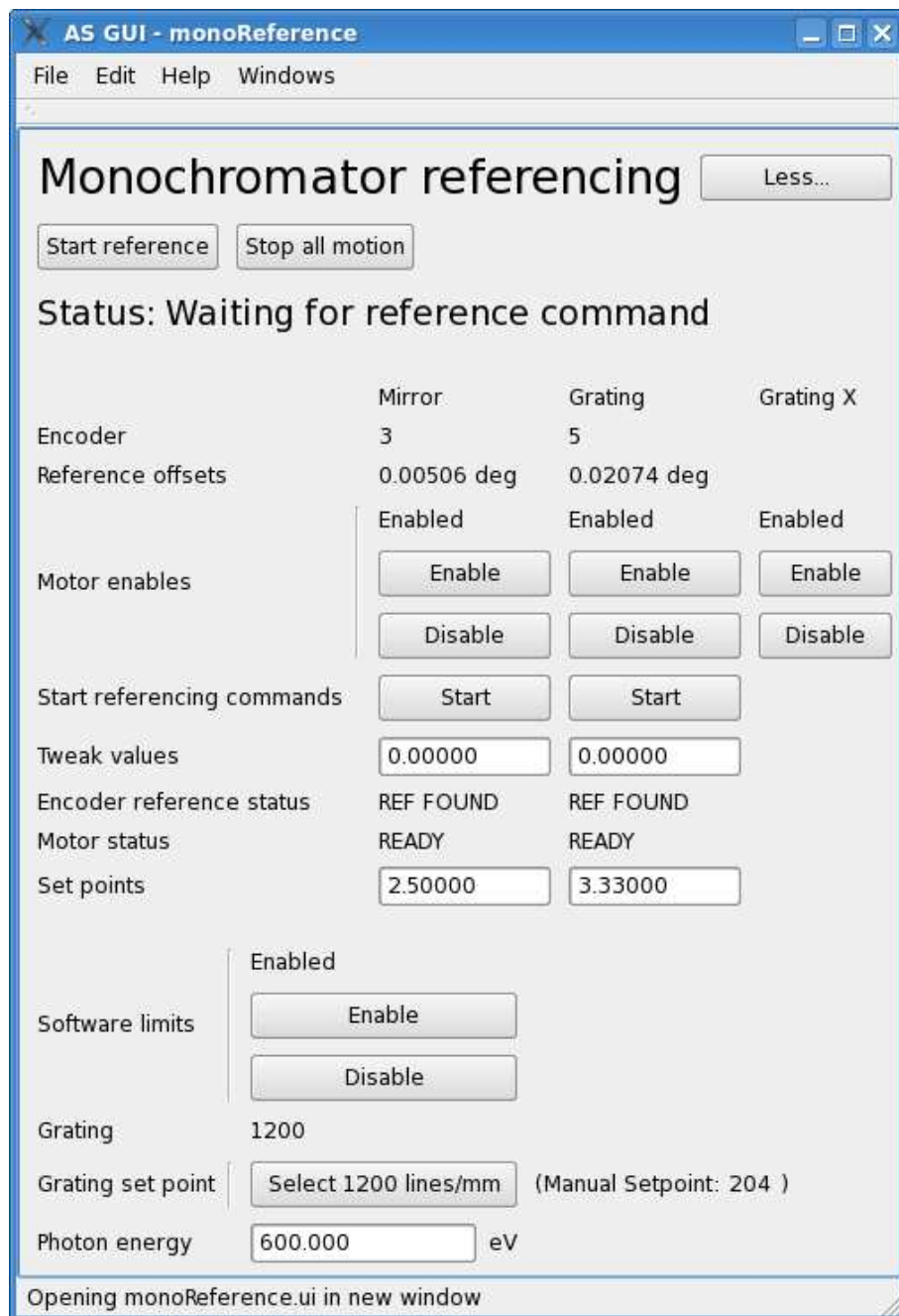


Figure 3.2: Monochromator referencing

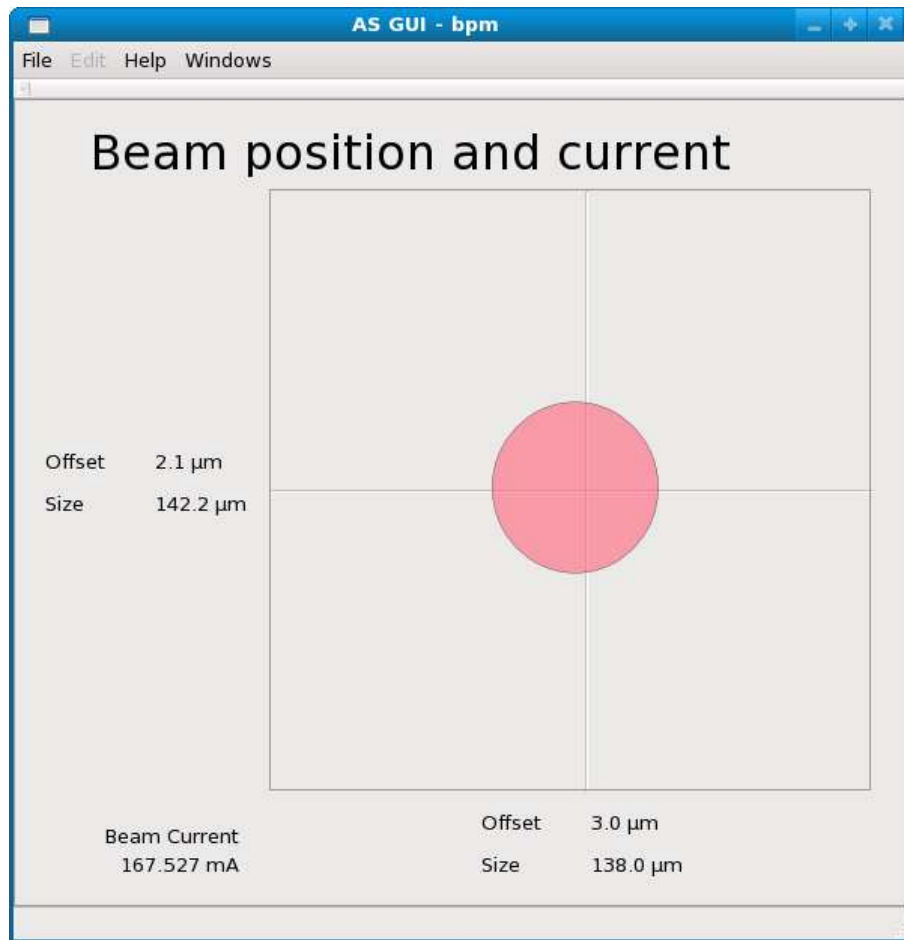


Figure 3.3: Beam position monitor

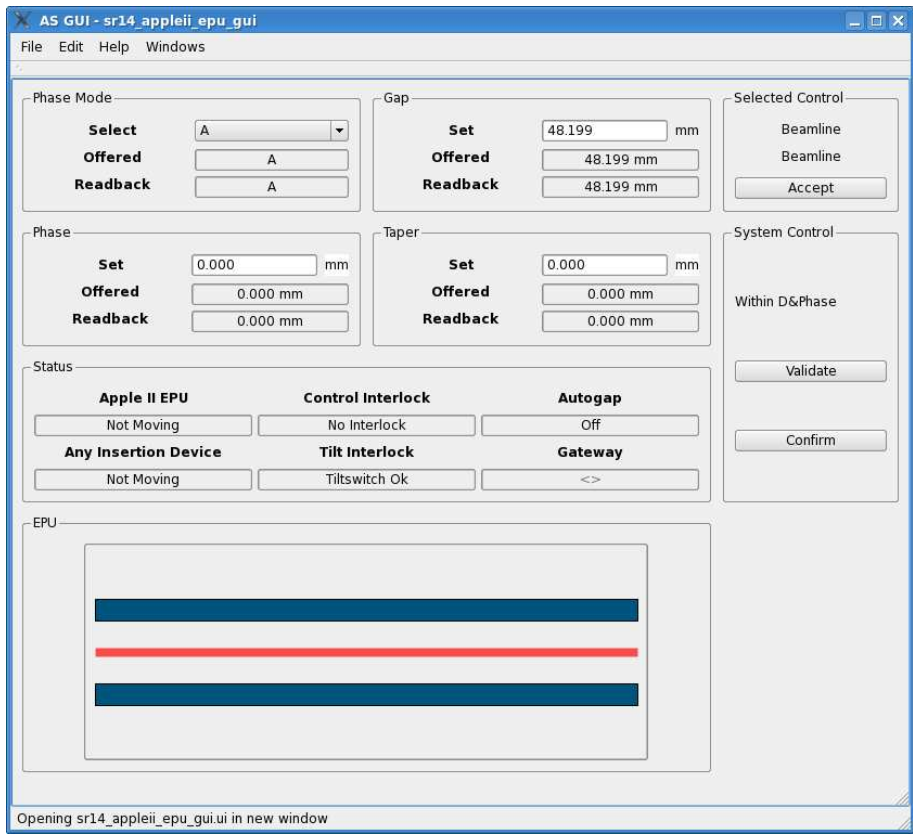


Figure 3.4: Insertion device



Figure 3.5: Injection efficiency monitor



## Chapter 4

# other applications using epicsqt widgets

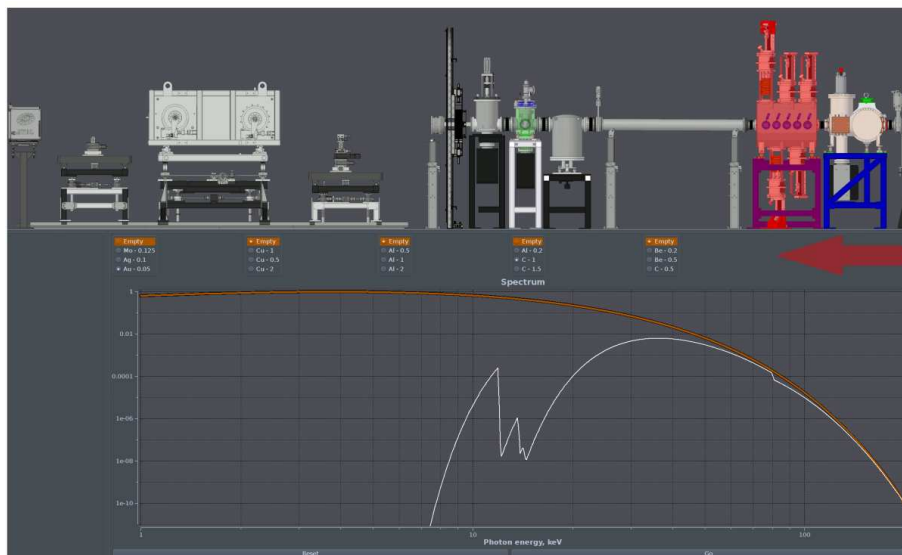


Figure 4.1: Medical Imaging beamline

2B SampleTable Z <@SR08ID01OPI01>

PV name: SR08ID01:MTR32B View mode: Macro

Description: 2B SampleTable Z

Precision: 5 Units: mm

Message: Connection established. clean

User: 6mm Move absolutely Raw: -80932

JOG< UNDO >JOG

LIMIT< Move relatively >LIMIT

< 1mm >

step/10 step/2 step\*2 step\*10

User: 6mm = Hi limit

Resolution: 0.0001mm/step \* 42271.2068mm

Raw: -80932 + Lo limit

Offset: -2.0932mm -42275.3932mm

Speed Acceleration

Maximum: 1.2mm/s

Normal: 0.8mm/s 1s

Backlash: 0mm/s 1s

log: 1mm/s 10s

Backlash: 0mm

Figure 4.2: Motor controller

MotorMx <@SR08ID01OPI01>

- ▲ ▼	DEI Theta Mono	109.5mm	<	0.1	>	UNDO
- ▲ ▼	DEI Mono Z	-0.3mm	<	0.1	>	UNDO
- ▲ ▼	2B Sample Table Y	0mm	<	1	>	UNDO
- ▲ ▼	2B SampleTable Z	6mm	<	1	>	UNDO
- ▲ ▼	2B Detector Table Z	42mm	<	5	>	UNDO
- ▲ ▼	2B Sample Rotate	-1deg	<	1	>	UNDO
- ▲ ▼	2B Detector Table Y	13.9025mm	<	1	>	UNDO
- ▲ ▼	SETUP	0	<	relative	>	STOP
- ▲ ▼	SLW01:LEFT	9.99975mm	<	3456	>	UNDO

Add motor

Figure 4.3: Motor controller



## Chapter 5

# Qt Designer

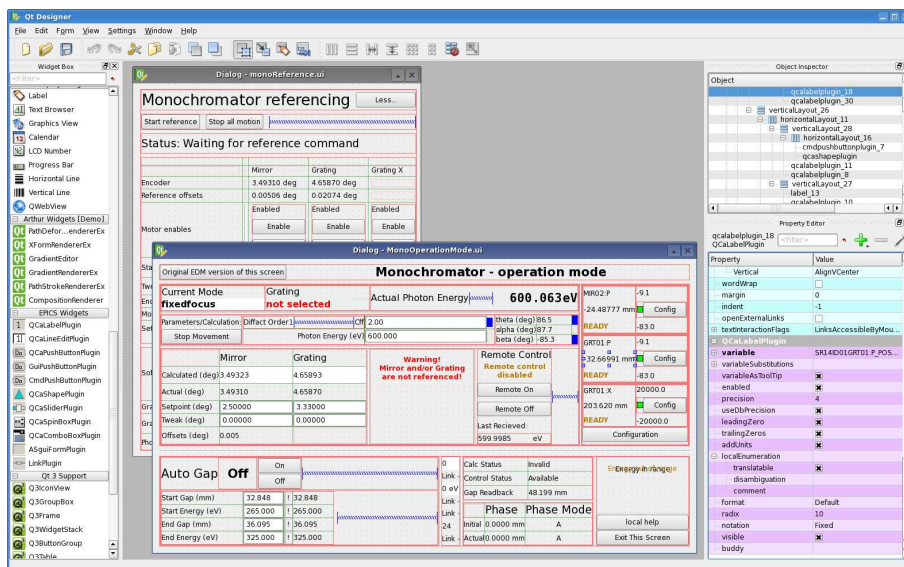


Figure 5.1: Editing multiple GUIs

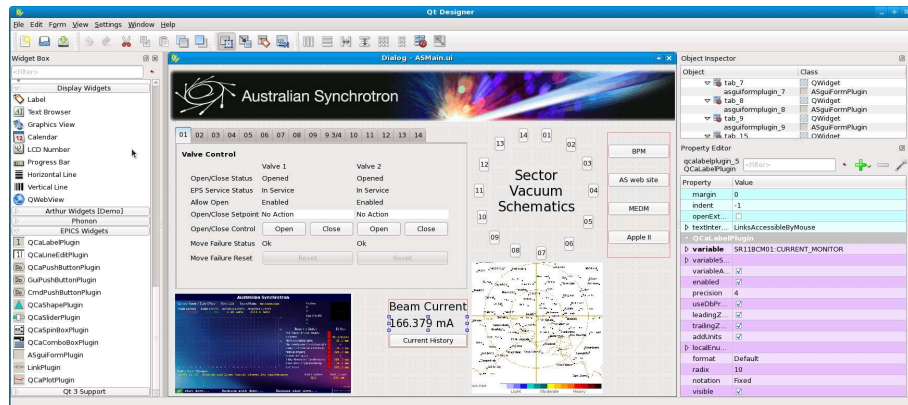


Figure 5.2: Editing a GUI

## Chapter 6

# Qt Creator

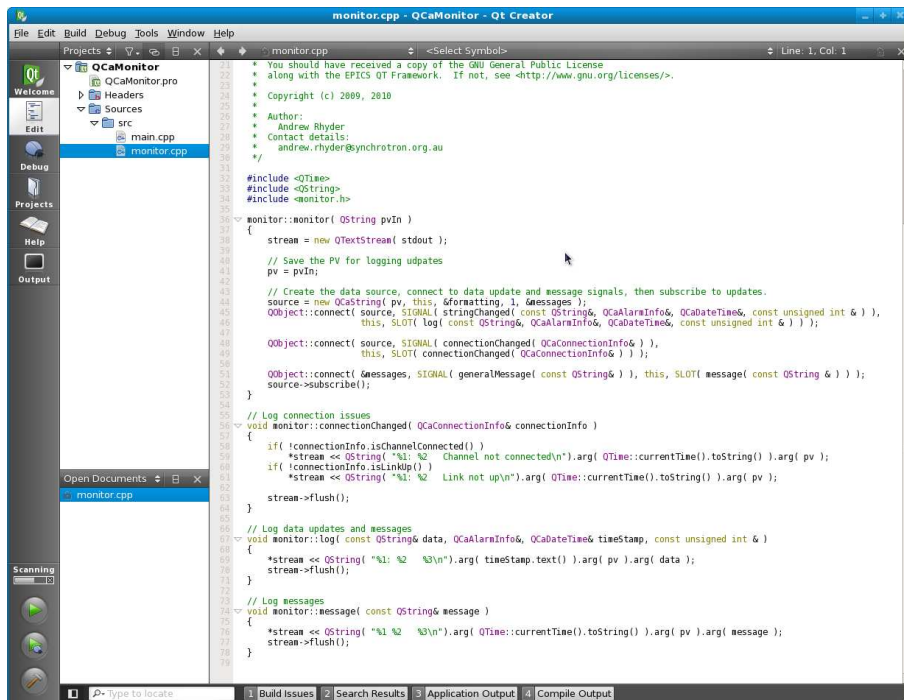


Figure 6.1: Application using epicsqt data source classes



## Chapter 7

# Class Index

### 7.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

_Field . . . . .	27
_Item . . . . .	28
_QDialogItem . . . . .	28
_QDialogLogin . . . . .	28
_QPushButtonGroup . . . . .	29
_QTableWidgetFileBrowser . . . . .	29
_QTableWidgetLog . . . . .	30
_QTableWidgetScript . . . . .	30
QEEAnalogIndicator::Band . . . . .	30
QEEAnalogIndicator::BandList . . . . .	31
ContainerProfile . . . . .	31
QEEWidget . . . . .	230
QEEAnalogProgressBar . . . . .	61
QEEBitStatus . . . . .	70
QEEComboBox . . . . .	76
QEEConfiguredLayout . . . . .	81
QEEFileBrowser . . . . .	85
QEEForm . . . . .	88
QEEFrame . . . . .	89
QEEGenericButton . . . . .	93
QEEPUSHBUTTON . . . . .	160
QEERadioBUTTON . . . . .	179
QEEGroupBox . . . . .	95
QEEImage . . . . .	98
QEELabel . . . . .	118
QEELineEdit . . . . .	129
QEELink . . . . .	139
QEELog . . . . .	141
QEELogin . . . . .	143

QEPeriodic	145
QEPlot	153
QEPvProperties	174
QERecipe	193
QEScript	195
QEShape	197
QESlider	210
QESpinBox	215
QEStripChart	224
QESubstitutedLabel	227
contextMenu	33
QEWidget	230
contextMenuObject	35
QEPeriodic::elementInfoStruct	35
flipRotateMenu	36
imageContextMenu	36
imageMarkup	37
VideoWidget	245
localEnumerationItem	38
managePixmap	39
QEGenericButton	93
QELabel	118
markupItem	41
markupBeam	39
markupHLine	40
markupLine	42
markupRegion	43
markupTarget	44
markupText	44
markupVLine	45
PeriodicDialog	46
PeriodicElementSetupForm	46
PeriodicSetupDialog	46
QEStripChart::PrivateData	47
QEStripChartItem::PrivateData	47
profilePlot	48
PushButtonSpecifications	48
QBitStatus	48
QEBitStatus	70
QCaAlarmInfo	50
QCaConnectionInfo	51
QCaDataPoint	51
QCaDataPointList	51
QCaDateTime	51
QCaEventFilter	52
QCaEventItem	52
QCaEventUpdate	52
QCaInstalledFiltersListItem	53

qcaobject::QCaObject . . . . .	53
QEByteArray . . . . .	75
QEFloating . . . . .	86
QEInteger . . . . .	116
QEString . . . . .	220
QCaVariableNamePropertyManager . . . . .	55
QEAnalogIndicator . . . . .	56
QEAnalogProgressBar . . . . .	61
QEConfiguredLayoutManager . . . . .	83
QEDragDrop . . . . .	83
QEWidget . . . . .	230
QEFloatingFormatting . . . . .	87
QEIntegerFormatting . . . . .	116
QELineEditManager . . . . .	139
QEPeriodicComponentData . . . . .	151
QEPeriodicTaskMenu . . . . .	152
QEPeriodicTaskMenuFactory . . . . .	152
QEpicsPV . . . . .	152
QEPvPropertiesManager . . . . .	179
QEStringFormatting . . . . .	221
QEStringFormattingMethods . . . . .	223
QEAnalogProgressBar . . . . .	61
QEGenericButton . . . . .	93
QELabel . . . . .	118
QELineEdit . . . . .	129
QEStripChartItem . . . . .	226
QEStripChartItemDialog . . . . .	226
QEStripChartTimeDialog . . . . .	227
QEToolTip . . . . .	228
QEWidget . . . . .	230
QEWidgets . . . . .	234
RecordSpec . . . . .	234
RecordSpecList . . . . .	235
selectMenu . . . . .	235
standardProperties . . . . .	235
QEWidget . . . . .	230
StateMachineTemplate . . . . .	237
qcastatemachine::QCaStateMachine . . . . .	55
qcastatemachine::ConnectionQCaStateMachine . . . . .	31
qcastatemachine::ReadQCaStateMachine . . . . .	234
qcastatemachine::SubscriptionQCaStateMachine . . . . .	237
qcastatemachine::WriteQCaStateMachine . . . . .	246
trace . . . . .	238
TrackRange . . . . .	238
userInfoStruct . . . . .	239
QEPeriodic::userInfoStructArray . . . . .	239
userLevelSignal . . . . .	239

userLevelSlot . . . . .	<a href="#">240</a>
UserMessage . . . . .	<a href="#">240</a>
QEWidget . . . . .	<a href="#">230</a>
UserMessageSignal . . . . .	<a href="#">243</a>
UserMessageSlot . . . . .	<a href="#">244</a>
WidgetRef . . . . .	<a href="#">246</a>
zoomMenu . . . . .	<a href="#">246</a>



## Chapter 8

# Class Index

### 8.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

<a href="#">_Field</a>	27
<a href="#">_Item</a>	28
<a href="#">_QDialogItem</a>	28
<a href="#">_QDialogLogin</a>	28
<a href="#">_QPushButtonGroup</a>	29
<a href="#">_QTableWidgetFileBrowser</a>	29
<a href="#">_QTableWidgetLog</a>	30
<a href="#">_QTableWidgetScript</a>	30
<a href="#">QEAnalogIndicator::Band</a>	30
<a href="#">QEAnalogIndicator::BandList</a>	31
<a href="#">qcastatemachine::ConnectionQCaStateMachine</a>	31
<a href="#">ContainerProfile</a>	31
<a href="#">contextMenu</a>	33
<a href="#">contextMenuObject</a>	35
<a href="#">QEPeriodic::elementInfoStruct</a>	35
<a href="#">flipRotateMenu</a>	36
<a href="#">imageContextMenu</a>	36
<a href="#">imageMarkup</a>	37
<a href="#">localEnumerationItem</a>	38
<a href="#">managePixmaps</a>	39
<a href="#">markupBeam</a>	39
<a href="#">markupHLine</a>	40
<a href="#">markupItem</a>	41
<a href="#">markupLine</a>	42
<a href="#">markupRegion</a>	43
<a href="#">markupTarget</a>	44
<a href="#">markupText</a>	44
<a href="#">markupVLine</a>	45
<a href="#">PeriodicDialog</a>	46

PeriodicElementSetupForm	46
PeriodicSetupDialog	46
QEStripChart::PrivateData	47
QEStripChartItem::PrivateData	47
profilePlot	48
PushButtonSpecifications	48
QBitStatus	48
QCaAlarmInfo	50
QCaConnectionInfo	51
QCaDataPoint	51
QCaDataPointList	51
QCaDateTime	51
QCaEventFilter	52
QCaEventItem	52
QCaEventUpdate	52
QCaInstalledFiltersListItem	53
qcaobject::QCaObject	53
qcastatemachine::QCaStateMachine	55
QCaVariableNamePropertyManager	55
QERAnalogIndicator	56
QERAnalogProgressBar	61
QERBitStatus	70
QERByteArray	75
QERComboBox	76
QERConfiguredLayout	81
QERConfiguredLayoutManager	83
QERDragDrop	83
QERFileBrowser	85
QERFloating	86
QERFloatingFormatting	87
QERForm	88
QERFrame	89
QERGenericButton	93
QERGroupBox	95
QERImage	98
QERInteger	116
QERIntegerFormatting	116
QERLabel	118
QERLineEdit	129
QERLineEditManager	139
QERLink	139
QERLog	141
QERLogin	143
QERPeriodic	145
QERPeriodicComponentData	151
QERPeriodicTaskMenu	152
QERPeriodicTaskMenuFactory	152
QERpicsPV	152
QERPlot	153
QERPushButton	160

QEPvProperties	174
QEPvPropertiesManager	179
QERadioButton	179
QERecipe	193
QEScript	195
QEShape	197
QESlider	210
QESpinBox	215
QEStrng	220
QEStrngFormatting	221
QEStrngFormattingMethods	223
QEStrngChart	224
QEStrngChartItem	226
QEStrngChartItemDialog	226
QEStrngChartTimeDialog	227
QESubstitutedLabel	227
QEToolTip	228
QEWidget	230
QEWidgets	234
qcastatemachine::ReadQCaStateMachine	234
RecordSpec	234
RecordSpecList	235
selectMenu	235
standardProperties	235
StateMachineTemplate	237
qcastatemachine::SubscriptionQCaStateMachine	237
trace	238
TrackRange	238
userInfoStruct	239
QEPeriodic::userInfoStructArray	239
userLevelSignal	239
userLevelSlot	240
UserMessage	240
UserMessageSignal	243
UserMessageSlot	244
VideoWidget	245
WidgetRef	246
qcastatemachine::WriteQCaStateMachine	246
zoomMenu	246



## Chapter 9

# Class Documentation

### 9.1 \_Field Class Reference

#### Public Member Functions

- [QEObject](#) \* **getObject** ()
- void **setObject** (QString \*pValue)
- QString **getName** ()
- void **setName** (QString pValue)
- QString **getProcessVariable** ()
- void **setProcessVariable** (QString pValue)
- void **setJoin** (bool pValue)
- bool **getJoin** ()
- int **getType** ()
- void **setType** (int pValue)
- QString **getGroup** ()
- void **setGroup** (QString pValue)
- QString **getVisible** ()
- void **setVisible** (QString pValue)
- QString **getEditable** ()
- void **setEditable** (QString pValue)
- bool **getVisibility** ()
- void **setVisibility** (bool pValue)

#### Public Attributes

- [QEObject](#) \* **qCaWidget**

The documentation for this class was generated from the following files:

- /home/rhydera/epicsqt/framework/widgets/QEObjectifiedLayout/QEObjectifiedLayout.h
- /home/rhydera/epicsqt/framework/widgets/QEObjectifiedLayout/QEObjectifiedLayout.cpp

## 9.2 `_Item` Class Reference

### Public Member Functions

- void **setName** (QString pValue)
- QString **getName** ()
- void **setSubstitution** (QString pValue)
- QString **getSubstitution** ()
- void **setVisible** (QString pValue)
- QString **getVisible** ()

### Public Attributes

- QList< [\\_Field](#) \* > **fieldList**

The documentation for this class was generated from the following files:

- /home/rhydera/epicsqt/framework/widgets/QEConfiguredLayout/QEConfiguredLayout.h
- /home/rhydera/epicsqt/framework/widgets/QEConfiguredLayout/QEConfiguredLayout.cpp

## 9.3 `_QDialogItem` Class Reference

### Public Member Functions

- **\_QDialogItem** (QWidget \*pParent=0, QString pItemName="", QString pGroupName="", QList< [\\_Field](#) \* > \*pCurrentFieldList=0, Qt::WindowFlags pF=0)

The documentation for this class was generated from the following files:

- /home/rhydera/epicsqt/framework/widgets/QEConfiguredLayout/QEConfiguredLayout.h
- /home/rhydera/epicsqt/framework/widgets/QEConfiguredLayout/QEConfiguredLayout.cpp

## 9.4 `_QDialogLogin` Class Reference

### Public Member Functions

- **\_QDialogLogin** (QWidget \*pParent=0, int pUserType=-1, Qt::WindowFlags pF=0)
- void **setCurrentUserType** (int pValue)
- void **setPassword** (QString pValue)

### Protected Attributes

- `QGridLayout` \* **`qGridLayout`**
- `QVBoxLayout` \* **`qVBoxLayout`**
- `QGroupBox` \* **`qGroupBox`**
- `QRadioButton` \* **`qRadioButtonUser`**
- `QRadioButton` \* **`qRadioButtonScientist`**
- `QRadioButton` \* **`qRadioButtonEngineer`**
- `QLabel` \* **`qLabelType`**
- `QLineEdit` \* **`qLineEditPassword`**
- `QPushButton` \* **`qPushButtonOk`**
- `QPushButton` \* **`qPushButtonCancel`**
- `int` **`userType`**

The documentation for this class was generated from the following files:

- `/home/rhydera/epicsqt/framework/widgets/QELogin/QELogin.h`
- `/home/rhydera/epicsqt/framework/widgets/QELogin/QELogin.cpp`

## 9.5 `_QPushButtonGroup` Class Reference

### Public Slots

- `void` **`buttonGroupClicked`** ()

### Public Member Functions

- **`_QPushButtonGroup`** (`QWidget` \*`pParent`=0, `QString` `pltemName`="", `QString` `pGroupName`="", `QList`< `_Field` \* > \*`pCurrentFieldList`=0)
- `void` **`mouseReleaseEvent`** (`QMouseEvent` \*`qMouseEvent`)
- `void` **`keyPressEvent`** (`QKeyEvent` \*`pKeyEvent`)
- `void` **`showDialogGroup`** ()

The documentation for this class was generated from the following files:

- `/home/rhydera/epicsqt/framework/widgets/QEConfiguredLayout/QEConfiguredLayout.h`
- `/home/rhydera/epicsqt/framework/widgets/QEConfiguredLayout/QEConfiguredLayout.cpp`

## 9.6 `_QTableWidgetFileBrowser` Class Reference

### Public Member Functions

- **`_QTableWidgetFileBrowser`** (`QWidget` \*`pParent`=0)

- void **refreshSize** ()
- void **resizeEvent** (QResizeEvent \*)
- void **resize** (int w, int h)

The documentation for this class was generated from the following files:

- /home/rhydera/epicsqt/framework/widgets/QFileBrowser/QFileBrowser.h
- /home/rhydera/epicsqt/framework/widgets/QFileBrowser/QFileBrowser.cpp

## 9.7 \_QTableWidgetLog Class Reference

### Public Member Functions

- **\_QTableWidgetLog** (QWidget \*pParent=0)
- void **refreshSize** ()
- void **resizeEvent** (QResizeEvent \*)
- void **resize** (int w, int h)

The documentation for this class was generated from the following files:

- /home/rhydera/epicsqt/framework/widgets/QELog/QELog.h
- /home/rhydera/epicsqt/framework/widgets/QELog/QELog.cpp

## 9.8 \_QTableWidgetScript Class Reference

### Public Member Functions

- **\_QTableWidgetScript** (QWidget \*pParent=0)
- void **refreshSize** ()
- void **resizeEvent** (QResizeEvent \*)
- void **resize** (int w, int h)

The documentation for this class was generated from the following files:

- /home/rhydera/epicsqt/framework/widgets/QEScript/QEScript.h
- /home/rhydera/epicsqt/framework/widgets/QEScript/QEScript.cpp

## 9.9 QEAnalogIndicator::Band Struct Reference

### Public Attributes

- double **lower**



- double **upper**
- QColor **colour**

The documentation for this struct was generated from the following file:

- /home/rhydera/epicsqt/framework/widgets/QEAnalogIndicator/QEAnalogIndicator.h

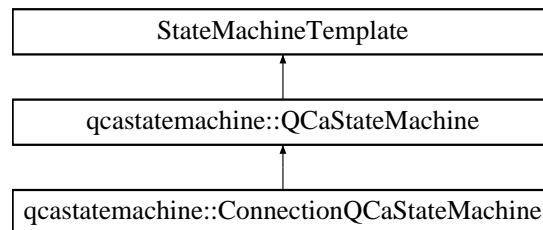
## 9.10 QEAnalogIndicator::BandList Class Reference

The documentation for this class was generated from the following file:

- /home/rhydera/epicsqt/framework/widgets/QEAnalogIndicator/QEAnalogIndicator.h

## 9.11 qcastatemachine::ConnectionQCaStateMachine Class Reference

Inheritance diagram for qcastatemachine::ConnectionQCaStateMachine:



### Public Member Functions

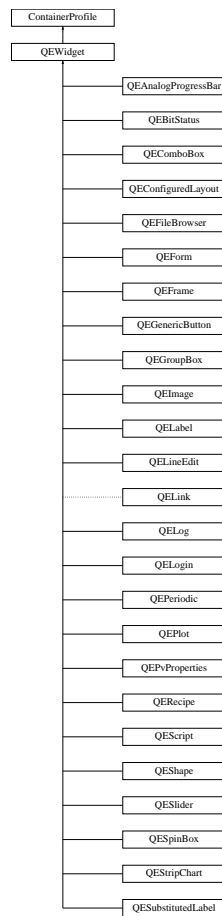
- **ConnectionQCaStateMachine** (void \*parent)
- bool **process** (int requestedState)

The documentation for this class was generated from the following files:

- /home/rhydera/epicsqt/framework/data/include/QCaStateMachine.h
- /home/rhydera/epicsqt/framework/data/src/QCaStateMachine.cpp

## 9.12 ContainerProfile Class Reference

Inheritance diagram for ContainerProfile:



## Public Member Functions

- void **takeLocalCopy** ()
- void **setupProfile** (QObject \*guiLaunchConsumerIn, QStringList pathListIn, QString parentPathIn, QString macroSubstitutionsIn)
- void **setupLocalProfile** (QObject \*guiLaunchConsumerIn, QStringList pathListIn, QString parentPathIn, QString macroSubstitutionsIn)
- void **updateConsumers** (QObject \*guiLaunchConsumerIn)
- QObject \* **replaceGuiLaunchConsumer** (QObject \*newGuiLaunchConsumerIn)
- void **addMacroSubstitutions** (QString macroSubstitutionsIn)
- void **removeMacroSubstitutions** ()
- QObject \* **getGuiLaunchConsumer** ()
- QString **getPath** ()
- QStringList **getPathList** ()
- QString **getParentPath** ()
- void **setPublishedParentPath** (QString publishedParentPathIn)
- QString **getMacroSubstitutions** ()

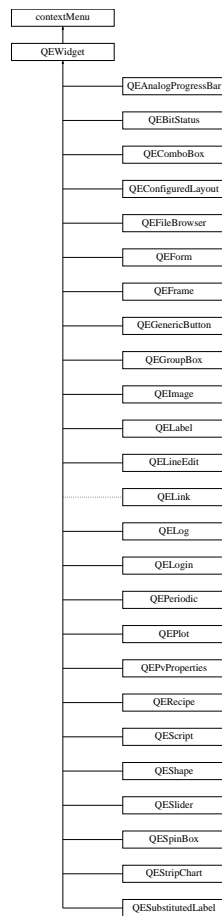
- bool **isProfileDefined** ()
- void **addContainedWidget** (QEWWidget \*containedWidget)
- QEWWidget \* **getNextContainedWidget** ()
- void **removeContainedWidget** (QEWWidget \*containedWidget)
- unsigned int **getMessageFormId** ()
- unsigned int **getPublishedMessageFormId** ()
- void **setPublishedMessageFormId** (unsigned int publishedMessageFormIdIn)
- void **releaseProfile** ()
- void **publishOwnProfile** ()
- void **setUserLevel** (userLevels level)
- userLevels **getUserLevel** ()
- virtual void **userLevelChanged** (userLevels)

The documentation for this class was generated from the following files:

- /home/rhydera/epicsqt/framework/widgets/include/ContainerProfile.h
- /home/rhydera/epicsqt/framework/widgets/src/ContainerProfile.cpp

## 9.13 contextMenu Class Reference

Inheritance diagram for contextMenu:



## Public Types

- enum **contextMenuOptions** {  
**CM\_NONE**, **CM\_COPY\_VARIABLE**, **CM\_COPY\_DATA**, **CM\_PASTE**,  
**CM\_DRAG\_VARIABLE**, **CM\_DRAG\_DATA**, **CM\_SPECIFIC\_WIDGETS\_START\_HERE** }

## Public Member Functions

- void **addContextMenuToWidget** (QWidget \*w)
- bool **isDraggingVariable** ()
- QMenu \* **getContextMenu** ()
- virtual QString **copyVariable** ()
- virtual QVariant **copyData** ()
- virtual void **paste** (QVariant)

### Friends

- class [contextMenuObject](#)

The documentation for this class was generated from the following files:

- /home/rhydera/epicsqt/framework/widgets/include/contextMenu.h
- /home/rhydera/epicsqt/framework/widgets/src/contextMenu.cpp

## 9.14 contextMenuObject Class Reference

### Public Slots

- void **contextMenuTriggered** (QAction \*selectedItem)
- void **showContextMenu** (const QPoint &pos)
- void **setChecked** ()

### Public Member Functions

- void **addContextMenuToWidget** (QWidget \*w)
- void **manageChecked** (bool draggingVariable)
- void **setMenu** ([contextMenu](#) \*menuIn)
- bool **isDraggingVariable** ()

The documentation for this class was generated from the following files:

- /home/rhydera/epicsqt/framework/widgets/include/contextMenu.h
- /home/rhydera/epicsqt/framework/widgets/src/contextMenu.cpp

## 9.15 QEP periodic::elementInfoStruct Struct Reference

### Public Attributes

- unsigned int **number**
- double **atomicWeight**
- QString **name**
- QString **symbol**
- double **meltingPoint**
- double **boilingPoint**
- double **density**
- unsigned int **group**
- double **ionizationEnergy**
- unsigned int **tableRow**

- unsigned int **tableCol**

The documentation for this struct was generated from the following file:

- /home/rhydera/epicsqt/framework/widgets/QEPeriodic/QEPeriodic.h

## 9.16 flipRotateMenu Class Reference

### Public Member Functions

- **flipRotateMenu** (QWidget \*parent=0)
- imageContextMenu::imageContextMenuOptions **getFlipRotate** (const QPoint &pos)
- void **setChecked** (const int rotation, const bool flipH, const bool flipV)

The documentation for this class was generated from the following files:

- /home/rhydera/epicsqt/framework/widgets/QEImage/flipRotateMenu.h
- /home/rhydera/epicsqt/framework/widgets/QEImage/flipRotateMenu.cpp

## 9.17 imageContextMenu Class Reference

### Public Types

- enum **imageContextMenuOptions** {  
**ICM\_NONE** = contextMenu::CM\_SPECIFIC\_WIDGETS\_START\_HERE, **ICM\_SAVE**,  
**ICM\_PAUSE**, **ICM\_ENABLE\_TIME**,  
**ICM\_ENABLE\_CURSOR\_PIXEL**, **ICM\_ENABLE\_CONTRAST\_REVERSAL**, **ICM\_-**  
**ENABLE\_VERT**, **ICM\_ENABLE\_HOZ**,  
**ICM\_ENABLE\_AREA**, **ICM\_ENABLE\_LINE**, **ICM\_ENABLE\_TARGET**, **ICM\_DISPLAY\_-**  
**BUTTON\_BAR**,  
**ICM\_ZOOM\_SELECTED**, **ICM\_ZOOM\_FIT**, **ICM\_ZOOM\_10**, **ICM\_ZOOM\_25**,  
**ICM\_ZOOM\_50**, **ICM\_ZOOM\_75**, **ICM\_ZOOM\_100**, **ICM\_ZOOM\_150**,  
**ICM\_ZOOM\_200**, **ICM\_ZOOM\_300**, **ICM\_ZOOM\_400**, **ICM\_ROTATE\_NONE**,  
**ICM\_ROTATE\_RIGHT**, **ICM\_ROTATE\_LEFT**, **ICM\_ROTATE\_180**, **ICM\_FLIP\_HORIZONTAL**,  
**ICM\_FLIP\_VERTICAL**, **ICM\_SELECT\_PAN**, **ICM\_SELECT\_HSLICE**, **ICM\_SELECT\_-**  
**VSLICE**,  
**ICM\_SELECT\_AREA1**, **ICM\_SELECT\_AREA2**, **ICM\_SELECT\_AREA3**, **ICM\_-**  
**SELECT\_AREA4**,  
**ICM\_SELECT\_PROFILE**, **ICM\_SELECT\_TARGET**, **ICM\_SELECT\_BEAM**, **ICM\_-**  
**CLEAR\_MARKUP** }

### Public Member Functions

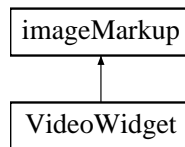
- **imageContextMenu** (QWidget \*parent=0)
- void **getContextMenuOption** (const QPoint &, imageContextMenuOptions \*option, bool \*checked)
- void **addMenuItem** (const QString &title, const bool checkable, const bool checked, const imageContextMenuOptions option)
- void **addOptionMenuItem** (const QString &title, const bool checkable, const bool checked, const imageContextMenuOptions option)

The documentation for this class was generated from the following files:

- /home/rhydera/epicsqt/framework/widgets/QEImage/imageContextMenu.h
- /home/rhydera/epicsqt/framework/widgets/QEImage/imageContextMenu.cpp

## 9.18 imageMarkup Class Reference

Inheritance diagram for imageMarkup:



### Public Types

- enum **markupIds** {  
**MARKUP\_ID\_REGION1**, **MARKUP\_ID\_REGION2**, **MARKUP\_ID\_REGION3**, **MARKUP\_ID\_REGION4**,  
**MARKUP\_ID\_H\_SLICE**, **MARKUP\_ID\_V\_SLICE**, **MARKUP\_ID\_LINE**, **MARKUP\_ID\_TARGET**,  
**MARKUP\_ID\_BEAM**, **MARKUP\_ID\_TIMESTAMP**, **MARKUP\_ID\_COUNT**, **MARKUP\_ID\_NONE** }

### Public Member Functions

- void **setShowTime** (bool visibleIn)
- bool **getShowTime** ()
- markupIds **getMode** ()
- void **setMode** (markupIds modeIn)
- void **setMarkupColor** (markupIds mode, QColor markupColorIn)
- QColor **getMarkupColor** (markupIds mode)

- bool **showMarkupMenu** (const QPoint &pos, const QPoint &globalPos)
- QCursor **getCircleCursor** ()
- QCursor **getTargetCursor** ()
- virtual void **markupSetCursor** (QCursor cursor)=0

### Public Attributes

- QImage \* **markupImage**
- QVector< [markupItem](#) \* > **items**
- QPoint **grabOffset**
- bool **markupAreasStale**
- QFont **legendFont**
- QFontMetrics \* **legendFontMetrics**

### Protected Member Functions

- bool **anyVisibleMarkups** ()
- QVector< QRect > & **getMarkupAreas** ()
- QCursor **getDefaultMarkupCursor** ()
- void **setMarkupTime** ([QCaDateTime](#) &time)
- bool **markupMousePressEvent** (QMouseEvent \*event, bool panning)
- bool **markupMouseReleaseEvent** (QMouseEvent \*event, bool panning)
- bool **markupMouseMoveEvent** (QMouseEvent \*event, bool panning)
- void **markupResize** (QSize newSize)
- virtual void **markupChange** (QImage &markups, QVector< QRect > &changedAreas)=0
- virtual void **markupAction** (markupIds mode, bool complete, bool clearing, QPoint point1, QPoint point2)=0

The documentation for this class was generated from the following files:

- /home/rhydera/epicsqt/framework/widgets/QEImage/imageMarkup.h
- /home/rhydera/epicsqt/framework/widgets/QEImage/imageMarkup.cpp

## 9.19 localEnumerationItem Class Reference

### Public Types

- enum **operations** {  
**LESS, LESS\_EQUAL, EQUAL, NOT\_EQUAL,**  
**GREATER\_EQUAL, GREATER, ALWAYS, UNKNOWN }**



### Public Attributes

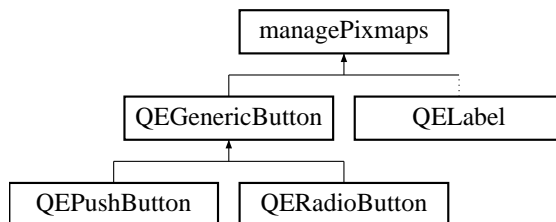
- double **dValue**
- QString **sValue**
- operations **op**
- QString **text**

The documentation for this class was generated from the following file:

- /home/rhydera/epicsqt/framework/data/include/QCStringFormatting.h

## 9.20 managePixmaps Class Reference

Inheritance diagram for managePixmaps:



### Public Member Functions

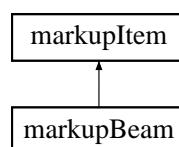
- void **setDataPixmap** (const QPixmap &Pixmap, const unsigned int index)
- QPixmap **getDataPixmap** (const unsigned int index)
- QPixmap **getDataPixmap** (const QString value)

The documentation for this class was generated from the following files:

- /home/rhydera/epicsqt/framework/widgets/include/managePixmaps.h
- /home/rhydera/epicsqt/framework/widgets/src/managePixmaps.cpp

## 9.21 markupBeam Class Reference

Inheritance diagram for markupBeam:



## Public Member Functions

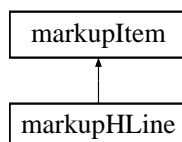
- **markupBeam** ([imageMarkup](#) \*ownerIn, bool interactiveIn, bool reportOnMoveIn, const QString legendIn)
- void **startDrawing** (QPoint pos)
- void **setArea** ()
- void **drawMarkup** (QPainter &p)
- void **moveTo** (QPoint pos)
- bool **isOver** (QPoint point, QCursor \*cursor)
- QPoint **origin** ()
- QPoint **getPoint1** ()
- QPoint **getPoint2** ()
- QCursor **defaultCursor** ()
- void **scaleSpecific** (double xScale, double yScale)

The documentation for this class was generated from the following files:

- /home/rhydera/epicsqt/framework/widgets/QEImage/imageMarkup.h
- /home/rhydera/epicsqt/framework/widgets/QEImage/imageMarkup.cpp

## 9.22 markupHLine Class Reference

Inheritance diagram for markupHLine:



## Public Member Functions

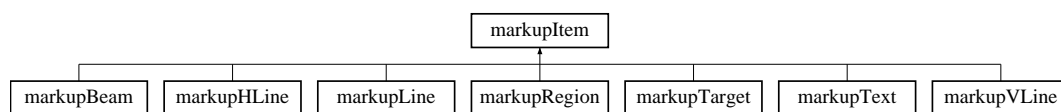
- **markupHLine** ([imageMarkup](#) \*ownerIn, bool interactiveIn, bool reportOnMoveIn, const QString legendIn)
- void **startDrawing** (QPoint pos)
- void **setArea** ()
- void **drawMarkup** (QPainter &p)
- void **moveTo** (QPoint pos)
- bool **isOver** (QPoint point, QCursor \*cursor)
- QPoint **origin** ()
- QPoint **getPoint1** ()
- QPoint **getPoint2** ()
- QCursor **defaultCursor** ()
- void **scaleSpecific** (double xScale, double yScale)

The documentation for this class was generated from the following files:

- /home/rhydera/epicsqt/framework/widgets/QEImage/imageMarkup.h
- /home/rhydera/epicsqt/framework/widgets/QEImage/imageMarkup.cpp

## 9.23 markupItem Class Reference

Inheritance diagram for markupItem:



### Public Member Functions

- void **erase** ()
- void **drawMarkupIn** ()
- void **drawMarkupOut** ()
- void **setColor** (QColor colorIn)
- void **scale** (double xScale, double yScale)
- virtual QPoint **origin** ()=0
- virtual void **moveTo** (QPoint pos)=0
- virtual void **startDrawing** (QPoint pos)=0
- virtual bool **isOver** (const QPoint point, QCursor \*cursor)=0
- virtual QPoint **getPoint1** ()=0
- virtual QPoint **getPoint2** ()=0
- virtual QCursor **defaultCursor** ()=0

### Public Attributes

- QRect **area**
- bool **visible**
- bool **interactive**
- bool **reportOnMove**
- QColor **color**

### Protected Types

- enum **isOverOptions** { **OVER\_LINE**, **OVER\_BORDER**, **OVER\_AREA** }

- enum **markupHandles** {  
**MARKUP\_HANDLE\_NONE**, **MARKUP\_HANDLE\_START**, **MARKUP\_HANDLE\_END**, **MARKUP\_HANDLE\_TL**,  
**MARKUP\_HANDLE\_TR**, **MARKUP\_HANDLE\_BL**, **MARKUP\_HANDLE\_BR**, **MARKUP\_HANDLE\_T**,  
**MARKUP\_HANDLE\_B**, **MARKUP\_HANDLE\_L**, **MARKUP\_HANDLE\_R** }
- enum **legendJustification** { **ABOVE\_RIGHT**, **BELOW\_LEFT**, **BELOW\_RIGHT** }

### Protected Member Functions

- **markupItem** ([imageMarkup](#) \*ownerIn, isOverOptions over, bool interactiveIn, bool reportOnMoveIn, const QString legendIn)
- virtual void **setArea** ()=0
- virtual void **drawMarkup** (QPainter &p)=0
- bool **pointIsNear** (QPoint p1, QPoint p)
- QColor **getColor** ()
- const QString **getLegend** ()
- const QSize **getLegendSize** ()
- void **addLegendArea** ()
- const QPoint **setLegendPos** (QPoint pos, legendJustification just)
- const QPoint **getLegendPos** ()
- void **drawLegend** (QPainter &p, QPoint pos, legendJustification just)

### Protected Attributes

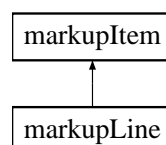
- markupHandles **activeHandle**
- isOverOptions **isOverType**
- bool **highlighted**
- int **highlightMargin**
- [imageMarkup](#) \* **owner**

The documentation for this class was generated from the following files:

- /home/rhydera/epicsqt/framework/widgets/QEImage/imageMarkup.h
- /home/rhydera/epicsqt/framework/widgets/QEImage/imageMarkup.cpp

## 9.24 markupLine Class Reference

Inheritance diagram for markupLine:



### Public Member Functions

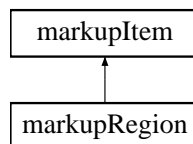
- **markupLine** ([imageMarkup](#) \*ownerIn, bool interactiveIn, bool reportOnMoveIn, const QString legendIn)
- void **startDrawing** (QPoint pos)
- void **setArea** ()
- void **drawMarkup** (QPainter &p)
- void **moveTo** (QPoint pos)
- bool **isOver** (QPoint point, QCursor \*cursor)
- QPoint **origin** ()
- QPoint **getPoint1** ()
- QPoint **getPoint2** ()
- QCursor **defaultCursor** ()
- void **scaleSpecific** (double xScale, double yScale)

The documentation for this class was generated from the following files:

- /home/rhydera/epicsqt/framework/widgets/QEImage/imageMarkup.h
- /home/rhydera/epicsqt/framework/widgets/QEImage/imageMarkup.cpp

## 9.25 markupRegion Class Reference

Inheritance diagram for markupRegion:



### Public Member Functions

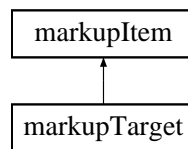
- **markupRegion** ([imageMarkup](#) \*ownerIn, bool interactiveIn, bool reportOnMoveIn, const QString legendIn)
- void **startDrawing** (QPoint pos)
- void **setArea** ()
- void **drawMarkup** (QPainter &p)
- void **moveTo** (QPoint pos)
- bool **isOver** (QPoint point, QCursor \*cursor)
- QPoint **origin** ()
- QPoint **getPoint1** ()
- QPoint **getPoint2** ()
- QCursor **defaultCursor** ()
- void **scaleSpecific** (double xScale, double yScale)

The documentation for this class was generated from the following files:

- /home/rhydera/epicsqt/framework/widgets/QEImage/imageMarkup.h
- /home/rhydera/epicsqt/framework/widgets/QEImage/imageMarkup.cpp

## 9.26 markupTarget Class Reference

Inheritance diagram for markupTarget:



### Public Member Functions

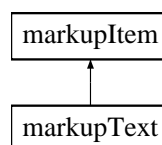
- **markupTarget** ([imageMarkup](#) \*ownerIn, bool interactiveIn, bool reportOnMoveIn, const QString legendIn)
- void **startDrawing** (QPoint pos)
- void **setArea** ()
- void **drawMarkup** (QPainter &p)
- void **moveTo** (QPoint pos)
- bool **isOver** (QPoint point, QCursor \*cursor)
- QPoint **origin** ()
- QPoint **getPoint1** ()
- QPoint **getPoint2** ()
- QCursor **defaultCursor** ()
- void **scaleSpecific** (double xScale, double yScale)

The documentation for this class was generated from the following files:

- /home/rhydera/epicsqt/framework/widgets/QEImage/imageMarkup.h
- /home/rhydera/epicsqt/framework/widgets/QEImage/imageMarkup.cpp

## 9.27 markupText Class Reference

Inheritance diagram for markupText:



### Public Member Functions

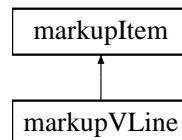
- **markupText** ([imageMarkup](#) \*ownerIn, bool interactiveIn, bool reportOnMoveIn, const QString legendIn)
- void **setText** (QString textIn, bool draw)
- void **startDrawing** (QPoint pos)
- void **setArea** ()
- void **drawMarkup** (QPainter &p)
- void **moveTo** (QPoint pos)
- bool **isOver** (QPoint point, QCursor \*cursor)
- QPoint **origin** ()
- QPoint **getPoint1** ()
- QPoint **getPoint2** ()
- QCursor **defaultCursor** ()
- void **scaleSpecific** (double xScale, double yScale)

The documentation for this class was generated from the following files:

- /home/rhydera/epicsqt/framework/widgets/QEImage/imageMarkup.h
- /home/rhydera/epicsqt/framework/widgets/QEImage/imageMarkup.cpp

## 9.28 markupVLine Class Reference

Inheritance diagram for markupVLine:



### Public Member Functions

- **markupVLine** ([imageMarkup](#) \*ownerIn, bool interactiveIn, bool reportOnMoveIn, const QString legendIn)
- void **startDrawing** (QPoint pos)
- void **setArea** ()
- void **drawMarkup** (QPainter &p)
- void **moveTo** (QPoint pos)
- bool **isOver** (QPoint point, QCursor \*cursor)
- QPoint **origin** ()
- QPoint **getPoint1** ()
- QPoint **getPoint2** ()
- QCursor **defaultCursor** ()

- void **scaleSpecific** (double xScale, double yScale)

The documentation for this class was generated from the following files:

- /home/rhydera/epicsqt/framework/widgets/QEImage/imageMarkup.h
- /home/rhydera/epicsqt/framework/widgets/QEImage/imageMarkup.cpp

## 9.29 PeriodicDialog Class Reference

### Public Member Functions

- **PeriodicDialog** (QWidget \*parent=0)
- QString **getElement** ()
- void **setElement** (QString elementIn, QList< bool > &enabledList, QList< QString > &elementList)

### Protected Member Functions

- void **changeEvent** (QEvent \*e)

The documentation for this class was generated from the following files:

- /home/rhydera/epicsqt/framework/widgets/QEPeriodic/PeriodicDialog.h
- /home/rhydera/epicsqt/framework/widgets/QEPeriodic/PeriodicDialog.cpp

## 9.30 PeriodicElementSetupForm Class Reference

### Public Member Functions

- **PeriodicElementSetupForm** (QWidget \*parent=0)

The documentation for this class was generated from the following files:

- /home/rhydera/epicsqt/framework/widgets/QEPeriodic/PeriodicElementSetupForm.h
- /home/rhydera/epicsqt/framework/widgets/QEPeriodic/PeriodicElementSetupForm.cpp

## 9.31 PeriodicSetupDialog Class Reference

### Public Member Functions

- **PeriodicSetupDialog** (QWidget \*parent=0)



The documentation for this class was generated from the following files:

- /home/rhydera/epicsqt/framework/widgets/QEPeriodic/PeriodicSetupDialog.h
- /home/rhydera/epicsqt/framework/widgets/QEPeriodic/PeriodicSetupDialog.cpp

## 9.32 QEStripChart::PrivateData Class Reference

### Public Member Functions

- **PrivateData** ([QEStripChart](#) \*chartIn)
- [QEStripChartItem](#) \* **getItem** (unsigned int slot)
- [QwtPlotCurve](#) \* **allocateCurve** ()
- void **calcDisplayMinMax** ()
- void **plotData** ()
- void **setReadOut** (QString text)

### Public Attributes

- enum ChartYScale **chartYScale**
- enum ChartTimeMode **chartTimeMode**

### Protected Member Functions

- bool **eventFilter** (QObject \*obj, QEvent \*event)

The documentation for this class was generated from the following file:

- /home/rhydera/epicsqt/framework/widgets/QEStripChart/QEStripChart.cpp

## 9.33 QEStripChartItem::PrivateData Class Reference

### Public Attributes

- [QEStripChart](#) \* **chart**
- [QLabel](#) \* **pvName**
- [QELabel](#) \* **caLabel**

The documentation for this class was generated from the following file:

- /home/rhydera/epicsqt/framework/widgets/QEStripChart/QEStripChartItem.cpp

### 9.34 profilePlot Class Reference

#### Public Member Functions

- **profilePlot** (QWidget \*parent=0)
- void **setProfile** (QVector< QPointF > &profile, double minX, double maxX, double minY, double maxY)

The documentation for this class was generated from the following files:

- /home/rhydera/epicsqt/framework/widgets/QEImage/profilePlot.h
- /home/rhydera/epicsqt/framework/widgets/QEImage/profilePlot.cpp

### 9.35 PushButtonSpecifications Struct Reference

#### Public Attributes

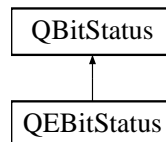
- int **width**
- const QString **caption**
- const QString **iconName**
- const QString **toolTip**
- const char \* **member**

The documentation for this struct was generated from the following file:

- /home/rhydera/epicsqt/framework/widgets/QEStripChart/QEStripChart.cpp

### 9.36 QBitStatus Class Reference

Inheritance diagram for QBitStatus:



#### Public Types

- enum **Orientations** { **LSB\_On\_Right**, **LSB\_On\_Bottom**, **LSB\_On\_Left**, **LSB\_On\_Top** }
- enum **Shapes** { **Rectangle**, **Circle** }

## Public Slots

- void **setValue** (const int value)

## Public Member Functions

- **QBitStatus** (QWidget \*parent=0)
- virtual QSize **sizeHint** () const
- void **setBorderColour** (const QColor value)
- QColor **getBorderColour** ()
- void **setOnColour** (const QColor value)
- QColor **getOnColour** ()
- void **setOffColour** (const QColor value)
- QColor **getOffColour** ()
- void **setInvalidColour** (const QColor value)
- QColor **getInvalidColour** ()
- void **setClearColour** (const QColor value)
- QColor **getClearColour** ()
- void **setDrawBorder** (const bool value)
- bool **getDrawBorder** ()
- void **setNumberOfBits** (const int value)
- int **getNumberOfBits** ()
- void **setGap** (const int value)
- int **getGap** ()
- void **setShift** (const int value)
- int **getShift** ()
- void **setOnClearMask** (const QString value)
- QString **getOnClearMask** ()
- void **setOffClearMask** (const QString value)
- QString **getOffClearMask** ()
- void **setReversePolarityMask** (const QString value)
- QString **getReversePolarityMask** ()
- void **setIsValid** (const bool value)
- bool **getIsValid** ()
- void **setOrientation** (const enum Orientations value)
- enum Orientations **getOrientation** ()
- void **setShape** (const enum Shapes value)
- enum Shapes **getShape** ()
- int **getValue** ()

## Properties

- int **value**
- int **numberOfBits**
- int **shift**
- Orientations **Orientation**
- Shapes **shape**
- int **gap**
- QString **reversePolarityMask**
- QString **onClearMask**
- QString **offClearMask**
- QColor **boarderColour**
- QColor **invalidColour**
- QColor **onColour**
- QColor **offColour**
- QColor **clearColour**
- bool **drawBorder**
- bool **isValid**

The documentation for this class was generated from the following files:

- /home/rhydera/epicsqt/framework/widgets/QEBitStatus/QEBitStatus.h
- /home/rhydera/epicsqt/framework/widgets/QEBitStatus/QEBitStatus.cpp

## 9.37 QCaAlarmInfo Class Reference

### Public Member Functions

- **QCaAlarmInfo** (unsigned short statusIn, unsigned short severityIn)
- QString **statusName** ()
- QString **severityName** ()
- bool **isInAlarm** ()
- bool **isMinor** ()
- bool **isMajor** ()
- bool **isInvalid** ()
- QString **style** ()
- QString **getColorName** ()
- QCAALARMINFO\_SEVERITY **getSeverity** ()

### Static Public Member Functions

- static QCAALARMINFO\_SEVERITY **getInvalidSeverity** ()

The documentation for this class was generated from the following files:

- /home/rhydera/epicsqt/framework/data/include/QCaAlarmInfo.h
- /home/rhydera/epicsqt/framework/data/src/QCaAlarmInfo.cpp

## 9.38 QCaConnectionInfo Class Reference

### Public Member Functions

- **QCaConnectionInfo** (unsigned short channelStateIn, unsigned short linkStateIn)
- bool **isChannelConnected** ()
- bool **isLinkUp** ()

The documentation for this class was generated from the following files:

- /home/rhydera/epicsqt/framework/data/include/QCaConnectionInfo.h
- /home/rhydera/epicsqt/framework/data/src/QCaConnectionInfo.cpp

## 9.39 QCaDataPoint Struct Reference

### Public Attributes

- double **value**
- [QCaDateTime](#) **datetime**
- [QCaAlarmInfo](#) **alarm**

The documentation for this struct was generated from the following file:

- /home/rhydera/epicsqt/framework/data/include/QCaDataPoint.h

## 9.40 QCaDataPointList Class Reference

The documentation for this class was generated from the following file:

- /home/rhydera/epicsqt/framework/data/include/QCaDataPoint.h

## 9.41 QCaDateTime Class Reference

### Public Member Functions

- **QCaDateTime** (QDateTime dt)
- void **operator=** (const [QCaDateTime](#) &other)
- **QCaDateTime** (unsigned long seconds, unsigned long nanoseconds)
- QString **text** ()
- double **floating** (QDateTime base)

### Public Attributes

- unsigned long **nSec**

The documentation for this class was generated from the following files:

- /home/rhydera/epicsqt/framework/data/include/QCaDateTime.h
- /home/rhydera/epicsqt/framework/data/src/QCaDateTime.cpp

## 9.42 QCaEventFilter Class Reference

### Public Member Functions

- void **addFilter** (QObject \*objectIn)
- void **deleteFilter** (QObject \*objectIn)
- bool **eventFilter** (QObject \*watched, QEvent \*e)

The documentation for this class was generated from the following files:

- /home/rhydera/epicsqt/framework/data/include/QCaEventFilter.h
- /home/rhydera/epicsqt/framework/data/src/QCaEventFilter.cpp

## 9.43 QCaEventItem Class Reference

### Public Member Functions

- **QCaEventItem** ([QCaEventUpdate](#) \*newEvent)

### Public Attributes

- [QCaEventUpdate](#) \* **event**

The documentation for this class was generated from the following file:

- /home/rhydera/epicsqt/framework/data/include/QCaEventUpdate.h

## 9.44 QCaEventUpdate Class Reference

### Public Member Functions

- **QCaEventUpdate** ([qcaobject::QCaObject](#) \*emitterObjectIn, long newReason, void \*newDataPtr)

### Public Attributes

- bool **acceptThisEvent**
- [qcaobject::QCaObject](#) \* **emitterObject**
- long **reason**
- void \* **dataPtr**

### Static Public Attributes

- static QEvent::Type **EVENT\_UPDATE\_TYPE** = QEvent::User

The documentation for this class was generated from the following files:

- /home/rhydera/epicsqt/framework/data/include/QCaEventUpdate.h
- /home/rhydera/epicsqt/framework/data/src/QCaEventUpdate.cpp

## 9.45 QCalInstalledFiltersListItem Class Reference

### Public Member Functions

- **QCalInstalledFiltersListItem** (QObject \*eventObjectIn)

### Public Attributes

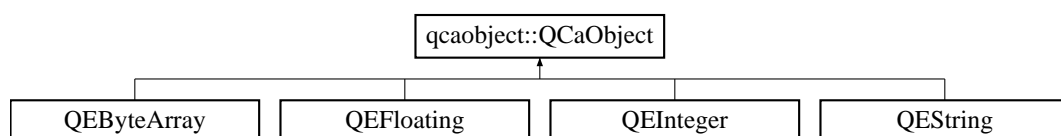
- QObject \* **eventObject**
- long **referenceCount**

The documentation for this class was generated from the following file:

- /home/rhydera/epicsqt/framework/data/include/QCaEventFilter.h

## 9.46 qcaobject::QCaObject Class Reference

Inheritance diagram for qcaobject::QCaObject:



## Public Slots

- bool **writeData** (const QVariant &value)
- void **resendLastData** ()

## Signals

- void **dataChanged** (const QVariant &value, [QCaAlarmInfo](#) &alarmInfo, [QCaDateTime](#) &timeStamp)
- void **dataChanged** (const QByteArray &value, unsigned long dataSize, [QCaAlarmInfo](#) &alarmInfo, [QCaDateTime](#) &timeStamp)
- void **connectionChanged** ([QCaConnectionInfo](#) &connectionInfo)

## Public Member Functions

- **QCaObject** (const QString &recordName, QObject \*eventObject, unsigned char signalsToSendIn=SIG\_VARIANT)
- **QCaObject** (const QString &recordName, QObject \*eventObject, [UserMessage](#) \*userMessageIn, unsigned char signalsToSendIn=SIG\_VARIANT)
- bool **subscribe** ()
- bool **singleShotRead** ()
- bool **dataTypeKnown** ()
- bool **createChannel** ()
- void **deleteChannel** ()
- bool **createSubscription** ()
- bool **getChannel** ()
- bool **putChannel** ()
- bool **isChannelConnected** ()
- void **startConnectionTimer** ()
- void **stopConnectionTimer** ()
- void **setUserMessage** ([UserMessage](#) \*userMessageIn)
- void **enableWriteCallbacks** (bool enable)
- bool **isWriteCallbacksEnabled** ()
- QString **getEgu** ()
- QStringList **getEnumerations** ()
- unsigned int **getPrecision** ()
- double **getDisplayLimitUpper** ()
- double **getDisplayLimitLower** ()
- double **getAlarmLimitUpper** ()
- double **getAlarmLimitLower** ()
- double **getWarningLimitUpper** ()
- double **getWarningLimitLower** ()
- double **getControlLimitUpper** ()
- double **getControlLimitLower** ()
- generic::generic\_types **getDataType** ()



### Static Public Member Functions

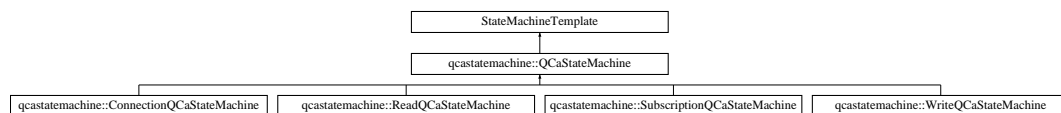
- static void **processEventStatic** (QCaEventUpdate \*dataUpdateEvent)

The documentation for this class was generated from the following files:

- /home/rhydera/epicsqt/framework/data/include/QCaObject.h
- /home/rhydera/epicsqt/framework/data/src/QCaObject.cpp

## 9.47 qcastatemachine::QCaStateMachine Class Reference

Inheritance diagram for qcastatemachine::QCaStateMachine:



### Public Member Functions

- **QCaStateMachine** (void \*parent)
- virtual bool **process** (int requestedState)=0

### Public Attributes

- QMutex **lock**
- bool **pending**
- bool **active**
- bool **expired**
- void \* **myWorker**

The documentation for this class was generated from the following files:

- /home/rhydera/epicsqt/framework/data/include/QCaStateMachine.h
- /home/rhydera/epicsqt/framework/data/src/QCaStateMachine.cpp

## 9.48 QCaVariableNamePropertyManager Class Reference

### Signals

- void **newVariableNameProperty** (QString variable, QString Substitutions, unsigned int variableIndex)

### Public Member Functions

- QString **getVariableNameProperty** ()
- void **setVariableNameProperty** (QString variableNamePropertyIn)
- QString **getSubstitutionsProperty** ()
- void **setSubstitutionsProperty** (QString substitutionsPropertyIn)
- void **setVariableIndex** (unsigned int variableIndexIn)

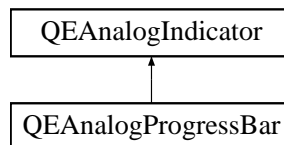
The documentation for this class was generated from the following files:

- /home/rhydera/epicsqt/framework/data/include/QCaVariableNamePropertyManager.h
- /home/rhydera/epicsqt/framework/data/src/QCaVariableNamePropertyManager.cpp

## 9.49 QEAnalogIndicator Class Reference

```
#include <QEAnalogIndicator.h>
```

Inheritance diagram for QEAnalogIndicator:



### Classes

- struct [Band](#)
- class [BandList](#)

### Public Types

- enum [Orientations](#) { [Left\\_To\\_Right](#), [Top\\_To\\_Bottom](#), [Right\\_To\\_Left](#), [Bottom\\_To\\_Top](#) }
- enum [Modes](#) { [Bar](#), [Scale](#), [Meter](#) }

### Public Slots

- void **setRange** (const double MinimumIn, const double MaximumIn)
- void **setValue** (const double ValueIn)

## Public Member Functions

- [QEAnalogIndicator](#) (QWidget \*parent=0)  
*Constructor.*
- virtual [~QEAnalogIndicator](#) ()  
*Destructor.*
- virtual QSize [sizeHint](#) () const  
*Size hint.*
- double [getValue](#) ()  
*Access function for [value](#) property - refer to [value](#) property for details.*
- void [setMinimum](#) (const double value)  
*Access function for [minimum](#) - refer to [minimum](#) property for details.*
- double [getMinimum](#) ()  
*Access function for [minimum](#) - refer to [minimum](#) property for details.*
- void [setMaximum](#) (const double value)  
*Access function for [maximum](#) - refer to [maximum](#) property for details.*
- double [getMaximum](#) ()  
*Access function for [maximum](#) - refer to [maximum](#) property for details.*
- void [setOrientation](#) (const enum [Orientations](#) value)  
*Access function for [orientation](#) - refer to [orientation](#) property for details.*
- enum [Orientations](#) [getOrientation](#) ()  
*Access function for [orientation](#) - refer to [orientation](#) property for details.*
- void [setMode](#) (const enum [Modes](#) value)  
*Access function for [mode](#) - refer to [mode](#) property for details.*
- enum [Modes](#) [getMode](#) ()  
*Access function for [mode](#) - refer to [mode](#) property for details.*
- void [setCentreAngle](#) (const int value)  
*Access function for [centreAngle](#) - refer to [centreAngle](#) property for details.*
- int [getCentreAngle](#) ()  
*Access function for [centreAngle](#) - refer to [centreAngle](#) property for details.*
- void [setSpanAngle](#) (const int value)  
*Access function for [spanAngle](#) - refer to [spanAngle](#) property for details.*
- int [getSpanAngle](#) ()  
*Access function for [spanAngle](#) - refer to [spanAngle](#) property for details.*
- void [setMinorInterval](#) (const double value)  
*Access function for [minorInterval](#) - refer to [minorInterval](#) property for details.*
- double [getMinorInterval](#) ()  
*Access function for [minorInterval](#) - refer to [minorInterval](#) property for details.*
- void [setMajorInterval](#) (const double value)  
*Access function for [majorInterval](#) - refer to [majorInterval](#) property for details.*
- double [getMajorInterval](#) ()  
*Access function for [majorInterval](#) - refer to [majorInterval](#) property for details.*
- void [setLogScaleInterval](#) (const int value)

- Access function for [logScaleInterval](#) - refer to [logScaleInterval](#) property for details.

  - int [getLogScaleInterval](#) ()
- Access function for [logScaleInterval](#) - refer to [logScaleInterval](#) property for details.

  - void [setBorderColour](#) (const QColor value)
- Access function for [borderColour](#) - refer to [borderColour](#) property for details.

  - QColor [getBorderColour](#) ()
- Access function for [borderColour](#) - refer to [borderColour](#) property for details.

  - void [setForegroundColour](#) (const QColor value)
- Access function for [foregroundColour](#) - refer to [foregroundColour](#) property for details.

  - QColor [getForegroundColour](#) ()
- Access function for [foregroundColour](#) - refer to [foregroundColour](#) property for details.

  - void [setBackgroundColour](#) (const QColor value)
- Access function for [backgroundColour](#) - refer to [backgroundColour](#) property for details.

  - QColor [getBackgroundColour](#) ()
- Access function for [backgroundColour](#) - refer to [backgroundColour](#) property for details.

  - void [setFontColour](#) (const QColor value)
- Access function for [fontColour](#) - refer to [fontColour](#) property for details.

  - QColor [getFontColour](#) ()
- Access function for [fontColour](#) - refer to [fontColour](#) property for details.

  - void [setShowText](#) (const bool value)
- Access function for [showText](#) - refer to [showText](#) property for details.

  - bool [getShowText](#) ()
- Access function for [showText](#) - refer to [showText](#) property for details.

  - void [setShowScale](#) (const bool value)
- Access function for [showScale](#) - refer to [showScale](#) property for details.

  - bool [getShowScale](#) ()
- Access function for [showScale](#) - refer to [showScale](#) property for details.

  - void [setLogScale](#) (const bool value)
- Access function for [logScale](#) - refer to [logScale](#) property for details.

  - bool [getLogScale](#) ()
- Access function for [logScale](#) - refer to [logScale](#) property for details.

## Protected Member Functions

- virtual QString [getTextImage](#) ()
- virtual [BandList](#) [getBandList](#) ()

## Properties

- double [value](#)
- double [minimum](#)
- double [maximum](#)
- double [minorInterval](#)

- double [majorInterval](#)
- int [logScaleInterval](#)
- bool [showText](#)
- bool [showScale](#)
- bool [logScale](#)
- [Modes](#) [mode](#)
- [Orientations](#) [orientation](#)
- int [centreAngle](#)
- int [spanAngle](#)
- QColor [borderColour](#)
- QColor [backgroundColour](#)
- QColor [foregroundColour](#)
- QColor [fontColour](#)

### 9.49.1 Detailed Description

This class provides a non CA aware graphical analog indicator base class. It supports a number of display modes including Bar, Scale and Meter.

When in Bar mode, it mimics QProgressBar and provides an analog progress bar widget.

### 9.49.2 Member Enumeration Documentation

#### 9.49.2.1 enum QEAnalogIndicator::Modes

The type of analog indicator used to represent the value

##### Enumerator:

- Bar*** Bar (solid bar from minimum up to current value)
- Scale*** Scale (diamond marker tracks current value)
- Meter*** Meter (Needle moving across an arc scale)

#### 9.49.2.2 enum QEAnalogIndicator::Orientations

The orientation of Bar and Scale indicators

##### Enumerator:

- Left\_To\_Right*** Left to right.
- Top\_To\_Bottom*** Top to bottom.
- Right\_To\_Left*** Right to left.
- Bottom\_To\_Top*** Bottom to top.

### 9.49.3 Property Documentation

9.49.3.1 QColor QEAnalogIndicator::backgroundColour [read, write]

Background colour

9.49.3.2 QColor QEAnalogIndicator::borderColour [read, write]

Border colour

9.49.3.3 int QEAnalogIndicator::centreAngle [read, write]

The angle in degrees of the line that Meter indicators are centered around. Zero represents a vertical centerline and angles increment clockwise.

9.49.3.4 QColor QEAnalogIndicator::fontColour [read, write]

Font colour

9.49.3.5 QColor QEAnalogIndicator::foregroundColour [read, write]

Foreground colour

9.49.3.6 bool QEAnalogIndicator::logScale [read, write]

If set, use a logarithmic scale. If clear, use a linear scale

9.49.3.7 int QEAnalogIndicator::logScaleInterval [read, write]

Log scale interval.

9.49.3.8 double QEAnalogIndicator::majorInterval [read, write]

Minor scale interval. Only applies for linear scale (not log scale)

9.49.3.9 double QEAnalogIndicator::maximum [read, write]

Maximum indicated value.

9.49.3.10 double QEAnalogIndicator::minimum [read, write]

Minimum indicated value.

9.49.3.11 `double QEAnalogIndicator::minorInterval` [read, write]

Minor scale interval. Only applies for linear scale (not log scale)

9.49.3.12 **Modes** `QEAnalogIndicator::mode` [read, write]

Selects what type of indicator is used (refer to Modes)

9.49.3.13 **Orientations** `QEAnalogIndicator::orientation` [read, write]

The orientation of Bar and Scale indicators (refer to Orientations)

9.49.3.14 `bool QEAnalogIndicator::showScale` [read, write]

If set, show the scale

9.49.3.15 `bool QEAnalogIndicator::showText` [read, write]

If set, show textual representation of value on the indicator

9.49.3.16 `int QEAnalogIndicator::spanAngle` [read, write]

The span of the Meter scale arc in degrees Typical meters are 180 deg and 270 deg

9.49.3.17 `double QEAnalogIndicator::value` [read, write]

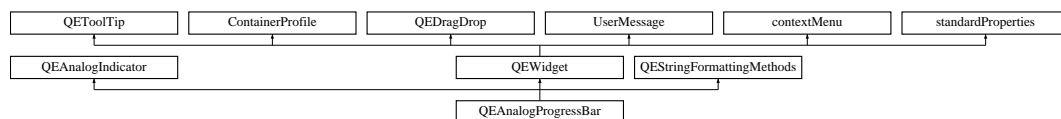
Current indicated value.

The documentation for this class was generated from the following files:

- /home/rhydera/epicsqt/framework/widgets/QEAnalogIndicator/QEAnalogIndicator.h
- /home/rhydera/epicsqt/framework/widgets/QEAnalogIndicator/QEAnalogIndicator.cpp

## 9.50 QEAnalogProgressBar Class Reference

Inheritance diagram for QEAnalogProgressBar:



## Public Types

- enum [UserLevels](#) { [User](#) = USERLEVEL\_USER, [Scientist](#) = USERLEVEL\_SCIENTIST, [Engineer](#) = USERLEVEL\_ENGINEER }
- enum [Formats](#) {  
[Default](#) = QStringFormatting::FORMAT\_DEFAULT, [Floating](#) = QStringFormatting::FORMAT\_FLOATING, [Integer](#) = QStringFormatting::FORMAT\_INTEGER, [UnsignedInteger](#) = QStringFormatting::FORMAT\_UNSIGNEDINTEGER,  
[Time](#) = QStringFormatting::FORMAT\_TIME, [LocalEnumeration](#) = QStringFormatting::FORMAT\_LOCAL\_ENUMERATE }
- enum [Notations](#) { [Fixed](#) = QStringFormatting::NOTATION\_FIXED, [Scientific](#) = QStringFormatting::NOTATION\_SCIENTIFIC, [Automatic](#) = QStringFormatting::NOTATION\_AUTOMATIC }
- enum [ArrayActions](#) { [Append](#) = QStringFormatting::APPEND, [Ascii](#) = QStringFormatting::ASCII, [Index](#) = QStringFormatting::INDEX }
- enum [AlarmSeverityDisplayModes](#) { [none](#), [foreground](#), [background](#) }

## Public Slots

- void [requestEnabled](#) (const bool &state)

## Signals

- void [dbValueChanged](#) (const double &out)
- void [requestResend](#) ()  
*Internal use only. Used when changing a property value to force a re-display to reflect the new property value.*

## Public Member Functions

- bool [isEnabled](#) () const  
*Access function for [enabled](#) property - refer to [enabled](#) property for details.*
- void [setEnabled](#) (bool state)  
*Access function for [enabled](#) property - refer to [enabled](#) property for details.*
- [UserLevels](#) [getUserLevelVisibilityProperty](#) ()  
*Access function for [userLevel/Visibility](#) property - refer to [userLevel/Visibility](#) property for details.*
- void [setUserLevelVisibilityProperty](#) ([UserLevels](#) level)  
*Access function for [userLevel/Visibility](#) property - refer to [userLevel/Visibility](#) property for details.*
- [UserLevels](#) [getUserLevelEnabledProperty](#) ()  
*Access function for [userLevel/Enabled](#) property - refer to [userLevel/Enabled](#) property for details.*
- void [setUserLevelEnabledProperty](#) ([UserLevels](#) level)



Access function for [userLevelEnabled](#) property - refer to [userLevelEnabled](#) property for details.

- void [setFormatProperty](#) ([Formats](#) format)
 

Access function for [format](#) property - refer to [format](#) property for details.
- [Formats](#) [getFormatProperty](#) ()
 

Access function for [format](#) property - refer to [format](#) property for details.
- void [setNotationProperty](#) ([Notations](#) notation)
 

Access function for [notation](#) property - refer to [notation](#) property for details.
- [Notations](#) [getNotationProperty](#) ()
 

Access function for [notation](#) property - refer to [notation](#) property for details.
- void [setArrayActionProperty](#) ([ArrayActions](#) arrayAction)
 

Access function for [arrayAction](#) property - refer to [arrayAction](#) property for details.
- [ArrayActions](#) [getArrayActionProperty](#) ()
 

Access function for [arrayAction](#) property - refer to [arrayAction](#) property for details.
- [QEAnalogProgressBar](#) (QWidget \*parent=0)
- [QEAnalogProgressBar](#) (const QString &variableName, QWidget \*parent=0)
- virtual [~QEAnalogProgressBar](#) ()
 

Destruction.
- void [setUseDbDisplayLimits](#) (bool useDbDisplayLimitsIn)
 

Access function for [useDbDisplayLimits](#) property - refer to [useDbDisplayLimits](#) property for details.
- bool [getUseDbDisplayLimits](#) ()
 

Access function for [useDbDisplayLimits](#) property - refer to [useDbDisplayLimits](#) property for details.
- void [setAlarmSeverityDisplayMode](#) (AlarmSeverityDisplayModes value)
 

Access function for [#AlarmSeverityDisplayModes](#) property - refer to [#AlarmSeverityDisplayModes](#) property for details.
- AlarmSeverityDisplayModes [getAlarmSeverityDisplayMode](#) ()
 

Access function for [#AlarmSeverityDisplayModes](#) property - refer to [#AlarmSeverityDisplayModes](#) property for details.

### Protected Member Functions

- QString [getTextImage](#) ()
- [BandList](#) [getBandList](#) ()
- void [establishConnection](#) (unsigned int variableIndex)
- void [stringFormattingChange](#) ()
- void [dragEnterEvent](#) (QDragEnterEvent \*event)
- void [dropEvent](#) (QDropEvent \*event)
- void [mousePressEvent](#) (QMouseEvent \*event)
- void [setDrop](#) (QVariant drop)
- QVariant [getDrop](#) ()
- QString [copyVariable](#) ()
- QVariant [copyData](#) ()

## Protected Attributes

- [QEFloatingFormatting](#) **floatingFormatting**

## Properties

- QString [variable](#)
- QString [variableSubstitutions](#)
- bool [variableAsToolTip](#)
- bool [enabled](#)
- bool [allowDrop](#)
- bool [visible](#)
- unsigned int
- QString [userLevelUserStyle](#)
- QString [userLevelScientistStyle](#)
- QString [userLevelEngineerStyle](#)
- [UserLevels](#) [userLevelVisibility](#)
- [UserLevels](#) [userLevelEnabled](#)
- int [precision](#)
- bool [useDbPrecision](#)
- bool [leadingZero](#)
- bool [trailingZeros](#)
- bool [addUnits](#)
- QString [localEnumeration](#)
- [Formats](#) [format](#)
- [Notations](#) [notation](#)
- [ArrayActions](#) [arrayAction](#)
- bool [useDbDisplayLimits](#)
- [AlarmSeverityDisplayModes](#) [alarmSeverityDisplayMode](#)

### 9.50.1 Member Enumeration Documentation

#### 9.50.1.1 enum [QEAnalogProgressBar::ArrayActions](#)

User friendly enumerations for arrayAction property - refer to [QEStringFormatting::arrayActions](#) for details.

#### Enumerator:

***Append*** Refer to [QEStringFormatting::APPEND](#) for details.

***Ascii*** Refer to [QEStringFormatting::ASCII](#) for details.

***Index*** Refer to [QEStringFormatting::INDEX](#) for details.

## 9.50.1.2 enum QEAnalogProgressBar::Formats

User friendly enumerations for format property - refer to [QCStringFormatting::formats](#) for details.

**Enumerator:**

**Default** Format according to the EPICS database record type.

**Floating** Format as a floating point number.

**Integer** Format as an integer.

**UnsignedInteger** Format as an unsigned integer.

**Time** Format as a time.

**LocalEnumeration** Format as a selection from the [localEnumeration](#) property.

## 9.50.1.3 enum QEAnalogProgressBar::Notations

User friendly enumerations for notation property - refer to [QCStringFormatting::notations](#) for details.

**Enumerator:**

**Fixed** Refer to [QCStringFormatting::NOTATION\\_FIXED](#) for details.

**Scientific** Refer to [QCStringFormatting::NOTATION\\_SCIENTIFIC](#) for details.

**Automatic** Refer to [QCStringFormatting::NOTATION\\_AUTOMATIC](#) for details.

## 9.50.1.4 enum QEAnalogProgressBar::UserLevels

User friendly enumerations for [userLevelVisibility](#) and [userLevelEnabled](#) properties - refer to [userLevelVisibility](#) and [userLevelEnabled](#) properties and [userLevel](#) enumeration for details.

**Enumerator:**

**User** Refer to [USERLEVEL\\_USER](#) for details.

**Scientist** Refer to [USERLEVEL\\_SCIENTIST](#) for details.

**Engineer** Refer to [USERLEVEL\\_ENGINEER](#) for details.

## 9.50.2 Constructor &amp; Destructor Documentation

## 9.50.2.1 QEAnalogProgressBar::QEAnalogProgressBar ( QWidget \* parent = 0 )

Create without a variable. Use [setVariableNameProperty\(\)](#) and [setSubstitutionsProperty\(\)](#) to define a variable and, optionally, macro substitutions later.

### 9.50.2.2 `QEAAnalogProgressBar::QEAAnalogProgressBar ( const QString & variableName, QWidget * parent = 0 )`

Create with a variable. A connection is automatically established. If macro substitutions are required, create without a variable and set the variable and macro substitutions after creation.

## 9.50.3 Member Function Documentation

### 9.50.3.1 `void QEAAnalogProgressBar::dbValueChanged ( const double & out ) [signal]`

Sent when the widget is updated following a data change Can be used to pass on EPICS data (as presented in this widget) to other widgets. For example a QList widget could log updates from this widget.

### 9.50.3.2 `void QEAAnalogProgressBar::requestEnabled ( const bool & state ) [inline, slot]`

Similar to standard `setEnabled` slot, but allows QE widget to determine if the widget remains disabled due to invalid data. If disabled due to invalid data, a request to enable the widget will be honoured when the data is no longer invalid.

## 9.50.4 Property Documentation

### 9.50.4.1 `bool QEAAnalogProgressBar::addUnits [read, write]`

If true (default), add engineering units supplied with the data.

### 9.50.4.2 `AlarmSeverityDisplayModes QEAAnalogProgressBar::alarmSeverityDisplayMode [read, write]`

Visualise the EPICS alarm severity

### 9.50.4.3 `bool QEAAnalogProgressBar::allowDrop [read, write]`

Allow drag/drops operations to this widget. Default is false. Any dropped text will be used as a new variable name.

Reimplemented from [QEDragDrop](#).

### 9.50.4.4 `ArrayActions QEAAnalogProgressBar::arrayAction [read, write]`

Text formatting option for array data. Default is ASCII. Options are:

- ASCII - treat array as a single text string. For example an array of three characters 'a' 'b' 'c' will be formatted as 'abc'.
- APPEND - treat array as an array of numbers and format a string containing them all with a space between each. For example, an array of three numbers 10, 11 and 12 will be formatted as '10 11 12'.
- INDEX - Extract a single item from the array. The item is then formatted as any other non array data would be. The item selected is determined by the `arrayIndex` property. For example, if `arrayIndex` property is 1, an array of three numbers 10, 11 and 12 will be formatted as '11'.

#### 9.50.4.5 `bool QEAnalogProgressBar::enabled` [read, write]

Set the preferred 'enabled' state. Default is true. This property is copied to the standard Qt 'enabled' property if the data being displayed is valid. If the data being displayed is invalid the standard Qt 'enabled' property will always be set to false to indicate invalid data. The value of this property will only be copied to the standard Qt 'enabled' property once data is valid.

#### 9.50.4.6 `Formats QEAnalogProgressBar::format` [read, write]

Format to apply to data. Default is 'Default' in which case the data type supplied with the data determines how the data is formatted. For all other options, an attempt is made to format the data as requested (whatever its native form).

#### 9.50.4.7 `unsigned QEAnalogProgressBar::int` [read, write]

Set the ID used by the message filtering system. Default is zero. Widgets or applications that use messages from the framework have the option of filtering on this ID. For example, by using a unique message source ID a [QELog](#) widget may be set up to only log messages from a select set of widgets.

Base used for when formatting integers. Default is 10 (duh!)

Index used to select a single item of data for formatting from an array of data. Default is 0. Only used when the `arrayAction` property is INDEX. Refer to the `arrayAction` property for more details.

#### 9.50.4.8 `bool QEAnalogProgressBar::leadingZero` [read, write]

If true (default), always add a leading zero when formatting numbers.

#### 9.50.4.9 `QString QEAnalogProgressBar::localEnumeration` [read, write]

An enumeration list used to data values. Used only when the formatting option is 'local enumeration'. Value is converted to an integer and used to select a string from this list.

Format is:

```
[[<|<=|!=|>|=|>]value1|*]: string1 , [[<|<=|!=|>|=|>]value2|*]: string2 , [[<|<=|!=|>|=|>]value3|*]: string3 , ...
```

Where: < Less than <= Less than or equal = Equal (default if no operator specified)  
 >= Greather than or equal > Greater than Always match (used to specify default text)

Values may be numeric or textual Values do not have to be in any order, but first match wins Values may be quoted Strings may be quoted Consecutive values do not have to be present. Operator is assumed to be equality if not present. White space is ignored except within quoted strings.

may be included in a string to indicate a line break

Examples are:

```
0:Off,1:On 0 : "Pump Running", 1 : "Pump not running" 0:"", 1:"Warning!\nAlarm"
<2:"Value is less than two", =2:"Value is equal to two", >2:"Value is grater than 2"
3:"Beamline Available", *:"" "Pump Off": "OH NO!, the pump is OFF!","Pump On": "It's OK, the pump is on"
```

The data value is converted to a string if no enumeration for that value is available. For example, if the local enumeration is '0:off,1:on', and a value of 10 is processed, the text generated is '10'. If a blank string is required, this should be explicit. for example, '0:off,1:on,10: ""'

A range of numbers can be covered by a pair of values as in the following example:  
 >=4:"Between 4 and 8", <=8:"Between 4 and 8"

#### 9.50.4.10 Notations QEAnalogProgressBar::notation [read, write]

Notation used for numerical formatting. Default is fixed.

#### 9.50.4.11 int QEAnalogProgressBar::precision [read, write]

Precision used when formatting floating point numbers. The default is 4. This is only used if useDbPrecision is false.

#### 9.50.4.12 bool QEAnalogProgressBar::trailingZeros [read, write]

If true (default), always remove any trailing zeros when formatting numbers.

#### 9.50.4.13 bool QEAnalogProgressBar::useDbDisplayLimits [read, write]

Use the EPICS database display limits

**9.50.4.14** `bool QEAnalogProgressBar::useDbPrecision` [read, write]

If true (default), format floating point numbers using the precision supplied with the data. If false, the precision property is used.

**9.50.4.15** `UserLevels QEAnalogProgressBar::userLevelEnabled` [read, write]

Lowest user level at which the widget is enabled. Default is 'User'. Used when designing GUIs that allow access to more and more detail according to the user mode. The user mode is set application wide through the [QELogin](#) widget, or programatically through `setUserLevel()`. Widgets that are always accessible should be visible at 'User'. Widgets that are only accessible to scientists managing the facility should be visible at 'Scientist'. Widgets that are only accessible to engineers maintaining the facility should be visible at 'Engineer'.

**9.50.4.16** `QString QEAnalogProgressBar::userLevelEngineerStyle` [read, write]

Style Sheet string to be applied when the widget is displayed in 'Engineer' mode. Default is an empty string. The syntax is the standard Qt Style Sheet syntax. For example, 'background-color: red'. This Style Sheet string will be applied by the `styleManager` class. Refer to the `styleManager` class for details about how this Style Sheet string will be merged with any pre-existing Style Sheet string and any Style Sheet strings generated during the display of data.

**9.50.4.17** `QString QEAnalogProgressBar::userLevelScientistStyle` [read, write]

Style Sheet string to be applied when the widget is displayed in 'Scientist' mode. Default is an empty string. The syntax is the standard Qt Style Sheet syntax. For example, 'background-color: red'. This Style Sheet string will be applied by the `styleManager` class. Refer to the `styleManager` class for details about how this Style Sheet string will be merged with any pre-existing Style Sheet string and any Style Sheet strings generated during the display of data.

**9.50.4.18** `QString QEAnalogProgressBar::userLevelUserStyle` [read, write]

Style Sheet string to be applied when the widget is displayed in 'User' mode. Default is an empty string. The syntax is the standard Qt Style Sheet syntax. For example, 'background-color: red'. This Style Sheet string will be applied by the `styleManager` class. Refer to the `styleManager` class for details about how this Style Sheet string will be merged with any pre-existing Style Sheet string and any Style Sheet strings generated during the display of data.

**9.50.4.19** `UserLevels QEAnalogProgressBar::userLevelVisibility` [read, write]

Lowest user level at which the widget is visible. Default is 'User'. Used when designing GUIs that display more and more detail according to the user mode. The user mode is

set application wide through the [QELogin](#) widget, or programatically through `setUserLevel()` Widgets that are always visible should be visible at 'User'. Widgets that are only used by scientists managing the facility should be visible at 'Scientist'. Widgets that are only used by engineers maintaining the facility should be visible at 'Engineer'.

#### 9.50.4.20 `QString QEAnalogProgressBar::variable` [read, write]

EPICS variable name (CA PV)

#### 9.50.4.21 `bool QEAnalogProgressBar::variableAsToolTip` [read, write]

Use the variable as the tool tip. Default is true. Tool tip property will be overwritten by the variable name.

Reimplemented from [QEToolTip](#).

#### 9.50.4.22 `QString QEAnalogProgressBar::variableSubstitutions` [read, write]

Macro substitutions. The default is no substitutions. The format is NAME1=VALUE1[,] NAME2=VALUE2... Values may be quoted strings. For example, 'PUMP=PMP3, NAME = "My Pump"' These substitutions are applied to variable names for all QE widgets. In some widgets are are also used for other purposes.

#### 9.50.4.23 `bool QEAnalogProgressBar::visible` [read, write]

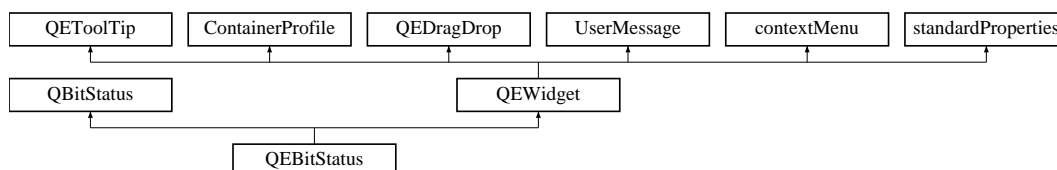
Display the widget. Default is true. Setting this property false is usefull if widget is only used to provide a signal - for example, when supplying data to a [QELink](#) widget. Note, when false the widget will still be visible in Qt Designer.

The documentation for this class was generated from the following files:

- /home/rhydera/epicsqt/framework/widgets/QEAnalogProgressBar/QEAnalogProgressBar.h
- /home/rhydera/epicsqt/framework/widgets/QEAnalogProgressBar/QEAnalogProgressBar.cpp

## 9.51 QEBitStatus Class Reference

Inheritance diagram for QEBitStatus:





## Public Types

- enum [UserLevels](#) { [User](#) = USERLEVEL\_USER, [Scientist](#) = USERLEVEL\_SCIENTIST, [Engineer](#) = USERLEVEL\_ENGINEER }

## Public Slots

- void [requestEnabled](#) (const bool &state)

## Signals

- void [dbValueChanged](#) (const long &out)

## Public Member Functions

- bool [isEnabled](#) () const  
*Access function for [enabled](#) property - refer to [enabled](#) property for details.*
- void [setEnabled](#) (bool state)  
*Access function for [enabled](#) property - refer to [enabled](#) property for details.*
- [UserLevels](#) [getUserLevelVisibilityProperty](#) ()  
*Access function for [userLevelVisibility](#) property - refer to [userLevelVisibility](#) property for details.*
- void [setUserLevelVisibilityProperty](#) ([UserLevels](#) level)  
*Access function for [userLevelVisibility](#) property - refer to [userLevelVisibility](#) property for details.*
- [UserLevels](#) [getUserLevelEnabledProperty](#) ()  
*Access function for [userLevelEnabled](#) property - refer to [userLevelEnabled](#) property for details.*
- void [setUserLevelEnabledProperty](#) ([UserLevels](#) level)  
*Access function for [userLevelEnabled](#) property - refer to [userLevelEnabled](#) property for details.*
- QEBitStatus** (QWidget \*parent=0)
- QEBitStatus** (const QString &variableName, QWidget \*parent=0)
- void [setVariableNameAndSubstitutions](#) (QString variableNameIn, QString variableNameSubstitutionsIn, unsigned int variableIndex)

## Protected Member Functions

- void **establishConnection** (unsigned int variableIndex)
- void **dragEnterEvent** (QDragEnterEvent \*event)
- void **dropEvent** (QDropEvent \*event)
- void **mousePressEvent** (QMouseEvent \*event)
- void **setDrop** (QVariant drop)
- QVariant **getDrop** ()
- QString **copyVariable** ()
- QVariant **copyData** ()

## Protected Attributes

- [QEIntegerFormatting](#) `integerFormatting`

## Properties

- [QString](#) `variable`
- [QString](#) `variableSubstitutions`
- [bool](#) `variableAsToolTip`
- [bool](#) `enabled`
- [bool](#) `allowDrop`
- [bool](#) `visible`
- [unsigned int](#)
- [QString](#) `userLevelUserStyle`
- [QString](#) `userLevelScientistStyle`
- [QString](#) `userLevelEngineerStyle`
- [UserLevels](#) `userLevelVisibility`
- [UserLevels](#) `userLevelEnabled`

## 9.51.1 Member Enumeration Documentation

### 9.51.1.1 enum `QEBitStatus::UserLevels`

User friendly enumerations for [userLevelVisibility](#) and [userLevelEnabled](#) properties - refer to [userLevelVisibility](#) and [userLevelEnabled](#) properties and `userLevel` enumeration for details.

#### Enumerator:

***User*** Refer to `USERLEVEL_USER` for details.

***Scientist*** Refer to `USERLEVEL_SCIENTIST` for details.

***Engineer*** Refer to `USERLEVEL_ENGINEER` for details.

## 9.51.2 Member Function Documentation

### 9.51.2.1 void `QEBitStatus::dbValueChanged ( const long & out )` [`signal`]

Sent when the widget is updated following a data change Can be used to pass on EPICS data (as presented in this widget) to other widgets. For example a `QList` widget could log updates from this widget.

### 9.51.2.2 void `QEBitStatus::requestEnabled ( const bool & state )` [`inline`, `slot`]

Similar to standard `setEnabled` slot, but allows QE widget to determine if the widget remains disabled due to invalid data. If disabled due to invalid data, a request to enable the widget will be honoured when the data is no longer invalid.

9.51.2.3 void QEBitStatus::setVariableNameAndSubstitutions ( QString *variableNameIn*,  
QString *variableNameSubstitutionsIn*, unsigned int *variableIndex* ) [virtual]

Virtual function that may be implemented by users of [QEWidget](#) to update variable names and macro substitutions. A default is provided that is suitable in most cases.

Reimplemented from [QEWidget](#).

### 9.51.3 Property Documentation

9.51.3.1 bool QEBitStatus::allowDrop [read, write]

Allow drag/drops operations to this widget. Default is false. Any dropped text will be used as a new variable name.

Reimplemented from [QEDragDrop](#).

9.51.3.2 bool QEBitStatus::enabled [read, write]

Set the preferred 'enabled' state. Default is true. This property is copied to the standard Qt 'enabled' property if the data being displayed is valid. If the data being displayed is invalid the standard Qt 'enabled' property will always be set to false to indicate invalid data. The value of this property will only be copied to the standard Qt 'enabled' property once data is valid.

9.51.3.3 unsigned QEBitStatus::int [read, write]

Set the ID used by the message filtering system. Default is zero. Widgets or applications that use messages from the framework have the option of filtering on this ID. For example, by using a unique message source ID a [QELog](#) widget may be set up to only log messages from a select set of widgets.

9.51.3.4 UserLevels QEBitStatus::userLevelEnabled [read, write]

Lowest user level at which the widget is enabled. Default is 'User'. Used when designing GUIs that allow access to more and more detail according to the user mode. The user mode is set application wide through the [QELogin](#) widget, or programatically through setUserLevel() Widgets that are always accessible should be visible at 'User'. Widgets that are only accessible to scientists managing the facility should be visible at 'Scientist'. Widgets that are only accessible to engineers maintaining the facility should be visible at 'Engineer'.

9.51.3.5 QString QEBitStatus::userLevelEngineerStyle [read, write]

Style Sheet string to be applied when the widget is displayed in 'Engineer' mode. Default is an empty string. The syntax is the standard Qt Style Sheet syntax. For example,

'background-color: red' This Style Sheet string will be applied by the styleManager class. Refer to the styleManager class for details about how this Style Sheet string will be merged with any pre-existing Style Sheet string and any Style Sheet strings generated during the display of data.

#### 9.51.3.6 QString QEBitStatus::userLevelScientistStyle [read, write]

Style Sheet string to be applied when the widget is displayed in 'Scientist' mode. Default is an empty string. The syntax is the standard Qt Style Sheet syntax. For example, 'background-color: red' This Style Sheet string will be applied by the styleManager class. Refer to the styleManager class for details about how this Style Sheet string will be merged with any pre-existing Style Sheet string and any Style Sheet strings generated during the display of data.

#### 9.51.3.7 QString QEBitStatus::userLevelUserStyle [read, write]

Style Sheet string to be applied when the widget is displayed in 'User' mode. Default is an empty string. The syntax is the standard Qt Style Sheet syntax. For example, 'background-color: red' This Style Sheet string will be applied by the styleManager class. Refer to the styleManager class for details about how this Style Sheet string will be merged with any pre-existing Style Sheet string and any Style Sheet strings generated during the display of data.

#### 9.51.3.8 UserLevels QEBitStatus::userLevelVisibility [read, write]

Lowest user level at which the widget is visible. Default is 'User'. Used when designing GUIs that display more and more detail according to the user mode. The user mode is set application wide through the [QELogin](#) widget, or programatically through setUserLevel(). Widgets that are always visible should be visible at 'User'. Widgets that are only used by scientists managing the facility should be visible at 'Scientist'. Widgets that are only used by engineers maintaining the facility should be visible at 'Engineer'.

#### 9.51.3.9 QString QEBitStatus::variable [read, write]

EPICS variable name (CA PV)

#### 9.51.3.10 bool QEBitStatus::variableAsToolTip [read, write]

Use the variable as the tool tip. Default is true. Tool tip property will be overwritten by the variable name.

Reimplemented from [QEToolTip](#).

### 9.51.3.11 QString QEBitStatus::variableSubstitutions [read, write]

Macro substitutions. The default is no substitutions. The format is NAME1=VALUE1[, NAME2=VALUE2... Values may be quoted strings. For example, 'PUMP=PMP3, NAME = "My Pump"' These substitutions are applied to variable names for all QE widgets. In some widgets are are also used for other purposes.

### 9.51.3.12 bool QEBitStatus::visible [read, write]

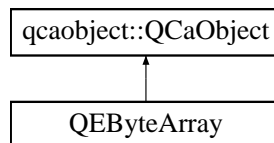
Display the widget. Default is true. Setting this property false is usefull if widget is only used to provide a signal - for example, when supplying data to a [QELink](#) widget. Note, when false the widget will still be visible in Qt Designer.

The documentation for this class was generated from the following files:

- /home/rhydera/epicsqt/framework/widgets/QEBitStatus/QEBitStatus.h
- /home/rhydera/epicsqt/framework/widgets/QEBitStatus/QEBitStatus.cpp

## 9.52 QEByteArray Class Reference

Inheritance diagram for QEByteArray:



### Public Slots

- void **writeByteArray** (const QByteArray &data)

### Signals

- void **byteArrayConnectionChanged** ([QCaConnectionInfo](#) &connectionInfo, const unsigned int &variableIndex)
- void **byteArrayChanged** (const QByteArray &value, unsigned long dataSize, [QCaAlarmInfo](#) &alarmInfo, [QCaDateTime](#) &timeStamp, const unsigned int &variableIndex)

### Public Member Functions

- **QEByteArray** (QString recordName, QObject \*eventObject, unsigned int variableIndexIn)

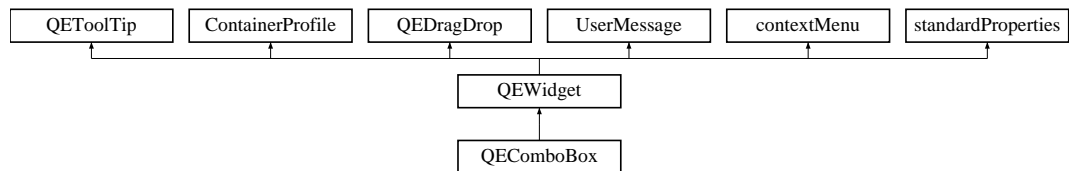
- **QEByteArray** (QString recordName, QObject \*eventObject, unsigned int variableIndexIn, [UserMessage](#) \*userMessageIn)

The documentation for this class was generated from the following files:

- /home/rhydera/epicsqt/framework/data/include/QEByteArray.h
- /home/rhydera/epicsqt/framework/data/src/QEByteArray.cpp

## 9.53 QEComboBox Class Reference

Inheritance diagram for QEComboBox:



### Public Types

- enum [UserLevels](#) { [User](#) = USERLEVEL\_USER, [Scientist](#) = USERLEVEL\_SCIENTIST, [Engineer](#) = USERLEVEL\_ENGINEER }

### Public Slots

- void [requestEnabled](#) (const bool &state)

### Signals

- void [dbValueChanged](#) (const qlonglong &out)
- void [userChange](#) (const QString &oldValue, const QString &newValue, const QString &lastValue)

*Internal use only. Used by [QEConfiguredLayout](#) to be notified when one of its widgets has written something.*

### Public Member Functions

- **QEComboBox** (QWidget \*parent=0)
- **QEComboBox** (const QString &variableName, QWidget \*parent=0)
- void **setWriteOnChange** (bool writeOnChangeIn)
- bool **getWriteOnChange** ()
- void **setSubscribe** (bool subscribe)

- bool **getSubscribe** ()
- void **setUseDbEnumerations** (bool useDbEnumerations)
- bool **getUseDbEnumerations** ()
- bool **isEnabled** () const  
*Access function for [enabled](#) property - refer to [enabled](#) property for details.*
- void **setEnabled** (bool state)  
*Access function for [enabled](#) property - refer to [enabled](#) property for details.*
- [UserLevels](#) **getUserLevelVisibilityProperty** ()  
*Access function for [userLevelVisibility](#) property - refer to [userLevelVisibility](#) property for details.*
- void **setUserLevelVisibilityProperty** ([UserLevels](#) level)  
*Access function for [userLevelVisibility](#) property - refer to [userLevelVisibility](#) property for details.*
- [UserLevels](#) **getUserLevelEnabledProperty** ()  
*Access function for [userLevelEnabled](#) property - refer to [userLevelEnabled](#) property for details.*
- void **setUserLevelEnabledProperty** ([UserLevels](#) level)  
*Access function for [userLevelEnabled](#) property - refer to [userLevelEnabled](#) property for details.*

### Protected Member Functions

- void **establishConnection** (unsigned int variableIndex)
- void **dragEnterEvent** (QDragEnterEvent \*event)
- void **dropEvent** (QDropEvent \*event)
- void **setDrop** (QVariant drop)
- QVariant **getDrop** ()

### Protected Attributes

- [QEIntegerFormatting](#) **integerFormatting**
- bool **useDbEnumerations**
- bool [writeOnChange](#)

### Properties

- QString [variable](#)
- QString [variableSubstitutions](#)
- bool [subscribe](#)
- bool [variableAsToolTip](#)
- bool [enabled](#)
- bool [allowDrop](#)
- bool [visible](#)
- unsigned [int](#)

- QString [userLevelUserStyle](#)
- QString [userLevelScientistStyle](#)
- QString [userLevelEngineerStyle](#)
- UserLevels [userLevelVisibility](#)
- UserLevels [userLevelEnabled](#)

### 9.53.1 Member Enumeration Documentation

#### 9.53.1.1 enum QComboBox::UserLevels

User friendly enumerations for [userLevelVisibility](#) and [userLevelEnabled](#) properties - refer to [userLevelVisibility](#) and [userLevelEnabled](#) properties and userLevel enumeration for details.

##### Enumerator:

**User** Refer to USERLEVEL\_USER for details.

**Scientist** Refer to USERLEVEL\_SCIENTIST for details.

**Engineer** Refer to USERLEVEL\_ENGINEER for details.

### 9.53.2 Member Function Documentation

#### 9.53.2.1 void QComboBox::dbValueChanged ( const qlonglong & *out* ) [signal]

Sent when the widget is updated following a data change Can be used to pass on EPICS data (as presented in this widget) to other widgets. For example a QList widget could log updates from this widget.

#### 9.53.2.2 void QComboBox::requestEnabled ( const bool & *state* ) [inline, slot]

Similar to standard setEnabled slot, but allows QE widget to determine if the widget remains disabled due to invalid data. If disabled due to invalid data, a request to enable the widget will be honoured when the data is no longer invalid.

### 9.53.3 Member Data Documentation

#### 9.53.3.1 bool QComboBox::writeOnChange [read, write, protected]

Sets if this widget writes any changes as the user selects values (the QComboBox 'activated' signal is emitted). Default is 'true' (writes any changes when the QComboBox 'activated' signal is emitted).

### 9.53.4 Property Documentation



#### 9.53.4.1 `bool QComboBox::allowDrop` [read, write]

Allow drag/drops operations to this widget. Default is false. Any dropped text will be used as a new variable name.

Reimplemented from [QEDragDrop](#).

#### 9.53.4.2 `bool QComboBox::enabled` [read, write]

Set the preferred 'enabled' state. Default is true. This property is copied to the standard Qt 'enabled' property if the data being displayed is valid. If the data being displayed is invalid the standard Qt 'enabled' property will always be set to false to indicate invalid data. The value of this property will only be copied to the standard Qt 'enabled' property once data is valid.

#### 9.53.4.3 `unsigned QComboBox::int` [read, write]

Set the ID used by the message filtering system. Default is zero. Widgets or applications that use messages from the framework have the option of filtering on this ID. For example, by using a unique message source ID a [QELog](#) widget may be set up to only log messages from a select set of widgets.

#### 9.53.4.4 `bool QComboBox::subscribe` [read, write]

Sets if this widget subscribes for data updates and displays current data. Default is 'true' (subscribes for and displays data updates)

Reimplemented from [QEWidget](#).

#### 9.53.4.5 `UserLevels QComboBox::userLevelEnabled` [read, write]

Lowest user level at which the widget is enabled. Default is 'User'. Used when designing GUIs that allow access to more and more detail according to the user mode. The user mode is set application wide through the [QELogin](#) widget, or programatically through `setUserLevel()` Widgets that are always accessible should be visible at 'User'. Widgets that are only accessible to scientists managing the facility should be visible at 'Scientist'. Widgets that are only accessible to engineers maintaining the facility should be visible at 'Engineer'.

#### 9.53.4.6 `QString QComboBox::userLevelEngineerStyle` [read, write]

Style Sheet string to be applied when the widget is displayed in 'Engineer' mode. Default is an empty string. The syntax is the standard Qt Style Sheet syntax. For example, 'background-color: red' This Style Sheet string will be applied by the `styleManager` class. Refer to the `styleManager` class for details about how this Style Sheet string will be merged with any pre-existing Style Sheet string and any Style Sheet strings generated during the display of data.

#### 9.53.4.7 QString QEComboBox::userLevelScientistStyle [read, write]

Style Sheet string to be applied when the widget is displayed in 'Scientist' mode. Default is an empty string. The syntax is the standard Qt Style Sheet syntax. For example, 'background-color: red' This Style Sheet string will be applied by the styleManager class. Refer to the styleManager class for details about how this Style Sheet string will be merged with any pre-existing Style Sheet string and any Style Sheet strings generated during the display of data.

#### 9.53.4.8 QString QEComboBox::userLevelUserStyle [read, write]

Style Sheet string to be applied when the widget is displayed in 'User' mode. Default is an empty string. The syntax is the standard Qt Style Sheet syntax. For example, 'background-color: red' This Style Sheet string will be applied by the styleManager class. Refer to the styleManager class for details about how this Style Sheet string will be merged with any pre-existing Style Sheet string and any Style Sheet strings generated during the display of data.

#### 9.53.4.9 UserLevels QEComboBox::userLevelVisibility [read, write]

Lowest user level at which the widget is visible. Default is 'User'. Used when designing GUIs that display more and more detail according to the user mode. The user mode is set application wide through the [QELogin](#) widget, or programatically through setUserLevel() Widgets that are always visible should be visible at 'User'. Widgets that are only used by scientists managing the facility should be visible at 'Scientist'. Widgets that are only used by engineers maintaining the facility should be visible at 'Engineer'.

#### 9.53.4.10 QString QEComboBox::variable [read, write]

EPICS variable name (CA PV)

#### 9.53.4.11 bool QEComboBox::variableAsToolTip [read, write]

Use the variable as the tool tip. Default is true. Tool tip property will be overwritten by the variable name.

Reimplemented from [QEToolTip](#).

#### 9.53.4.12 QString QEComboBox::variableSubstitutions [read, write]

Macro substitutions. The default is no substitutions. The format is NAME1=VALUE1[, NAME2=VALUE2... Values may be quoted strings. For example, 'PUMP=PMP3, NAME = "My Pump"' These substitutions are applied to variable names for all QE widgets. In some widgets are also used for other purposes.

#### 9.53.4.13 bool QEComboBox::visible [read, write]

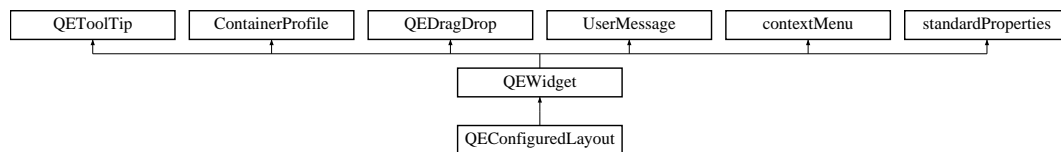
Display the widget. Default is true. Setting this property false is usefull if widget is only used to provide a signal - for example, when supplying data to a [QELink](#) widget. Note, when false the widget will still be visible in Qt Designer.

The documentation for this class was generated from the following files:

- /home/rhydera/epicsqt/framework/widgets/QEComboBox/QEComboBox.h
- /home/rhydera/epicsqt/framework/widgets/QEComboBox/QEComboBox.cpp

## 9.54 QEConfiguredLayout Class Reference

Inheritance diagram for QEConfiguredLayout:



### Public Types

- enum **configurationTypesProperty** { **File** = FROM\_FILE, **Text** = FROM\_TEXT }
- enum **detailsLayoutProperty** { **Top** = TOP, **Bottom** = BOTTOM, **Left** = LEFT, **Right** = RIGHT }
- enum **userTypesProperty** { **User** = USERLEVEL\_USER, **Scientist** = USERLEVEL\_SCIENTIST, **Engineer** = USERLEVEL\_ENGINEER }

### Public Member Functions

- **QEConfiguredLayout** (QWidget \*pParent=0, bool pSubscription=true)
- void **setItemDescription** (QString pValue)
- QString **getItemDescription** ()
- void **setShowItemList** (bool pValue)
- bool **getShowItemList** ()
- void **setConfigurationType** (int pValue)
- int **getConfigurationType** ()
- void **setConfigurationFile** (QString pValue)
- QString **getConfigurationFile** ()
- void **setConfigurationText** (QString pValue)
- QString **getConfigurationText** ()
- void **setDetailsLayout** (int pValue)
- int **getDetailsLayout** ()
- void **setCurrentUserType** (int pValue)

- int **getCurrentUserType** ()
- void **refreshFields** ()
- void **userLevelChanged** (userLevels pValue)
- void **setConfigurationTypeProperty** (configurationTypesProperty pConfigurationType)
- configurationTypesProperty **getConfigurationTypeProperty** ()
- void **setDetailsLayoutProperty** (detailsLayoutProperty pDetailsLayout)
- detailsLayoutProperty **getDetailsLayoutProperty** ()
- void **setCurrentUserTypeProperty** (userTypesProperty pUserType)
- userTypesProperty **getCurrentUserTypeProperty** ()

### Public Attributes

- QList< [\\_Item](#) \* > **itemList**
- QList< [\\_Field](#) \* > **currentFieldList**

### Protected Attributes

- QLabel \* **qLabelItemDescription**
- QComboBox \* **qComboBoxItemList**
- QVBoxLayout \* **qVBoxLayoutFields**
- QScrollArea \* **qScrollArea**
- QString **configurationFile**
- QString **configurationText**
- int **configurationType**
- int **detailsLayout**
- int **currentUserType**
- bool **subscription**

### Properties

- QString **itemDescription**
- bool **showItemList**
- configurationTypesProperty **configurationType**
- detailsLayoutProperty **detailsLayout**
- userTypesProperty **currentUserType**

The documentation for this class was generated from the following files:

- /home/rhydera/epicsqt/framework/widgets/QEConfiguredLayout/QEConfiguredLayout.h
- /home/rhydera/epicsqt/framework/widgets/QEConfiguredLayout/QEConfiguredLayout.cpp

## 9.55 QEConfiguredLayoutManager Class Reference

### Public Member Functions

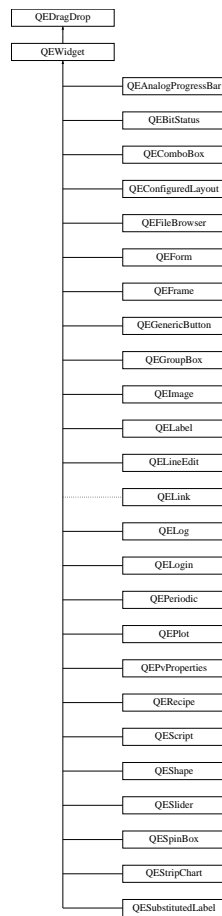
- **QEConfiguredLayoutManager** (QObject \*pParent=0)
- bool **isContainer** () const
- bool **isInitialized** () const
- QIcon **icon** () const
- QString **group** () const
- QString **includeFile** () const
- QString **name** () const
- QString **toolTip** () const
- QString **whatsThis** () const
- QWidget \* **createWidget** (QWidget \*pParent)
- void **initialize** (QDesignerFormEditorInterface \*pCore)

The documentation for this class was generated from the following files:

- /home/rhydera/epicsqt/framework/widgets/QEConfiguredLayout/QEConfiguredLayoutManager.h
- /home/rhydera/epicsqt/framework/widgets/QEConfiguredLayout/QEConfiguredLayoutManager.cpp

## 9.56 QEDragDrop Class Reference

Inheritance diagram for QEDragDrop:



## Public Member Functions

- **QEDragDrop** (QWidget \*ownerIn)

## Protected Member Functions

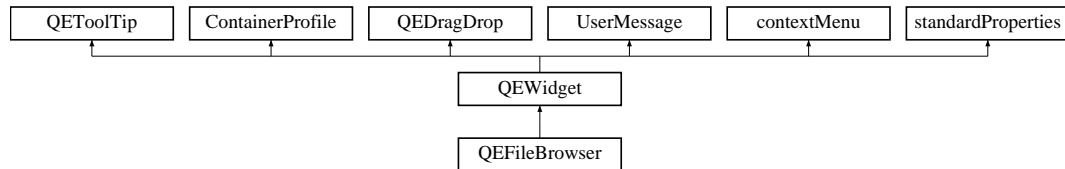
- void **qcaDragEnterEvent** (QDragEnterEvent \*event)
- void **qcaDropEvent** (QDropEvent \*event)
- void **qcaMousePressEvent** (QMouseEvent \*event)
- virtual void **setDrop** (QVariant)
- virtual QVariant **getDrop** ()
- void **setAllowDrop** (bool allowDropIn)
- bool **getAllowDrop** ()

The documentation for this class was generated from the following files:

- /home/rhydera/epicsqt/framework/widgets/include/QEDragDrop.h
- /home/rhydera/epicsqt/framework/widgets/src/QEDragDrop.cpp

## 9.57 QFileDialog Class Reference

Inheritance diagram for QFileDialog:



### Public Types

- enum **detailsLayoutProperty** { **Top** = TOP, **Bottom** = BOTTOM, **Left** = LEFT, **Right** = RIGHT }

### Signals

- void **selected** (QString pFilename)

### Public Member Functions

- **QFileDialog** (QWidget \*pParent=0)
- void **setDirectoryPath** (QString pValue)
- QString **getDirectoryPath** ()
- void **setShowDirectoryPath** (bool pValue)
- bool **getShowDirectoryPath** ()
- void **setShowDirectoryBrowser** (bool pValue)
- bool **getShowDirectoryBrowser** ()
- void **setShowRefresh** (bool pValue)
- bool **getShowRefresh** ()
- void **setShowColumnTime** (bool pValue)
- bool **getShowColumnTime** ()
- void **setShowColumnSize** (bool pValue)
- bool **getShowColumnSize** ()
- void **setShowColumnFilename** (bool pValue)
- bool **getShowColumnFilename** ()
- void **setShowFileExtension** (bool pValue)
- bool **getShowFileExtension** ()
- void **setFileFilter** (QString pValue)
- QString **getFileFilter** ()
- void **setDetailsLayout** (int pValue)
- int **getDetailsLayout** ()
- void **updateTable** ()
- void **setDetailsLayoutProperty** (detailsLayoutProperty pDetailsLayout)
- detailsLayoutProperty **getDetailsLayoutProperty** ()

### Protected Attributes

- QLineEdit \* **qlineEditDirectoryPath**
- QPushButton \* **qPushButtonDirectoryBrowser**
- QPushButton \* **qPushButtonRefresh**
- [\\_QTableWidgetFileBrowser](#) \* **qTableWidgetFileBrowser**
- QString **fileFilter**
- bool **showFileExtension**
- int **detailsLayout**

### Properties

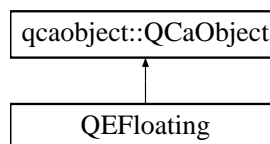
- QString **directoryPath**
- bool **showDirectoryPath**
- bool **showDirectoryBrowser**
- bool **showRefresh**
- bool **showColumnTime**
- bool **showColumnSize**
- bool **showColumnFilename**
- detailsLayoutProperty **detailsLayout**

The documentation for this class was generated from the following files:

- /home/rhydera/epicsqt/framework/widgets/QEFileBrowser/QEFileBrowser.h
- /home/rhydera/epicsqt/framework/widgets/QEFileBrowser/QEFileBrowser.cpp

## 9.58 QEFloating Class Reference

Inheritance diagram for QEFloating:



### Public Slots

- void **writeFloating** (const double &data)



## Signals

- void **floatingConnectionChanged** ([QCaConnectionInfo](#) &connectionInfo, const unsigned int &variableIndex)
- void **floatingChanged** (const double &value, [QCaAlarmInfo](#) &alarmInfo, [QCaDateTime](#) &timeStamp, const unsigned int &variableIndex)
- void **floatingArrayChanged** (const QVector< double > &values, [QCaAlarmInfo](#) &alarmInfo, [QCaDateTime](#) &timeStamp, const unsigned int &variableIndex)

## Public Member Functions

- **QEFloating** (QString recordName, QObject \*eventObject, [QEFloatingFormatting](#) \*floatingFormattingIn, unsigned int variableIndexIn)
- **QEFloating** (QString recordName, QObject \*eventObject, [QEFloatingFormatting](#) \*floatingFormattingIn, unsigned int variableIndexIn, [UserMessage](#) \*userMessageIn)

The documentation for this class was generated from the following files:

- /home/rhydera/epicsqt/framework/data/include/QEFloating.h
- /home/rhydera/epicsqt/framework/data/src/QEFloating.cpp

## 9.59 QEFloatingFormatting Class Reference

### Public Types

- enum **formats** {  
**FORMAT\_e** = 'e', **FORMAT\_E** = 'E', **FORMAT\_f** = 'f', **FORMAT\_g** = 'g',  
**FORMAT\_G** = 'G' }

### Public Member Functions

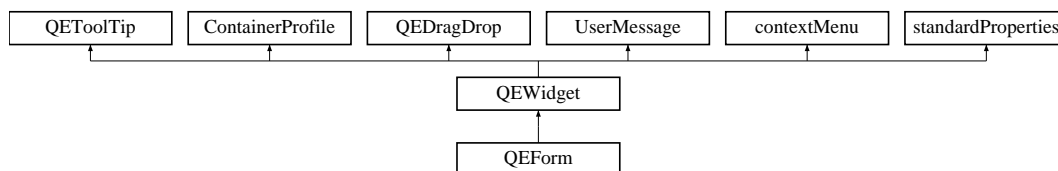
- double **formatFloating** (const QVariant &value)
- QVector< double > **formatFloatingArray** (const QVariant &value)
- QVariant **formatValue** (const double &floatingValue, generic::generic\_types valueType)
- void **setPrecision** (unsigned int precision)
- void **setFormat** (formats format)
- unsigned int **getPrecision** ()
- int **getFormat** ()

The documentation for this class was generated from the following files:

- /home/rhydera/epicsqt/framework/data/include/QEFloatingFormatting.h
- /home/rhydera/epicsqt/framework/data/src/QEFloatingFormatting.cpp

## 9.60 QForm Class Reference

Inheritance diagram for QForm:



### Public Types

- enum **creationOptions** { **CREATION\_OPTION\_OPEN**, **CREATION\_OPTION\_NEW\_TAB**, **CREATION\_OPTION\_NEW\_WINDOW** }
- enum **MessageFilterOptions** { **Match** = UserMessage::MESSAGE\_FILTER\_MATCH, **None** = UserMessage::MESSAGE\_FILTER\_NONE }

### Public Slots

- bool **readUiFile** ()
- void **launchGui** (QString guiName, QForm::creationOptions createOption)

### Public Member Functions

- **QForm** (QWidget \*parent=0)
- **QForm** (const QString &uifileNameIn, QWidget \*parent=0)
- void **commonInit** (const bool alertIfUINoFoundIn)
- QString **getQEGuiTitle** ()
- QString **getFullFileName** ()
- void **setVariableNameAndSubstitutions** (QString variableNameIn, QString variableNameSubstitutionsIn, unsigned int variableIndex)
- void **setUiFileName** (QString uiFile)
- QString **getUiFileName** ()
- void **setHandleGuiLaunchRequests** (bool handleGuiLaunchRequests)
- bool **getHandleGuiLaunchRequests** ()
- void **setResizeContents** (bool resizeContentsIn)
- bool **getResizeContents** ()
- void **setVariableNameSubstitutionsProperty** (QString variableNameSubstitutions)
- QString **getVariableNameSubstitutionsProperty** ()
- MessageFilterOptions **getMessageFormFilter** ()
- void **setMessageFormFilter** (MessageFilterOptions messageFormFilter)
- MessageFilterOptions **getMessageSourceFilter** ()
- void **setMessageSourceFilter** (MessageFilterOptions messageSourceFilter)

### Protected Member Functions

- void **setVariableNameSubstitutions** (QString variableNameSubstitutionsIn)

### Protected Attributes

- QString **uiFileName**
- QString **fullUiFileName**
- bool **handleGuiLaunchRequests**
- bool **resizeContents**

### Properties

- QString **uiFile**
- QString **variableSubstitutions**
- unsigned int
- MessageFilterOptions **messageFormFilter**
- MessageFilterOptions **messageSourceFilter**

#### 9.60.1 Member Function Documentation

9.60.1.1 void QEForm::setVariableNameAndSubstitutions ( QString *variableNameIn*, QString *variableNameSubstitutionsIn*, unsigned int *variableIndex* ) [virtual]

Virtual function that may be implimented by users of [QEWidget](#) to update variable names and macro substitutions. A default is provided that is suitable in most cases.

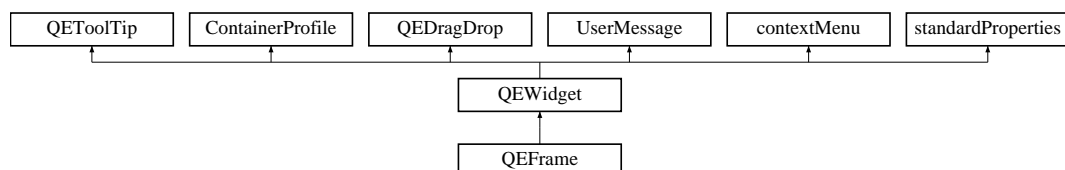
Reimplemented from [QEWidget](#).

The documentation for this class was generated from the following files:

- /home/rhydera/epicsqt/framework/widgets/QEForm/QEForm.h
- /home/rhydera/epicsqt/framework/widgets/QEForm/QEForm.cpp

## 9.61 QEFrame Class Reference

Inheritance diagram for QEFrame:



## Public Types

- enum [UserLevels](#) { [User](#) = USERLEVEL\_USER, [Scientist](#) = USERLEVEL\_SCIENTIST, [Engineer](#) = USERLEVEL\_ENGINEER }

## Public Slots

- void [requestEnabled](#) (const bool &state)

## Public Member Functions

- bool [isEnabled](#) () const  
*Access function for [enabled](#) property - refer to [enabled](#) property for details.*
- void [setEnabled](#) (bool state)  
*Access function for [enabled](#) property - refer to [enabled](#) property for details.*
- [UserLevels](#) [getUserLevelVisibilityProperty](#) ()  
*Access function for [userLevelVisibility](#) property - refer to [userLevelVisibility](#) property for details.*
- void [setUserLevelVisibilityProperty](#) ([UserLevels](#) level)  
*Access function for [userLevelVisibility](#) property - refer to [userLevelVisibility](#) property for details.*
- [UserLevels](#) [getUserLevelEnabledProperty](#) ()  
*Access function for [userLevelEnabled](#) property - refer to [userLevelEnabled](#) property for details.*
- void [setUserLevelEnabledProperty](#) ([UserLevels](#) level)  
*Access function for [userLevelEnabled](#) property - refer to [userLevelEnabled](#) property for details.*
- **QEFrame** (QWidget \*parent=0)
- QSize **sizeHint** () const

## Properties

- bool [variableAsToolTip](#)
- bool [enabled](#)
- bool [allowDrop](#)
- bool [visible](#)
- unsigned int
- QString [userLevelUserStyle](#)
- QString [userLevelScientistStyle](#)
- QString [userLevelEngineerStyle](#)
- [UserLevels](#) [userLevelVisibility](#)
- [UserLevels](#) [userLevelEnabled](#)

## 9.61.1 Member Enumeration Documentation

### 9.61.1.1 enum QEFrame::UserLevels

User friendly enumerations for [userLevelVisibility](#) and [userLevelEnabled](#) properties - refer to [userLevelVisibility](#) and [userLevelEnabled](#) properties and [userLevel](#) enumeration for details.

#### Enumerator:

**User** Refer to `USERLEVEL_USER` for details.

**Scientist** Refer to `USERLEVEL_SCIENTIST` for details.

**Engineer** Refer to `USERLEVEL_ENGINEER` for details.

## 9.61.2 Member Function Documentation

### 9.61.2.1 void QEFrame::requestEnabled ( const bool & state ) [inline, slot]

Similar to standard `setEnabled` slot, but allows QE widget to determine if the widget remains disabled due to invalid data. If disabled due to invalid data, a request to enable the widget will be honoured when the data is no longer invalid.

## 9.61.3 Property Documentation

### 9.61.3.1 bool QEFrame::allowDrop [read, write]

Allow drag/drops operations to this widget. Default is false. Any dropped text will be used as a new variable name.

Reimplemented from [QEDragDrop](#).

### 9.61.3.2 bool QEFrame::enabled [read, write]

Set the preferred 'enabled' state. Default is true. This property is copied to the standard Qt 'enabled' property if the data being displayed is valid. If the data being displayed is invalid the standard Qt 'enabled' property will always be set to false to indicate invalid data. The value of this property will only be copied to the standard Qt 'enabled' property once data is valid.

### 9.61.3.3 unsigned QEFrame::int [read, write]

Set the ID used by the message filtering system. Default is zero. Widgets or applications that use messages from the framework have the option of filtering on this ID. For example, by using a unique message source ID a [QELog](#) widget may be set up to only log messages from a select set of widgets.

#### 9.61.3.4 UserLevels QFrame::userLevelEnabled [read, write]

Lowest user level at which the widget is enabled. Default is 'User'. Used when designing GUIs that allow access to more and more detail according to the user mode. The user mode is set application wide through the [QELogin](#) widget, or programatically through `setUserLevel()` Widgets that are always accessible should be visible at 'User'. Widgets that are only accessible to scientists managing the facility should be visible at 'Scientist'. Widgets that are only accessible to engineers maintaining the facility should be visible at 'Engineer'.

#### 9.61.3.5 QString QFrame::userLevelEngineerStyle [read, write]

Style Sheet string to be applied when the widget is displayed in 'Engineer' mode. Default is an empty string. The syntax is the standard Qt Style Sheet syntax. For example, 'background-color: red' This Style Sheet string will be applied by the styleManager class. Refer to the styleManager class for details about how this Style Sheet string will be merged with any pre-existing Style Sheet string and any Style Sheet strings generated during the display of data.

#### 9.61.3.6 QString QFrame::userLevelScientistStyle [read, write]

Style Sheet string to be applied when the widget is displayed in 'Scientist' mode. Default is an empty string. The syntax is the standard Qt Style Sheet syntax. For example, 'background-color: red' This Style Sheet string will be applied by the styleManager class. Refer to the styleManager class for details about how this Style Sheet string will be merged with any pre-existing Style Sheet string and any Style Sheet strings generated during the display of data.

#### 9.61.3.7 QString QFrame::userLevelUserStyle [read, write]

Style Sheet string to be applied when the widget is displayed in 'User' mode. Default is an empty string. The syntax is the standard Qt Style Sheet syntax. For example, 'background-color: red' This Style Sheet string will be applied by the styleManager class. Refer to the styleManager class for details about how this Style Sheet string will be merged with any pre-existing Style Sheet string and any Style Sheet strings generated during the display of data.

#### 9.61.3.8 UserLevels QFrame::userLevelVisibility [read, write]

Lowest user level at which the widget is visible. Default is 'User'. Used when designing GUIs that display more and more detail according to the user mode. The user mode is set application wide through the [QELogin](#) widget, or programatically through `setUserLevel()` Widgets that are always visible should be visible at 'User'. Widgets that are only used by scientists managing the facility should be visible at 'Scientist'. Widgets that are only used by engineers maintaining the facility should be visible at 'Engineer'.

### 9.61.3.9 bool QFrame::variableAsToolTip [read, write]

Use the variable as the tool tip. Default is true. Tool tip property will be overwritten by the variable name.

Reimplemented from [QEToolTip](#).

### 9.61.3.10 bool QFrame::visible [read, write]

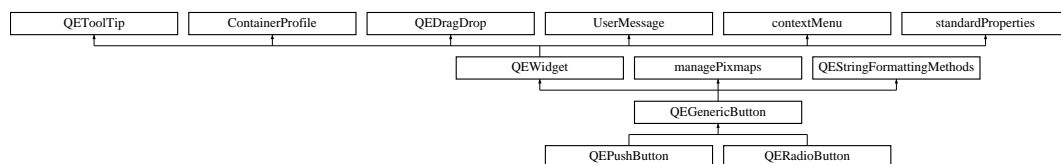
Display the widget. Default is true. Setting this property false is usefull if widget is only used to provide a signal - for example, when supplying data to a [QELink](#) widget. Note, when false the widget will still be visible in Qt Designer.

The documentation for this class was generated from the following files:

- /home/rhydera/epicsqt/framework/widgets/QFrame/QFrame.h
- /home/rhydera/epicsqt/framework/widgets/QFrame/QFrame.cpp

## 9.62 QEGenericButton Class Reference

Inheritance diagram for QEGenericButton:



### Public Types

- enum **updateOptions** { **UPDATE\_TEXT**, **UPDATE\_ICON**, **UPDATE\_TEXT\_AND\_ICON**, **UPDATE\_STATE** }

### Public Member Functions

- **QEGenericButton** (QWidget \*owner)
- void **setSubscribe** (bool subscribe)
- bool **getSubscribe** ()
- void **setUpdateOption** (updateOptions updateOptionIn)
- updateOptions **getUpdateOption** ()
- void **setTextAlignment** (Qt::Alignment alignment)
- Qt::Alignment **getTextAlignment** ()
- void **setPassword** (QString password)
- QString **getPassword** ()
- void **setConfirmAction** (bool confirmRequiredIn)

- bool **getConfirmAction** ()
- void **setWriteOnPress** (bool writeOnPress)
- bool **getWriteOnPress** ()
- void **setWriteOnRelease** (bool writeOnRelease)
- bool **getWriteOnRelease** ()
- void **setWriteOnClick** (bool writeOnClick)
- bool **getWriteOnClick** ()
- void **setPressText** (QString pressText)
- QString **getPressText** ()
- void **setReleaseText** (QString releaseTextIn)
- QString **getReleaseText** ()
- void **setClickText** (QString clickTextIn)
- QString **getClickText** ()
- void **setClickCheckedText** (QString clickCheckedTextIn)
- QString **getClickCheckedText** ()
- void **setProgram** (QString program)
- QString **getProgram** ()
- void **setArguments** (QStringList arguments)
- QStringList **getArguments** ()
- void **setGuiName** (QString guiName)
- QString **getGuiName** ()
- void **setCreationOption** (QForm::creationOptions creationOption)
- QForm::creationOptions **getCreationOption** ()
- void **setLabelTextProperty** (QString labelTextIn)
- QString **getLabelTextProperty** ()

### Protected Member Functions

- void **connectionChanged** ([QCaConnectionInfo](#) &connectionInfo)
- void **setGenericButtonText** (const QString &text, [QCaAlarmInfo](#) &alarmInfo, [QCaDateTime](#) &, const unsigned int &variableIndex)
- void **userPressed** ()
- void **userReleased** ()
- void **userClicked** (bool checked)
- void **launchGui** (QString guiName, QForm::creationOptions creationOption)
- virtual updateOptions **getDefaultUpdateOption** ()=0
- void **setup** ()
- void **establishConnection** (unsigned int variableIndex)

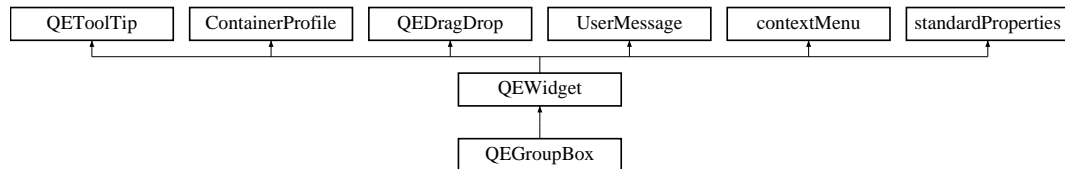
The documentation for this class was generated from the following files:

- /home/rhydera/epicsqt/framework/widgets/QEButton/QEGenericButton.h
- /home/rhydera/epicsqt/framework/widgets/QEButton/QEGenericButton.cpp



## 9.63 QEGroupBox Class Reference

Inheritance diagram for QEGroupBox:



### Public Types

- enum `UserLevels` { `User` = USERLEVEL\_USER, `Scientist` = USERLEVEL\_SCIENTIST, `Engineer` = USERLEVEL\_ENGINEER }

### Public Slots

- void `requestEnabled` (const bool &state)

### Public Member Functions

- bool `isEnabled` () const  
Access function for `enabled` property - refer to `enabled` property for details.
- void `setEnabled` (bool state)  
Access function for `enabled` property - refer to `enabled` property for details.
- `UserLevels` `getUserLevelVisibilityProperty` ()  
Access function for `userLevelVisibility` property - refer to `userLevelVisibility` property for details.
- void `setUserLevelVisibilityProperty` (`UserLevels` level)  
Access function for `userLevelVisibility` property - refer to `userLevelVisibility` property for details.
- `UserLevels` `getUserLevelEnabledProperty` ()  
Access function for `userLevelEnabled` property - refer to `userLevelEnabled` property for details.
- void `setUserLevelEnabledProperty` (`UserLevels` level)  
Access function for `userLevelEnabled` property - refer to `userLevelEnabled` property for details.
- QEGroupBox** (QWidget \*parent=0)
- QSize `sizeHint` () const

## Properties

- bool [variableAsToolTip](#)
- bool [enabled](#)
- bool [allowDrop](#)
- bool [visible](#)
- unsigned [int](#)
- QString [userLevelUserStyle](#)
- QString [userLevelScientistStyle](#)
- QString [userLevelEngineerStyle](#)
- [UserLevels](#) [userLevelVisibility](#)
- [UserLevels](#) [userLevelEnabled](#)

### 9.63.1 Member Enumeration Documentation

#### 9.63.1.1 enum [QEGroupBox::UserLevels](#)

User friendly enumerations for [userLevelVisibility](#) and [userLevelEnabled](#) properties - refer to [userLevelVisibility](#) and [userLevelEnabled](#) properties and [userLevel](#) enumeration for details.

#### Enumerator:

***User*** Refer to `USERLEVEL_USER` for details.

***Scientist*** Refer to `USERLEVEL_SCIENTIST` for details.

***Engineer*** Refer to `USERLEVEL_ENGINEER` for details.

### 9.63.2 Member Function Documentation

#### 9.63.2.1 void [QEGroupBox::requestEnabled](#) ( const bool & *state* ) [[inline](#), [slot](#)]

Similar to standard `setEnabled` slot, but allows QE widget to determine if the widget remains disabled due to invalid data. If disabled due to invalid data, a request to enable the widget will be honoured when the data is no longer invalid.

### 9.63.3 Property Documentation

#### 9.63.3.1 bool [QEGroupBox::allowDrop](#) [[read](#), [write](#)]

Allow drag/drops operations to this widget. Default is false. Any dropped text will be used as a new variable name.

Reimplemented from [QEDragDrop](#).

**9.63.3.2 bool QEGroupBox::enabled** [read, write]

Set the preferred 'enabled' state. Default is true. This property is copied to the standard Qt 'enabled' property if the data being displayed is valid. If the data being displayed is invalid the standard Qt 'enabled' property will always be set to false to indicate invalid data. The value of this property will only be copied to the standard Qt 'enabled' property once data is valid.

**9.63.3.3 unsigned QEGroupBox::int** [read, write]

Set the ID used by the message filtering system. Default is zero. Widgets or applications that use messages from the framework have the option of filtering on this ID. For example, by using a unique message source ID a [QELog](#) widget may be set up to only log messages from a select set of widgets.

**9.63.3.4 UserLevels QEGroupBox::userLevelEnabled** [read, write]

Lowest user level at which the widget is enabled. Default is 'User'. Used when designing GUIs that allow access to more and more detail according to the user mode. The user mode is set application wide through the [QELogin](#) widget, or programatically through `setUserLevel()`. Widgets that are always accessible should be visible at 'User'. Widgets that are only accessible to scientists managing the facility should be visible at 'Scientist'. Widgets that are only accessible to engineers maintaining the facility should be visible at 'Engineer'.

**9.63.3.5 QString QEGroupBox::userLevelEngineerStyle** [read, write]

Style Sheet string to be applied when the widget is displayed in 'Engineer' mode. Default is an empty string. The syntax is the standard Qt Style Sheet syntax. For example, 'background-color: red' This Style Sheet string will be applied by the `styleManager` class. Refer to the `styleManager` class for details about how this Style Sheet string will be merged with any pre-existing Style Sheet string and any Style Sheet strings generated during the display of data.

**9.63.3.6 QString QEGroupBox::userLevelScientistStyle** [read, write]

Style Sheet string to be applied when the widget is displayed in 'Scientist' mode. Default is an empty string. The syntax is the standard Qt Style Sheet syntax. For example, 'background-color: red' This Style Sheet string will be applied by the `styleManager` class. Refer to the `styleManager` class for details about how this Style Sheet string will be merged with any pre-existing Style Sheet string and any Style Sheet strings generated during the display of data.

### 9.63.3.7 QString QEGroupBox::userLevelUserStyle [read, write]

Style Sheet string to be applied when the widget is displayed in 'User' mode. Default is an empty string. The syntax is the standard Qt Style Sheet syntax. For example, 'background-color: red' This Style Sheet string will be applied by the styleManager class. Refer to the styleManager class for details about how this Style Sheet string will be merged with any pre-existing Style Sheet string and any Style Sheet strings generated during the display of data.

### 9.63.3.8 UserLevels QEGroupBox::userLevelVisibility [read, write]

Lowest user level at which the widget is visible. Default is 'User'. Used when designing GUIs that display more and more detail according to the user mode. The user mode is set application wide through the [QELogin](#) widget, or programmatically through setUserLevel(). Widgets that are always visible should be visible at 'User'. Widgets that are only used by scientists managing the facility should be visible at 'Scientist'. Widgets that are only used by engineers maintaining the facility should be visible at 'Engineer'.

### 9.63.3.9 bool QEGroupBox::variableAsToolTip [read, write]

Use the variable as the tool tip. Default is true. Tool tip property will be overwritten by the variable name.

Reimplemented from [QEToolTip](#).

### 9.63.3.10 bool QEGroupBox::visible [read, write]

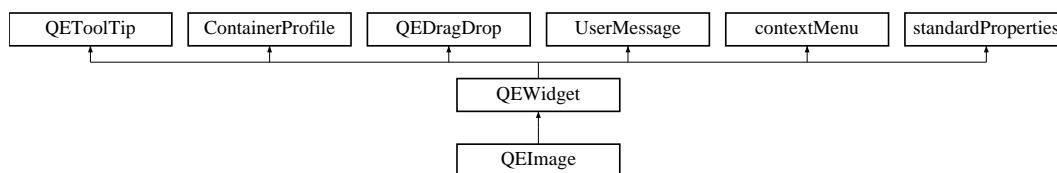
Display the widget. Default is true. Setting this property false is usefull if widget is only used to provide a signal - for example, when supplying data to a [QELink](#) widget. Note, when false the widget will still be visible in Qt Designer.

The documentation for this class was generated from the following files:

- /home/rhydera/epicsqt/framework/widgets/QEGroupBox/QEGroupBox.h
- /home/rhydera/epicsqt/framework/widgets/QEGroupBox/QEGroupBox.cpp

## 9.64 QEImage Class Reference

Inheritance diagram for QEImage:



## Public Types

- enum `selectOptions` {  
`SO_NONE`, `SO_PANNING`, `SO_VSLICE`, `SO_HSLICE`,  
`SO_AREA1`, `SO_AREA2`, `SO_AREA3`, `SO_AREA4`,  
`SO_PROFILE`, `SO_TARGET`, `SO_BEAM` }
- enum `formatOptions` { `GREY8`, `GREY12`, `GREY16`, `RGB_888` }
- enum `resizeOptions` { `RESIZE_OPTION_ZOOM`, `RESIZE_OPTION_FIT` }
- enum `rotationOptions` { `ROTATION_0`, `ROTATION_90_RIGHT`, `ROTATION_90_LEFT`, `ROTATION_180` }
- enum `UserLevels` { `User` = `USERLEVEL_USER`, `Scientist` = `USERLEVEL_SCIENTIST`,  
`Engineer` = `USERLEVEL_ENGINEER` }
- enum `FormatOptions` { `Grey_8` = `QImage::GREY8`, `Grey_12` = `QImage::GREY12`,  
`Grey_16` = `QImage::GREY16`, `RGB` = `QImage::RGB_888` }
- enum `ResizeOptions` { `Zoom` = `QImage::RESIZE_OPTION_ZOOM`, `Fit` = `QImage::RESIZE_OPTION_FIT` }
- enum `RotationOptions` { `NoRotation` = `QImage::ROTATION_0`, `Rotate90Right`  
= `QImage::ROTATION_90_RIGHT`, `Rotate90Left` = `QImage::ROTATION_90_LEFT`, `Rotate180`  
= `QImage::ROTATION_180` }

## Public Slots

- void `setSelectPanMode` ()  
*Framework use only. Slot to allow external setting of selection menu options.*
- void `setSelectVSliceMode` ()  
*Framework use only. Slot to allow external setting of selection menu options.*
- void `setSelectHSliceMode` ()  
*Framework use only. Slot to allow external setting of selection menu options.*
- void `setSelectArea1Mode` ()  
*Framework use only. Slot to allow external setting of selection menu options.*
- void `setSelectArea2Mode` ()  
*Framework use only. Slot to allow external setting of selection menu options.*
- void `setSelectArea3Mode` ()  
*Framework use only. Slot to allow external setting of selection menu options.*
- void `setSelectArea4Mode` ()  
*Framework use only. Slot to allow external setting of selection menu options.*
- void `setSelectProfileMode` ()  
*Framework use only. Slot to allow external setting of selection menu options.*
- void `setSelectTargetMode` ()  
*Framework use only. Slot to allow external setting of selection menu options.*
- void `setSelectBeamMode` ()  
*Framework use only. Slot to allow external setting of selection menu options.*
- void `pauseClicked` ()  
*Framework use only. Slot to allow external setting of selection menu options.*

- void [saveClicked](#) ()  
*Framework use only. Slot to allow external setting of selection menu options.*
- void [roi1Changed](#) ()  
*Framework use only. Slot to allow external setting of selection menu options.*
- void [roi2Changed](#) ()  
*Framework use only. Slot to allow external setting of selection menu options.*
- void [roi3Changed](#) ()  
*Framework use only. Slot to allow external setting of selection menu options.*
- void [roi4Changed](#) ()  
*Framework use only. Slot to allow external setting of selection menu options.*
- void [targetClicked](#) ()  
*Framework use only. Slot to allow external setting of selection menu options.*
- void [requestEnabled](#) (const bool &state)

## Signals

- void [dbValueChanged](#) (const QString &out)
- void [requestResend](#) ()  
*Internal use only. Used when changing a property value to force a re-display to reflect the new property value.*

## Public Member Functions

- [QEImage](#) (QWidget \*parent=0)
- [QEImage](#) (const QString &variableName, QWidget \*parent=0)
- [~QEImage](#) ()  
*Destructor.*
- [selectOptions](#) [getSelectionOption](#) ()
- void [setFormatOption](#) ([formatOptions](#) formatOption)  
*Access function for #formatOption property - refer to #formatOption property for details.*
- [formatOptions](#) [getFormatOption](#) ()  
*Access function for #formatOption property - refer to #formatOption property for details.*
- void [setResizeOption](#) ([resizeOptions](#) resizeModeIn)  
*Access function for #resizeOption property - refer to #resizeOption property for details.*
- [resizeOptions](#) [getResizeOption](#) ()  
*Access function for #resizeOption property - refer to #resizeOption property for details.*
- void [setZoom](#) (int zoomIn)  
*Access function for zoom property - refer to zoom property for details.*
- int [getZoom](#) ()  
*Access function for zoom property - refer to zoom property for details.*
- void [setRotation](#) ([rotationOptions](#) rotationIn)

- Access function for `#rotation` property - refer to `#rotation` property for details.

  - [rotationOptions getRotation](#) ()
- Access function for `#rotation` property - refer to `#rotation` property for details.

  - void [setHorizontalFlip](#) (bool flipHozIn)
- Access function for `horizontalFlip` property - refer to `horizontalFlip` property for details.

  - bool [getHorizontalFlip](#) ()
- Access function for `horizontalFlip` property - refer to `horizontalFlip` property for details.

  - void [setVerticalFlip](#) (bool flipVertIn)
- Access function for `verticalFlip` property - refer to `verticalFlip` property for details.

  - bool [getVerticalFlip](#) ()
- Access function for `verticalFlip` property - refer to `verticalFlip` property for details.

  - void [setInitialHozScrollPos](#) (int initialHosScrollPosIn)
- Access function for `initialHosScrollPos` property - refer to `initialHosScrollPos` property for details.

  - int [getInitialHozScrollPos](#) ()
- Access function for `initialHosScrollPos` property - refer to `initialHosScrollPos` property for details.

  - void [setInitialVertScrollPos](#) (int initialVertScrollPosIn)
- Access function for `initialVertScrollPos` property - refer to `initialVertScrollPos` property for details.

  - int [getInitialVertScrollPos](#) ()
- Access function for `initialVertScrollPos` property - refer to `initialVertScrollPos` property for details.

  - void [setDisplayButtonBar](#) (bool displayButtonBarIn)
- Access function for `displayButtonBar` property - refer to `displayButtonBar` property for details.

  - bool [getDisplayButtonBar](#) ()
- Access function for `displayButtonBar` property - refer to `displayButtonBar` property for details.

  - void [setShowTime](#) (bool pValue)
- Access function for `showTime` property - refer to `showTime` property for details.

  - bool [getShowTime](#) ()
- Access function for `showTime` property - refer to `showTime` property for details.

  - void [setVertSliceMarkupColor](#) (QColor pValue)
- Access function for `vertSliceColor` property - refer to `vertSliceColor` property for details.

  - QColor [getVertSliceMarkupColor](#) ()
- Access function for `vertSliceColor` property - refer to `vertSliceColor` property for details.

  - void [setHozSliceMarkupColor](#) (QColor pValue)
- Access function for `hozSliceColor` property - refer to `hozSliceColor` property for details.

  - QColor [getHozSliceMarkupColor](#) ()
- Access function for `hozSliceColor` property - refer to `hozSliceColor` property for details.

  - void [setProfileMarkupColor](#) (QColor pValue)
- Access function for `profileColor` property - refer to `profileColor` property for details.

  - QColor [getProfileMarkupColor](#) ()

- Access function for [profileColor](#) property - refer to [profileColor](#) property for details.

  - void [setAreaMarkupColor](#) (QColor pValue)
- Access function for [areaColor](#) property - refer to [areaColor](#) property for details.

  - QColor [getAreaMarkupColor](#) ()
- Access function for [areaColor](#) property - refer to [areaColor](#) property for details.

  - void [setTargetMarkupColor](#) (QColor pValue)
- Access function for [targetColor](#) property - refer to [targetColor](#) property for details.

  - QColor [getTargetMarkupColor](#) ()
- Access function for [targetColor](#) property - refer to [targetColor](#) property for details.

  - void [setBeamMarkupColor](#) (QColor pValue)
- Access function for [beamColor](#) property - refer to [beamColor](#) property for details.

  - QColor [getBeamMarkupColor](#) ()
- Access function for [beamColor](#) property - refer to [beamColor](#) property for details.

  - void [setTimeMarkupColor](#) (QColor pValue)
- Access function for [timeColor](#) property - refer to [timeColor](#) property for details.

  - QColor [getTimeMarkupColor](#) ()
- Access function for [timeColor](#) property - refer to [timeColor](#) property for details.

  - void [setDisplayCursorPixelInfo](#) (bool displayCursorPixelInfoIn)
- Access function for [#displayCursorPixelInfo](#) property - refer to [#displayCursorPixelInfo](#) property for details.

  - bool [getDisplayCursorPixelInfo](#) ()
- Access function for [#displayCursorPixelInfo](#) property - refer to [#displayCursorPixelInfo](#) property for details.

  - void [setContrastReversal](#) (bool contrastReversalIn)
- Access function for [#contrastReversal](#) property - refer to [#contrastReversal](#) property for details.

  - bool [getContrastReversal](#) ()
- Access function for [#contrastReversal](#) property - refer to [#contrastReversal](#) property for details.

  - void [setEnableVertSliceSelection](#) (bool enableVSliceSelectionIn)
- Access function for [enableVertSliceSelection](#) property - refer to [enableVertSliceSelection](#) property for details.

  - bool [getEnableVertSliceSelection](#) ()
- Access function for [enableVertSliceSelection](#) property - refer to [enableVertSliceSelection](#) property for details.

  - void [setEnableHozSliceSelection](#) (bool enableHSliceSelectionIn)
- Access function for [enableHozSliceSelection](#) property - refer to [enableHozSliceSelection](#) property for details.

  - bool [getEnableHozSliceSelection](#) ()
- Access function for [enableHozSliceSelection](#) property - refer to [enableHozSliceSelection](#) property for details.

  - void [setEnableAreaSelection](#) (bool enableAreaSelectionIn)
- Access function for [#enableAreaSelection](#) property - refer to [#enableAreaSelection](#) property for details.

  - bool [getEnableAreaSelection](#) ()



- Access function for #enableAreaSelection property - refer to #enableAreaSelection property for details.*
- void [setEnableProfileSelection](#) (bool enableProfileSelectionIn)
 

*Access function for #enableProfileSelection property - refer to #enableProfileSelection property for details.*
  - bool [getEnableProfileSelection](#) ()
 

*Access function for #enableProfileSelection property - refer to #enableProfileSelection property for details.*
  - void [setEnableTargetSelection](#) (bool enableTargetSelectionIn)
 

*Access function for #enableTargetSelection property - refer to #enableTargetSelection property for details.*
  - bool [getEnableTargetSelection](#) ()
 

*Access function for #enableTargetSelection property - refer to #enableTargetSelection property for details.*
  - bool [isEnabled](#) () const
 

*Access function for [enabled](#) property - refer to [enabled](#) property for details.*
  - void [setEnabled](#) (bool state)
 

*Access function for [enabled](#) property - refer to [enabled](#) property for details.*
  - [UserLevels](#) [getUserLevelVisibilityProperty](#) ()
 

*Access function for [userLevelVisibility](#) property - refer to [userLevelVisibility](#) property for details.*
  - void [setUserLevelVisibilityProperty](#) ([UserLevels](#) level)
 

*Access function for [userLevelVisibility](#) property - refer to [userLevelVisibility](#) property for details.*
  - [UserLevels](#) [getUserLevelEnabledProperty](#) ()
 

*Access function for [userLevelEnabled](#) property - refer to [userLevelEnabled](#) property for details.*
  - void [setUserLevelEnabledProperty](#) ([UserLevels](#) level)
 

*Access function for [userLevelEnabled](#) property - refer to [userLevelEnabled](#) property for details.*
  - void [setFormatOptionProperty](#) ([FormatOptions](#) formatOption)
 

*Access function for #formatOption property - refer to #formatOption property for details.*
  - [FormatOptions](#) [getFormatOptionProperty](#) ()
 

*Access function for #formatOption property - refer to #formatOption property for details.*
  - void [setResizeOptionProperty](#) ([ResizeOptions](#) resizeOption)
 

*Access function for #resizeOption property - refer to #resizeOption property for details.*
  - [ResizeOptions](#) [getResizeOptionProperty](#) ()
 

*Access function for #resizeOption property - refer to #resizeOption property for details.*
  - void [setRotationProperty](#) ([RotationOptions](#) rotation)
 

*Access function for #rotation property - refer to #rotation property for details.*
  - [RotationOptions](#) [getRotationProperty](#) ()
 

*Access function for #rotation property - refer to #rotation property for details.*

## Protected Types

- enum **variableIndexes** {  
**IMAGE\_VARIABLE**, **WIDTH\_VARIABLE**, **HEIGHT\_VARIABLE**, **ROI1\_X\_VARIABLE**,  
**ROI1\_Y\_VARIABLE**, **ROI1\_W\_VARIABLE**, **ROI1\_H\_VARIABLE**, **ROI2\_X\_VARIABLE**,  
**ROI2\_Y\_VARIABLE**, **ROI2\_W\_VARIABLE**, **ROI2\_H\_VARIABLE**, **ROI3\_X\_VARIABLE**,  
**ROI3\_Y\_VARIABLE**, **ROI3\_W\_VARIABLE**, **ROI3\_H\_VARIABLE**, **ROI4\_X\_VARIABLE**,  
**ROI4\_Y\_VARIABLE**, **ROI4\_W\_VARIABLE**, **ROI4\_H\_VARIABLE**, **TARGET\_X\_VARIABLE**,  
**TARGET\_Y\_VARIABLE**, **BEAM\_X\_VARIABLE**, **BEAM\_Y\_VARIABLE**, **TARGET\_TRIGGER\_VARIABLE**,  
**CLIPPING\_ONOFF\_VARIABLE**, **CLIPPING\_LOW\_VARIABLE**, **CLIPPING\_HIGH\_VARIABLE**, **QEIMAGE\_NUM\_VARIABLES** }

## Protected Member Functions

- void **establishConnection** (unsigned int variableIndex)
- void **dragEnterEvent** (QDragEnterEvent \*event)
- void **dropEvent** (QDropEvent \*event)
- void **setDrop** (QVariant drop)
- QVariant **getDrop** ()
- QString **copyVariable** ()
- QVariant **copyData** ()
- void **paste** (QVariant v)
- void **resizeEvent** (QResizeEvent \*)

## Protected Attributes

- [QEIntegerFormatting](#) **integerFormatting**
- [resizeOptions](#) **resizeOption**
- int [zoom](#)  
*Zoom percentage. Used when #resizeOption is [Zoom](#).*
- [rotationOptions](#) **rotation**
- bool **flipVert**
- bool **flipHoz**
- int **initialHozScrollPos**
- int [initialVertScrollPos](#)
- bool [displayButtonBar](#)

## Properties

- QString [imageVariable](#)
- QString [widthVariable](#)
- QString [heightVariable](#)
- QString [regionOfInterest1XVariable](#)
- QString [regionOfInterest1YVariable](#)
- QString [regionOfInterest1WVariable](#)
- QString [regionOfInterest1HVariable](#)
- QString [regionOfInterest2XVariable](#)
- QString [regionOfInterest2YVariable](#)
- QString [regionOfInterest2WVariable](#)
- QString [regionOfInterest2HVariable](#)
- QString [regionOfInterest3XVariable](#)
- QString [regionOfInterest3YVariable](#)
- QString [regionOfInterest3WVariable](#)
- QString [regionOfInterest3HVariable](#)
- QString [regionOfInterest4XVariable](#)
- QString [regionOfInterest4YVariable](#)
- QString [regionOfInterest4WVariable](#)
- QString [regionOfInterest4HVariable](#)
- QString [targetXVariable](#)
- QString [targetYVariable](#)
- QString [beamXVariable](#)
- QString [beamYVariable](#)
- QString [targetTriggerVariable](#)
- QString [clippingOnOffVariable](#)
- QString [clippingLowVariable](#)
- QString [clippingHighVariable](#)
- QString [variableSubstitutions](#)
- bool [variableAsToolTip](#)
- bool [enabled](#)
- bool [allowDrop](#)
- bool [visible](#)
- unsigned int
- QString [userLevelUserStyle](#)
- QString [userLevelScientistStyle](#)
- QString [userLevelEngineerStyle](#)
- UserLevels [userLevelVisibility](#)
- UserLevels [userLevelEnabled](#)
- FormatOptions [formatOption](#)
- bool [enableVertSliceSelection](#)
- bool [enableHozSliceSelection](#)
- bool [showTime](#)
- QColor [vertSliceColor](#)
- QColor [hozSliceColor](#)
- QColor [profileColor](#)

- QColor [areaColor](#)
- QColor [beamColor](#)
- QColor [targetColor](#)
- QColor [timeColor](#)
- [ResizeOptions](#) [resizeOption](#)
- [RotationOptions](#) [rotation](#)
- bool [verticalFlip](#)
- bool [horizontalFlip](#)
- int [initialHosScrollPos](#)

### 9.64.1 Member Enumeration Documentation

#### 9.64.1.1 enum QImage::formatOptions

Video format options

##### Enumerator:

- GREY8*** 8 bit grey scale
- GREY12*** 12 bit grey scale
- GREY16*** 16 bit grey scale
- RGB\_888*** 24 bit RGB

#### 9.64.1.2 enum QImage::FormatOptions

User friendly enumerations for #formatOption property - refer to #formatOption property and [formatOptions](#) enumeration for details.

##### Enumerator:

- Grey\_8*** 8 bit grey scale
- Grey\_12*** 12 bit grey scale
- Grey\_16*** 16 bit grey scale

#### 9.64.1.3 enum QImage::ResizeOptions

User friendly enumerations for #resizeOption property

##### Enumerator:

- Zoom*** Zoom to selected percentage.
- Fit*** Zoom to fit the current window size.

## 9.64.1.4 enum QImage::resizeOptions

Image resize options

Enumerator:

**RESIZE\_OPTION\_ZOOM** Zoom to selected percentage.  
**RESIZE\_OPTION\_FIT** Zoom to fit the current window size.

## 9.64.1.5 enum QImage::rotationOptions

Image rotation options

Enumerator:

**ROTATION\_0** No image rotation.  
**ROTATION\_90\_RIGHT** Rotate image 90 degrees clockwise.  
**ROTATION\_90\_LEFT** Rotate image 90 degrees anticlockwise.  
**ROTATION\_180** Rotate image 180 degrees.

## 9.64.1.6 enum QImage::RotationOptions

User friendly enumerations for #rotation property

Enumerator:

**NoRotation** No image rotation.  
**Rotate90Right** Rotate image 90 degrees clockwise.  
**Rotate90Left** Rotate image 90 degrees anticlockwise.  
**Rotate180** Rotate image 180 degrees.

## 9.64.1.7 enum QImage::selectOptions

Internal use only. Selection options. What will happen when the user interacts with the image area

Enumerator:

**SO\_NONE** Do nothing.  
**SO\_PANNING** User is panning.  
**SO\_VSLICE** Select the vertical slice point.  
**SO\_HSLICE** Select the horizontal slice point.  
**SO\_AREA4** User is selecting an area (for region of interest)  
**SO\_PROFILE** Select an arbitrary line across the image (to determine a profile)  
**SO\_TARGET** Mark the target point.  
**SO\_BEAM** Mark the current beam location.

### 9.64.1.8 enum QImage::UserLevels

User friendly enumerations for [userLevelVisibility](#) and [userLevelEnabled](#) properties - refer to [userLevelVisibility](#) and [userLevelEnabled](#) properties and [userLevel](#) enumeration for details.

#### Enumerator:

**User** Refer to USERLEVEL\_USER for details.

**Scientist** Refer to USERLEVEL\_SCIENTIST for details.

**Engineer** Refer to USERLEVEL\_ENGINEER for details.

## 9.64.2 Constructor & Destructor Documentation

### 9.64.2.1 QImage::QImage ( QWidget \* parent = 0 )

Create without a variable. Use `setVariableName'n'Property()` - where 'n' is a number from 0 to 26 - and `setSubstitutionsProperty()` to define variables and, optionally, macro substitutions later. Note, each variable property is named by function (such as `imageVariable` and `widthVariable`) but given a numeric get and set property access function such as `setVariableName22Property()`. Refer to the property definitions to determine what 'set' and 'get' function is used for each variable, or use Qt library functions to set or get the variable names by name.

### 9.64.2.2 QImage::QImage ( const QString & variableName, QWidget \* parent = 0 )

Create with a variable. A connection is automatically established. The variable is set up as the first variable. This is consistent with other widgets, but will not result in an updating image as the width and height variables are required as a minimum.

## 9.64.3 Member Function Documentation

### 9.64.3.1 void QImage::dbValueChanged ( const QString & out ) [signal]

Sent when the widget is updated following a data change Can be used to pass on EPICS data (as presented in this widget) to other widgets. For example a QList widget could log updates from this widget.

### 9.64.3.2 void QImage::requestEnabled ( const bool & state ) [inline, slot]

Similar to standard `setEnabled` slot, but allows QE widget to determine if the widget remains disabled due to invalid data. If disabled due to invalid data, a request to enable the widget will be honoured when the data is no longer invalid.

### 9.64.4 Member Data Documentation

9.64.4.1 **bool QImage::displayButtonBar** [read, write, protected]

If true, a button bar will be displayed above the image. If not displayed, all buttons in the button bar are still available in the right click menu.

9.64.4.2 **int QImage::initialVertScrollPos** [read, write, protected]

Sets the initial position of the vertical scroll bar, if present. Used to set up an initial view when zoomed in.

### 9.64.5 Property Documentation

9.64.5.1 **bool QImage::allowDrop** [read, write]

Allow drag/drops operations to this widget. Default is false. Any dropped text will be used as a new variable name.

Reimplemented from [QEDragDrop](#).

9.64.5.2 **QColor QImage::areaColor** [read, write]

Used to select the color of the area selection markups.

9.64.5.3 **QColor QImage::beamColor** [read, write]

Used to select the color of the beam marker.

9.64.5.4 **QString QImage::beamXVariable** [read, write]

EPICS variable name (CA PV). This variable is used to write the selected beam X position.

9.64.5.5 **QString QImage::beamYVariable** [read, write]

EPICS variable name (CA PV). This variable is used to write the selected beam Y position.

9.64.5.6 **QString QImage::clippingHighVariable** [read, write]

EPICS variable name (CA PV). This variable is used to write the areadetector clipping high level.

**9.64.5.7 QString QEImage::clippingLowVariable** [read, write]

EPICS variable name (CA PV). This variable is used to write the areadetector clipping low level.

**9.64.5.8 QString QEImage::clippingOnOffVariable** [read, write]

EPICS variable name (CA PV). This variable is used to write the areadetector clipping on/off command.

**9.64.5.9 bool QEImage::enabled** [read, write]

Set the preferred 'enabled' state. Default is true. This property is copied to the standard Qt 'enabled' property if the data being displayed is valid. If the data being displayed is invalid the standard Qt 'enabled' property will always be set to false to indicate invalid data. The value of this property will only be copied to the standard Qt 'enabled' property once data is valid.

**9.64.5.10 bool QEImage::enableHozSliceSelection** [read, write]

If true, the option to select a horizontal slice through the image will be available to the user. This will be used to generate a horizontal pixel profile.

**9.64.5.11 bool QEImage::enableVertSliceSelection** [read, write]

If true, the option to select a vertical slice through the image will be available to the user. This will be used to generate a vertical pixel profile.

**9.64.5.12 FormatOptions QEImage::formatOption** [read, write]

Video format. EPICS data type size will typically be adequate for the number of bits required (one byte for 8 bits, 2 bytes for 12 and 16 bits), but can be larger (4 bytes for 24 bits.)

**9.64.5.13 QString QEImage::heightVariable** [read, write]

EPICS variable name (CA PV). This variable is used to read the height of the image.

**9.64.5.14 bool QEImage::horizontalFlip** [read, write]

If true, flip image horizontally.



**9.64.5.15** `QColor QEImage::hozSliceColor` [read, write]

Used to select the color of the horizontal slice markup.

**9.64.5.16** `QString QEImage::imageVariable` [read, write]

EPICS variable name (CA PV). This variable is used as the source the image waveform.

**9.64.5.17** `int QEImage::initialHosScrollPos` [read, write]

Sets the initial position of the horizontal scroll bar, if present. Used to set up an initial view when zoomed in.

**9.64.5.18** `unsigned QEImage::int` [read, write]

Set the ID used by the message filtering system. Default is zero. Widgets or applications that use messages from the framework have the option of filtering on this ID. For example, by using a unique message source ID a [QELog](#) widget may be set up to only log messages from a select set of widgets.

**9.64.5.19** `QColor QEImage::profileColor` [read, write]

Used to select the color of the arbitrary profile line markup.

**9.64.5.20** `QString QEImage::regionOfInterest1HVariable` [read, write]

EPICS variable name (CA PV). This variable is used to write the first region of interest height.

**9.64.5.21** `QString QEImage::regionOfInterest1WVariable` [read, write]

EPICS variable name (CA PV). This variable is used to write the first region of interest width.

**9.64.5.22** `QString QEImage::regionOfInterest1XVariable` [read, write]

EPICS variable name (CA PV). This variable is used to write the first region of interest X position.

**9.64.5.23** `QString QEImage::regionOfInterest1YVariable` [read, write]

EPICS variable name (CA PV). This variable is used to write the first region of interest Y position.

**9.64.5.24 QString QEImage::regionOfInterest2HVariable** [read, write]

EPICS variable name (CA PV). This variable is used to write the second region of interest height.

**9.64.5.25 QString QEImage::regionOfInterest2WVariable** [read, write]

EPICS variable name (CA PV). This variable is used to write the second region of interest width.

**9.64.5.26 QString QEImage::regionOfInterest2XVariable** [read, write]

EPICS variable name (CA PV). This variable is used to write the second region of interest X position.

**9.64.5.27 QString QEImage::regionOfInterest2YVariable** [read, write]

EPICS variable name (CA PV). This variable is used to write the second region of interest Y position.

**9.64.5.28 QString QEImage::regionOfInterest3HVariable** [read, write]

EPICS variable name (CA PV). This variable is used to write the third region of interest height.

**9.64.5.29 QString QEImage::regionOfInterest3WVariable** [read, write]

EPICS variable name (CA PV). This variable is used to write the third region of interest width.

**9.64.5.30 QString QEImage::regionOfInterest3XVariable** [read, write]

EPICS variable name (CA PV). This variable is used to write the third region of interest X position.

**9.64.5.31 QString QEImage::regionOfInterest3YVariable** [read, write]

EPICS variable name (CA PV). This variable is used to write the third region of interest Y position.

**9.64.5.32 QString QEImage::regionOfInterest4HVariable** [read, write]

EPICS variable name (CA PV). This variable is used to write the fourth region of interest height.

**9.64.5.33** `QString QElImage::regionOfInterest4WVariable` [read, write]

EPICS variable name (CA PV). This variable is used to write the fourth region of interest width.

**9.64.5.34** `QString QElImage::regionOfInterest4XVariable` [read, write]

EPICS variable name (CA PV). This variable is used to write the fourth region of interest X position.

**9.64.5.35** `QString QElImage::regionOfInterest4YVariable` [read, write]

EPICS variable name (CA PV). This variable is used to write the fourth region of interest Y position.

**9.64.5.36** `ResizeOptions QElImage::resizeOption` [read, write]

Resize option. Zoom to zoom to the percentage given by the [zoom](#) property, or fit to the window size.

**9.64.5.37** `RotationOptions QElImage::rotation` [read, write]

Image rotation option.

**9.64.5.38** `bool QElImage::showTime` [read, write]

If true, the image timestamp will be written in the top left of the image.

**9.64.5.39** `QColor QElImage::targetColor` [read, write]

Used to select the color of the target marker.

**9.64.5.40** `QString QElImage::targetTriggerVariable` [read, write]

EPICS variable name (CA PV). This variable is used to write a 'trigger' to initiate movement of the target into the beam as defined by the target and beam X and Y positions.

**9.64.5.41** `QString QElImage::targetXVariable` [read, write]

EPICS variable name (CA PV). This variable is used to write the selected target X position.

**9.64.5.42 QString QEImage::targetYVariable** [read, write]

EPICS variable name (CA PV). This variable is used to write the selected target Y position.

**9.64.5.43 QColor QEImage::timeColor** [read, write]

Used to select the color of the timestamp.

**9.64.5.44 UserLevels QEImage::userLevelEnabled** [read, write]

Lowest user level at which the widget is enabled. Default is 'User'. Used when designing GUIs that allow access to more and more detail according to the user mode. The user mode is set application wide through the [QELogin](#) widget, or programatically through `setUserLevel()` Widgets that are always accessible should be visible at 'User'. Widgets that are only accessible to scientists managing the facility should be visible at 'Scientist'. Widgets that are only accessible to engineers maintaining the facility should be visible at 'Engineer'.

**9.64.5.45 QString QEImage::userLevelEngineerStyle** [read, write]

Style Sheet string to be applied when the widget is displayed in 'Engineer' mode. Default is an empty string. The syntax is the standard Qt Style Sheet syntax. For example, 'background-color: red' This Style Sheet string will be applied by the `styleManager` class. Refer to the `styleManager` class for details about how this Style Sheet string will be merged with any pre-existing Style Sheet string and any Style Sheet strings generated during the display of data.

**9.64.5.46 QString QEImage::userLevelScientistStyle** [read, write]

Style Sheet string to be applied when the widget is displayed in 'Scientist' mode. Default is an empty string. The syntax is the standard Qt Style Sheet syntax. For example, 'background-color: red' This Style Sheet string will be applied by the `styleManager` class. Refer to the `styleManager` class for details about how this Style Sheet string will be merged with any pre-existing Style Sheet string and any Style Sheet strings generated during the display of data.

**9.64.5.47 QString QEImage::userLevelUserStyle** [read, write]

Style Sheet string to be applied when the widget is displayed in 'User' mode. Default is an empty string. The syntax is the standard Qt Style Sheet syntax. For example, 'background-color: red' This Style Sheet string will be applied by the `styleManager` class. Refer to the `styleManager` class for details about how this Style Sheet string will be merged with any pre-existing Style Sheet string and any Style Sheet strings generated during the display of data.

**9.64.5.48 UserLevels QEImage::userLevelVisibility** [read, write]

Lowest user level at which the widget is visible. Default is 'User'. Used when designing GUIs that display more and more detail according to the user mode. The user mode is set application wide through the [QELogin](#) widget, or programatically through setUserLevel(). Widgets that are always visible should be visible at 'User'. Widgets that are only used by scientists managing the facility should be visible at 'Scientist'. Widgets that are only used by engineers maintaining the facility should be visible at 'Engineer'.

**9.64.5.49 bool QEImage::variableAsToolTip** [read, write]

Use the variable as the tool tip. Default is true. Tool tip property will be overwritten by the variable name.

Reimplemented from [QEToolTip](#).

**9.64.5.50 QString QEImage::variableSubstitutions** [read, write]

Macro substitutions. The default is no substitutions. The format is NAME1=VALUE1[, NAME2=VALUE2... Values may be quoted strings. For example, 'CAM=1, NAME = "Image 1"' These substitutions are applied to all the variable names.

**9.64.5.51 bool QEImage::verticalFlip** [read, write]

If true, flip image vertically.

**9.64.5.52 QColor QEImage::vertSliceColor** [read, write]

Used to select the color of the vertical slice markup.

**9.64.5.53 bool QEImage::visible** [read, write]

Display the widget. Default is true. Setting this property false is usefull if widget is only used to provide a signal - for example, when supplying data to a [QELink](#) widget. Note, when false the widget will still be visible in Qt Designer.

**9.64.5.54 QString QEImage::widthVariable** [read, write]

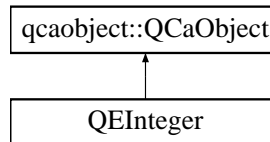
EPICS variable name (CA PV). This variable is used to read the width of the image.

The documentation for this class was generated from the following files:

- /home/rhydera/epicsqt/framework/widgets/QEImage/QEImage.h
- /home/rhydera/epicsqt/framework/widgets/QEImage/QEImage.cpp

## 9.65 QEInteger Class Reference

Inheritance diagram for QEInteger:



### Public Slots

- void **writeInteger** (const long &data)

### Signals

- void **integerConnectionChanged** ([QCaConnectionInfo](#) &connectionInfo, const unsigned int &variableIndex)
- void **integerChanged** (const long &value, [QCaAlarmInfo](#) &alarmInfo, [QCaDateTime](#) &timeStamp, const unsigned int &variableIndex)
- void **integerArrayChanged** (const QVector< long > &values, [QCaAlarmInfo](#) &alarmInfo, [QCaDateTime](#) &timeStamp, const unsigned int &variableIndex)

### Public Member Functions

- **QEInteger** (QString recordName, QObject \*eventObject, [QEIntegerFormatting](#) \*integerFormattingIn, unsigned int variableIndexIn)
- **QEInteger** (QString recordName, QObject \*eventObject, [QEIntegerFormatting](#) \*integerFormattingIn, unsigned int variableIndexIn, [UserMessage](#) \*userMessageIn)

The documentation for this class was generated from the following files:

- /home/rhydera/epicsqt/framework/data/include/QEInteger.h
- /home/rhydera/epicsqt/framework/data/src/QEInteger.cpp

## 9.66 QEIntegerFormatting Class Reference

```
#include <QEIntegerFormatting.h>
```

## Public Member Functions

- [QEIntegerFormatting](#) ()  
*Constructor.*
- long [formatInteger](#) (const QVariant &value)
- QVector< long > [formatIntegerArray](#) (const QVariant &value)
- QVariant [formatValue](#) (const long &integerValue, generic::generic\_types value-Type)
- void [setRadix](#) (unsigned int radix)  
*Set the radix used for all conversions. Default is 10.*
- unsigned int [getPrecision](#) ()  
*Get the precision used for all conversions.*
- unsigned int [getRadix](#) ()  
*Get the radix used for all conversions.*

### 9.66.1 Detailed Description

This class holds formatting instructions and uses them to convert between an integer and a QVariant of any type. It is generally set up with its formatting instructions and then passed to a [QEInteger](#) class that will sink and source integer data to widgets or other code. It is used to convert data to and from a QCaObject (which sources and sinks data in the form of a QVariant where the QVariant reflects the underlying variable data type) and the [QEInteger](#) class. An example of a requirement for integer data is a combo box which must determine an integer index to select a menu option.

### 9.66.2 Member Function Documentation

#### 9.66.2.1 long QEIntegerFormatting::formatInteger ( const QVariant & value )

Given a data value of any type, format it as an integer according to the formatting instructions held by the class. This is used to convert the QVariant value received from a QCaObject, which is still based on the data variable type, to an integer.

#### 9.66.2.2 QVector< long > QEIntegerFormatting::formatIntegerArray ( const QVariant & value )

Given a data value of any type, format it as an array of integers according to the formatting instructions held by the class. This is used to convert the QVariant value received from a QCaObject, which is still based on the data variable type, to an integer array. Typically used where the input QVariant value is an array of data values, but will work for any QVariant type.

### 9.66.2.3 QVariant QEIntegerFormatting::formatValue ( const long & *integerValue*, generic::generic\_types *valueType* )

Given an integer value, format it as a data value of the specified type, according to the formatting instructions held by the class. This is used when writing integer data to a QCaObject.

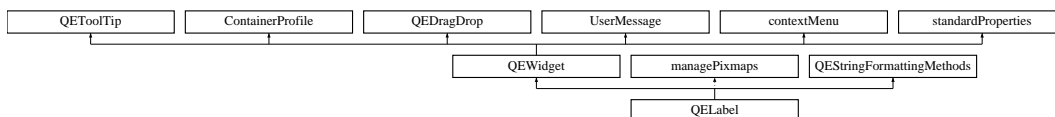
The documentation for this class was generated from the following files:

- /home/rhydera/epicsqt/framework/data/include/QEIntegerFormatting.h
- /home/rhydera/epicsqt/framework/data/src/QEIntegerFormatting.cpp

## 9.67 QELabel Class Reference

```
#include <QELabel.h>
```

Inheritance diagram for QELabel:



### Public Types

- enum [updateOptions](#) { [UPDATE\\_TEXT](#), [UPDATE\\_PIXMAP](#) }
- enum [UserLevels](#) { [User](#) = USERLEVEL\_USER, [Scientist](#) = USERLEVEL\_SCIENTIST, [Engineer](#) = USERLEVEL\_ENGINEER }
- enum [Formats](#) {  
[Default](#) = QEStringFormatting::FORMAT\_DEFAULT, [Floating](#) = QEStringFormatting::FORMAT\_FLOATING, [Integer](#) = QEStringFormatting::FORMAT\_INTEGER, [UnsignedInteger](#) = QEStringFormatting::FORMAT\_UNSIGNEDINTEGER,  
[Time](#) = QEStringFormatting::FORMAT\_TIME, [LocalEnumeration](#) = QEStringFormatting::FORMAT\_LOCAL\_ENUMERATE }
- enum [Notations](#) { [Fixed](#) = QEStringFormatting::NOTATION\_FIXED, [Scientific](#) = QEStringFormatting::NOTATION\_SCIENTIFIC, [Automatic](#) = QEStringFormatting::NOTATION\_AUTOMATIC }
- enum [ArrayActions](#) { [Append](#) = QEStringFormatting::APPEND, [Ascii](#) = QEStringFormatting::ASCII, [Index](#) = QEStringFormatting::INDEX }
- enum [UpdateOptions](#) { [Text](#) = QELabel::UPDATE\_TEXT, [Picture](#) = QELabel::UPDATE\_PIXMAP }

*User friendly enumerations for updateOption property - refer to [QELabel::updateOptions](#) for details.*



## Public Slots

- void [requestEnabled](#) (const bool &state)

## Signals

- void [dbValueChanged](#) (const QString &out)
- void [requestResend](#) ()  
*Internal use only. Used when changing a property value to force a re-display to reflect the new property value.*

## Public Member Functions

- [QELabel](#) (QWidget \*parent=0)
- [QELabel](#) (const QString &variableName, QWidget \*parent=0)
- bool [isEnabled](#) () const  
*Access function for [enabled](#) property - refer to [enabled](#) property for details.*
- void [setEnabled](#) (bool state)  
*Access function for [enabled](#) property - refer to [enabled](#) property for details.*
- [UserLevels](#) [getUserLevelVisibilityProperty](#) ()  
*Access function for [userLevelVisibility](#) property - refer to [userLevelVisibility](#) property for details.*
- void [setUserLevelVisibilityProperty](#) ([UserLevels](#) level)  
*Access function for [userLevelVisibility](#) property - refer to [userLevelVisibility](#) property for details.*
- [UserLevels](#) [getUserLevelEnabledProperty](#) ()  
*Access function for [userLevelEnabled](#) property - refer to [userLevelEnabled](#) property for details.*
- void [setUserLevelEnabledProperty](#) ([UserLevels](#) level)  
*Access function for [userLevelEnabled](#) property - refer to [userLevelEnabled](#) property for details.*
- void [setFormatProperty](#) ([Formats](#) format)  
*Access function for [format](#) property - refer to [format](#) property for details.*
- [Formats](#) [getFormatProperty](#) ()  
*Access function for [format](#) property - refer to [format](#) property for details.*
- void [setNotationProperty](#) ([Notations](#) notation)  
*Access function for [notation](#) property - refer to [notation](#) property for details.*
- [Notations](#) [getNotationProperty](#) ()  
*Access function for [notation](#) property - refer to [notation](#) property for details.*
- void [setArrayActionProperty](#) ([ArrayActions](#) arrayAction)  
*Access function for [arrayAction](#) property - refer to [arrayAction](#) property for details.*
- [ArrayActions](#) [getArrayActionProperty](#) ()  
*Access function for [arrayAction](#) property - refer to [arrayAction](#) property for details.*
- void [setUpdateOptionProperty](#) ([UpdateOptions](#) updateOption)

Access function for `#updateOption` property - refer to `#updateOption` property for details.

- [UpdateOptions getUpdateOptionProperty \(\)](#)

Access function for `#updateOption` property - refer to `#updateOption` property for details.

- void [setPixmap0Property](#) (QPixmap pixmap)

'Set' access function for [pixmap0](#) properties. Refer to [pixmap0](#) property for details

- void [setPixmap1Property](#) (QPixmap pixmap)

'Set' access function for [pixmap1](#) properties. Refer to [pixmap1](#) property for details

- void [setPixmap2Property](#) (QPixmap pixmap)

'Set' access function for [pixmap2](#) properties. Refer to [pixmap2](#) property for details

- void [setPixmap3Property](#) (QPixmap pixmap)

'Set' access function for [pixmap3](#) properties. Refer to [pixmap3](#) property for details

- void [setPixmap4Property](#) (QPixmap pixmap)

'Set' access function for [pixmap4](#) properties. Refer to [pixmap4](#) property for details

- void [setPixmap5Property](#) (QPixmap pixmap)

'Set' access function for [pixmap5](#) properties. Refer to [pixmap5](#) property for details

- void [setPixmap6Property](#) (QPixmap pixmap)

'Set' access function for [pixmap6](#) properties. Refer to [pixmap6](#) property for details

- void [setPixmap7Property](#) (QPixmap pixmap)

'Set' access function for [pixmap7](#) properties. Refer to [pixmap7](#) property for details

- QPixmap [getPixmap0Property](#) ()

'Get' access function for [pixmap0](#) properties. Refer to [pixmap0](#) property for details

- QPixmap [getPixmap1Property](#) ()

'Get' access function for [pixmap1](#) properties. Refer to [pixmap1](#) property for details

- QPixmap [getPixmap2Property](#) ()

'Get' access function for [pixmap2](#) properties. Refer to [pixmap2](#) property for details

- QPixmap [getPixmap3Property](#) ()

'Get' access function for [pixmap3](#) properties. Refer to [pixmap3](#) property for details

- QPixmap [getPixmap4Property](#) ()

'Get' access function for [pixmap4](#) properties. Refer to [pixmap4](#) property for details

- QPixmap [getPixmap5Property](#) ()

'Get' access function for [pixmap5](#) properties. Refer to [pixmap5](#) property for details

- QPixmap [getPixmap6Property](#) ()

'Get' access function for [pixmap6](#) properties. Refer to [pixmap6](#) property for details

- QPixmap [getPixmap7Property](#) ()

'Get' access function for [pixmap7](#) properties. Refer to [pixmap7](#) property for details

## Properties

- QString [variable](#)
- QString [variableSubstitutions](#)
- bool [variableAsToolTip](#)
- bool [enabled](#)
- bool [allowDrop](#)
- bool [visible](#)
- unsigned [int](#)
- QString [userLevelUserStyle](#)
- QString [userLevelScientistStyle](#)
- QString [userLevelEngineerStyle](#)
- [UserLevels](#) [userLevelVisibility](#)
- [UserLevels](#) [userLevelEnabled](#)
- int [precision](#)
- bool [useDbPrecision](#)
- bool [leadingZero](#)
- bool [trailingZeros](#)
- bool [addUnits](#)
- QString [localEnumeration](#)
- [Formats](#) [format](#)
- [Notations](#) [notation](#)
- [ArrayActions](#) [arrayAction](#)
- [UpdateOptions](#) [updateOption](#)
- QPixmap [pixmap0](#)
- QPixmap [pixmap1](#)
- QPixmap [pixmap2](#)
- QPixmap [pixmap3](#)
- QPixmap [pixmap4](#)
- QPixmap [pixmap5](#)
- QPixmap [pixmap6](#)
- QPixmap [pixmap7](#)

### 9.67.1 Detailed Description

This class is a EPICS aware label widget based on the Qt label widget. When a variable is defined, the label text (or optionally the background pixmap) will be updated. The label will be disabled if the variable is invalid. It is tightly integrated with the base class [QEWidget](#) which provides generic support such as macro substitutions, drag/drop, and standard properties.

## 9.67.2 Member Enumeration Documentation

### 9.67.2.1 enum QELabel::ArrayActions

User friendly enumerations for arrayAction property - refer to [QCStringFormatting::arrayActions](#) for details.

#### Enumerator:

**Append** Refer to [QCStringFormatting::APPEND](#) for details.

**Ascii** Refer to [QCStringFormatting::ASCII](#) for details.

**Index** Refer to [QCStringFormatting::INDEX](#) for details.

### 9.67.2.2 enum QELabel::Formats

User friendly enumerations for format property - refer to [QCStringFormatting::formats](#) for details.

#### Enumerator:

**Default** Format as best appropriate for the data type.

**Floating** Format as a floating point number.

**Integer** Format as an integer.

**UnsignedInteger** Format as an unsigned integer.

**Time** Format as a time.

**LocalEnumeration** Format as a selection from the [localEnumeration](#) property.

### 9.67.2.3 enum QELabel::Notations

User friendly enumerations for notation property - refer to [QCStringFormatting::notations](#) for details.

#### Enumerator:

**Fixed** Refer to [QCStringFormatting::NOTATION\\_FIXED](#) for details.

**Scientific** Refer to [QCStringFormatting::NOTATION\\_SCIENTIFIC](#) for details.

**Automatic** Refer to [QCStringFormatting::NOTATION\\_AUTOMATIC](#) for details.

### 9.67.2.4 enum QELabel::UpdateOptions

User friendly enumerations for updateOption property - refer to [QELabel::updateOptions](#) for details.

#### Enumerator:

**Text** Data updates will update the label text.

**Picture** Data updates will update the label icon.

### 9.67.2.5 enum QELabel::updateOptions

Options for updating the label. The formatted text is used to update the label text, or select a background pixmap.

#### Enumerator:

**UPDATE\_TEXT** Update the label text.

**UPDATE\_PIXMAP** Update the label background pixmap.

### 9.67.2.6 enum QELabel::UserLevels

User friendly enumerations for [userLevelVisibility](#) and [userLevelEnabled](#) properties - refer to [userLevelVisibility](#) and [userLevelEnabled](#) properties and userLevel enumeration for details.

#### Enumerator:

**User** Refer to USERLEVEL\_USER for details.

**Scientist** Refer to USERLEVEL\_SCIENTIST for details.

**Engineer** Refer to USERLEVEL\_ENGINEER for details.

## 9.67.3 Constructor & Destructor Documentation

### 9.67.3.1 QELabel::QELabel ( QWidget \* *parent* = 0 )

Create without a variable. Use `setVariableNameProperty()` and `setSubstitutionsProperty()` to define a variable and, optionally, macro substitutions later.

### 9.67.3.2 QELabel::QELabel ( const QString & *variableName*, QWidget \* *parent* = 0 )

Create with a variable. A connection is automatically established. If macro substitutions are required, create without a variable and set the variable and macro substitutions after creation.

## 9.67.4 Member Function Documentation

### 9.67.4.1 void QELabel::dbValueChanged ( const QString & *out* ) [signal]

Sent when the widget is updated following a data change Can be used to pass on EPICS data (as presented in this widget) to other widgets. For example a QList widget could log updates from this widget.

#### 9.67.4.2 void QELabel::requestEnabled ( const bool & state ) [inline, slot]

Similar to standard setEnabled slot, but allows QE widget to determine if the widget remains disabled due to invalid data. If disabled due to invalid data, a request to enable the widget will be honoured when the data is no longer invalid.

### 9.67.5 Property Documentation

#### 9.67.5.1 bool QELabel::addUnits [read, write]

If true (default), add engineering units supplied with the data.

#### 9.67.5.2 bool QELabel::allowDrop [read, write]

Allow drag/drops operations to this widget. Default is false. Any dropped text will be used as a new variable name.

Reimplemented from [QEDragDrop](#).

#### 9.67.5.3 ArrayActions QELabel::arrayAction [read, write]

Text formatting option for array data. Default is ASCII. Options are:

- ASCII - treat array as a single text string. For example an array of three characters 'a' 'b' 'c' will be formatted as 'abc'.
- APPEND - treat array as an array of numbers and format a string containing them all with a space between each. For example, an array of three numbers 10, 11 and 12 will be formatted as '10 11 12'.
- INDEX - Extract a single item from the array. The item is then formatted as any other non array data would be. The item selected is determined by the arrayIndex property. For example, if arrayIndex property is 1, an array of three numbers 10, 11 and 12 will be formatted as '11'.

#### 9.67.5.4 bool QELabel::enabled [read, write]

Set the preferred 'enabled' state. Default is true. This property is copied to the standard Qt 'enabled' property if the data being displayed is valid. If the data being displayed is invalid the standard Qt 'enabled' property will always be set to false to indicate invalid data. The value of this property will only be copied to the standard Qt 'enabled' property once data is valid.

**9.67.5.5 Formats QELabel::format** [read, write]

Format to apply to data. Default is 'Default' in which case the data type supplied with the data determines how the data is formatted. For all other options, an attempt is made to format the data as requested (whatever its native form).

**9.67.5.6 unsigned QELabel::int** [read, write]

Set the ID used by the message filtering system. Default is zero. Widgets or applications that use messages from the framework have the option of filtering on this ID. For example, by using a unique message source ID a [QELog](#) widget may be set up to only log messages from a select set of widgets.

Base used for when formatting integers. Default is 10 (duh!)

Index used to select a single item of data for formatting from an array of data. Default is 0. Only used when the arrayAction property is INDEX. Refer to the arrayAction property for more details.

**9.67.5.7 bool QELabel::leadingZero** [read, write]

If true (default), always add a leading zero when formatting numbers.

**9.67.5.8 QString QELabel::localEnumeration** [read, write]

An enumeration list used to data values. Used only when the formatting option is 'local enumeration'. Value is converted to an integer and used to select a string from this list.

Format is:

```
[[<|<=|!=|>|=|>]value1|*]: string1 , [[<|<=|!=|>|=|>]value2|*]: string2 , [[<|<=|!=|>|=|>]value3|*]: string3 , ...
```

Where: < Less than <= Less than or equal = Equal (default if no operator specified)  
>= Greather than or equal > Greater than Always match (used to specify default text)

Values may be numeric or textual Values do not have to be in any order, but first match wins Values may be quoted Strings may be quoted Consecutive values do not have to be present. Operator is assumed to be equality if not present. White space is ignored except within quoted strings.

may be included in a string to indicate a line break

Examples are:

```
0:Off,1:On 0 : "Pump Running", 1 : "Pump not running" 0:"", 1:"Warning!\nAlarm"
<2:"Value is less than two", =2:"Value is equal to two", >2:"Value is grater than 2"
3:"Beamline Available", *:"" "Pump Off": "OH NO!, the pump is OFF!","Pump On": "It's OK, the pump is on"
```

The data value is converted to a string if no enumeration for that value is available. For example, if the local enumeration is '0:off,1:on', and a value of 10 is processed, the

text generated is '10'. If a blank string is required, this should be explicit. for example, '0:off,1:on,10:'''

A range of numbers can be covered by a pair of values as in the following example:  
>=4:"Between 4 and 8",<=8:"Between 4 and 8"

#### 9.67.5.9 Notations QELabel::notation [read, write]

Notation used for numerical formatting. Default is fixed.

#### 9.67.5.10 QPixmap QELabel::pixmap0 [read, write]

Pixmap displayed when updateOption property is 'Picture' and data is interpreted as 0.

#### 9.67.5.11 QPixmap QELabel::pixmap1 [read, write]

Pixmap displayed when updateOption property is 'Picture' and data is interpreted as 1.

#### 9.67.5.12 QPixmap QELabel::pixmap2 [read, write]

Pixmap displayed when updateOption property is 'Picture' and data is interpreted as 2.

#### 9.67.5.13 QPixmap QELabel::pixmap3 [read, write]

Pixmap displayed when updateOption property is 'Picture' and data is interpreted as 3.

#### 9.67.5.14 QPixmap QELabel::pixmap4 [read, write]

Pixmap displayed when updateOption property is 'Picture' and data is interpreted as 4.

#### 9.67.5.15 QPixmap QELabel::pixmap5 [read, write]

Pixmap displayed when updateOption property is 'Picture' and data is interpreted as 5.

#### 9.67.5.16 QPixmap QELabel::pixmap6 [read, write]

Pixmap displayed when updateOption property is 'Picture' and data is interpreted as 6.



**9.67.5.17 QPixmap QELabel::pixmap7** [read, write]

Pixmap displayed when updateOption property is 'Picture' and data is interpreted as 7.

**9.67.5.18 int QELabel::precision** [read, write]

Precision used when formatting floating point numbers. The default is 4. This is only used if useDbPrecision is false.

**9.67.5.19 bool QELabel::trailingZeros** [read, write]

If true (default), always remove any trailing zeros when formatting numbers.

**9.67.5.20 UpdateOptions QELabel::updateOption** [read, write]

Determines if data updates the label text, or the label pixmap. For both options all normal string formatting is applied. If Text, the formatted text is simply presented as the label text. If Picture, the FORMATTED text is then interpreted as an integer and used to select one of the pixmaps specified by properties pixmap0 through to pixmap7.

**9.67.5.21 bool QELabel::useDbPrecision** [read, write]

If true (default), format floating point numbers using the precision supplied with the data. If false, the precision property is used.

**9.67.5.22 UserLevels QELabel::userLevelEnabled** [read, write]

Lowest user level at which the widget is enabled. Default is 'User'. Used when designing GUIs that allow access to more and more detail according to the user mode. The user mode is set application wide through the [QELogin](#) widget, or programatically through setUserLevel() Widgets that are always accessible should be visible at 'User'. Widgets that are only accessible to scientists managing the facility should be visible at 'Scientist'. Widgets that are only accessible to engineers maintaining the facility should be visible at 'Engineer'.

**9.67.5.23 QString QELabel::userLevelEngineerStyle** [read, write]

Style Sheet string to be applied when the widget is displayed in 'Engineer' mode. Default is an empty string. The syntax is the standard Qt Style Sheet syntax. For example, 'background-color: red' This Style Sheet string will be applied by the styleManager class. Refer to the styleManager class for details about how this Style Sheet string will be merged with any pre-existing Style Sheet string and any Style Sheet strings generated during the display of data.

#### 9.67.5.24 `QString QELabel::userLevelScientistStyle` [read, write]

Style Sheet string to be applied when the widget is displayed in 'Scientist' mode. Default is an empty string. The syntax is the standard Qt Style Sheet syntax. For example, 'background-color: red' This Style Sheet string will be applied by the styleManager class. Refer to the styleManager class for details about how this Style Sheet string will be merged with any pre-existing Style Sheet string and any Style Sheet strings generated during the display of data.

#### 9.67.5.25 `QString QELabel::userLevelUserStyle` [read, write]

Style Sheet string to be applied when the widget is displayed in 'User' mode. Default is an empty string. The syntax is the standard Qt Style Sheet syntax. For example, 'background-color: red' This Style Sheet string will be applied by the styleManager class. Refer to the styleManager class for details about how this Style Sheet string will be merged with any pre-existing Style Sheet string and any Style Sheet strings generated during the display of data.

#### 9.67.5.26 `UserLevels QELabel::userLevelVisibility` [read, write]

Lowest user level at which the widget is visible. Default is 'User'. Used when designing GUIs that display more and more detail according to the user mode. The user mode is set application wide through the [QELogin](#) widget, or programatically through setUserLevel() Widgets that are always visible should be visible at 'User'. Widgets that are only used by scientists managing the facility should be visible at 'Scientist'. Widgets that are only used by engineers maintaining the facility should be visible at 'Engineer'.

#### 9.67.5.27 `QString QELabel::variable` [read, write]

EPICS variable name (CA PV)

#### 9.67.5.28 `bool QELabel::variableAsToolTip` [read, write]

Use the variable as the tool tip. Default is true. Tool tip property will be overwritten by the variable name.

Reimplemented from [QEToolTip](#).

#### 9.67.5.29 `QString QELabel::variableSubstitutions` [read, write]

Macro substitutions. The default is no substitutions. The format is NAME1=VALUE1[, NAME2=VALUE2... Values may be quoted strings. For example, 'PUMP=PMP3, NAME = "My Pump"' These substitutions are applied to variable names for all QE widgets. In some widgets are also used for other purposes.

### 9.67.5.30 bool QELabel::visible [read, write]

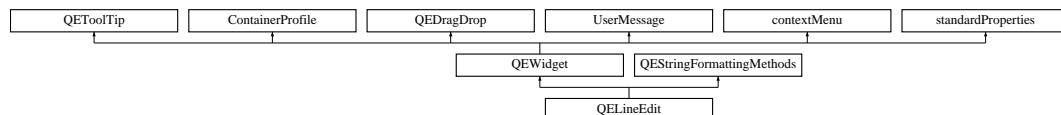
Display the widget. Default is true. Setting this property false is usefull if widget is only used to provide a signal - for example, when supplying data to a [QELink](#) widget. Note, when false the widget will still be visible in Qt Designer.

The documentation for this class was generated from the following files:

- /home/rhydera/epicsqt/framework/widgets/QELabel/QELabel.h
- /home/rhydera/epicsqt/framework/widgets/QELabel/QELabel.cpp

## 9.68 QELineEdit Class Reference

Inheritance diagram for QELineEdit:



### Public Types

- enum [UserLevels](#) { [User](#) = USERLEVEL\_USER, [Scientist](#) = USERLEVEL\_SCIENTIST, [Engineer](#) = USERLEVEL\_ENGINEER }
- enum [Formats](#) {  
[Default](#) = QEStrFormatting::FORMAT\_DEFAULT, [Floating](#) = QEStrFormatting::FORMAT\_FLOATING, [Integer](#) = QEStrFormatting::FORMAT\_INTEGER, [UnsignedInteger](#) = QEStrFormatting::FORMAT\_UNSIGNEDINTEGER,  
[Time](#) = QEStrFormatting::FORMAT\_TIME, [LocalEnumeration](#) = QEStrFormatting::FORMAT\_LOCAL\_ENUMERATE }
- enum [Notations](#) { [Fixed](#) = QEStrFormatting::NOTATION\_FIXED, [Scientific](#) = QEStrFormatting::NOTATION\_SCIENTIFIC, [Automatic](#) = QEStrFormatting::NOTATION\_AUTOMATIC }
- enum [ArrayActions](#) { [Append](#) = QEStrFormatting::APPEND, [Ascii](#) = QEStrFormatting::ASCII, [Index](#) = QEStrFormatting::INDEX }

### Public Slots

- void [requestEnabled](#) (const bool &state)

### Signals

- void [dbValueChanged](#) (const QString &out)

- void [userChange](#) (const QString &oldValue, const QString &newValue, const QString &lastValue)  
*Internal use only. Used by [QEConfiguredLayout](#) to be notified when one of its widgets has written something.*
- void [requestResend](#) ()  
*Internal use only. Used when changing a property value to force a re-display to reflect the new property value.*

## Public Member Functions

- bool [isEnabled](#) () const  
*Access function for [enabled](#) property - refer to [enabled](#) property for details.*
- void [setEnabled](#) (bool state)  
*Access function for [enabled](#) property - refer to [enabled](#) property for details.*
- [UserLevels](#) [getUserLevelVisibilityProperty](#) ()  
*Access function for [userLevelVisibility](#) property - refer to [userLevelVisibility](#) property for details.*
- void [setUserLevelVisibilityProperty](#) ([UserLevels](#) level)  
*Access function for [userLevelVisibility](#) property - refer to [userLevelVisibility](#) property for details.*
- [UserLevels](#) [getUserLevelEnabledProperty](#) ()  
*Access function for [userLevelEnabled](#) property - refer to [userLevelEnabled](#) property for details.*
- void [setUserLevelEnabledProperty](#) ([UserLevels](#) level)  
*Access function for [userLevelEnabled](#) property - refer to [userLevelEnabled](#) property for details.*
- void [setFormatProperty](#) ([Formats](#) format)  
*Access function for [format](#) property - refer to [format](#) property for details.*
- [Formats](#) [getFormatProperty](#) ()  
*Access function for [format](#) property - refer to [format](#) property for details.*
- void [setNotationProperty](#) ([Notations](#) notation)  
*Access function for [notation](#) property - refer to [notation](#) property for details.*
- [Notations](#) [getNotationProperty](#) ()  
*Access function for [notation](#) property - refer to [notation](#) property for details.*
- void [setArrayActionProperty](#) ([ArrayActions](#) arrayAction)  
*Access function for [arrayAction](#) property - refer to [arrayAction](#) property for details.*
- [ArrayActions](#) [getArrayActionProperty](#) ()  
*Access function for [arrayAction](#) property - refer to [arrayAction](#) property for details.*
- [QELineEdit](#) (QWidget \*parent=0)
- [QELineEdit](#) (const QString &variableName, QWidget \*parent=0)
- void [setWriteOnLoseFocus](#) (bool writeOnLoseFocus)
- bool [getWriteOnLoseFocus](#) ()
- void [setWriteOnEnter](#) (bool writeOnEnter)
- bool [getWriteOnEnter](#) ()
- void [setWriteOnFinish](#) (bool writeOnFinish)

- bool [getWriteOnFinish](#) ()
- void [setConfirmWrite](#) (bool confirmWrite)
- bool [getConfirmWrite](#) ()
- void [setSubscribe](#) (bool subscribe)
- bool [getSubscribe](#) ()

## Properties

- QString [variable](#)
- QString [variableSubstitutions](#)
- bool [subscribe](#)
- bool [writeOnLoseFocus](#)
- bool [writeOnEnter](#)
- bool [writeOnFinish](#)
- bool [confirmWrite](#)
- bool [variableAsToolTip](#)
- bool [enabled](#)
- bool [allowDrop](#)
- bool [visible](#)
- unsigned int
- QString [userLevelUserStyle](#)
- QString [userLevelScientistStyle](#)
- QString [userLevelEngineerStyle](#)
- [UserLevels](#) [userLevelVisibility](#)
- [UserLevels](#) [userLevelEnabled](#)
- int [precision](#)
- bool [useDbPrecision](#)
- bool [leadingZero](#)
- bool [trailingZeros](#)
- bool [addUnits](#)
- QString [localEnumeration](#)
- [Formats](#) [format](#)
- [Notations](#) [notation](#)
- [ArrayActions](#) [arrayAction](#)

### 9.68.1 Member Enumeration Documentation

#### 9.68.1.1 enum QELineEdit::ArrayActions

User friendly enumerations for arrayAction property - refer to [QStringFormatting::arrayActions](#) for details.

#### Enumerator:

**Append** Refer to [QStringFormatting::APPEND](#) for details.

**Ascii** Refer to [QStringFormatting::ASCII](#) for details.

**Index** Refer to [QStringFormatting::INDEX](#) for details.

### 9.68.1.2 enum `QLineEdit::Formats`

User friendly enumerations for format property - refer to [QStringFormatting::formats](#) for details.

#### Enumerator:

- Default** Refer to [QStringFormatting::FORMAT\\_DEFAULT](#) for details.
- Floating** Refer to [QStringFormatting::FORMAT\\_FLOATING](#) for details.
- Integer** Refer to [QStringFormatting::FORMAT\\_INTEGER](#) for details.
- UnsignedInteger** Refer to [QStringFormatting::FORMAT\\_UNSIGNEDINTEGER](#) for details.
- Time** Refer to [QStringFormatting::FORMAT\\_TIME](#) for details.
- LocalEnumeration** Refer to [QStringFormatting::FORMAT\\_LOCAL\\_ENUMERATE](#) for details (and the [localEnumeration](#) property)

### 9.68.1.3 enum `QLineEdit::Notations`

User friendly enumerations for notation property - refer to [QStringFormatting::notations](#) for details.

#### Enumerator:

- Fixed** Refer to [QStringFormatting::NOTATION\\_FIXED](#) for details.
- Scientific** Refer to [QStringFormatting::NOTATION\\_SCIENTIFIC](#) for details.
- Automatic** Refer to [QStringFormatting::NOTATION\\_AUTOMATIC](#) for details.

### 9.68.1.4 enum `QLineEdit::UserLevels`

User friendly enumerations for [userLevelVisibility](#) and [userLevelEnabled](#) properties - refer to [userLevelVisibility](#) and [userLevelEnabled](#) properties and `::userLevel` enumeration for details.

#### Enumerator:

- User** Refer to `::USERLEVEL_USER` for details.
- Scientist** Refer to `::USERLEVEL_SCIENTIST` for details.
- Engineer** Refer to `::USERLEVEL_ENGINEER` for details.

## 9.68.2 Constructor & Destructor Documentation

### 9.68.2.1 `QLineEdit::QLineEdit ( QWidget * parent = 0 )`

Create without a variable. Use `setVariableNameProperty()` and `setSubstitutionsProperty()` to define a variable and, optionally, macro substitutions later.

### 9.68.2.2 QELineEdit::QELineEdit ( const QString & *variableName*, QWidget \* *parent* = 0 )

Create with a variable. A connection is automatically established. If macro substitutions are required, create without a variable and set the variable and macro substitutions after creation.

## 9.68.3 Member Function Documentation

### 9.68.3.1 void QELineEdit::dbValueChanged ( const QString & *out* ) [signal]

Sent when the widget is updated following a data change Can be used to pass on EPICS data (as presented in this widget) to other widgets. For example a QList widget could log updates from this widget.

### 9.68.3.2 bool QELineEdit::getConfirmWrite ( )

Returns 'true' if this widget will ask for confirmation (using a dialog box) prior to writing data.

### 9.68.3.3 bool QELineEdit::getSubscribe ( )

Returns 'true' if this widget subscribes for data updates and displays current data.

### 9.68.3.4 bool QELineEdit::getWriteOnEnter ( )

Returns 'true' if this widget writes any changes when the user presses 'enter'.

### 9.68.3.5 bool QELineEdit::getWriteOnFinish ( )

Returns 'true' if this widget writes any changes when the user finished editing (the QLineEdit 'editingFinished' signal is emitted).

### 9.68.3.6 bool QELineEdit::getWriteOnLoseFocus ( )

Returns 'true' if this widget automatically writes any changes when it loses focus.

### 9.68.3.7 void QELineEdit::requestEnabled ( const bool & *state* ) [inline, slot]

Similar to standard setEnabled slot, but allows QE widget to determine if the widget remains disabled due to invalid data. If disabled due to invalid data, a request to enable the widget will be honoured when the data is no longer invalid.

#### 9.68.3.8 void QLEdit::setConfirmWrite ( bool *confirmWrite* )

Sets if this widget will ask for confirmation (using a dialog box) prior to writing data. Default is 'false' (will not ask for confirmation (using a dialog box) prior to writing data).

#### 9.68.3.9 void QLEdit::setSubscribe ( bool *subscribe* )

Sets if this widget subscribes for data updates and displays current data. Default is 'true' (subscribes for and displays data updates)

#### 9.68.3.10 void QLEdit::setWriteOnEnter ( bool *writeOnEnter* )

Sets if this widget writes any changes when the user presses 'enter'. Note, the current value will be written even if the user has not changed it. Default is 'true' (writes any changes when the user presses 'enter').

#### 9.68.3.11 void QLEdit::setWriteOnFinish ( bool *writeOnFinish* )

Sets if this widget writes any changes when the user finished editing (the QLEdit 'editingFinished' signal is emitted). No writing occurs if no changes were made. Default is 'true' (writes any changes when the QLEdit 'editingFinished' signal is emitted).

#### 9.68.3.12 void QLEdit::setWriteOnLoseFocus ( bool *writeOnLoseFocus* )

Sets if this widget automatically writes any changes when it loses focus. Default is 'false' (does not write any changes when it loses focus).

### 9.68.4 Property Documentation

#### 9.68.4.1 bool QLEdit::addUnits [read, write]

If true (default), add engineering units supplied with the data.

#### 9.68.4.2 bool QLEdit::allowDrop [read, write]

Allow drag/drops operations to this widget. Default is false. Any dropped text will be used as a new variable name.

Reimplemented from [QEDragDrop](#).

#### 9.68.4.3 ArrayActions QLEdit::arrayAction [read, write]

Text formatting option for array data. Default is ASCII. Options are:



- ASCII - treat array as a single text string. For example an array of three characters 'a' 'b' 'c' will be formatted as 'abc'.
- APPEND - treat array as an array of numbers and format a string containing them all with a space between each. For example, an array of three numbers 10, 11 and 12 will be formatted as '10 11 12'.
- INDEX - Extract a single item from the array. The item is then formatted as any other non array data would be. The item selected is determined by the arrayIndex property. For example, if arrayIndex property is 1, an array of three numbers 10, 11 and 12 will be formatted as '11'.

#### 9.68.4.4 bool QLELineEdit::confirmWrite [read, write]

Sets if this widget will ask for confirmation (using a dialog box) prior to writing data. Default is 'false' (will not ask for confirmation (using a dialog box) prior to writing data).

#### 9.68.4.5 bool QLELineEdit::enabled [read, write]

Set the preferred 'enabled' state. Default is true. This property is copied to the standard Qt 'enabled' property if the data being displayed is valid. If the data being displayed is invalid the standard Qt 'enabled' property will always be set to false to indicate invalid data. The value of this property will only be copied to the standard Qt 'enabled' property once data is valid.

#### 9.68.4.6 Formats QLELineEdit::format [read, write]

Format to apply to data. Default is 'Default' in which case the data type supplied with the data determines how the data is formatted. For all other options, an attempt is made to format the data as requested (whatever its native form).

#### 9.68.4.7 unsigned QLELineEdit::int [read, write]

Set the ID used by the message filtering system. Default is zero. Widgets or applications that use messages from the framework have the option of filtering on this ID. For example, by using a unique message source ID a [QELog](#) widget may be set up to only log messages from a select set of widgets.

Base used for when formatting integers. Default is 10 (duh!)

Index used to select a single item of data for formatting from an array of data. Default is 0. Only used when the arrayAction property is INDEX. Refer to the arrayAction property for more details.

#### 9.68.4.8 bool QLELineEdit::leadingZero [read, write]

If true (default), always add a leading zero when formatting numbers.

#### 9.68.4.9 QString QLEdit::localEnumeration [read, write]

An enumeration list used to data values. Used only when the formatting option is 'local enumeration'. Value is converted to an integer and used to select a string from this list.

Format is:

```
[[<|<=|=|!=|>|=|>]value1|*]: string1 , [[<|<=|=|!=|>|=|>]value2|*]: string2 , [[<|<=|=|!=|>|=|>]value3|*]: string3 , ...
```

Where: < Less than <= Less than or equal = Equal (default if no operator specified)

>= Greather than or equal > Greater than Always match (used to specify default text)

Values may be numeric or textual Values do not have to be in any order, but first match wins Values may be quoted Strings may be quoted Consecutive values do not have to be present. Operator is assumed to be equality if not present. White space is ignored except within quoted strings.

may be included in a string to indicate a line break

Examples are:

```
0:Off,1:On 0 : "Pump Running", 1 : "Pump not running" 0:"", 1:"Warning!\nAlarm"
<2:"Value is less than two", =2:"Value is equal to two", >2:"Value is grater than 2"
3:"Beamline Available", *:"" "Pump Off": "OH NO!, the pump is OFF!", "Pump On": "It's OK, the pump is on"
```

The data value is converted to a string if no enumeration for that value is available. For example, if the local enumeration is '0:off,1:on', and a value of 10 is processed, the text generated is '10'. If a blank string is required, this should be explicit. for example, '0:off,1:on,10:'''

A range of numbers can be covered by a pair of values as in the following example:  
>=4:"Between 4 and 8", <=8:"Between 4 and 8"

#### 9.68.4.10 Notations QLEdit::notation [read, write]

Notation used for numerical formatting. Default is fixed.

#### 9.68.4.11 int QLEdit::precision [read, write]

Precision used when formatting floating point numbers. The default is 4. This is only used if useDbPrecision is false.

#### 9.68.4.12 bool QLEdit::subscribe [read, write]

Sets if this widget subscribes for data updates and displays current data. Default is 'true' (subscribes for and displays data updates)

Reimplemented from [QEWidget](#).

**9.68.4.13** `bool QLEdit::trailingZeros` [read, write]

If true (default), always remove any trailing zeros when formatting numbers.

**9.68.4.14** `bool QLEdit::useDbPrecision` [read, write]

If true (default), format floating point numbers using the precision supplied with the data. If false, the precision property is used.

**9.68.4.15** `UserLevels QLEdit::userLevelEnabled` [read, write]

Lowest user level at which the widget is enabled. Default is 'User'. Used when designing GUIs that allow access to more and more detail according to the user mode. The user mode is set application wide through the [QELogin](#) widget, or programatically through `setUserLevel()`. Widgets that are always accessible should be visible at 'User'. Widgets that are only accessible to scientists managing the facility should be visible at 'Scientist'. Widgets that are only accessible to engineers maintaining the facility should be visible at 'Engineer'.

**9.68.4.16** `QString QLEdit::userLevelEngineerStyle` [read, write]

Style Sheet string to be applied when the widget is displayed in 'Engineer' mode. Default is an empty string. The syntax is the standard Qt Style Sheet syntax. For example, 'background-color: red'. This Style Sheet string will be applied by the `styleManager` class. Refer to the `styleManager` class for details about how this Style Sheet string will be merged with any pre-existing Style Sheet string and any Style Sheet strings generated during the display of data.

**9.68.4.17** `QString QLEdit::userLevelScientistStyle` [read, write]

Style Sheet string to be applied when the widget is displayed in 'Scientist' mode. Default is an empty string. The syntax is the standard Qt Style Sheet syntax. For example, 'background-color: red'. This Style Sheet string will be applied by the `styleManager` class. Refer to the `styleManager` class for details about how this Style Sheet string will be merged with any pre-existing Style Sheet string and any Style Sheet strings generated during the display of data.

**9.68.4.18** `QString QLEdit::userLevelUserStyle` [read, write]

Style Sheet string to be applied when the widget is displayed in 'User' mode. Default is an empty string. The syntax is the standard Qt Style Sheet syntax. For example, 'background-color: red'. This Style Sheet string will be applied by the `styleManager` class. Refer to the `styleManager` class for details about how this Style Sheet string will be merged with any pre-existing Style Sheet string and any Style Sheet strings generated during the display of data.

#### 9.68.4.19 **UserLevels QLineEdit::userLevelVisibility** [read, write]

Lowest user level at which the widget is visible. Default is 'User'. Used when designing GUIs that display more and more detail according to the user mode. The user mode is set application wide through the [QELogin](#) widget, or programatically through `setUserLevel()`. Widgets that are always visible should be visible at 'User'. Widgets that are only used by scientists managing the facility should be visible at 'Scientist'. Widgets that are only used by engineers maintaining the facility should be visible at 'Engineer'.

#### 9.68.4.20 **QString QLineEdit::variable** [read, write]

EPICS variable name (CA PV)

#### 9.68.4.21 **bool QLineEdit::variableAsToolTip** [read, write]

Use the variable as the tool tip. Default is true. Tool tip property will be overwritten by the variable name.

Reimplemented from [QEToolTip](#).

#### 9.68.4.22 **QString QLineEdit::variableSubstitutions** [read, write]

Macro substitutions. The default is no substitutions. The format is NAME1=VALUE1[, NAME2=VALUE2... Values may be quoted strings. For example, 'PUMP=PMP3, NAME = "My Pump"' These substitutions are applied to variable names for all QE widgets. In some widgets are are also used for other purposes.

#### 9.68.4.23 **bool QLineEdit::visible** [read, write]

Display the widget. Default is true. Setting this property false is usefull if widget is only used to provide a signal - for example, when supplying data to a [QELink](#) widget. Note, when false the widget will still be visible in Qt Designer.

#### 9.68.4.24 **bool QLineEdit::writeOnEnter** [read, write]

Sets if this widget writes any changes when the user presses 'enter'. Note, the current value will be written even if the user has not changed it. Default is 'true' (writes any changes when the user presses 'enter').

#### 9.68.4.25 **bool QLineEdit::writeOnFinish** [read, write]

Sets if this widget writes any changes when the user finished editing (the `QLineEdit::editingFinished` signal is emitted). No writing occurs if no changes were made. Default is 'true' (writes any changes when the `QLineEdit::editingFinished` signal is emitted).

9.68.4.26 `bool QELineEdit::writeOnLoseFocus [read, write]`

Sets if this widget automatically writes any changes when it loses focus. Default is 'false' (does not write any changes when it loses focus).

The documentation for this class was generated from the following files:

- /home/rhydera/epicsqt/framework/widgets/QELineEdit/QELineEdit.h
- /home/rhydera/epicsqt/framework/widgets/QELineEdit/QELineEdit.cpp

## 9.69 QELineEditManager Class Reference

### Public Member Functions

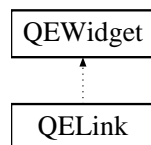
- **QELineEditManager** (QObject \*parent=0)
- `bool isContainer () const`
- `bool isInitialized () const`
- `QIcon icon () const`
- `QString group () const`
- `QString includeFile () const`
- `QString name () const`
- `QString toolTip () const`
- `QString whatsThis () const`
- `QWidget * createWidget (QWidget *parent)`
- `void initialize (QDesignerFormEditorInterface *core)`

The documentation for this class was generated from the following file:

- /home/rhydera/epicsqt/framework/widgets/QELineEdit/QELineEditManager.h

## 9.70 QELink Class Reference

Inheritance diagram for QELink:



### Public Types

- `enum conditions {`  
`CONDITION_EQ, CONDITION_NE, CONDITION_GT, CONDITION_GE,`  
`CONDITION_LT, CONDITION_LE }`

- enum **ConditionNames** {  
**Equal** = QELink::CONDITION\_EQ, **NotEqual** = QELink::CONDITION\_NE, **GreaterThan**  
= QELink::CONDITION\_GT, **GreaterThanOrEqual** = QELink::CONDITION\_GE,  
**LessThan** = QELink::CONDITION\_LT, **LessThanOrEqual** = QELink::CONDITION\_-  
LE }

### Public Slots

- void **in** (const bool &in)
- void **in** (const qlonglong &in)
- void **in** (const double &in)
- void **in** (const QString &in)
- void **autoFillBackground** (const bool &enable)

### Signals

- void **out** (const bool &out)
- void **out** (const qlonglong &out)
- void **out** (const double &out)
- void **out** (const QString &out)

### Public Member Functions

- **QELink** (QWidget \*parent=0)
- void **setCondition** (conditions conditionIn)
- conditions **getCondition** ()
- void **setComparisonValue** (QString comparisonValue)
- QString **getComparisonValue** ()
- void **setSignalTrue** (bool signalTrue)
- bool **getSignalTrue** ()
- void **setSignalFalse** (bool signalFalse)
- bool **getSignalFalse** ()
- void **setOutTrueValue** (QString outTrueValue)
- QString **getOutTrueValue** ()
- void **setOutFalseValue** (QString outFalseValue)
- QString **getOutFalseValue** ()
- void **setRunVisible** (bool visibleIn)
- bool **getRunVisible** ()
- void **setConditionProperty** (ConditionNames condition)
- ConditionNames **getConditionProperty** ()

### Protected Attributes

- conditions **condition**
- QVariant **comparisonValue**
- bool **signalTrue**
- bool **signalFalse**
- QVariant **outTrueValue**
- QVariant **outFalseValue**
- bool **visible**

### Properties

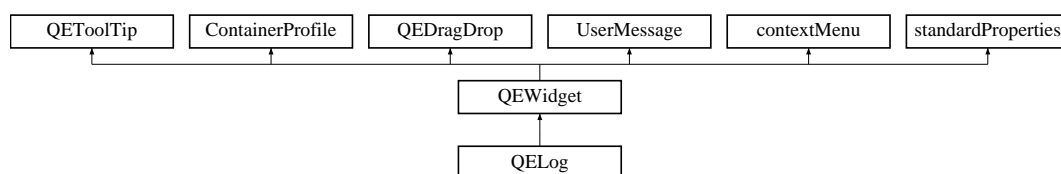
- ConditionNames **condition**
- QString **comparisonValue**
- QString **outTrueValue**
- QString **outFalseValue**
- bool **runVisible**

The documentation for this class was generated from the following files:

- /home/rhydera/epicsqt/framework/widgets/QELink/QELink.h
- /home/rhydera/epicsqt/framework/widgets/QELink/QELink.cpp

## 9.71 QELog Class Reference

Inheritance diagram for QELog:



### Public Types

- enum **detailsLayoutProperty** { **Top** = TOP, **Bottom** = BOTTOM, **Left** = LEFT, **Right** = RIGHT }
- enum **MessageFilterOptions** { **Any** = UserMessage::MESSAGE\_FILTER\_ANY, **Match** = UserMessage::MESSAGE\_FILTER\_MATCH, **None** = UserMessage::MESSAGE\_FILTER\_NONE }

## Public Member Functions

- **QELog** (QWidget \*pParent=0)
- void **setShowColumnTime** (bool pValue)
- bool **getShowColumnTime** ()
- void **setShowColumnType** (bool pValue)
- bool **getShowColumnType** ()
- void **setShowColumnMessage** (bool pValue)
- bool **getShowColumnMessage** ()
- void **setShowMessageFilter** (bool pValue)
- bool **getShowMessageFilter** ()
- void **setShowClear** (bool pValue)
- bool **getShowClear** ()
- void **setShowSave** (bool pValue)
- bool **getShowSave** ()
- void **setDetailsLayout** (int pValue)
- int **getDetailsLayout** ()
- void **setScrollToBottom** (bool pValue)
- bool **getScrollToBottom** ()
- void **setInfoColor** (QColor pValue)
- QColor **getInfoColor** ()
- void **setWarningColor** (QColor pValue)
- QColor **getWarningColor** ()
- void **setErrorColor** (QColor pValue)
- QColor **getErrorColor** ()
- void **clearLog** ()
- void **addLog** (int pType, QString pMessage)
- void **refreshLog** ()
- void **setDetailsLayoutProperty** (detailsLayoutProperty pDetailsLayout)
- detailsLayoutProperty **getDetailsLayoutProperty** ()
- MessageFilterOptions **getMessageFormFilter** ()
- void **setMessageFormFilter** (MessageFilterOptions messageFormFilter)
- MessageFilterOptions **getMessageSourceFilter** ()
- void **setMessageSourceFilter** (MessageFilterOptions messageSourceFilter)

## Protected Attributes

- [\\_QTableWidgetLog](#) \* **qTableWidgetLog**
- QCheckBox \* **qCheckBoxInfoMessage**
- QCheckBox \* **qCheckBoxWarningMessage**
- QCheckBox \* **qCheckBoxErrorMessage**
- QPushButton \* **qPushButtonClear**
- QPushButton \* **qPushButtonSave**
- QColor **qColorInfo**
- QColor **qColorWarning**
- QColor **qColorError**
- bool **scrollToBottom**
- int **detailsLayout**



## Properties

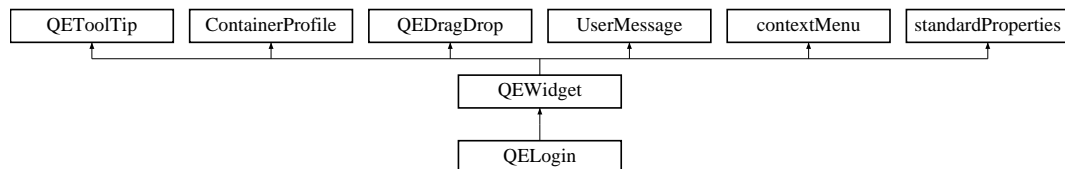
- bool **showColumnTime**
- bool **showColumnType**
- bool **showColumnMessage**
- bool **showMessageFilter**
- bool **showClear**
- bool **showSave**
- detailsLayoutProperty **detailsLayout**
- QColor **infoColor**
- QColor **warningColor**
- QColor **errorColor**
- MessageFilterOptions **messageFormFilter**
- MessageFilterOptions **messageSourceFilter**
- unsigned int

The documentation for this class was generated from the following files:

- /home/rhydera/epicsqt/framework/widgets/QELog/QELog.h
- /home/rhydera/epicsqt/framework/widgets/QELog/QELog.cpp

## 9.72 QELogin Class Reference

Inheritance diagram for QELogin:



## Public Types

- enum **userTypesProperty** { **User** = USERLEVEL\_USER, **Scientist** = USERLEVEL\_SCIENTIST, **Engineer** = USERLEVEL\_ENGINEER }
- enum **detailsLayoutProperty** { **Top** = TOP, **Bottom** = BOTTOM, **Left** = LEFT, **Right** = RIGHT }

## Public Member Functions

- **QELogin** (QWidget \*pParent=0)
- void **setShowUserType** (bool pValue)
- bool **getShowUserType** ()

- void **setShowLogin** (bool pValue)
- bool **getShowButtonLogin** ()
- void **setShowLogout** (bool pValue)
- bool **getShowButtonLogout** ()
- void **setUserPassword** (QString pValue)
- QString **getUserPassword** ()
- void **setScientistPassword** (QString pValue)
- QString **getScientistPassword** ()
- void **setEngineerPassword** (QString pValue)
- QString **getEngineerPassword** ()
- void **setCurrentUserType** (int pValue)
- int **getCurrentUserType** ()
- void **setDetailsLayout** (int pValue)
- int **getDetailsLayout** ()
- QString **getUserTypeName** (userLevels type)
- void **logoutCurrentUserType** ()
- void **setCurrentUserTypeProperty** (userTypesProperty pUserType)
- userTypesProperty **getCurrentUserTypeProperty** ()
- void **setDetailsLayoutProperty** (detailsLayoutProperty pDetailsLayout)
- detailsLayoutProperty **getDetailsLayoutProperty** ()

### Protected Attributes

- QStack< int > **loginHistory**
- QPushButton \* **qPushButtonLogin**
- QPushButton \* **qPushButtonLogout**
- QLabel \* **qLabelUserType**
- QString **userPassword**
- QString **scientistPassword**
- QString **engineerPassword**
- int **currentUserType**
- int **detailsLayout**

### Properties

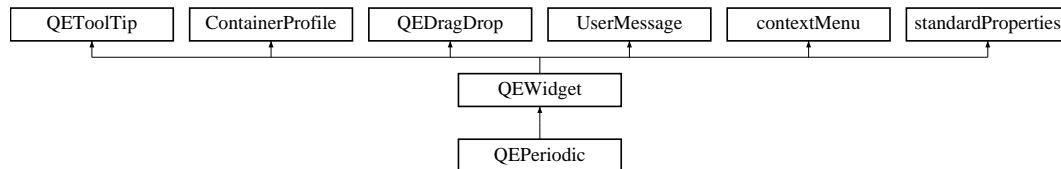
- bool **showUserType**
- bool **showLogin**
- bool **showLogout**
- userTypesProperty **currentUserType**
- detailsLayoutProperty **detailsLayout**

The documentation for this class was generated from the following files:

- /home/rhydera/epicsqt/framework/widgets/QELogin/QELogin.h
- /home/rhydera/epicsqt/framework/widgets/QELogin/QELogin.cpp

## 9.73 QEPeriodic Class Reference

Inheritance diagram for QEPeriodic:



### Classes

- struct [elementInfoStruct](#)
- struct [userInfoStructArray](#)

### Public Types

- enum **variableTypes** {  
**VARIABLE\_TYPE\_NUMBER**, **VARIABLE\_TYPE\_ATOMIC\_WEIGHT**, **VARIABLE\_TYPE\_MELTING\_POINT**, **VARIABLE\_TYPE\_BOILING\_POINT**,  
**VARIABLE\_TYPE\_DENSITY**, **VARIABLE\_TYPE\_GROUP**, **VARIABLE\_TYPE\_IONIZATION\_ENERGY**, **VARIABLE\_TYPE\_USER\_VALUE\_1**,  
**VARIABLE\_TYPE\_USER\_VALUE\_2** }
- enum **presentationOptions** { **PRESENTATION\_BUTTON\_AND\_LABEL**, **PRESENTATION\_BUTTON\_ONLY**, **PRESENTATION\_LABEL\_ONLY** }
- enum **UserLevels** { **User** = USERLEVEL\_USER, **Scientist** = USERLEVEL\_SCIENTIST, **Engineer** = USERLEVEL\_ENGINEER }
- enum **PresentationOptions** { **buttonAndLabel** = QEPeriodic::PRESENTATION\_BUTTON\_AND\_LABEL, **buttonOnly** = QEPeriodic::PRESENTATION\_BUTTON\_ONLY, **labelOnly** = QEPeriodic::PRESENTATION\_LABEL\_ONLY }
- enum **VariableTypes** {  
**Number** = QEPeriodic::VARIABLE\_TYPE\_NUMBER, **atomicWeight** = QEPeriodic::VARIABLE\_TYPE\_ATOMIC\_WEIGHT, **meltingPoint** = QEPeriodic::VARIABLE\_TYPE\_MELTING\_POINT, **boilingPoint** = QEPeriodic::VARIABLE\_TYPE\_BOILING\_POINT,  
**density** = QEPeriodic::VARIABLE\_TYPE\_DENSITY, **group** = QEPeriodic::VARIABLE\_TYPE\_GROUP, **ionizationEnergy** = QEPeriodic::VARIABLE\_TYPE\_IONIZATION\_ENERGY, **userValue1** = QEPeriodic::VARIABLE\_TYPE\_USER\_VALUE\_1,  
**userValue2** = QEPeriodic::VARIABLE\_TYPE\_USER\_VALUE\_2 }

### Public Slots

- void [requestEnabled](#) (const bool &state)

## Signals

- void [dbValueChanged](#) (const double &out)
- void [dbElementChanged](#) (const QString &out)
- void [requestResend](#) ()

*Internal use only. Used when changing a property value to force a re-display to reflect the new property value.*

## Public Member Functions

- **QEPeriodic** (QWidget \*parent=0)
- **QEPeriodic** (const QString &variableName, QWidget \*parent=0)
- void **setSubscribe** (bool subscribe)
- bool **getSubscribe** ()
- void **setPresentationOption** (presentationOptions presentationOptionIn)
- presentationOptions **getPresentationOption** ()
- void **setVariableType1** (variableTypes variableType1In)
- variableTypes **getVariableType1** ()
- void **setVariableType2** (variableTypes variableType2In)
- variableTypes **getVariableType2** ()
- void **setVariableTolerance1** (double variableTolerance1In)
- double **getVariableTolerance1** ()
- void **setVariableTolerance2** (double variableTolerance2In)
- double **getVariableTolerance2** ()
- void **setUserInfo** (QString userInfo)
- QString **getUserInfo** ()
- bool **isEnabled** () const
- *Access function for [enabled](#) property - refer to [enabled](#) property for details.*
- void **setEnabled** (bool state)
- *Access function for [enabled](#) property - refer to [enabled](#) property for details.*
- **UserLevels** **getUserLevelVisibilityProperty** ()
- *Access function for [userLevelVisibility](#) property - refer to [userLevelVisibility](#) property for details.*
- void **setUserLevelVisibilityProperty** (UserLevels level)
- *Access function for [userLevelVisibility](#) property - refer to [userLevelVisibility](#) property for details.*
- **UserLevels** **getUserLevelEnabledProperty** ()
- *Access function for [userLevelEnabled](#) property - refer to [userLevelEnabled](#) property for details.*
- void **setUserLevelEnabledProperty** (UserLevels level)
- *Access function for [userLevelEnabled](#) property - refer to [userLevelEnabled](#) property for details.*
- void **setPresentationOptionProperty** (PresentationOptions presentationOption)
- PresentationOptions **getPresentationOptionProperty** ()
- void **setVariableType1Property** (VariableTypes variableType)

- void **setVariableType2Property** (VariableTypes variableType)
- VariableTypes **getVariableType1Property** ()
- VariableTypes **getVariableType2Property** ()

### Public Attributes

- [userInfoStruct](#) **userInfo** [NUM\_ELEMENTS]

### Static Public Attributes

- static [elementInfoStruct](#) **elementInfo** [NUM\_ELEMENTS]

### Protected Member Functions

- void **establishConnection** (unsigned int variableIndex)
- void **dragEnterEvent** (QDragEnterEvent \*event)
- void **dropEvent** (QDropEvent \*event)
- void **setDrop** (QVariant drop)
- QVariant **getDrop** ()

### Protected Attributes

- [QEFloatingFormatting](#) **floatingFormatting**
- bool **localEnabled**
- bool [allowDrop](#)
- variableTypes **variableType1**
- variableTypes **variableType2**
- double **variableTolerance1**
- double **variableTolerance2**

### Properties

- QString [writeButtonVariable1](#)
- QString [writeButtonVariable2](#)
- QString [readbackLabelVariable1](#)
- QString [readbackLabelVariable2](#)
- QString [variableSubstitutions](#)
- bool [subscribe](#)
- bool [variableAsToolTip](#)
- bool [enabled](#)
- bool [visible](#)
- unsigned int
- QString [userLevelUserStyle](#)
- QString [userLevelScientistStyle](#)

- QString [userLevelEngineerStyle](#)
- [UserLevels](#) [userLevelVisibility](#)
- [UserLevels](#) [userLevelEnabled](#)
- PresentationOptions **presentationOption**
- VariableTypes **variableType1**
- VariableTypes **variableType2**
- QString **userInfo**

### 9.73.1 Member Enumeration Documentation

#### 9.73.1.1 enum QEPeriodic::UserLevels

User friendly enumerations for [userLevelVisibility](#) and [userLevelEnabled](#) properties - refer to [userLevelVisibility](#) and [userLevelEnabled](#) properties and userLevel enumeration for details.

#### Enumerator:

**User** Refer to USERLEVEL\_USER for details.

**Scientist** Refer to USERLEVEL\_SCIENTIST for details.

**Engineer** Refer to USERLEVEL\_ENGINEER for details.

### 9.73.2 Member Function Documentation

#### 9.73.2.1 void QEPeriodic::dbElementChanged ( const QString & out ) [signal]

Sent when the widget is updated following a data change Can be used to pass on EPICS data (as presented in this widget) to other widgets. For example a QList widget could log updates from this widget.

#### 9.73.2.2 void QEPeriodic::dbValueChanged ( const double & out ) [signal]

Sent when the widget is updated following a data change Can be used to pass on EPICS data (as presented in this widget) to other widgets. For example a QList widget could log updates from this widget.

#### 9.73.2.3 void QEPeriodic::requestEnabled ( const bool & state ) [inline, slot]

Similar to standard setEnabled slot, but allows QE widget to determine if the widget remains disabled due to invalid data. If disabled due to invalid data, a request to enable the widget will be honoured when the data is no longer invalid.

### 9.73.3 Member Data Documentation

#### 9.73.3.1 `bool QEPeiodic::allowDrop` [read, write, protected]

Allow drag/drops operations to this widget. Default is false. Any dropped text will be used as a new variable name.

Reimplemented from [QEDragDrop](#).

### 9.73.4 Property Documentation

#### 9.73.4.1 `bool QEPeiodic::enabled` [read, write]

Set the preferred 'enabled' state. Default is true. This property is copied to the standard Qt 'enabled' property if the data being displayed is valid. If the data being displayed is invalid the standard Qt 'enabled' property will always be set to false to indicate invalid data. The value of this property will only be copied to the standard Qt 'enabled' property once data is valid.

#### 9.73.4.2 `unsigned QEPeiodic::int` [read, write]

Set the ID used by the message filtering system. Default is zero. Widgets or applications that use messages from the framework have the option of filtering on this ID. For example, by using a unique message source ID a [QELog](#) widget may be set up to only log messages from a select set of widgets.

#### 9.73.4.3 `QString QEPeiodic::readbackLabelVariable1` [read, write]

EPICS variable name (CA PV). This variable is used to read the value to the first of two positioners to determine which (if any) element is currently selected.

#### 9.73.4.4 `QString QEPeiodic::readbackLabelVariable2` [read, write]

EPICS variable name (CA PV). This variable is used to read the value to the second of two positioners to determine which (if any) element is currently selected.

#### 9.73.4.5 `bool QEPeiodic::subscribe` [read, write]

Sets if this widget subscribes for data updates and displays current data. Default is 'true' (subscribes for and displays data updates)

Reimplemented from [QEWidget](#).

#### 9.73.4.6 UserLevels QEP periodic::userLevelEnabled [read, write]

Lowest user level at which the widget is enabled. Default is 'User'. Used when designing GUIs that allow access to more and more detail according to the user mode. The user mode is set application wide through the [QELogin](#) widget, or programatically through `setUserLevel()`. Widgets that are always accessible should be visible at 'User'. Widgets that are only accessible to scientists managing the facility should be visible at 'Scientist'. Widgets that are only accessible to engineers maintaining the facility should be visible at 'Engineer'.

#### 9.73.4.7 QString QEP periodic::userLevelEngineerStyle [read, write]

Style Sheet string to be applied when the widget is displayed in 'Engineer' mode. Default is an empty string. The syntax is the standard Qt Style Sheet syntax. For example, 'background-color: red'. This Style Sheet string will be applied by the `styleManager` class. Refer to the `styleManager` class for details about how this Style Sheet string will be merged with any pre-existing Style Sheet string and any Style Sheet strings generated during the display of data.

#### 9.73.4.8 QString QEP periodic::userLevelScientistStyle [read, write]

Style Sheet string to be applied when the widget is displayed in 'Scientist' mode. Default is an empty string. The syntax is the standard Qt Style Sheet syntax. For example, 'background-color: red'. This Style Sheet string will be applied by the `styleManager` class. Refer to the `styleManager` class for details about how this Style Sheet string will be merged with any pre-existing Style Sheet string and any Style Sheet strings generated during the display of data.

#### 9.73.4.9 QString QEP periodic::userLevelUserStyle [read, write]

Style Sheet string to be applied when the widget is displayed in 'User' mode. Default is an empty string. The syntax is the standard Qt Style Sheet syntax. For example, 'background-color: red'. This Style Sheet string will be applied by the `styleManager` class. Refer to the `styleManager` class for details about how this Style Sheet string will be merged with any pre-existing Style Sheet string and any Style Sheet strings generated during the display of data.

#### 9.73.4.10 UserLevels QEP periodic::userLevelVisibility [read, write]

Lowest user level at which the widget is visible. Default is 'User'. Used when designing GUIs that display more and more detail according to the user mode. The user mode is set application wide through the [QELogin](#) widget, or programatically through `setUserLevel()`. Widgets that are always visible should be visible at 'User'. Widgets that are only used by scientists managing the facility should be visible at 'Scientist'. Widgets that are only used by engineers maintaining the facility should be visible at 'Engineer'.



**9.73.4.11 bool QEPERIODIC::variableAsToolTip** [read, write]

Use the variable as the tool tip. Default is true. Tool tip property will be overwritten by the variable name.

Reimplemented from [QEToolTip](#).

**9.73.4.12 QString QEPERIODIC::variableSubstitutions** [read, write]

Macro substitutions. The default is no substitutions. The format is NAME1=VALUE1[, NAME2=VALUE2... Values may be quoted strings. For example, 'SAMPLE=SAM1, NAME = "Ref foil"' These substitutions are applied to all the variable names.

**9.73.4.13 bool QEPERIODIC::visible** [read, write]

Display the widget. Default is true. Setting this property false is usefull if widget is only used to provide a signal - for example, when supplying data to a [QELink](#) widget. Note, when false the widget will still be visible in Qt Designer.

**9.73.4.14 QString QEPERIODIC::writeButtonVariable1** [read, write]

EPICS variable name (CA PV). This variable is used to write a value to the first of two positioners that will position the select element.

**9.73.4.15 QString QEPERIODIC::writeButtonVariable2** [read, write]

EPICS variable name (CA PV). This variable is used to write a value to the second of two positioners that will position the select element.

The documentation for this class was generated from the following files:

- /home/rhydera/epicsqt/framework/widgets/QEPERIODIC/QEPERIODIC.h
- /home/rhydera/epicsqt/framework/widgets/QEPERIODIC/QEPERIODIC.cpp

**9.74 QEPERIODICCOMPONENTDATA Class Reference****Public Attributes**

- unsigned int **variableIndex1**
- double **lastData1**
- bool **haveLastData1**
- unsigned int **variableIndex2**
- double **lastData2**
- bool **haveLastData2**

The documentation for this class was generated from the following file:

- /home/rhydera/epicsqt/framework/widgets/QEPeriodic/QEPeriodic.h

## 9.75 QEPeriodicTaskMenu Class Reference

### Public Member Functions

- **QEPeriodicTaskMenu** ([QEPeriodic](#) \*periodic, QObject \*parent)
- QAction \* **preferredEditAction** () const
- QList< QAction \* > **taskActions** () const

The documentation for this class was generated from the following files:

- /home/rhydera/epicsqt/framework/widgets/QEPeriodic/QEPeriodicTaskMenu.h
- /home/rhydera/epicsqt/framework/widgets/QEPeriodic/QEPeriodicTaskMenuExtension.cpp

## 9.76 QEPeriodicTaskMenuFactory Class Reference

### Public Member Functions

- **QEPeriodicTaskMenuFactory** (QExtensionManager \*parent=0)

### Protected Member Functions

- QObject \* **createExtension** (QObject \*object, const QString &iid, QObject \*parent) const

The documentation for this class was generated from the following files:

- /home/rhydera/epicsqt/framework/widgets/QEPeriodic/QEPeriodicTaskMenu.h
- /home/rhydera/epicsqt/framework/widgets/QEPeriodic/QEPeriodicTaskMenuExtension.cpp

## 9.77 QEpicsPV Class Reference

### Public Slots

- const QVariant & **set** (QVariant value, int delay=-1)
- void **setPV** (const QString &\_pvName="")

## Signals

- void **connectionChanged** (bool connected)
- void **connected** ()
- void **disconnected** ()
- void **valueChanged** (const QVariant &value)
- void **valueUpdated** (const QVariant &value)
- void **valueInitd** (const QVariant &value)

## Public Member Functions

- **QEpicsPV** (const QString &\_pvName, QObject \*parent=0)
- **QEpicsPV** (QObject \*parent=0)
- const QVariant & **get** () const
- void **needUpdated** () const
- const QVariant & **getUpdated** (int delay=defaultDelay) const
- bool **isConnected** () const
- const QStringList & **getEnum** () const
- const QString & **pv** () const
- const QVariant & **getReady** (int delay=defaultDelay) const

## Static Public Member Functions

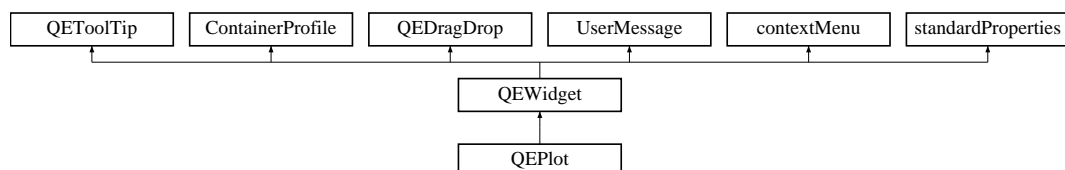
- static void **setDebugLevel** (unsigned level=0)
- static QVariant **get** (const QString &\_pvName, int delay=defaultDelay)
- static QVariant **set** (QString &\_pvName, const QVariant &value, int delay=-1)

The documentation for this class was generated from the following files:

- /home/rhydera/epicsqt/framework/data/include/qepicspv.h
- /home/rhydera/epicsqt/framework/data/src/qepicspv.cpp

## 9.78 QEPlot Class Reference

Inheritance diagram for QEPlot:



## Public Types

- enum **UserLevels** { **User** = USERLEVEL\_USER, **Scientist** = USERLEVEL\_SCIENTIST, **Engineer** = USERLEVEL\_ENGINEER }
- enum **TraceStyles** { **Lines** = QwtPlotCurve::Lines, **Sticks** = QwtPlotCurve::Sticks, **Steps** = QwtPlotCurve::Steps, **Dots** = QwtPlotCurve::Dots }

## Public Slots

- void **requestEnabled** (const bool &state)

## Signals

- void **dbValueChanged** (const double &out)
- void **dbValueChanged** (const QVector< double > &out)

## Public Member Functions

- **QEPlot** (QWidget \*parent=0)
- **QEPlot** (const QString &variableName, QWidget \*parent=0)
- void **setYMin** (double yMin)
- double **getYMin** ()
- void **setYMax** (double yMax)
- double **getYMax** ()
- void **setAutoScale** (bool autoScale)
- bool **getAutoScale** ()
- void **setAxisEnableX** (bool axisEnableXIn)
- bool **getAxisEnableX** ()
- void **setAxisEnableY** (bool axisEnableYIn)
- bool **getAxisEnableY** ()
- QString **getTitle** ()
- void **setBackgroundColor** (QColor backgroundColor)
- QColor **getBackgroundColor** ()
- void **setTraceStyle** (QwtPlotCurve::CurveStyle traceStyle, const unsigned int variableIndex)
- QwtPlotCurve::CurveStyle **getTraceStyle** (const unsigned int variableIndex)
- void **setTraceColor** (QColor traceColor, const unsigned int variableIndex)
- void **setTraceColor1** (QColor traceColor)
- void **setTraceColor2** (QColor traceColor)
- void **setTraceColor3** (QColor traceColor)
- void **setTraceColor4** (QColor traceColor)
- QColor **getTraceColor** (const unsigned int variableIndex)
- QColor **getTraceColor1** ()
- QColor **getTraceColor2** ()
- QColor **getTraceColor3** ()

- QColor **getTraceColor4** ()
- void **setTraceLegend1** (QString traceLegend)
- void **setTraceLegend2** (QString traceLegend)
- void **setTraceLegend3** (QString traceLegend)
- void **setTraceLegend4** (QString traceLegend)
- QString **getTraceLegend1** ()
- QString **getTraceLegend2** ()
- QString **getTraceLegend3** ()
- QString **getTraceLegend4** ()
- void **setXUnit** (QString xUnit)
- QString **getXUnit** ()
- void **setYUnit** (QString yUnit)
- QString **getYUnit** ()
- void **setGridEnableMajorX** (bool gridEnableMajorXIn)
- void **setGridEnableMajorY** (bool gridEnableMajorYIn)
- void **setGridEnableMinorX** (bool gridEnableMinorXIn)
- void **setGridEnableMinorY** (bool gridEnableMinorYIn)
- bool **getGridEnableMajorX** ()
- bool **getGridEnableMajorY** ()
- bool **getGridEnableMinorX** ()
- bool **getGridEnableMinorY** ()
- void **setGridMajorColor** (QColor gridMajorColorIn)
- void **setGridMinorColor** (QColor gridMinorColorIn)
- QColor **getGridMajorColor** ()
- QColor **getGridMinorColor** ()
- void **setXStart** (double xStart)
- double **getXStart** ()
- void **setXIncrement** (double xIncrement)
- double **getXIncrement** ()
- void **setTimeSpan** (unsigned int timeSpan)
- unsigned int **getTimeSpan** ()
- void **setTickRate** (unsigned int tickRate)
- unsigned int **getTickRate** ()
- bool **isEnabled** () const  
*Access function for [enabled](#) property - refer to [enabled](#) property for details.*
- void **setEnabled** (bool state)  
*Access function for [enabled](#) property - refer to [enabled](#) property for details.*
- [UserLevels](#) **getUserLevelVisibilityProperty** ()  
*Access function for [userLevelVisibility](#) property - refer to [userLevelVisibility](#) property for details.*
- void **setUserLevelVisibilityProperty** ([UserLevels](#) level)  
*Access function for [userLevelVisibility](#) property - refer to [userLevelVisibility](#) property for details.*
- [UserLevels](#) **getUserLevelEnabledProperty** ()  
*Access function for [userLevelEnabled](#) property - refer to [userLevelEnabled](#) property for details.*

- void [setUserLevelEnabledProperty](#) ([UserLevels](#) level)  
*Access function for [userLevelEnabled](#) property - refer to [userLevelEnabled](#) property for details.*
- void **setTraceStyle1** (TraceStyles traceStyle)
- void **setTraceStyle2** (TraceStyles traceStyle)
- void **setTraceStyle3** (TraceStyles traceStyle)
- void **setTraceStyle4** (TraceStyles traceStyle)
- TraceStyles **getTraceStyle1** ()
- TraceStyles **getTraceStyle2** ()
- TraceStyles **getTraceStyle3** ()
- TraceStyles **getTraceStyle4** ()

### Protected Member Functions

- void **establishConnection** (unsigned int variableIndex)
- void **dragEnterEvent** (QDragEnterEvent \*event)
- void **dropEvent** (QDropEvent \*event)
- void **mousePressEvent** (QMouseEvent \*event)
- void **setDrop** (QVariant drop)
- QVariant **getDrop** ()

### Protected Attributes

- [QEFloatingFormatting](#) **floatingFormatting**
- bool **localEnabled**
- bool [allowDrop](#)

### Properties

- QString [variable1](#)
- QString [variable2](#)
- QString [variable3](#)
- QString [variable4](#)
- QString [variableSubstitutions](#)
- bool [variableAsToolTip](#)
- bool [enabled](#)
- bool [visible](#)
- unsigned int
- QString [userLevelUserStyle](#)
- QString [userLevelScientistStyle](#)
- QString [userLevelEngineerStyle](#)
- [UserLevels](#) [userLevelVisibility](#)
- [UserLevels](#) [userLevelEnabled](#)
- QColor **traceColor1**
- QColor **traceColor2**

- QColor **traceColor3**
- QColor **traceColor4**
- TraceStyles **traceStyle1**
- TraceStyles **traceStyle2**
- TraceStyles **traceStyle3**
- TraceStyles **traceStyle4**
- QString **traceLegend1**
- QString **traceLegend2**
- QString **traceLegend3**
- QString **traceLegend4**
- QString **title**
- QColor **backgroundColor**
- QString **xUnit**
- QString **yUnit**

### 9.78.1 Member Enumeration Documentation

#### 9.78.1.1 enum QEPlot::UserLevels

User friendly enumerations for [userLevelVisibility](#) and [userLevelEnabled](#) properties - refer to [userLevelVisibility](#) and [userLevelEnabled](#) properties and [userLevel](#) enumeration for details.

##### Enumerator:

**User** Refer to USERLEVEL\_USER for details.

**Scientist** Refer to USERLEVEL\_SCIENTIST for details.

**Engineer** Refer to USERLEVEL\_ENGINEER for details.

### 9.78.2 Member Function Documentation

#### 9.78.2.1 void QEPlot::dbValueChanged ( const double & *out* ) [signal]

Sent when the widget is updated following a data change Can be used to pass on EPICS data (as presented in this widget) to other widgets. For example a QList widget could log updates from this widget.

#### 9.78.2.2 void QEPlot::dbValueChanged ( const QVector< double > & *out* ) [signal]

Sent when the widget is updated following a data change Can be used to pass on EPICS data (as presented in this widget) to other widgets. For example a QList widget could log updates from this widget.

### 9.78.2.3 void QEPlot::requestEnabled ( const bool & state ) [inline, slot]

Similar to standard setEnabled slot, but allows QE widget to determine if the widget remains disabled due to invalid data. If disabled due to invalid data, a request to enable the widget will be honoured when the data is no longer invalid.

## 9.78.3 Member Data Documentation

### 9.78.3.1 bool QEPlot::allowDrop [read, write, protected]

Allow drag/drops operations to this widget. Default is false. Any dropped text will be used as a new variable name.

Reimplemented from [QEDragDrop](#).

## 9.78.4 Property Documentation

### 9.78.4.1 bool QEPlot::enabled [read, write]

Set the preferred 'enabled' state. Default is true. This property is copied to the standard Qt 'enabled' property if the data being displayed is valid. If the data being displayed is invalid the standard Qt 'enabled' property will always be set to false to indicate invalid data. The value of this property will only be copied to the standard Qt 'enabled' property once data is valid.

### 9.78.4.2 unsigned QEPlot::int [read, write]

Set the ID used by the message filtering system. Default is zero. Widgets or applications that use messages from the framework have the option of filtering on this ID. For example, by using a unique message source ID a [QELog](#) widget may be set up to only log messages from a select set of widgets.

### 9.78.4.3 UserLevels QEPlot::userLevelEnabled [read, write]

Lowest user level at which the widget is enabled. Default is 'User'. Used when designing GUIs that allow access to more and more detail according to the user mode. The user mode is set application wide through the [QELogin](#) widget, or programatically through setUserLevel(). Widgets that are always accessible should be visible at 'User'. Widgets that are only accessible to scientists managing the facility should be visible at 'Scientist'. Widgets that are only accessible to engineers maintaining the facility should be visible at 'Engineer'.

### 9.78.4.4 QString QEPlot::userLevelEngineerStyle [read, write]

Style Sheet string to be applied when the widget is displayed in 'Engineer' mode. Default is an empty string. The syntax is the standard Qt Style Sheet syntax. For example,



'background-color: red' This Style Sheet string will be applied by the styleManager class. Refer to the styleManager class for details about how this Style Sheet string will be merged with any pre-existing Style Sheet string and any Style Sheet strings generated during the display of data.

#### 9.78.4.5 QString QEPlot::userLevelScientistStyle [read, write]

Style Sheet string to be applied when the widget is displayed in 'Scientist' mode. Default is an empty string. The syntax is the standard Qt Style Sheet syntax. For example, 'background-color: red' This Style Sheet string will be applied by the styleManager class. Refer to the styleManager class for details about how this Style Sheet string will be merged with any pre-existing Style Sheet string and any Style Sheet strings generated during the display of data.

#### 9.78.4.6 QString QEPlot::userLevelUserStyle [read, write]

Style Sheet string to be applied when the widget is displayed in 'User' mode. Default is an empty string. The syntax is the standard Qt Style Sheet syntax. For example, 'background-color: red' This Style Sheet string will be applied by the styleManager class. Refer to the styleManager class for details about how this Style Sheet string will be merged with any pre-existing Style Sheet string and any Style Sheet strings generated during the display of data.

#### 9.78.4.7 UserLevels QEPlot::userLevelVisibility [read, write]

Lowest user level at which the widget is visible. Default is 'User'. Used when designing GUIs that display more and more detail according to the user mode. The user mode is set application wide through the [QELogin](#) widget, or programatically through setUserLevel() Widgets that are always visible should be visible at 'User'. Widgets that are only used by scientists managing the facility should be visible at 'Scientist'. Widgets that are only used by engineers maintaining the facility should be visible at 'Engineer'.

#### 9.78.4.8 QString QEPlot::variable1 [read, write]

EPICS variable name (CA PV). This variable is used to read updating values or waveforms for plotting in the first trace.

#### 9.78.4.9 QString QEPlot::variable2 [read, write]

EPICS variable name (CA PV). This variable is used to read updating values or waveforms for plotting in the second trace.

#### 9.78.4.10 QString QEPlot::variable3 [read, write]

EPICS variable name (CA PV). This variable is used to read updating values or waveforms for plotting in the third trace.

#### 9.78.4.11 QString QEPlot::variable4 [read, write]

EPICS variable name (CA PV). This variable is used to read updating values or waveforms for plotting in the fourth trace.

#### 9.78.4.12 bool QEPlot::variableAsToolTip [read, write]

Use the variable as the tool tip. Default is true. Tool tip property will be overwritten by the variable name.

Reimplemented from [QEToolTip](#).

#### 9.78.4.13 QString QEPlot::variableSubstitutions [read, write]

Macro substitutions. The default is no substitutions. The format is NAME1=VALUE1[, NAME2=VALUE2... Values may be quoted strings. For example, 'SAMPLE=SAM1, NAME = "Ref foil"' These substitutions are applied to all the variable names.

#### 9.78.4.14 bool QEPlot::visible [read, write]

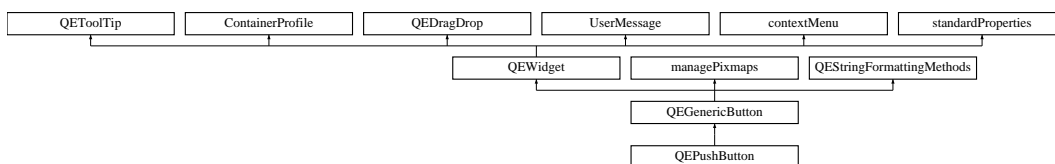
Display the widget. Default is true. Setting this property false is usefull if widget is only used to provide a signal - for example, when supplying data to a [QELink](#) widget. Note, when false the widget will still be visible in Qt Designer.

The documentation for this class was generated from the following files:

- /home/rhydera/epicsqt/framework/widgets/QEPlot/QEPlot.h
- /home/rhydera/epicsqt/framework/widgets/QEPlot/QEPlot.cpp

## 9.79 QEPushButton Class Reference

Inheritance diagram for QEPushButton:



## Public Types

- enum `UserLevels` { `User` = USERLEVEL\_USER, `Scientist` = USERLEVEL\_SCIENTIST, `Engineer` = USERLEVEL\_ENGINEER }
- enum `Formats` {  
`Default` = QQStringFormatting::FORMAT\_DEFAULT, `Floating` = QQStringFormatting::FORMAT\_FLOATING, `Integer` = QQStringFormatting::FORMAT\_INTEGER, `UnsignedInteger` = QQStringFormatting::FORMAT\_UNSIGNEDINTEGER,  
`Time` = QQStringFormatting::FORMAT\_TIME, `LocalEnumeration` = QQStringFormatting::FORMAT\_LOCAL\_ENUMERATE }
- enum `Notations` { `Fixed` = QQStringFormatting::NOTATION\_FIXED, `Scientific` = QQStringFormatting::NOTATION\_SCIENTIFIC, `Automatic` = QQStringFormatting::NOTATION\_AUTOMATIC }
- enum `ArrayActions` { `Append` = QQStringFormatting::APPEND, `Ascii` = QQStringFormatting::ASCII, `Index` = QQStringFormatting::INDEX }
- enum `UpdateOptions` { `Text` = QEGenericButton::UPDATE\_TEXT, `Icon` = QEGenericButton::UPDATE\_ICON, `TextAndIcon` = QEGenericButton::UPDATE\_TEXT\_AND\_ICON, `State` = QEGenericButton::UPDATE\_STATE }

*User friendly enumerations for updateOption property - refer to QEGenericButton::updateOptions for details.*

- enum `CreationOptionNames` { `Open` = QForm::CREATION\_OPTION\_OPEN, `NewTab` = QForm::CREATION\_OPTION\_NEW\_TAB, `NewWindow` = QForm::CREATION\_OPTION\_NEW\_WINDOW }

*Creation options. Used to indicate how to present a GUI when requesting a new GUI be created. Open a new window, open a new tab, or replace the current window.*

## Public Slots

- void `launchGui` (QString guiName, QForm::creationOptions creationOption)
- void `requestEnabled` (const bool &state)

## Signals

- void `dbValueChanged` (const QString &out)
- void `requestResend` ()  
*Internal use only. Used when changing a property value to force a re-display to reflect the new property value.*
- void `newGui` (QString guiName, QForm::creationOptions creationOption)  
*Internal use only. Request a new GUI is created. Typically, this is caught by the QEGui application.*
- void `pressed` (int value)
- void `released` (int value)
- void `clicked` (int value)

## Public Member Functions

- [QEPushButton](#) (QWidget \*parent=0)
- [QEPushButton](#) (const QString &variableName, QWidget \*parent=0)
- bool [isEnabled](#) () const  
Access function for [enabled](#) property - refer to [enabled](#) property for details.
- void [setEnabled](#) (bool state)  
Access function for [enabled](#) property - refer to [enabled](#) property for details.
- [UserLevels](#) [getUserLevelVisibilityProperty](#) ()  
Access function for [userLevelVisibility](#) property - refer to [userLevelVisibility](#) property for details.
- void [setUserLevelVisibilityProperty](#) ([UserLevels](#) level)  
Access function for [userLevelVisibility](#) property - refer to [userLevelVisibility](#) property for details.
- [UserLevels](#) [getUserLevelEnabledProperty](#) ()  
Access function for [userLevelEnabled](#) property - refer to [userLevelEnabled](#) property for details.
- void [setUserLevelEnabledProperty](#) ([UserLevels](#) level)  
Access function for [userLevelEnabled](#) property - refer to [userLevelEnabled](#) property for details.
- void [setFormatProperty](#) ([Formats](#) format)  
Access function for [format](#) property - refer to [format](#) property for details.
- [Formats](#) [getFormatProperty](#) ()  
Access function for [format](#) property - refer to [format](#) property for details.
- void [setNotationProperty](#) ([Notations](#) notation)  
Access function for [notation](#) property - refer to [notation](#) property for details.
- [Notations](#) [getNotationProperty](#) ()  
Access function for [notation](#) property - refer to [notation](#) property for details.
- void [setArrayActionProperty](#) ([ArrayActions](#) arrayAction)  
Access function for [arrayAction](#) property - refer to [arrayAction](#) property for details.
- [ArrayActions](#) [getArrayActionProperty](#) ()  
Access function for [arrayAction](#) property - refer to [arrayAction](#) property for details.

## Properties

- QString [variable](#)
- QString [altReadbackVariable](#)
- QString [variableSubstitutions](#)
- bool [subscribe](#)
- bool [variableAsToolTip](#)
- bool [enabled](#)
- bool [allowDrop](#)
- bool [visible](#)
- unsigned [int](#)
- QString [userLevelUserStyle](#)

- QString [userLevelScientistStyle](#)
- QString [userLevelEngineerStyle](#)
- UserLevels [userLevelVisibility](#)
- UserLevels [userLevelEnabled](#)
- int [precision](#)
- bool [useDbPrecision](#)
- bool [leadingZero](#)
- bool [trailingZeros](#)
- bool [addUnits](#)
- QString [localEnumeration](#)
- Formats [format](#)
- Notations [notation](#)
- ArrayActions [arrayAction](#)
- Qt::Alignment [alignment](#)
- UpdateOptions [updateOption](#)
- QPixmap [pixmap0](#)
- QPixmap [pixmap1](#)
- QPixmap [pixmap2](#)
- QPixmap [pixmap3](#)
- QPixmap [pixmap4](#)
- QPixmap [pixmap5](#)
- QPixmap [pixmap6](#)
- QPixmap [pixmap7](#)
- QString [password](#)
- bool [confirmAction](#)
- bool [writeOnPress](#)
- bool [writeOnRelease](#)
- bool [writeOnClick](#)
- QString [pressText](#)
- QString [releaseText](#)
- QString [clickText](#)
- QString [clickCheckedText](#)
- QString [labelText](#)
- QString [program](#)
- QStringList [arguments](#)
- QString [guiFile](#)
- CreationOptionNames [creationOption](#)

### 9.79.1 Member Enumeration Documentation

#### 9.79.1.1 enum QEPushButton::ArrayActions

User friendly enumerations for arrayAction property - refer to [QStringFormatting::arrayActions](#) for details.

**Enumerator:**

**Append** Refer to [QStringFormatting::APPEND](#) for details.

**Ascii** Refer to [QStringFormatting::ASCII](#) for details.

**Index** Refer to [QStringFormatting::INDEX](#) for details.

**9.79.1.2 enum QEPushButton::CreationOptionNames**

Creation options. Used to indicate how to present a GUI when requesting a new GUI be created. Open a new window, open a new tab, or replace the current window.

**Enumerator:**

**Open** Replace the current GUI with the new GUI.

**NewTab** Open new GUI in a new tab.

**NewWindow** Open new GUI in a new window.

**9.79.1.3 enum QEPushButton::Formats**

User friendly enumerations for format property - refer to [QStringFormatting::formats](#) for details.

**Enumerator:**

**Default** Format as best appropriate for the data type.

**Floating** Format as a floating point number.

**Integer** Format as an integer.

**UnsignedInteger** Format as an unsigned integer.

**Time** Format as a time.

**LocalEnumeration** Format as a selection from the [localEnumeration](#) property.

**9.79.1.4 enum QEPushButton::Notations**

User friendly enumerations for notation property - refer to [QStringFormatting::notations](#) for details.

**Enumerator:**

**Fixed** Refer to [QStringFormatting::NOTATION\\_FIXED](#) for details.

**Scientific** Refer to [QStringFormatting::NOTATION\\_SCIENTIFIC](#) for details.

**Automatic** Refer to [QStringFormatting::NOTATION\\_AUTOMATIC](#) for details.

## 9.79.1.5 enum QEPushButton::UpdateOptions

User friendly enumerations for updateOption property - refer to QEGenericButton::updateOptions for details.

**Enumerator:**

**Text** Data updates will update the button text.

**Icon** Data updates will update the button icon.

**TextAndIcon** Data updates will update the button text and icon.

**State** Data updates will update the button state (checked or unchecked)

## 9.79.1.6 enum QEPushButton::UserLevels

User friendly enumerations for [userLevelVisibility](#) and [userLevelEnabled](#) properties - refer to [userLevelVisibility](#) and [userLevelEnabled](#) properties and #userLevel enumeration for details.

**Enumerator:**

**User** Refer to USERLEVEL\_USER for details.

**Scientist** Refer to USERLEVEL\_SCIENTIST for details.

**Engineer** Refer to USERLEVEL\_ENGINEER for details.

## 9.79.2 Constructor &amp; Destructor Documentation

9.79.2.1 QEPushButton::QEPushButton ( QWidget \* *parent* = 0 )

Create without a variable. Use setVariableNameProperty() and setSubstitutionsProperty() to define a variable and, optionally, macro substitutions later.

9.79.2.2 QEPushButton::QEPushButton ( const QString & *variableName*, QWidget \* *parent* = 0 )

Create with a variable. A connection is automatically established. If macro substitutions are required, create without a variable and set the variable and macro substitutions after creation.

## 9.79.3 Member Function Documentation

9.79.3.1 void QEPushButton::clicked ( int *value* ) [signal]

Button has been Clicked. The value emitted is the integer interpretation of the clickText property (or the clickCheckedText property if the button was checked)

### 9.79.3.2 void QEPushButton::dbValueChanged ( const QString & out ) [signal]

Sent when the widget is updated following a data change Can be used to pass on EPICS data (as presented in this widget) to other widgets. For example a QList widget could log updates from this widget.

### 9.79.3.3 void QEPushButton::launchGui ( QString guiName, QForm::creationOptions creationOption ) [inline, slot]

Default slot used to create a new GUI if there is no slot indicated in the [ContainerProfile](#) class. This slot is typically used when the button is pressed within the Designer preview window to allow the operation of the button to be tested. If an application does not specify a slot to use for creating new windows (through the [ContainerProfile](#) class) a window will still be created through this slot, but it will not respect the window creation options or any other window related application constraints. For example, the QEGui application does provide a slot for creating new GUIs in the [ContainerProfile](#) class which respects the creation options, knows how to add tabs in the application, and extend the application's window menu in the menu bar.

Reimplemented from [QEGenericButton](#).

### 9.79.3.4 void QEPushButton::pressed ( int value ) [signal]

Button has been Pressed. The value emitted is the integer interpretation of the press-Text property

### 9.79.3.5 void QEPushButton::released ( int value ) [signal]

Button has been Released The value emitted is the integer interpretation of the release-Text property

### 9.79.3.6 void QEPushButton::requestEnabled ( const bool & state ) [inline, slot]

Similar to standard setEnabled slot, but allows QE widget to determine if the widget remains disabled due to invalid data. If disabled due to invalid data, a request to enable the widget will be honoured when the data is no longer invalid.

## 9.79.4 Property Documentation

### 9.79.4.1 bool QEPushButton::addUnits [read, write]

If true (default), add engineering units supplied with the data.



**9.79.4.2 Qt::Alignment QEPushButton::alignment** [read, write]

Set the buttons text alignment. Left justification is particularly useful when displaying quickly changing numeric data updates.

**9.79.4.3 bool QEPushButton::allowDrop** [read, write]

Allow drag/drops operations to this widget. Default is false. Any dropped text will be used as a new variable name.

Reimplemented from [QEDragDrop](#).

**9.79.4.4 QString QEPushButton::altReadbackVariable** [read, write]

EPICS variable name (CA PV). This variable is used to provide a readback value when different to the variable written to by a button press.

**9.79.4.5 QStringList QEPushButton::arguments** [read, write]

Arguments for program specified in the 'program' property.

Reimplemented from [QEGenericButton](#).

**9.79.4.6 ArrayActions QEPushButton::arrayAction** [read, write]

Text formatting option for array data. Default is ASCII. Options are:

- ASCII - treat array as a single text string. For example an array of three characters 'a' 'b' 'c' will be formatted as 'abc'.
- APPEND - treat array as an array of numbers and format a string containing them all with a space between each. For example, an array of three numbers 10, 11 and 12 will be formatted as '10 11 12'.
- INDEX - Extract a single item from the array. The item is then formatted as any other non array data would be. The item selected is determined by the arrayIndex property. For example, if arrayIndex property is 1, an array of three numbers 10, 11 and 12 will be formatted as '11'.

**9.79.4.7 QString QEPushButton::clickCheckedText** [read, write]

Text used to compare with text written or read to determine if push button should be marked as checked. Note, must be an exact match following formatting of data updates. When writing values, the 'pressText', 'ReleaseText', or 'clickedtext' must match this property to cause the button to be checked when the write occurs.

Good example: formatting set to display a data value of '1' as 'On', clickCheckedText is 'On', clickText is 'On'. In this example, the push button will be checked when a data update occurs with a value of 1 or when the button is clicked.

Bad example: formatting set to display a data value of '1' as 'On', clickCheckedText is 'On', clickText is '1'. In this example, the push button will be checked when a data update occurs with a value of 1 but, although a valid value will be written when clicked, the button will not be checked when clicked as '1' is not the same as 'On'.

Reimplemented from [QEGenericButton](#).

#### 9.79.4.8 QString QEPushButton::clickText [read, write]

Value written when user clicks button if 'writeOnClick' property is true

Reimplemented from [QEGenericButton](#).

#### 9.79.4.9 bool QEPushButton::confirmAction [read, write]

If true, a dialog will be presented asking the user to confirm if the button action should be carried out

#### 9.79.4.10 CreationOptionNames QEPushButton::creationOption [read, write]

Creation options when opening a new GUI. Open a new window, open a new tab, or replace the current window. the creation option is supplied when the button generates a newGui signal. Application code connected to this signal should honour this request if possible. When used within the QEGui application, the QEGui application creates a new window, new tab, or replaces the current window as appropriate.

Reimplemented from [QEGenericButton](#).

#### 9.79.4.11 bool QEPushButton::enabled [read, write]

Set the preferred 'enabled' state. Default is true. This property is copied to the standard Qt 'enabled' property if the data being displayed is valid. If the data being displayed is invalid the standard Qt 'enabled' property will always be set to false to indicate invalid data. The value of this property will only be copied to the standard Qt 'enabled' property once data is valid.

#### 9.79.4.12 Formats QEPushButton::format [read, write]

Format to apply to data. Default is 'Default' in which case the data type supplied with the data determines how the data is formatted. For all other options, an attempt is made to format the data as requested (whatever its native form).

**9.79.4.13** `QString QEPushButton::guiFile` [read, write]

File name of GUI to be presented on button click. File name can be absolute, relative to the path of the QForm in which the [QEPushButton](#) is located, relative to the any path in the path list published in the [ContainerProfile](#) class, or relative to the current path. See [QEWidget::openQEFile\(\)](#) in `QEWidget.cpp` for details.

**9.79.4.14** `unsigned QEPushButton::int` [read, write]

Set the ID used by the message filtering system. Default is zero. Widgets or applications that use messages from the framework have the option of filtering on this ID. For example, by using a unique message source ID a [QELog](#) widget may be set up to only log messages from a select set of widgets.

Base used for when formatting integers. Default is 10 (duh!)

Index used to select a single item of data for formatting from an array of data. Default is 0. Only used when the `arrayAction` property is INDEX. Refer to the `arrayAction` property for more details.

**9.79.4.15** `QString QEPushButton::labelText` [read, write]

Button label text (prior to substitution). Macro substitutions will be applied to this text and the result will be set as the button text. Used when data updates are not being represented in the button text. For example, a button in a sub form may have a 'labelText' property of 'Turn Pump On'. When the sub form is used twice in a main form with substitutions `PUMPNUM=1` and `PUMPNUM=2` respectively, the two identical buttons in the sub forms will have the labels 'Turn Pump 1 On' and 'Turn Pump 2 On' respectively.

Reimplemented from [QEGenericButton](#).

**9.79.4.16** `bool QEPushButton::leadingZero` [read, write]

If true (default), always add a leading zero when formatting numbers.

**9.79.4.17** `QString QEPushButton::localEnumeration` [read, write]

An enumeration list used to data values. Used only when the formatting option is 'local enumeration'. Value is converted to an integer and used to select a string from this list.

Format is:

```
[[<|<=|!=|>|=|>]value1|*]: string1 , [[<|<=|!=|>|=|>]value2|*]: string2 , [[<|<=|!=|>|=|>]value3|*]: string3 , ...
```

Where: < Less than <= Less than or equal = Equal (default if no operator specified)  
>= Greather than or equal > Greater than Always match (used to specify default text)

Values may be numeric or textual Values do not have to be in any order, but first match wins Values may be quoted Strings may be quoted Consecutive values do not have to

be present. Operator is assumed to be equality if not present. White space is ignored except within quoted strings.

may be included in a string to indicate a line break

Examples are:

```
0:Off,1:On 0 : "Pump Running", 1 : "Pump not running" 0:"", 1:"Warning!\nAlarm"
<2:"Value is less than two", =2:"Value is equal to two", >2:"Value is grater than 2"
3:"Beamline Available", *:"" "Pump Off": "OH NO!, the pump is OFF!","Pump On": "It's
OK, the pump is on"
```

The data value is converted to a string if no enumeration for that value is available. For example, if the local enumeration is '0:off,1:on', and a value of 10 is processed, the text generated is '10'. If a blank string is required, this should be explicit. for example, '0:off,1:on,10: ""'

A range of numbers can be covered by a pair of values as in the following example:  
 >=4:"Between 4 and 8", <=8:"Between 4 and 8"

#### 9.79.4.18 Notations QEPushButton::notation [read, write]

Notation used for numerical formatting. Default is fixed.

#### 9.79.4.19 QString QEPushButton::password [read, write]

Password user will need to enter before any action is taken

Reimplemented from [QEGenericButton](#).

#### 9.79.4.20 QPixmap QEPushButton::pixmap0 [read, write]

Pixmap to display if updateOption is Icon or TextAndIcon and data value translates to an index of 0

#### 9.79.4.21 QPixmap QEPushButton::pixmap1 [read, write]

Pixmap to display if updateOption is Icon or TextAndIcon and data value translates to an index of 1

#### 9.79.4.22 QPixmap QEPushButton::pixmap2 [read, write]

Pixmap to display if updateOption is Icon or TextAndIcon and data value translates to an index of 2

**9.79.4.23 QPixmap QEPushButton::pixmap3** [read, write]

Pixmap to display if updateOption is Icon or TextAndIcon and data value translates to an index of 3

**9.79.4.24 QPixmap QEPushButton::pixmap4** [read, write]

Pixmap to display if updateOption is Icon or TextAndIcon and data value translates to an index of 4

**9.79.4.25 QPixmap QEPushButton::pixmap5** [read, write]

Pixmap to display if updateOption is Icon or TextAndIcon and data value translates to an index of 5

**9.79.4.26 QPixmap QEPushButton::pixmap6** [read, write]

Pixmap to display if updateOption is Icon or TextAndIcon and data value translates to an index of 6

**9.79.4.27 QPixmap QEPushButton::pixmap7** [read, write]

Pixmap to display if updateOption is Icon or TextAndIcon and data value translates to an index of 7

**9.79.4.28 int QEPushButton::precision** [read, write]

Precision used when formatting floating point numbers. The default is 4. This is only used if useDbPrecision is false.

**9.79.4.29 QString QEPushButton::pressText** [read, write]

Value written when user presses button if 'writeOnPress' property is true

Reimplemented from [QEGenericButton](#).

**9.79.4.30 QString QEPushButton::program** [read, write]

Program to run when the button is clicked. No attempt to run a program is made if this property is empty. Example: firefox

Reimplemented from [QEGenericButton](#).

**9.79.4.31 QString QEPushButton::releaseText** [read, write]

Value written when user releases button if 'writeOnRelease' property is true

Reimplemented from [QEGenericButton](#).

**9.79.4.32 bool QEPushButton::subscribe** [read, write]

Sets if this widget subscribes for data updates and displays current data. Default is 'true' (subscribes for and displays data updates)

Reimplemented from [QEWidget](#).

**9.79.4.33 bool QEPushButton::trailingZeros** [read, write]

If true (default), always remove any trailing zeros when formatting numbers.

**9.79.4.34 UpdateOptions QEPushButton::updateOption** [read, write]

Update options (text, pixmap, both, or state (checked or unchecked))

Reimplemented from [QEGenericButton](#).

**9.79.4.35 bool QEPushButton::useDbPrecision** [read, write]

If true (default), format floating point numbers using the precision supplied with the data. If false, the precision property is used.

**9.79.4.36 UserLevels QEPushButton::userLevelEnabled** [read, write]

Lowest user level at which the widget is enabled. Default is 'User'. Used when designing GUIs that allow access to more and more detail according to the user mode. The user mode is set application wide through the [QELogin](#) widget, or programatically through `setUserLevel()`. Widgets that are always accessible should be visible at 'User'. Widgets that are only accessible to scientists managing the facility should be visible at 'Scientist'. Widgets that are only accessible to engineers maintaining the facility should be visible at 'Engineer'.

**9.79.4.37 QString QEPushButton::userLevelEngineerStyle** [read, write]

Style Sheet string to be applied when the widget is displayed in 'Engineer' mode. Default is an empty string. The syntax is the standard Qt Style Sheet syntax. For example, 'background-color: red' This Style Sheet string will be applied by the styleManager class. Refer to the styleManager class for details about how this Style Sheet string will be merged with any pre-existing Style Sheet string and any Style Sheet strings generated during the display of data.

**9.79.4.38 QString QEPushButton::userLevelScientistStyle** [read, write]

Style Sheet string to be applied when the widget is displayed in 'Scientist' mode. Default is an empty string. The syntax is the standard Qt Style Sheet syntax. For example, 'background-color: red' This Style Sheet string will be applied by the styleManager class. Refer to the styleManager class for details about how this Style Sheet string will be merged with any pre-existing Style Sheet string and any Style Sheet strings generated during the display of data.

**9.79.4.39 QString QEPushButton::userLevelUserStyle** [read, write]

Style Sheet string to be applied when the widget is displayed in 'User' mode. Default is an empty string. The syntax is the standard Qt Style Sheet syntax. For example, 'background-color: red' This Style Sheet string will be applied by the styleManager class. Refer to the styleManager class for details about how this Style Sheet string will be merged with any pre-existing Style Sheet string and any Style Sheet strings generated during the display of data.

**9.79.4.40 UserLevels QEPushButton::userLevelVisibility** [read, write]

Lowest user level at which the widget is visible. Default is 'User'. Used when designing GUIs that display more and more detail according to the user mode. The user mode is set application wide through the [QELogin](#) widget, or programatically through setUserLevel() Widgets that are always visible should be visible at 'User'. Widgets that are only used by scientists managing the facility should be visible at 'Scientist'. Widgets that are only used by engineers maintaining the facility should be visible at 'Engineer'.

**9.79.4.41 QString QEPushButton::variable** [read, write]

EPICS variable name (CA PV). This variable is used for both writing (on button press), and reading if subscribed and no alternate readback variable is provided.

**9.79.4.42 bool QEPushButton::variableAsToolTip** [read, write]

Use the variable as the tool tip. Default is true. Tool tip property will be overwritten by the variable name.

Reimplemented from [QEToolTip](#).

**9.79.4.43 QString QEPushButton::variableSubstitutions** [read, write]

Macro substitutions. The default is no substitutions. The format is NAME1=VALUE1[, NAME2=VALUE2... Values may be quoted strings. For example, 'PUMP=PMP3, NAME = "My Pump"' These substitutions are applied to variable names for all QE widgets. In some widgets are also used for other purposes.

#### 9.79.4.44 bool QEPushButton::visible [read, write]

Display the widget. Default is true. Setting this property false is usefull if widget is only used to provide a signal - for example, when supplying data to a [QELink](#) widget. Note, when false the widget will still be visible in Qt Designer.

#### 9.79.4.45 bool QEPushButton::writeOnClick [read, write]

If true, the 'clickText' property is written when the button is clicked. Default is true

Reimplemented from [QEGenericButton](#).

#### 9.79.4.46 bool QEPushButton::writeOnPress [read, write]

If true, the 'pressText' property is written when the button is pressed. Default is false

Reimplemented from [QEGenericButton](#).

#### 9.79.4.47 bool QEPushButton::writeOnRelease [read, write]

If true, the 'releaseText' property is written when the button is released. Default is false

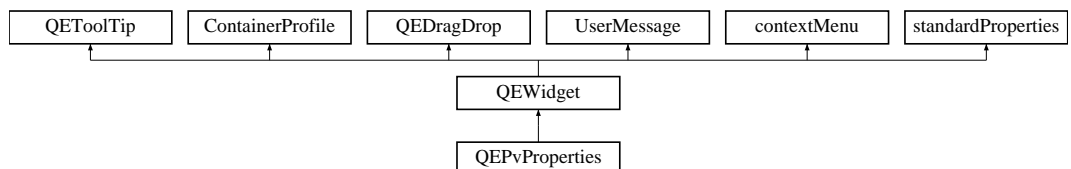
Reimplemented from [QEGenericButton](#).

The documentation for this class was generated from the following files:

- /home/rhydera/epicsqt/framework/widgets/QEButton/QEPushButton.h
- /home/rhydera/epicsqt/framework/widgets/QEButton/QEPushButton.cpp

## 9.80 QEPvProperties Class Reference

Inheritance diagram for QEPvProperties:



### Classes

- struct **WidgetHolder**



## Public Types

- enum [UserLevels](#) { [User](#) = USERLEVEL\_USER, [Scientist](#) = USERLEVEL\_SCIENTIST, [Engineer](#) = USERLEVEL\_ENGINEER }

## Public Slots

- void [requestEnabled](#) (const bool &state)

## Signals

- void [setCurrentBoxIndex](#) (int index)

## Public Member Functions

- bool [isEnabled](#) () const  
*Access function for [enabled](#) property - refer to [enabled](#) property for details.*
- void [setEnabled](#) (bool state)  
*Access function for [enabled](#) property - refer to [enabled](#) property for details.*
- [UserLevels](#) [getUserLevelVisibilityProperty](#) ()  
*Access function for [userLevelVisibility](#) property - refer to [userLevelVisibility](#) property for details.*
- void [setUserLevelVisibilityProperty](#) ([UserLevels](#) level)  
*Access function for [userLevelVisibility](#) property - refer to [userLevelVisibility](#) property for details.*
- [UserLevels](#) [getUserLevelEnabledProperty](#) ()  
*Access function for [userLevelEnabled](#) property - refer to [userLevelEnabled](#) property for details.*
- void [setUserLevelEnabledProperty](#) ([UserLevels](#) level)  
*Access function for [userLevelEnabled](#) property - refer to [userLevelEnabled](#) property for details.*
- QEPvProperties** (QWidget \*parent=0)
- QEPvProperties** (const QString &variableName, QWidget \*parent=0)
- QSize [sizeHint](#) () const
- void [establishConnection](#) (unsigned int variableIndex)

## Protected Member Functions

- void [setup](#) ()
- [qcaobject::QCaObject](#) \* [createQcaltem](#) (unsigned int variableIndex)
- void [dragEnterEvent](#) (QDragEnterEvent \*event)
- void [dropEvent](#) (QDropEvent \*event)
- void [mousePressEvent](#) (QMouseEvent \*event)
- QString [copyVariable](#) ()

- QVariant **copyData** ()
- void **paste** (QVariant s)
- void **setDrop** (QVariant drop)
- QVariant **getDrop** ()

## Properties

- QString [variable](#)
- QString [variableSubstitutions](#)
- bool [variableAsToolTip](#)
- bool [enabled](#)
- bool [allowDrop](#)
- bool [visible](#)
- unsigned int
- QString [userLevelUserStyle](#)
- QString [userLevelScientistStyle](#)
- QString [userLevelEngineerStyle](#)
- [UserLevels](#) [userLevelVisibility](#)
- [UserLevels](#) [userLevelEnabled](#)

## 9.80.1 Member Enumeration Documentation

### 9.80.1.1 enum QEPvProperties::UserLevels

User friendly enumerations for [userLevelVisibility](#) and [userLevelEnabled](#) properties - refer to [userLevelVisibility](#) and [userLevelEnabled](#) properties and [userLevel](#) enumeration for details.

#### Enumerator:

**User** Refer to USERLEVEL\_USER for details.

**Scientist** Refer to USERLEVEL\_SCIENTIST for details.

**Engineer** Refer to USERLEVEL\_ENGINEER for details.

## 9.80.2 Member Function Documentation

### 9.80.2.1 void QEPvProperties::requestEnabled ( const bool & *state* ) [inline, slot]

Similar to standard `setEnabled` slot, but allows QE widget to determine if the widget remains disabled due to invalid data. If disabled due to invalid data, a request to enable the widget will be honoured when the data is no longer invalid.

### 9.80.3 Property Documentation

#### 9.80.3.1 `bool QEPvProperties::allowDrop` [read, write]

Allow drag/drops operations to this widget. Default is false. Any dropped text will be used as a new variable name.

Reimplemented from [QEDragDrop](#).

#### 9.80.3.2 `bool QEPvProperties::enabled` [read, write]

Set the preferred 'enabled' state. Default is true. This property is copied to the standard Qt 'enabled' property if the data being displayed is valid. If the data being displayed is invalid the standard Qt 'enabled' property will always be set to false to indicate invalid data. The value of this property will only be copied to the standard Qt 'enabled' property once data is valid.

#### 9.80.3.3 `unsigned QEPvProperties::int` [read, write]

Set the ID used by the message filtering system. Default is zero. Widgets or applications that use messages from the framework have the option of filtering on this ID. For example, by using a unique message source ID a [QELog](#) widget may be set up to only log messages from a select set of widgets.

#### 9.80.3.4 `UserLevels QEPvProperties::userLevelEnabled` [read, write]

Lowest user level at which the widget is enabled. Default is 'User'. Used when designing GUIs that allow access to more and more detail according to the user mode. The user mode is set application wide through the [QELogin](#) widget, or programmatically through `setUserLevel()`. Widgets that are always accessible should be visible at 'User'. Widgets that are only accessible to scientists managing the facility should be visible at 'Scientist'. Widgets that are only accessible to engineers maintaining the facility should be visible at 'Engineer'.

#### 9.80.3.5 `QString QEPvProperties::userLevelEngineerStyle` [read, write]

Style Sheet string to be applied when the widget is displayed in 'Engineer' mode. Default is an empty string. The syntax is the standard Qt Style Sheet syntax. For example, 'background-color: red'. This Style Sheet string will be applied by the `styleManager` class. Refer to the `styleManager` class for details about how this Style Sheet string will be merged with any pre-existing Style Sheet string and any Style Sheet strings generated during the display of data.

#### 9.80.3.6 QString QEPvProperties::userLevelScientistStyle [read, write]

Style Sheet string to be applied when the widget is displayed in 'Scientist' mode. Default is an empty string. The syntax is the standard Qt Style Sheet syntax. For example, 'background-color: red' This Style Sheet string will be applied by the styleManager class. Refer to the styleManager class for details about how this Style Sheet string will be merged with any pre-existing Style Sheet string and any Style Sheet strings generated during the display of data.

#### 9.80.3.7 QString QEPvProperties::userLevelUserStyle [read, write]

Style Sheet string to be applied when the widget is displayed in 'User' mode. Default is an empty string. The syntax is the standard Qt Style Sheet syntax. For example, 'background-color: red' This Style Sheet string will be applied by the styleManager class. Refer to the styleManager class for details about how this Style Sheet string will be merged with any pre-existing Style Sheet string and any Style Sheet strings generated during the display of data.

#### 9.80.3.8 UserLevels QEPvProperties::userLevelVisibility [read, write]

Lowest user level at which the widget is visible. Default is 'User'. Used when designing GUIs that display more and more detail according to the user mode. The user mode is set application wide through the [QELogin](#) widget, or programatically through setUserLevel() Widgets that are always visible should be visible at 'User'. Widgets that are only used by scientists managing the facility should be visible at 'Scientist'. Widgets that are only used by engineers maintaining the facility should be visible at 'Engineer'.

#### 9.80.3.9 QString QEPvProperties::variable [read, write]

EPICS variable name (CA PV)

#### 9.80.3.10 bool QEPvProperties::variableAsToolTip [read, write]

Use the variable as the tool tip. Default is true. Tool tip property will be overwritten by the variable name.

Reimplemented from [QEToolTip](#).

#### 9.80.3.11 QString QEPvProperties::variableSubstitutions [read, write]

Macro substitutions. The default is no substitutions. The format is NAME1=VALUE1[, NAME2=VALUE2... Values may be quoted strings. For example, 'PUMP=PMP3, NAME = "My Pump"' These substitutions are applied to variable names for all QE widgets. In some widgets are also used for other purposes.

## 9.80.3.12 bool QEPvProperties::visible [read, write]

Display the widget. Default is true. Setting this property false is usefull if widget is only used to provide a signal - for example, when supplying data to a [QELink](#) widget. Note, when false the widget will still be visible in Qt Designer.

The documentation for this class was generated from the following files:

- /home/rhydera/epicsqt/framework/widgets/QEPvProperties/QEPvProperties.h
- /home/rhydera/epicsqt/framework/widgets/QEPvProperties/QEPvProperties.cpp

## 9.81 QEPvPropertiesManager Class Reference

## Public Member Functions

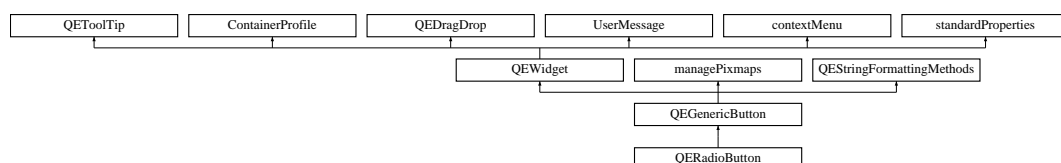
- **QEPvPropertiesManager** (QObject \*parent=0)
- bool **isContainer** () const
- bool **isInitialized** () const
- QIcon **icon** () const
- QString **group** () const
- QString **includeFile** () const
- QString **name** () const
- QString **toolTip** () const
- QString **whatsThis** () const
- QWidget \* **createWidget** (QWidget \*parent)
- void **initialize** (QDesignerFormEditorInterface \*core)

The documentation for this class was generated from the following files:

- /home/rhydera/epicsqt/framework/widgets/QEPvProperties/QEPvPropertiesManager.h
- /home/rhydera/epicsqt/framework/widgets/QEPvProperties/QEPvPropertiesManager.cpp

## 9.82 QERadioButton Class Reference

Inheritance diagram for QERadioButton:



## Public Types

- enum [UserLevels](#) { [User](#) = USERLEVEL\_USER, [Scientist](#) = USERLEVEL\_SCIENTIST, [Engineer](#) = USERLEVEL\_ENGINEER }
- enum [Formats](#) {  
[Default](#) = QStringFormatting::FORMAT\_DEFAULT, [Floating](#) = QStringFormatting::FORMAT\_FLOATING, [Integer](#) = QStringFormatting::FORMAT\_INTEGER, [UnsignedInteger](#) = QStringFormatting::FORMAT\_UNSIGNEDINTEGER,  
[Time](#) = QStringFormatting::FORMAT\_TIME, [LocalEnumeration](#) = QStringFormatting::FORMAT\_LOCAL\_ENUMERATE }
- enum [Notations](#) { [Fixed](#) = QStringFormatting::NOTATION\_FIXED, [Scientific](#) = QStringFormatting::NOTATION\_SCIENTIFIC, [Automatic](#) = QStringFormatting::NOTATION\_AUTOMATIC }
- enum [ArrayActions](#) { [Append](#) = QStringFormatting::APPEND, [Ascii](#) = QStringFormatting::ASCII, [Index](#) = QStringFormatting::INDEX }
- enum [UpdateOptions](#) { [Text](#) = QEGenericButton::UPDATE\_TEXT, [Icon](#) = QEGenericButton::UPDATE\_ICON, [TextAndIcon](#) = QEGenericButton::UPDATE\_TEXT\_AND\_ICON, [State](#) = QEGenericButton::UPDATE\_STATE }

*User friendly enumerations for updateOption property - refer to QEGenericButton::updateOptions for details.*

- enum [CreationOptionNames](#) { [Open](#) = QForm::CREATION\_OPTION\_OPEN, [NewTab](#) = QForm::CREATION\_OPTION\_NEW\_TAB, [NewWindow](#) = QForm::CREATION\_OPTION\_NEW\_WINDOW }

*Creation options. Used to indicate how to present a GUI when requesting a new GUI be created. Open a new window, open a new tab, or replace the current window.*

## Public Slots

- void [launchGui](#) (QString guiName, QForm::creationOptions creationOption)
- void [requestEnabled](#) (const bool &state)

## Signals

- void [dbValueChanged](#) (const QString &out)
- void [requestResend](#) ()  
*Internal use only. Used when changing a property value to force a re-display to reflect the new property value.*
- void [newGui](#) (QString guiName, QForm::creationOptions creationOption)  
*Internal use only. Request a new GUI is created. Typically, this is caught by the QEGui application.*
- void [pressed](#) (int value)
- void [released](#) (int value)
- void [clicked](#) (int value)

## Public Member Functions

- [QERadioButton](#) (QWidget \*parent=0)
- [QERadioButton](#) (const QString &variableName, QWidget \*parent=0)
- bool [isEnabled](#) () const  
Access function for [enabled](#) property - refer to [enabled](#) property for details.
- void [setEnabled](#) (bool state)  
Access function for [enabled](#) property - refer to [enabled](#) property for details.
- [UserLevels](#) [getUserLevelVisibilityProperty](#) ()  
Access function for [userLevelVisibility](#) property - refer to [userLevelVisibility](#) property for details.
- void [setUserLevelVisibilityProperty](#) ([UserLevels](#) level)  
Access function for [userLevelVisibility](#) property - refer to [userLevelVisibility](#) property for details.
- [UserLevels](#) [getUserLevelEnabledProperty](#) ()  
Access function for [userLevelEnabled](#) property - refer to [userLevelEnabled](#) property for details.
- void [setUserLevelEnabledProperty](#) ([UserLevels](#) level)  
Access function for [userLevelEnabled](#) property - refer to [userLevelEnabled](#) property for details.
- void [setFormatProperty](#) ([Formats](#) format)  
Access function for [format](#) property - refer to [format](#) property for details.
- [Formats](#) [getFormatProperty](#) ()  
Access function for [format](#) property - refer to [format](#) property for details.
- void [setNotationProperty](#) ([Notations](#) notation)  
Access function for [notation](#) property - refer to [notation](#) property for details.
- [Notations](#) [getNotationProperty](#) ()  
Access function for [notation](#) property - refer to [notation](#) property for details.
- void [setArrayActionProperty](#) ([ArrayActions](#) arrayAction)  
Access function for [arrayAction](#) property - refer to [arrayAction](#) property for details.
- [ArrayActions](#) [getArrayActionProperty](#) ()  
Access function for [arrayAction](#) property - refer to [arrayAction](#) property for details.

## Properties

- QString [variable](#)
- QString [variableSubstitutions](#)
- bool [subscribe](#)
- bool [variableAsToolTip](#)
- bool [enabled](#)
- bool [allowDrop](#)
- bool [visible](#)
- unsigned int
- QString [userLevelUserStyle](#)
- QString [userLevelScientistStyle](#)

- [QString userLevelEngineerStyle](#)
- [UserLevels userLevelVisibility](#)
- [UserLevels userLevelEnabled](#)
- [int precision](#)
- [bool useDbPrecision](#)
- [bool leadingZero](#)
- [bool trailingZeros](#)
- [bool addUnits](#)
- [QString localEnumeration](#)
- [Formats format](#)
- [Notations notation](#)
- [ArrayActions arrayAction](#)
- [Qt::Alignment alignment](#)
- [UpdateOptions updateOption](#)
- [QPixmap pixmap0](#)
- [QPixmap pixmap1](#)
- [QPixmap pixmap2](#)
- [QPixmap pixmap3](#)
- [QPixmap pixmap4](#)
- [QPixmap pixmap5](#)
- [QPixmap pixmap6](#)
- [QPixmap pixmap7](#)
- [QString password](#)
- [bool confirmAction](#)
- [bool writeOnPress](#)
- [bool writeOnRelease](#)
- [bool writeOnClick](#)
- [QString pressText](#)
- [QString releaseText](#)
- [QString clickText](#)
- [QString clickCheckedText](#)
- [QString labelText](#)
- [QString program](#)
- [QStringList arguments](#)
- [QString guiFile](#)
- [CreationOptionNames creationOption](#)

### 9.82.1 Member Enumeration Documentation

#### 9.82.1.1 enum `QERadioButton::ArrayActions`

User friendly enumerations for arrayAction property - refer to [QStringFormatting::arrayActions](#) for details.

##### Enumerator:

***Append*** Refer to [QStringFormatting::APPEND](#) for details.

***Ascii*** Refer to [QStringFormatting::ASCII](#) for details.

***Index*** Refer to [QStringFormatting::INDEX](#) for details.



## 9.82.1.2 enum QERadioButton::CreationOptionNames

Creation options. Used to indicate how to present a GUI when requesting a new GUI be created. Open a new window, open a new tab, or replace the current window.

**Enumerator:**

- Open** Replace the current GUI with the new GUI.
- NewTab** Open new GUI in a new tab.
- NewWindow** Open new GUI in a new window.

## 9.82.1.3 enum QERadioButton::Formats

User friendly enumerations for format property - refer to [QCStringFormatting::formats](#) for details.

**Enumerator:**

- Default** Format as best appropriate for the data type.
- Floating** Format as a floating point number.
- Integer** Format as an integer.
- UnsignedInteger** Format as an unsigned integer.
- Time** Format as a time.
- LocalEnumeration** Format as a selection from the [localEnumeration](#) property.

## 9.82.1.4 enum QERadioButton::Notations

User friendly enumerations for notation property - refer to [QCStringFormatting::notations](#) for details.

**Enumerator:**

- Fixed** Refer to [QCStringFormatting::NOTATION\\_FIXED](#) for details.
- Scientific** Refer to [QCStringFormatting::NOTATION\\_SCIENTIFIC](#) for details.
- Automatic** Refer to [QCStringFormatting::NOTATION\\_AUTOMATIC](#) for details.

## 9.82.1.5 enum QERadioButton::UpdateOptions

User friendly enumerations for updateOption property - refer to [QEGenericButton::updateOptions](#) for details.

**Enumerator:**

- Text** Data updates will update the button text.
- Icon** Data updates will update the button icon.
- TextAndIcon** Data updates will update the button text and icon.
- State** Data updates will update the button state (checked or unchecked)

### 9.82.1.6 enum QERadioButton::UserLevels

User friendly enumerations for [userLevelVisibility](#) and [userLevelEnabled](#) properties - refer to [userLevelVisibility](#) and [userLevelEnabled](#) properties and [userLevel](#) enumeration for details.

#### Enumerator:

**User** Refer to USERLEVEL\_USER for details.

**Scientist** Refer to USERLEVEL\_SCIENTIST for details.

**Engineer** Refer to USERLEVEL\_ENGINEER for details.

## 9.82.2 Constructor & Destructor Documentation

### 9.82.2.1 QERadioButton::QERadioButton ( QWidget \* parent = 0 )

Create without a variable. Use [setVariableNameProperty\(\)](#) and [setSubstitutionsProperty\(\)](#) to define a variable and, optionally, macro substitutions later.

### 9.82.2.2 QERadioButton::QERadioButton ( const QString & variableName, QWidget \* parent = 0 )

Create with a variable. A connection is automatically established. If macro substitutions are required, create without a variable and set the variable and macro substitutions after creation.

## 9.82.3 Member Function Documentation

### 9.82.3.1 void QERadioButton::clicked ( int value ) [signal]

Button has been Clicked. The value emitted is the integer interpretation of the [clickText](#) property (or the [clickCheckedText](#) property if the button was checked)

### 9.82.3.2 void QERadioButton::dbValueChanged ( const QString & out ) [signal]

Sent when the widget is updated following a data change Can be used to pass on EPICS data (as presented in this widget) to other widgets. For example a QList widget could log updates from this widget.

### 9.82.3.3 void QERadioButton::launchGui ( QString guiName, QEForm::creationOptions creationOption ) [inline, slot]

Default slot used to create a new GUI if there is no slot indicated in the [ContainerProfile](#) class. This slot is typically used when the button is pressed within the Designer preview window to allow the operation of the button to be tested. If an application does not

specify a slot to use for creating new windows (through the [ContainerProfile](#) class) a window will still be created through this slot, but it will not respect the window creation options or any other window related application constraints. For example, the QEGui application does provide a slot for creating new GUIs in the [ContainerProfile](#) class which respects the creation options, knows how to add tabs in the application, and extend the application's window menu in the menu bar.

Reimplemented from [QEGenericButton](#).

**9.82.3.4** void QERadioButton::pressed ( int *value* ) [signal]

Button has been Pressed. The value emitted is the integer interpretation of the press-Text property

**9.82.3.5** void QERadioButton::released ( int *value* ) [signal]

Button has been Released The value emitted is the integer interpretation of the release-Text property

**9.82.3.6** void QERadioButton::requestEnabled ( const bool & *state* ) [inline, slot]

Similar to standard setEnabled slot, but allows QE widget to determine if the widget remains disabled due to invalid data. If disabled due to invalid data, a request to enable the widget will be honoured when the data is no longer invalid.

## 9.82.4 Property Documentation

**9.82.4.1** bool QERadioButton::addUnits [read, write]

If true (default), add engineering units supplied with the data.

**9.82.4.2** Qt::Alignment QERadioButton::alignment [read, write]

Set the buttons text alignment. Left justification is particularly useful when displaying quickly changing numeric data updates.

**9.82.4.3** bool QERadioButton::allowDrop [read, write]

Allow drag/drops operations to this widget. Default is false. Any dropped text will be used as a new variable name.

Reimplemented from [QEDragDrop](#).

#### 9.82.4.4 QStringList QERadioButton::arguments [read, write]

Arguments for program specified in the 'program' property.

Reimplemented from [QEGenericButton](#).

#### 9.82.4.5 ArrayActions QERadioButton::arrayAction [read, write]

Text formatting option for array data. Default is ASCII. Options are:

- ASCII - treat array as a single text string. For example an array of three characters 'a' 'b' 'c' will be formatted as 'abc'.
- APPEND - treat array as an array of numbers and format a string containing them all with a space between each. For example, an array of three numbers 10, 11 and 12 will be formatted as '10 11 12'.
- INDEX - Extract a single item from the array. The item is then formatted as any other non array data would be. The item selected is determined by the arrayIndex property. For example, if arrayIndex property is 1, an array of three numbers 10, 11 and 12 will be formatted as '11'.

#### 9.82.4.6 QString QERadioButton::clickCheckedText [read, write]

Text used to compare with text written or read to determine if push button should be marked as checked. Note, must be an exact match following formatting of data updates. When writing values, the 'pressText', 'ReleaseText', or 'clickedtext' must match this property to cause the button to be checked when the write occurs.

Good example: formatting set to display a data value of '1' as 'On', clickCheckedText is 'On', clickText is 'On'. In this example, the push button will be checked when a data update occurs with a value of 1 or when the button is clicked.

Bad example: formatting set to display a data value of '1' as 'On', clickCheckedText is 'On', clickText is '1'. In this example, the push button will be checked when a data update occurs with a value of 1 but, although a valid value will be written when clicked, the button will not be checked when clicked as '1' is not the same as 'On'.

Reimplemented from [QEGenericButton](#).

#### 9.82.4.7 QString QERadioButton::clickText [read, write]

Value written when user clicks button if 'writeOnClick' property is true

Reimplemented from [QEGenericButton](#).

#### 9.82.4.8 bool QERadioButton::confirmAction [read, write]

If true, a dialog will be presented asking the user to confirm if the button action should be carried out

#### 9.82.4.9 CreationOptionNames QERadioButton::creationOption [read, write]

Creation options when opening a new GUI. Open a new window, open a new tab, or replace the current window. the creation option is supplied when the button generates a newGui signal. Application code connected to this signal should honour this request if possible. When used within the QEGui application, the QEGui application creates a new window, new tab, or replaces the current window as appropriate.

Reimplemented from [QEGenericButton](#).

#### 9.82.4.10 bool QERadioButton::enabled [read, write]

Set the preferred 'enabled' state. Default is true. This property is copied to the standard Qt 'enabled' property if the data being displayed is valid. If the data being displayed is invalid the standard Qt 'enabled' property will always be set to false to indicate invalid data. The value of this property will only be copied to the standard Qt 'enabled' property once data is valid.

#### 9.82.4.11 Formats QERadioButton::format [read, write]

Format to apply to data. Default is 'Default' in which case the data type supplied with the data determines how the data is formatted. For all other options, an attempt is made to format the data as requested (whatever its native form).

#### 9.82.4.12 QString QERadioButton::guiFile [read, write]

File name of GUI to be presented on button click. File name can be absolute, relative to the path of the QEform in which the [QEPushButton](#) is located, relative to the any path in the path list published in the [ContainerProfile](#) class, or relative to the current path. See [QEWidget::openQEFile\(\)](#) in QEWidget.cpp for details.

#### 9.82.4.13 unsigned QERadioButton::int [read, write]

Set the ID used by the message filtering system. Default is zero. Widgets or applications that use messages from the framework have the option of filtering on this ID. For example, by using a unique message source ID a [QELog](#) widget may be set up to only log messages from a select set of widgets.

Base used for when formatting integers. Default is 10 (duh!)

Index used to select a single item of data for formatting from an array of data. Default is 0. Only used when the arrayAction property is INDEX. Refer to the arrayAction property for more details.

#### 9.82.4.14 QString QERadioButton::labelText [read, write]

Button label text (prior to substitution). Macro substitutions will be applied to this text and the result will be set as the button text. Used when data updates are not being represented in the button text. For example, a button in a sub form may have a 'labelText' property of 'Turn Pump On'. When the sub form is used twice in a main form with substitutions PUMPNUM=1 and PUMPNUM=2 respectively, the two identical buttons in the sub forms will have the labels 'Turn Pump 1 On' and 'Turn Pump 2 On' respectively.

Reimplemented from [QEGenericButton](#).

#### 9.82.4.15 bool QERadioButton::leadingZero [read, write]

If true (default), always add a leading zero when formatting numbers.

#### 9.82.4.16 QString QERadioButton::localEnumeration [read, write]

An enumeration list used to data values. Used only when the formatting option is 'local enumeration'. Value is converted to an integer and used to select a string from this list.

Format is:

```
[[<|<=|!=|>|=|>]value1|*]: string1 , [[<|<=|!=|>|=|>]value2|*]: string2 , [[<|<=|!=|>|=|>]value3|*]: string3 , ...
```

Where: < Less than <= Less than or equal = Equal (default if no operator specified)  
 >= Greather than or equal > Greater than Always match (used to specify default text)

Values may be numeric or textual Values do not have to be in any order, but first match wins Values may be quoted Strings may be quoted Consecutive values do not have to be present. Operator is assumed to be equality if not present. White space is ignored except within quoted strings.

may be included in a string to indicate a line break

Examples are:

```
0:Off,1:On 0 : "Pump Running", 1 : "Pump not running" 0:"", 1:"Warning!\nAlarm"
<2:"Value is less than two", =2:"Value is equal to two", >2:"Value is grater than 2"
3:"Beamline Available", *: "" "Pump Off": "OH NO!, the pump is OFF!", "Pump On": "It's OK, the pump is on"
```

The data value is converted to a string if no enumeration for that value is available. For example, if the local enumeration is '0:off,1:on', and a value of 10 is processed, the text generated is '10'. If a blank string is required, this should be explicit. for example, '0:off,1:on,10: ""'

A range of numbers can be covered by a pair of values as in the following example:  
 >=4:"Between 4 and 8", <=8:"Between 4 and 8"

**9.82.4.17 Notations QERadioButton::notation** [read, write]

Notation used for numerical formatting. Default is fixed.

**9.82.4.18 QString QERadioButton::password** [read, write]

Password user will need to enter before any action is taken

Reimplemented from [QEGenericButton](#).

**9.82.4.19 QPixmap QERadioButton::pixmap0** [read, write]

Pixmap to display if updateOption is Icon or TextAndIcon and data value translates to an index of 0

**9.82.4.20 QPixmap QERadioButton::pixmap1** [read, write]

Pixmap to display if updateOption is Icon or TextAndIcon and data value translates to an index of 1

**9.82.4.21 QPixmap QERadioButton::pixmap2** [read, write]

Pixmap to display if updateOption is Icon or TextAndIcon and data value translates to an index of 2

**9.82.4.22 QPixmap QERadioButton::pixmap3** [read, write]

Pixmap to display if updateOption is Icon or TextAndIcon and data value translates to an index of 3

**9.82.4.23 QPixmap QERadioButton::pixmap4** [read, write]

Pixmap to display if updateOption is Icon or TextAndIcon and data value translates to an index of 4

**9.82.4.24 QPixmap QERadioButton::pixmap5** [read, write]

Pixmap to display if updateOption is Icon or TextAndIcon and data value translates to an index of 5

**9.82.4.25 QPixmap QERadioButton::pixmap6** [read, write]

Pixmap to display if updateOption is Icon or TextAndIcon and data value translates to an index of 6

**9.82.4.26 QPixmap QERadioButton::pixmap7** [read, write]

Pixmap to display if updateOption is Icon or TextAndIcon and data value translates to an index of 7

**9.82.4.27 int QERadioButton::precision** [read, write]

Precision used when formatting floating point numbers. The default is 4. This is only used if useDbPrecision is false.

**9.82.4.28 QString QERadioButton::pressText** [read, write]

Value written when user presses button if 'writeOnPress' property is true

Reimplemented from [QEGenericButton](#).

**9.82.4.29 QString QERadioButton::program** [read, write]

Program to run when the button is clicked. No attempt to run a program is made if this property is empty. Example: firefox

Reimplemented from [QEGenericButton](#).

**9.82.4.30 QString QERadioButton::releaseText** [read, write]

Value written when user releases button if 'writeOnRelease' property is true

Reimplemented from [QEGenericButton](#).

**9.82.4.31 bool QERadioButton::subscribe** [read, write]

Sets if this widget subscribes for data updates and displays current data. Default is 'true' (subscribes for and displays data updates)

Reimplemented from [QEWidget](#).

**9.82.4.32 bool QERadioButton::trailingZeros** [read, write]

If true (default), always remove any trailing zeros when formatting numbers.

**9.82.4.33 UpdateOptions QERadioButton::updateOption** [read, write]

Update options (text, pixmap, both, or state (checked or unchecked))

Reimplemented from [QEGenericButton](#).



**9.82.4.34** `bool QERadioButton::useDbPrecision` [read, write]

If true (default), format floating point numbers using the precision supplied with the data. If false, the precision property is used.

**9.82.4.35** `UserLevels QERadioButton::userLevelEnabled` [read, write]

Lowest user level at which the widget is enabled. Default is 'User'. Used when designing GUIs that allow access to more and more detail according to the user mode. The user mode is set application wide through the [QELogin](#) widget, or programatically through `setUserLevel()` Widgets that are always accessible should be visible at 'User'. Widgets that are only accessible to scientists managing the facility should be visible at 'Scientist'. Widgets that are only accessible to engineers maintaining the facility should be visible at 'Engineer'.

**9.82.4.36** `QString QERadioButton::userLevelEngineerStyle` [read, write]

Style Sheet string to be applied when the widget is displayed in 'Engineer' mode. Default is an empty string. The syntax is the standard Qt Style Sheet syntax. For example, 'background-color: red' This Style Sheet string will be applied by the `styleManager` class. Refer to the `styleManager` class for details about how this Style Sheet string will be merged with any pre-existing Style Sheet string and any Style Sheet strings generated during the display of data.

**9.82.4.37** `QString QERadioButton::userLevelScientistStyle` [read, write]

Style Sheet string to be applied when the widget is displayed in 'Scientist' mode. Default is an empty string. The syntax is the standard Qt Style Sheet syntax. For example, 'background-color: red' This Style Sheet string will be applied by the `styleManager` class. Refer to the `styleManager` class for details about how this Style Sheet string will be merged with any pre-existing Style Sheet string and any Style Sheet strings generated during the display of data.

**9.82.4.38** `QString QERadioButton::userLevelUserStyle` [read, write]

Style Sheet string to be applied when the widget is displayed in 'User' mode. Default is an empty string. The syntax is the standard Qt Style Sheet syntax. For example, 'background-color: red' This Style Sheet string will be applied by the `styleManager` class. Refer to the `styleManager` class for details about how this Style Sheet string will be merged with any pre-existing Style Sheet string and any Style Sheet strings generated during the display of data.

**9.82.4.39** `UserLevels QERadioButton::userLevelVisibility` [read, write]

Lowest user level at which the widget is visible. Default is 'User'. Used when designing GUIs that display more and more detail according to the user mode. The user mode is

set application wide through the [QELogin](#) widget, or programatically through `setUserLevel()`. Widgets that are always visible should be visible at 'User'. Widgets that are only used by scientists managing the facility should be visible at 'Scientist'. Widgets that are only used by engineers maintaining the facility should be visible at 'Engineer'.

#### 9.82.4.40 `QString QERadioButton::variable` [read, write]

EPICS variable name (CA PV)

#### 9.82.4.41 `bool QERadioButton::variableAsToolTip` [read, write]

Use the variable as the tool tip. Default is true. Tool tip property will be overwritten by the variable name.

Reimplemented from [QEToolTip](#).

#### 9.82.4.42 `QString QERadioButton::variableSubstitutions` [read, write]

Macro substitutions. The default is no substitutions. The format is `NAME1=VALUE1[, NAME2=VALUE2...`. Values may be quoted strings. For example, `'PUMP=PMP3, NAME = "My Pump"'`. These substitutions are applied to variable names for all QE widgets. In some widgets are also used for other purposes.

#### 9.82.4.43 `bool QERadioButton::visible` [read, write]

Display the widget. Default is true. Setting this property false is usefull if widget is only used to provide a signal - for example, when supplying data to a [QELink](#) widget. Note, when false the widget will still be visible in Qt Designer.

#### 9.82.4.44 `bool QERadioButton::writeOnClick` [read, write]

If true, the 'clickText' property is written when the button is clicked. Default is true

Reimplemented from [QEGenericButton](#).

#### 9.82.4.45 `bool QERadioButton::writeOnPress` [read, write]

If true, the 'pressText' property is written when the button is pressed. Default is false

Reimplemented from [QEGenericButton](#).

#### 9.82.4.46 `bool QERadioButton::writeOnRelease` [read, write]

If true, the 'releaseText' property is written when the button is released. Default is false

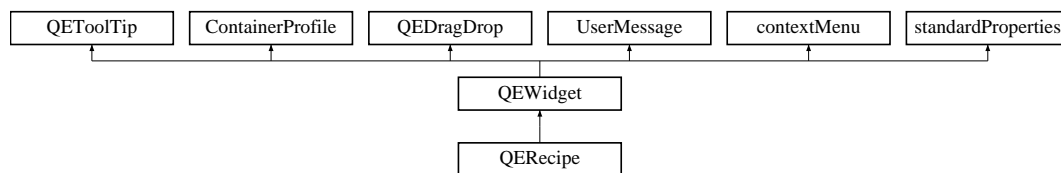
Reimplemented from [QEGenericButton](#).

The documentation for this class was generated from the following files:

- /home/rhydera/epicsqt/framework/widgets/QEButton/QERadioButton.h
- /home/rhydera/epicsqt/framework/widgets/QEButton/QERadioButton.cpp

## 9.83 QERecipe Class Reference

Inheritance diagram for QERecipe:



### Public Types

- enum **configurationTypesProperty** { **File** = FROM\_FILE, **Text** = FROM\_TEXT }
- enum **detailsLayoutProperty** { **Top** = TOP, **Bottom** = BOTTOM, **Left** = LEFT, **Right** = RIGHT }
- enum **userTypesProperty** { **User** = USERLEVEL\_USER, **Scientist** = USERLEVEL\_SCIENTIST, **Engineer** = USERLEVEL\_ENGINEER }

### Public Member Functions

- **QERecipe** (QWidget \*pParent=0)
- void **setRecipeDescription** (QString pValue)
- QString **getRecipeDescription** ()
- void **setShowRecipeList** (bool pValue)
- bool **getShowRecipeList** ()
- void **setShowNew** (bool pValue)
- bool **getShowNew** ()
- void **setShowSave** (bool pValue)
- bool **getShowSave** ()
- void **setShowDelete** (bool pValue)
- bool **getShowDelete** ()
- void **setShowApply** (bool pValue)
- bool **getShowApply** ()
- void **setShowRead** (bool pValue)
- bool **getShowRead** ()
- void **setShowFields** (bool pValue)
- bool **getShowFields** ()
- void **setConfigurationType** (int pValue)

- int **getConfigurationType** ()
- void **setConfigurationFile** (QString pValue)
- QString **getConfigurationFile** ()
- void **setRecipeFile** (QString pValue)
- QString **getRecipeFile** ()
- void **setConfigurationText** (QString pValue)
- QString **getConfigurationText** ()
- void **setDetailsLayout** (int pValue)
- int **getDetailsLayout** ()
- void **setCurrentUserType** (int pValue)
- int **getCurrentUserType** ()
- bool **saveRecipeList** ()
- void **refreshRecipeList** ()
- void **refreshButton** ()
- void **userLevelChanged** (userLevels pValue)
- void **setConfigurationTypeProperty** (configurationTypesProperty pConfigurationType)
- configurationTypesProperty **getConfigurationTypeProperty** ()
- void **setDetailsLayoutProperty** (detailsLayoutProperty pDetailsLayout)
- detailsLayoutProperty **getDetailsLayoutProperty** ()
- void **setCurrentUserTypeProperty** (userTypesProperty pUserType)
- userTypesProperty **getCurrentUserTypeProperty** ()

### Protected Attributes

- QLabel \* **qLabelRecipeDescription**
- QComboBox \* **qComboBoxRecipeList**
- QPushButton \* **qPushButtonNew**
- QPushButton \* **qPushButtonSave**
- QPushButton \* **qPushButtonDelete**
- QPushButton \* **qPushButtonApply**
- QPushButton \* **qPushButtonRead**
- [QEConfiguredLayout](#) \* **qEConfiguredLayoutRecipeFields**
- QDomDocument **document**
- QString **recipeFile**
- QString **filename**
- int **detailsLayout**
- int **currentUserType**

### Properties

- QString **recipeDescription**
- bool **showRecipeList**
- bool **showNew**
- bool **showSave**

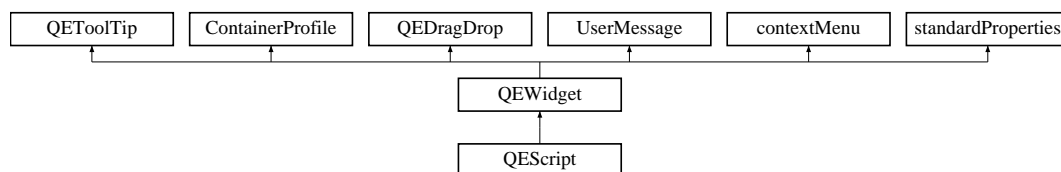
- bool **showDelete**
- bool **showApply**
- bool **showRead**
- bool **showFields**
- configurationTypesProperty **configurationType**
- QString **configurationFile**
- QString **configurationText**
- detailsLayoutProperty **detailsLayout**
- userTypesProperty **currentUserType**

The documentation for this class was generated from the following files:

- /home/rhydera/epicsqt/framework/widgets/QERecipe/QERecipe.h
- /home/rhydera/epicsqt/framework/widgets/QERecipe/QERecipe.cpp

## 9.84 QEScript Class Reference

Inheritance diagram for QEScript:



### Public Types

- enum **detailsLayoutProperty** { **Top** = TOP, **Bottom** = BOTTOM, **Left** = LEFT, **Right** = RIGHT }

### Signals

- void **selected** (QString pFilename)

### Public Member Functions

- **QEScript** (QWidget \*pParent=0)
- void **setDirectoryPath** (QString pValue)
- QString **getDirectoryPath** ()
- void **setShowDirectoryPath** (bool pValue)
- bool **getShowDirectoryPath** ()
- void **setShowDirectoryBrowser** (bool pValue)
- bool **getShowDirectoryBrowser** ()

- void **setShowRefresh** (bool pValue)
- bool **getShowRefresh** ()
- void **setShowColumnTime** (bool pValue)
- bool **getShowColumnTime** ()
- void **setShowColumnSize** (bool pValue)
- bool **getShowColumnSize** ()
- void **setShowColumnFilename** (bool pValue)
- bool **getShowColumnFilename** ()
- void **setShowFileExtension** (bool pValue)
- bool **getShowFileExtension** ()
- void **setFileFilter** (QString pValue)
- QString **getFileFilter** ()
- void **setDetailsLayout** (int pValue)
- int **getDetailsLayout** ()
- void **updateTable** ()
- void **setDetailsLayoutProperty** (detailsLayoutProperty pDetailsLayout)
- detailsLayoutProperty **getDetailsLayoutProperty** ()

### Protected Attributes

- QLineEdit \* **qlineEditDirectoryPath**
- QPushButton \* **qPushButtonDirectoryBrowser**
- QPushButton \* **qPushButtonRefresh**
- [\\_QTableWidgetScript](#) \* **qTableWidgetScript**
- QString **fileFilter**
- bool **showFileExtension**
- int **detailsLayout**

### Properties

- QString **directoryPath**
- bool **showDirectoryPath**
- bool **showDirectoryBrowser**
- bool **showRefresh**
- bool **showColumnTime**
- bool **showColumnSize**
- bool **showColumnFilename**
- detailsLayoutProperty **detailsLayout**

The documentation for this class was generated from the following files:

- /home/rhydera/epicsqt/framework/widgets/QEScript/QEScript.h
- /home/rhydera/epicsqt/framework/widgets/QEScript/QEScript.cpp

```
#include <QEShape.h>
```

```

graph BT
    QEShape[QEShape] --> QEWidget[QEWidget]
    QEWidget --> QEToolTip[QEToolTip]
    QEWidget --> ContainerProfile[ContainerProfile]
    QEWidget --> QEDragDrop[QEDragDrop]
    QEWidget --> UserMessage[UserMessage]
    QEWidget --> contextMenu[contextMenu]
    QEWidget --> standardProperties[standardProperties]

```

- enum `shapeOptions` {  
**Line, Points, Polyline, Polygon,**  
**Rect, RoundedRect, Ellipse, Arc,**  
**Chord, Pie, Path, Text,**  
**Pixmap }**
- enum `animationOptions` {  
**Width, Height, X, Y,**  
**Transparency, Rotation, ColourHue, ColourSaturation,**  
**ColourValue, ColourIndex, Penwidth }**
- enum `UserLevels` { `User` = USERLEVEL\_USER, `Scientist` = USERLEVEL\_SCIENTIST,  
`Engineer` = USERLEVEL\_ENGINEER }

- void **requestEnabled** (const bool &state)

- void [dbValueChanged1](#) (const qulonglong &out)
- void [dbValueChanged2](#) (const qulonglong &out)
- void [dbValueChanged3](#) (const qulonglong &out)
- void [dbValueChanged4](#) (const qulonglong &out)
- void [dbValueChanged5](#) (const qulonglong &out)
- void [dbValueChanged6](#) (const qulonglong &out)

## Public Member Functions

- [QEShape](#) (QWidget \*parent=0)
- [QEShape](#) (const QString &variableName, QWidget \*parent=0)
- void [setAnimation](#) ([animationOptions](#) animation, const int index)  
*Access function for #animation' properties - refer to animation' properties for details.*
- [animationOptions](#) [getAnimation](#) (const int index)  
*Access function for #animation' properties - refer to animation' properties for details.*
- void [setScale](#) (const double scale, const int index)  
*Access function for #scale' properties - refer to scale' properties for details.*
- double [getScale](#) (const int index)  
*Access function for #scale' properties - refer to scale' properties for details.*
- void [setOffset](#) (const double offset, const int index)  
*Access function for #offset' properties - refer to offset' properties for details.*
- double [getOffset](#) (const int index)  
*Access function for #offset' properties - refer to offset' properties for details.*
- void [setBorder](#) (const bool border)  
*Access function for #border' properties - refer to border' properties for details.*
- bool [getBorder](#) ()  
*Access function for #border' properties - refer to border' properties for details.*
- void [setFill](#) (const bool fill)  
*Access function for #fill' properties - refer to fill' properties for details.*
- bool [getFill](#) ()  
*Access function for #fill' properties - refer to fill' properties for details.*
- void [setShape](#) ([shapeOptions](#) shape)  
*Access function for #shape' properties - refer to shape' properties for details.*
- [shapeOptions](#) [getShape](#) ()  
*Access function for #shape' properties - refer to shape' properties for details.*
- void [setNumPoints](#) (const unsigned int numPoints)  
*Access function for #number of points' properties - refer to number of points' properties for details.*
- unsigned int [getNumPoints](#) ()  
*Access function for #number of points' properties - refer to number of points' properties for details.*
- void [setOriginTranslation](#) (const QPoint originTranslation)  
*Access function for #origin translation' properties - refer to origin translation' properties for details.*
- QPoint [getOriginTranslation](#) ()  
*Access function for #origin translation' properties - refer to origin translation' properties for details.*
- void [setPoint](#) (const QPoint point, const int index)  
*Access function for #point' properties - refer to point' properties for details.*
- QPoint [getPoint](#) (const int index)  
*Access function for #point' properties - refer to point' properties for details.*



- void [setColor](#) (const QColor color, const int index)  
*Access function for #colour' properties - refer to colour' properties for details.*
- QColor [getColor](#) (const int index)  
*Access function for #colour' properties - refer to colour' properties for details.*
- void [setDrawBorder](#) (const bool drawBorder)  
*Access function for #draw border' properties - refer to draw border' properties for details.*
- bool [getDrawBorder](#) ()  
*Access function for #draw border' properties - refer to draw border' properties for details.*
- void [setLineWidth](#) (const unsigned int lineWidth)  
*Access function for #line width' properties - refer to line width' properties for details.*
- unsigned int [getLineWidth](#) ()  
*Access function for #line width' properties - refer to line width' properties for details.*
- void [setStartAngle](#) (const double startAngle)  
*Access function for #start angle' properties - refer to start angle' properties for details.*
- double [getStartAngle](#) ()  
*Access function for #start angle' properties - refer to start angle' properties for details.*
- void [setRotation](#) (const double rotation)  
*Access function for #rotation' properties - refer to rotation' properties for details.*
- double [getRotation](#) ()  
*Access function for #rotation' properties - refer to rotation' properties for details.*
- void [setArcLength](#) (const double arcLength)  
*Access function for #arc length' properties - refer to arc length' properties for details.*
- double [getArcLength](#) ()  
*Access function for #arc length' properties - refer to arc length' properties for details.*
- void [setText](#) (const QString text)  
*Access function for #text' properties - refer to text' properties for details.*
- QString [getText](#) ()  
*Access function for #text' properties - refer to text' properties for details.*
- bool [isEnabled](#) () const  
*Access function for [enabled](#) property - refer to [enabled](#) property for details.*
- void [setEnabled](#) (bool state)  
*Access function for [enabled](#) property - refer to [enabled](#) property for details.*
- UserLevels [getUserLevelVisibilityProperty](#) ()  
*Access function for [userLevelVisibility](#) property - refer to [userLevelVisibility](#) property for details.*
- void [setUserLevelVisibilityProperty](#) (UserLevels level)  
*Access function for [userLevelVisibility](#) property - refer to [userLevelVisibility](#) property for details.*
- UserLevels [getUserLevelEnabledProperty](#) ()  
*Access function for [userLevelEnabled](#) property - refer to [userLevelEnabled](#) property for details.*
- void [setUserLevelEnabledProperty](#) (UserLevels level)  
*Access function for [userLevelEnabled](#) property - refer to [userLevelEnabled](#) property for details.*

## Properties

- QString [variable1](#)
- QString [variable2](#)
- QString [variable3](#)
- QString [variable4](#)
- QString [variable5](#)
- QString [variable6](#)
- QString [variableSubstitutions](#)
- bool [variableAsToolTip](#)
- bool [enabled](#)
- bool [allowDrop](#)
- bool [visible](#)
- unsigned int
- QString [userLevelUserStyle](#)
- QString [userLevelScientistStyle](#)
- QString [userLevelEngineerStyle](#)
- [UserLevels](#) [userLevelVisibility](#)
- [UserLevels](#) [userLevelEnabled](#)
- [animationOptions](#) [animation1](#)
- [animationOptions](#) [animation2](#)
- [animationOptions](#) [animation3](#)
- [animationOptions](#) [animation4](#)
- [animationOptions](#) [animation5](#)
- [animationOptions](#) [animation6](#)
- double [scale1](#)

*Scale factor applied to data from the 1st variable before it is used to animate the shape.*

- double [scale2](#)
- double [scale3](#)
- double [scale4](#)
- double [scale5](#)
- double [scale6](#)
- double [offset1](#)
- double [offset2](#)
- double [offset3](#)
- double [offset4](#)
- double [offset5](#)
- double [offset6](#)
- QPoint [point1](#)
- QPoint [point2](#)
- QPoint [point3](#)
- QPoint [point4](#)
- QPoint [point5](#)
- QPoint [point6](#)
- QPoint [point7](#)
- QPoint [point8](#)

- QPoint [point9](#)
- QPoint [point10](#)
- QColor [color1](#)
- QColor [color2](#)
- QColor [color3](#)
- QColor [color4](#)
- QColor [color5](#)
- QColor [color6](#)
- QColor [color7](#)
- QColor [color8](#)
- QColor [color9](#)
- QColor [color10](#)

### 9.85.1 Detailed Description

This class is a EPICS aware shape widget based on the Qt widget. One of several shapes can be drawn within the widget, and up to 6 variables can be used to animate various attributes of the shape. For example to represent beam positino and size, an ellipse can be drawn with four variables animating its vertcal and horizontal size and position. It is tightly integrated with the base class [QEWidget](#) which provides generic support such as macro substitutions, drag/drop, and standard properties.

### 9.85.2 Member Enumeration Documentation

#### 9.85.2.1 enum QEShape::animationOptions

Options for how a variable will animate the shape.

#### 9.85.2.2 enum QEShape::shapeOptions

Options for the type of shape.

#### 9.85.2.3 enum QEShape::UserLevels

User friendly enumerations for [userLevelVisibility](#) and [userLevelEnabled](#) properties - refer to [userLevelVisibility](#) and [userLevelEnabled](#) properties and userLevel enumeration for details.

#### Enumerator:

**User** Refer to USERLEVEL\_USER for details.

**Scientist** Refer to USERLEVEL\_SCIENTIST for details.

**Engineer** Refer to USERLEVEL\_ENGINEER for details.

### 9.85.3 Constructor & Destructor Documentation

#### 9.85.3.1 QEShape::QEShape ( QWidget \* *parent* = 0 )

Create without a variable. Use `setVariableNameProperty()` and `setSubstitutionsProperty()` to define a variable and, optionally, macro substitutions later.

#### 9.85.3.2 QEShape::QEShape ( const QString & *variableName*, QWidget \* *parent* = 0 )

Create with a single variable. (Note, the [QEShape](#) widget can use up to 6 variables) A connection is automatically established. If macro substitutions are required, create without a variable and set the variable and macro substitutions after creation.

### 9.85.4 Member Function Documentation

#### 9.85.4.1 void QEShape::dbValueChanged1 ( const qlonglong & *out* ) [signal]

Sent when the widget is updated following a data change for the first variable Can be used to pass on EPICS data (as presented in this widget) to other widgets. For example a QList widget could log updates from this widget.

#### 9.85.4.2 void QEShape::dbValueChanged2 ( const qlonglong & *out* ) [signal]

Sent when the widget is updated following a data change for the second variable Can be used to pass on EPICS data (as presented in this widget) to other widgets. For example a QList widget could log updates from this widget.

#### 9.85.4.3 void QEShape::dbValueChanged3 ( const qlonglong & *out* ) [signal]

Sent when the widget is updated following a data change for the third variable Can be used to pass on EPICS data (as presented in this widget) to other widgets. For example a QList widget could log updates from this widget.

#### 9.85.4.4 void QEShape::dbValueChanged4 ( const qlonglong & *out* ) [signal]

Sent when the widget is updated following a data change for the fourth variable Can be used to pass on EPICS data (as presented in this widget) to other widgets. For example a QList widget could log updates from this widget.

#### 9.85.4.5 void QEShape::dbValueChanged5 ( const qlonglong & *out* ) [signal]

Sent when the widget is updated following a data change for the fifth variable Can be used to pass on EPICS data (as presented in this widget) to other widgets. For example a QList widget could log updates from this widget.

**9.85.4.6** void QEShape::dbValueChanged6 ( const qlonglong & *out* ) [signal]

Sent when the widget is updated following a data change for the sixth variable Can be used to pass on EPICS data (as presented in this widget) to other widgets. For example a QList widget could log updates from this widget.

**9.85.4.7** void QEShape::requestEnabled ( const bool & *state* ) [inline, slot]

Similar to standard setEnabled slot, but allows QE widget to determine if the widget remains disabled due to invalid data. If disabled due to invalid data, a request to enable the widget will be honoured when the data is no longer invalid.

## 9.85.5 Property Documentation

**9.85.5.1** bool QEShape::allowDrop [read, write]

Allow drag/drops operations to this widget. Default is false. Any dropped text will be used as a new variable name.

Reimplemented from [QEDragDrop](#).

**9.85.5.2** animationOptions QEShape::animation1 [read, write]

Animation to be effected by the 1st variable. This is used to select what the effect changing data for the 1st variable will have on the shape.

**9.85.5.3** animationOptions QEShape::animation2 [read, write]

Animation to be effected by the 2nd variable. This is used to select what the effect changing data for the 2nd variable will have on the shape.

**9.85.5.4** animationOptions QEShape::animation3 [read, write]

Animation to be effected by the 3rd variable. This is used to select what the effect changing data for the 3rd variable will have on the shape.

**9.85.5.5** animationOptions QEShape::animation4 [read, write]

Animation to be effected by the 4th variable. This is used to select what the effect changing data for the 4th variable will have on the shape.

**9.85.5.6** animationOptions QEShape::animation5 [read, write]

Animation to be effected by the 5th variable. This is used to select what the effect changing data for the 5th variable will have on the shape.

**9.85.5.7 animationOptions QEShape::animation6** [read, write]

Animation to be effected by the 6th variable. This is used to select what the effect changing data for the 6th variable will have on the shape.

**9.85.5.8 QColor QEShape::color1** [read, write]

Used by the color animation to determine the color based on a data value. The scaled and offset data is used as an index to select color properties 'color1' to 'color10'.

**9.85.5.9 QColor QEShape::color10** [read, write]

Used by the color animation to determine the color based on a data value. The scaled and offset data is used as an index to select color properties 'color1' to 'color10'.

**9.85.5.10 QColor QEShape::color2** [read, write]

Used by the color animation to determine the color based on a data value. The scaled and offset data is used as an index to select color properties 'color1' to 'color10'.

**9.85.5.11 QColor QEShape::color3** [read, write]

Used by the color animation to determine the color based on a data value. The scaled and offset data is used as an index to select color properties 'color1' to 'color10'.

**9.85.5.12 QColor QEShape::color4** [read, write]

Used by the color animation to determine the color based on a data value. The scaled and offset data is used as an index to select color properties 'color1' to 'color10'.

**9.85.5.13 QColor QEShape::color5** [read, write]

Used by the color animation to determine the color based on a data value. The scaled and offset data is used as an index to select color properties 'color1' to 'color10'.

**9.85.5.14 QColor QEShape::color6** [read, write]

Used by the color animation to determine the color based on a data value. The scaled and offset data is used as an index to select color properties 'color1' to 'color10'.

**9.85.5.15 QColor QEShape::color7** [read, write]

Used by the color animation to determine the color based on a data value. The scaled and offset data is used as an index to select color properties 'color1' to 'color10'.

**9.85.5.16 QColor QEShape::color8** [read, write]

Used by the color animation to determine the color based on a data value. The scaled and offset data is used as an index to select color properties 'color1' to 'color10'.

**9.85.5.17 QColor QEShape::color9** [read, write]

Used by the color animation to determine the color based on a data value. The scaled and offset data is used as an index to select color properties 'color1' to 'color10'.

**9.85.5.18 bool QEShape::enabled** [read, write]

Set the preferred 'enabled' state. Default is true. This property is copied to the standard Qt 'enabled' property if the data being displayed is valid. If the data being displayed is invalid the standard Qt 'enabled' property will always be set to false to indicate invalid data. The value of this property will only be copied to the standard Qt 'enabled' property once data is valid.

**9.85.5.19 unsigned QEShape::int** [read, write]

Set the ID used by the message filtering system. Default is zero. Widgets or applications that use messages from the framework have the option of filtering on this ID. For example, by using a unique message source ID a [QELog](#) widget may be set up to only log messages from a select set of widgets.

The number of points to use when drawing shapes that are defined by a variable number of points, such as polyline, polygon, path, and series of points.

Sets the width of the pen. Used for the following shapes: Line, Points, Polyline, Polygon, Rect, RoundedRect, Ellipse, Arc, Chord, Pie, Path

**9.85.5.20 double QEShape::offset1** [read, write]

Offset applied to data from the 1st variable before it is used to animate the shape

**9.85.5.21 double QEShape::offset2** [read, write]

Offset applied to data from the 2nd variable before it is used to animate the shape

**9.85.5.22 double QEShape::offset3** [read, write]

Offset applied to data from the 3rd variable before it is used to animate the shape

**9.85.5.23** `double QEShape::offset4` [read, write]

Offset applied to data from the 4th variable before it is used to animate the shape

**9.85.5.24** `double QEShape::offset5` [read, write]

Offset applied to data from the 5th variable before it is used to animate the shape

**9.85.5.25** `double QEShape::offset6` [read, write]

Offset applied to data from the 6th variable before it is used to animate the shape

**9.85.5.26** `QPoint QEShape::point1` [read, write]

1st coordinate used when drawing the shape. Used for the following shapes: Line, Points, Polyline, Polygon, Rect, RoundedRect, Ellipse, Arc, Chord, Pie, Path, Text, QPixmap

**9.85.5.27** `QPoint QEShape::point10` [read, write]

10th coordinate used when drawing the shape. Used for the following shapes: Points, Polyline, Polygon, Path

**9.85.5.28** `QPoint QEShape::point2` [read, write]

2nd coordinate used when drawing the shape. Used for the following shapes: Line, Points, Polyline, Polygon, Rect, RoundedRect, Ellipse, Arc, Chord, Pie, Path, QPixmap

**9.85.5.29** `QPoint QEShape::point3` [read, write]

3rd coordinate used when drawing the shape. Used for the following shapes: Points, Polyline, Polygon, Path

**9.85.5.30** `QPoint QEShape::point4` [read, write]

4th coordinate used when drawing the shape. Used for the following shapes: Points, Polyline, Polygon, Path

**9.85.5.31** `QPoint QEShape::point5` [read, write]

5th coordinate used when drawing the shape. Used for the following shapes: Points, Polyline, Polygon, Path



**9.85.5.32** `QPoint QEShape::point6` [read, write]

6th coordinate used when drawing the shape. Used for the following shapes: Points, Polyline, Polygon, Path

**9.85.5.33** `QPoint QEShape::point7` [read, write]

7th coordinate used when drawing the shape. Used for the following shapes: Points, Polyline, Polygon, Path

**9.85.5.34** `QPoint QEShape::point8` [read, write]

8th coordinate used when drawing the shape. Used for the following shapes: Points, Polyline, Polygon, Path

**9.85.5.35** `QPoint QEShape::point9` [read, write]

9th coordinate used when drawing the shape. Used for the following shapes: Points, Polyline, Polygon, Path

**9.85.5.36** `double QEShape::scale2` [read, write]

Scale factor applied to data from the 2nd variable before it is used to animate the shape

**9.85.5.37** `double QEShape::scale3` [read, write]

Scale factor applied to data from the 3rd variable before it is used to animate the shape

**9.85.5.38** `double QEShape::scale4` [read, write]

Scale factor applied to data from the 4th variable before it is used to animate the shape

**9.85.5.39** `double QEShape::scale5` [read, write]

Scale factor applied to data from the 5th variable before it is used to animate the shape

**9.85.5.40** `double QEShape::scale6` [read, write]

Scale factor applied to data from the 6th variable before it is used to animate the shape

#### 9.85.5.41 UserLevels QEShape::userLevelEnabled [read, write]

Lowest user level at which the widget is enabled. Default is 'User'. Used when designing GUIs that allow access to more and more detail according to the user mode. The user mode is set application wide through the [QELogin](#) widget, or programatically through `setUserLevel()` Widgets that are always accessible should be visible at 'User'. Widgets that are only accessible to scientists managing the facility should be visible at 'Scientist'. Widgets that are only accessible to engineers maintaining the facility should be visible at 'Engineer'.

#### 9.85.5.42 QString QEShape::userLevelEngineerStyle [read, write]

Style Sheet string to be applied when the widget is displayed in 'Engineer' mode. Default is an empty string. The syntax is the standard Qt Style Sheet syntax. For example, 'background-color: red' This Style Sheet string will be applied by the styleManager class. Refer to the styleManager class for details about how this Style Sheet string will be merged with any pre-existing Style Sheet string and any Style Sheet strings generated during the display of data.

#### 9.85.5.43 QString QEShape::userLevelScientistStyle [read, write]

Style Sheet string to be applied when the widget is displayed in 'Scientist' mode. Default is an empty string. The syntax is the standard Qt Style Sheet syntax. For example, 'background-color: red' This Style Sheet string will be applied by the styleManager class. Refer to the styleManager class for details about how this Style Sheet string will be merged with any pre-existing Style Sheet string and any Style Sheet strings generated during the display of data.

#### 9.85.5.44 QString QEShape::userLevelUserStyle [read, write]

Style Sheet string to be applied when the widget is displayed in 'User' mode. Default is an empty string. The syntax is the standard Qt Style Sheet syntax. For example, 'background-color: red' This Style Sheet string will be applied by the styleManager class. Refer to the styleManager class for details about how this Style Sheet string will be merged with any pre-existing Style Sheet string and any Style Sheet strings generated during the display of data.

#### 9.85.5.45 UserLevels QEShape::userLevelVisibility [read, write]

Lowest user level at which the widget is visible. Default is 'User'. Used when designing GUIs that display more and more detail according to the user mode. The user mode is set application wide through the [QELogin](#) widget, or programatically through `setUserLevel()` Widgets that are always visible should be visible at 'User'. Widgets that are only used by scientists managing the facility should be visible at 'Scientist'. Widgets that are only used by engineers maintaining the facility should be visible at 'Engineer'.

**9.85.5.46** `QString QEShape::variable1` `[read, write]`

EPICS variable name (CA PV). This variable is read and used to animate an attribute of the shape. The value read is first scaled and offset by properties `scale1` and `offset1` then the attribute selected for animation is selected by the property `animation1`.

**9.85.5.47** `QString QEShape::variable2` `[read, write]`

EPICS variable name (CA PV). This variable is read and used to animate an attribute of the shape. The value read is first scaled and offset by properties `scale2` and `offset2` then the attribute selected for animation is selected by the property `animation2`.

**9.85.5.48** `QString QEShape::variable3` `[read, write]`

EPICS variable name (CA PV). This variable is read and used to animate an attribute of the shape. The value read is first scaled and offset by properties `scale3` and `offset3` then the attribute selected for animation is selected by the property `animation3`.

**9.85.5.49** `QString QEShape::variable4` `[read, write]`

EPICS variable name (CA PV). This variable is read and used to animate an attribute of the shape. The value read is first scaled and offset by properties `scale4` and `offset4` then the attribute selected for animation is selected by the property `animation4`.

**9.85.5.50** `QString QEShape::variable5` `[read, write]`

EPICS variable name (CA PV). This variable is read and used to animate an attribute of the shape. The value read is first scaled and offset by properties `scale5` and `offset5` then the attribute selected for animation is selected by the property `animation5`.

**9.85.5.51** `QString QEShape::variable6` `[read, write]`

EPICS variable name (CA PV). This variable is read and used to animate an attribute of the shape. The value read is first scaled and offset by properties `scale6` and `offset6` then the attribute selected for animation is selected by the property `animation6`.

**9.85.5.52** `bool QEShape::variableAsToolTip` `[read, write]`

Use the variable as the tool tip. Default is true. Tool tip property will be overwritten by the variable name.

Reimplemented from [QEToolTip](#).

### 9.85.5.53 QString QEShape::variableSubstitutions [read, write]

Macro substitutions. The default is no substitutions. The format is NAME1=VALUE1[, NAME2=VALUE2... Values may be quoted strings. For example, 'SAMPLE=SAM1, NAME = "Ref foil"' These substitutions are applied to all the variable names.

### 9.85.5.54 bool QEShape::visible [read, write]

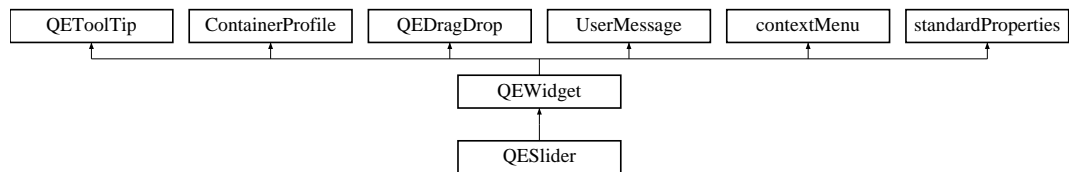
Display the widget. Default is true. Setting this property false is usefull if widget is only used to provide a signal - for example, when supplying data to a [QELink](#) widget. Note, when false the widget will still be visible in Qt Designer.

The documentation for this class was generated from the following files:

- /home/rhydera/epicsqt/framework/widgets/QEShape/QEShape.h
- /home/rhydera/epicsqt/framework/widgets/QEShape/QEShape.cpp

## 9.86 QESlider Class Reference

Inheritance diagram for QESlider:



### Public Types

- enum [UserLevels](#) { [User](#) = USERLEVEL\_USER, [Scientist](#) = USERLEVEL\_SCIENTIST, [Engineer](#) = USERLEVEL\_ENGINEER }

### Public Slots

- void [requestEnabled](#) (const bool &state)

### Signals

- void [dbValueChanged](#) (const qlonglong &out)

## Public Member Functions

- **QESlider** (QWidget \*parent=0)
- **QESlider** (const QString &variableName, QWidget \*parent=0)
- void **setWriteOnChange** (bool [writeOnChange](#))
- bool **getWriteOnChange** ()
- void **setSubscribe** (bool subscribe)
- bool **getSubscribe** ()
- void **setScale** (double scaleIn)
- double **getScale** ()
- void **setOffset** (double offsetIn)
- double **getOffset** ()
- bool **isEnabled** () const  
*Access function for [enabled](#) property - refer to [enabled](#) property for details.*
- void **setEnabled** (bool state)  
*Access function for [enabled](#) property - refer to [enabled](#) property for details.*
- [UserLevels](#) **getUserLevelVisibilityProperty** ()  
*Access function for [userLevelVisibility](#) property - refer to [userLevelVisibility](#) property for details.*
- void **setUserLevelVisibilityProperty** ([UserLevels](#) level)  
*Access function for [userLevelVisibility](#) property - refer to [userLevelVisibility](#) property for details.*
- [UserLevels](#) **getUserLevelEnabledProperty** ()  
*Access function for [userLevelEnabled](#) property - refer to [userLevelEnabled](#) property for details.*
- void **setUserLevelEnabledProperty** ([UserLevels](#) level)  
*Access function for [userLevelEnabled](#) property - refer to [userLevelEnabled](#) property for details.*

## Protected Member Functions

- void **establishConnection** (unsigned int variableIndex)
- void **dragEnterEvent** (QDragEnterEvent \*event)
- void **dropEvent** (QDropEvent \*event)
- void **setDrop** (QVariant drop)
- QVariant **getDrop** ()

## Protected Attributes

- [QEFloatingFormatting](#) **floatingFormatting**
- bool [writeOnChange](#)

## Properties

- QString [variable](#)
- QString [variableSubstitutions](#)
- bool [subscribe](#)
- bool [variableAsToolTip](#)
- bool [enabled](#)
- bool [allowDrop](#)
- bool [visible](#)
- unsigned [int](#)
- QString [userLevelUserStyle](#)
- QString [userLevelScientistStyle](#)
- QString [userLevelEngineerStyle](#)
- [UserLevels](#) [userLevelVisibility](#)
- [UserLevels](#) [userLevelEnabled](#)

## 9.86.1 Member Enumeration Documentation

### 9.86.1.1 enum QESlider::UserLevels

User friendly enumerations for [userLevelVisibility](#) and [userLevelEnabled](#) properties - refer to [userLevelVisibility](#) and [userLevelEnabled](#) properties and [userLevel](#) enumeration for details.

#### Enumerator:

**User** Refer to USERLEVEL\_USER for details.

**Scientist** Refer to USERLEVEL\_SCIENTIST for details.

**Engineer** Refer to USERLEVEL\_ENGINEER for details.

## 9.86.2 Member Function Documentation

### 9.86.2.1 void QESlider::dbValueChanged ( const qlonglong & out ) [signal]

Sent when the widget is updated following a data change Can be used to pass on EPICS data (as presented in this widget) to other widgets. For example a QList widget could log updates from this widget.

### 9.86.2.2 void QESlider::requestEnabled ( const bool & state ) [inline, slot]

Similar to standard setEnabled slot, but allows QE widget to determine if the widget remains disabled due to invalid data. If disabled due to invalid data, a request to enable the widget will be honoured when the data is no longer invalid.

### 9.86.3 Member Data Documentation

**9.86.3.1** `bool QESlider::writeOnChange` [read, write, protected]

Sets if this widget writes any changes as the user moves the slider (the QSlider 'valueChanged' signal is emitted). Default is 'true' (writes any changes when the QSlider 'valueChanged' signal is emitted).

### 9.86.4 Property Documentation

**9.86.4.1** `bool QESlider::allowDrop` [read, write]

Allow drag/drops operations to this widget. Default is false. Any dropped text will be used as a new variable name.

Reimplemented from [QEDragDrop](#).

**9.86.4.2** `bool QESlider::enabled` [read, write]

Set the preferred 'enabled' state. Default is true. This property is copied to the standard Qt 'enabled' property if the data being displayed is valid. If the data being displayed is invalid the standard Qt 'enabled' property will always be set to false to indicate invalid data. The value of this property will only be copied to the standard Qt 'enabled' property once data is valid.

**9.86.4.3** `unsigned QESlider::int` [read, write]

Set the ID used by the message filtering system. Default is zero. Widgets or applications that use messages from the framework have the option of filtering on this ID. For example, by using a unique message source ID a [QELog](#) widget may be set up to only log messages from a select set of widgets.

**9.86.4.4** `bool QESlider::subscribe` [read, write]

Sets if this widget subscribes for data updates and displays current data. Default is 'true' (subscribes for and displays data updates)

Reimplemented from [QEWidget](#).

**9.86.4.5** `UserLevels QESlider::userLevelEnabled` [read, write]

Lowest user level at which the widget is enabled. Default is 'User'. Used when designing GUIs that allow access to more and more detail according to the user mode. The user mode is set application wide through the [QELogin](#) widget, or programatically through setUserLevel() Widgets that are always accessible should be visible at 'User'. Widgets that are only accessible to scientists managing the facility should be visible at 'Scientist'.

Widgets that are only accessible to engineers maintaining the facility should be visible at 'Engineer'.

#### 9.86.4.6 QString QESlider::userLevelEngineerStyle [read, write]

Style Sheet string to be applied when the widget is displayed in 'Engineer' mode. Default is an empty string. The syntax is the standard Qt Style Sheet syntax. For example, 'background-color: red' This Style Sheet string will be applied by the styleManager class. Refer to the styleManager class for details about how this Style Sheet string will be merged with any pre-existing Style Sheet string and any Style Sheet strings generated during the display of data.

#### 9.86.4.7 QString QESlider::userLevelScientistStyle [read, write]

Style Sheet string to be applied when the widget is displayed in 'Scientist' mode. Default is an empty string. The syntax is the standard Qt Style Sheet syntax. For example, 'background-color: red' This Style Sheet string will be applied by the styleManager class. Refer to the styleManager class for details about how this Style Sheet string will be merged with any pre-existing Style Sheet string and any Style Sheet strings generated during the display of data.

#### 9.86.4.8 QString QESlider::userLevelUserStyle [read, write]

Style Sheet string to be applied when the widget is displayed in 'User' mode. Default is an empty string. The syntax is the standard Qt Style Sheet syntax. For example, 'background-color: red' This Style Sheet string will be applied by the styleManager class. Refer to the styleManager class for details about how this Style Sheet string will be merged with any pre-existing Style Sheet string and any Style Sheet strings generated during the display of data.

#### 9.86.4.9 UserLevels QESlider::userLevelVisibility [read, write]

Lowest user level at which the widget is visible. Default is 'User'. Used when designing GUIs that display more and more detail according to the user mode. The user mode is set application wide through the [QELogin](#) widget, or programatically through setUserLevel() Widgets that are always visible should be visible at 'User'. Widgets that are only used by scientists managing the facility should be visible at 'Scientist'. Widgets that are only used by engineers maintaining the facility should be visible at 'Engineer'.

#### 9.86.4.10 QString QESlider::variable [read, write]

EPICS variable name (CA PV)



9.86.4.11 `bool QESlider::variableAsToolTip` [read, write]

Use the variable as the tool tip. Default is true. Tool tip property will be overwritten by the variable name.

Reimplemented from [QEToolTip](#).

9.86.4.12 `QString QESlider::variableSubstitutions` [read, write]

Macro substitutions. The default is no substitutions. The format is NAME1=VALUE1[, NAME2=VALUE2... Values may be quoted strings. For example, 'PUMP=PMP3, NAME = "My Pump"' These substitutions are applied to variable names for all QE widgets. In some widgets are also used for other purposes.

9.86.4.13 `bool QESlider::visible` [read, write]

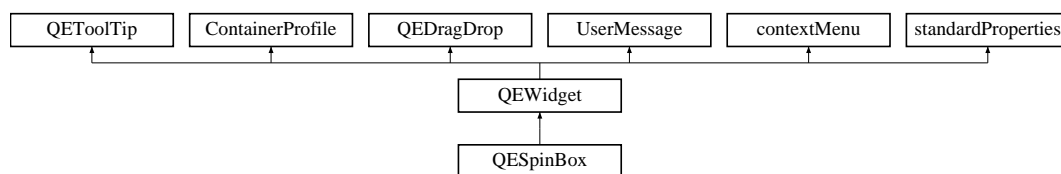
Display the widget. Default is true. Setting this property false is usefull if widget is only used to provide a signal - for example, when supplying data to a [QELink](#) widget. Note, when false the widget will still be visible in Qt Designer.

The documentation for this class was generated from the following files:

- /home/rhydera/epicsqt/framework/widgets/QESlider/QESlider.h
- /home/rhydera/epicsqt/framework/widgets/QESlider/QESlider.cpp

## 9.87 QESpinBox Class Reference

Inheritance diagram for QESpinBox:



### Public Types

- enum `UserLevels` { `User` = USERLEVEL\_USER, `Scientist` = USERLEVEL\_SCIENTIST, `Engineer` = USERLEVEL\_ENGINEER }

### Public Slots

- void `requestEnabled` (const bool &state)

## Signals

- void [dbValueChanged](#) (const double &out)
- void [userChange](#) (const QString &oldValue, const QString &newValue, const QString &lastValue)

*Internal use only. Used by [QEConfiguredLayout](#) to be notified when one of its widgets has written something.*

## Public Member Functions

- **QESpinBox** (QWidget \*parent=0)
- **QESpinBox** (const QString &variableName, QWidget \*parent=0)
- void **setWriteOnChange** (bool writeOnChangeIn)
- bool **getWriteOnChange** ()
- void **setSubscribe** (bool subscribe)
- bool **getSubscribe** ()
- void **setAddUnitsAsSuffix** (bool addUnitsAsSuffixIn)
- bool **getAddUnitsAsSuffix** ()
- void **setUseDbPrecisionForDecimals** (bool useDbPrecisionForDecimalln)
- bool **getUseDbPrecisionForDecimals** ()
- bool **isEnabled** () const  
*Access function for [enabled](#) property - refer to [enabled](#) property for details.*
- void **setEnabled** (bool state)  
*Access function for [enabled](#) property - refer to [enabled](#) property for details.*
- [UserLevels](#) **getUserLevelVisibilityProperty** ()  
*Access function for [userLevelVisibility](#) property - refer to [userLevelVisibility](#) property for details.*
- void **setUserLevelVisibilityProperty** ([UserLevels](#) level)  
*Access function for [userLevelVisibility](#) property - refer to [userLevelVisibility](#) property for details.*
- [UserLevels](#) **getUserLevelEnabledProperty** ()  
*Access function for [userLevelEnabled](#) property - refer to [userLevelEnabled](#) property for details.*
- void **setUserLevelEnabledProperty** ([UserLevels](#) level)  
*Access function for [userLevelEnabled](#) property - refer to [userLevelEnabled](#) property for details.*

## Protected Member Functions

- void **establishConnection** (unsigned int variableIndex)
- void **dragEnterEvent** (QDragEnterEvent \*event)
- void **dropEvent** (QDropEvent \*event)
- void **setDrop** (QVariant drop)
- QVariant **getDrop** ()

### Protected Attributes

- [QEFloatingFormatting](#) **floatingFormatting**
- bool **writeOnChange**
- bool **addUnitsAsSuffix**
- bool **useDbPrecisionForDecimal**

### Properties

- QString [variable](#)
- QString [variableSubstitutions](#)
- bool [variableAsToolTip](#)
- bool [enabled](#)
- bool [allowDrop](#)
- bool [visible](#)
- unsigned int
- QString [userLevelUserStyle](#)
- QString [userLevelScientistStyle](#)
- QString [userLevelEngineerStyle](#)
- [UserLevels](#) [userLevelVisibility](#)
- [UserLevels](#) [userLevelEnabled](#)
- bool [subscribe](#)
- bool **useDbPrecision**
- bool **addUnits**

## 9.87.1 Member Enumeration Documentation

### 9.87.1.1 enum QESpinBox::UserLevels

User friendly enumerations for [userLevelVisibility](#) and [userLevelEnabled](#) properties - refer to [userLevelVisibility](#) and [userLevelEnabled](#) properties and [userLevel](#) enumeration for details.

#### Enumerator:

**User** Refer to `USERLEVEL_USER` for details.

**Scientist** Refer to `USERLEVEL_SCIENTIST` for details.

**Engineer** Refer to `USERLEVEL_ENGINEER` for details.

## 9.87.2 Member Function Documentation

### 9.87.2.1 void QESpinBox::dbValueChanged ( const double & out ) [signal]

Sent when the widget is updated following a data change Can be used to pass on EPICS data (as presented in this widget) to other widgets. For example a QList widget could log updates from this widget.

### 9.87.2.2 void QESpinBox::requestEnabled ( const bool & state ) [inline, slot]

Similar to standard setEnabled slot, but allows QE widget to determine if the widget remains disabled due to invalid data. If disabled due to invalid data, a request to enable the widget will be honoured when the data is no longer invalid.

## 9.87.3 Property Documentation

### 9.87.3.1 bool QESpinBox::allowDrop [read, write]

Allow drag/drops operations to this widget. Default is false. Any dropped text will be used as a new variable name.

Reimplemented from [QEDragDrop](#).

### 9.87.3.2 bool QESpinBox::enabled [read, write]

Set the preferred 'enabled' state. Default is true. This property is copied to the standard Qt 'enabled' property if the data being displayed is valid. If the data being displayed is invalid the standard Qt 'enabled' property will always be set to false to indicate invalid data. The value of this property will only be copied to the standard Qt 'enabled' property once data is valid.

### 9.87.3.3 unsigned QESpinBox::int [read, write]

Set the ID used by the message filtering system. Default is zero. Widgets or applications that use messages from the framework have the option of filtering on this ID. For example, by using a unique message source ID a [QELog](#) widget may be set up to only log messages from a select set of widgets.

### 9.87.3.4 bool QESpinBox::subscribe [read, write]

Sets if this widget subscribes for data updates and displays current data. Default is 'true' (subscribes for and displays data updates)

Reimplemented from [QEWidget](#).

### 9.87.3.5 UserLevels QESpinBox::userLevelEnabled [read, write]

Lowest user level at which the widget is enabled. Default is 'User'. Used when designing GUIs that allow access to more and more detail according to the user mode. The user mode is set application wide through the [QELogin](#) widget, or programatically through setUserLevel() Widgets that are always accessible should be visible at 'User'. Widgets that are only accessible to scientists managing the facility should be visible at 'Scientist'. Widgets that are only accessible to engineers maintaining the facility should be visible at 'Engineer'.

**9.87.3.6 QString QESpinBox::userLevelEngineerStyle** [read, write]

Style Sheet string to be applied when the widget is displayed in 'Engineer' mode. Default is an empty string. The syntax is the standard Qt Style Sheet syntax. For example, 'background-color: red' This Style Sheet string will be applied by the styleManager class. Refer to the styleManager class for details about how this Style Sheet string will be merged with any pre-existing Style Sheet string and any Style Sheet strings generated during the display of data.

**9.87.3.7 QString QESpinBox::userLevelScientistStyle** [read, write]

Style Sheet string to be applied when the widget is displayed in 'Scientist' mode. Default is an empty string. The syntax is the standard Qt Style Sheet syntax. For example, 'background-color: red' This Style Sheet string will be applied by the styleManager class. Refer to the styleManager class for details about how this Style Sheet string will be merged with any pre-existing Style Sheet string and any Style Sheet strings generated during the display of data.

**9.87.3.8 QString QESpinBox::userLevelUserStyle** [read, write]

Style Sheet string to be applied when the widget is displayed in 'User' mode. Default is an empty string. The syntax is the standard Qt Style Sheet syntax. For example, 'background-color: red' This Style Sheet string will be applied by the styleManager class. Refer to the styleManager class for details about how this Style Sheet string will be merged with any pre-existing Style Sheet string and any Style Sheet strings generated during the display of data.

**9.87.3.9 UserLevels QESpinBox::userLevelVisibility** [read, write]

Lowest user level at which the widget is visible. Default is 'User'. Used when designing GUIs that display more and more detail according to the user mode. The user mode is set application wide through the [QELogin](#) widget, or programatically through setUserLevel() Widgets that are always visible should be visible at 'User'. Widgets that are only used by scientists managing the facility should be visible at 'Scientist'. Widgets that are only used by engineers maintaining the facility should be visible at 'Engineer'.

**9.87.3.10 QString QESpinBox::variable** [read, write]

EPICS variable name (CA PV)

**9.87.3.11 bool QESpinBox::variableAsToolTip** [read, write]

Use the variable as the tool tip. Default is true. Tool tip property will be overwritten by the variable name.

Reimplemented from [QEToolTip](#).

### 9.87.3.12 QString QESpinBox::variableSubstitutions [read, write]

Macro substitutions. The default is no substitutions. The format is NAME1=VALUE1[, NAME2=VALUE2... Values may be quoted strings. For example, 'PUMP=PMP3, NAME = "My Pump"' These substitutions are applied to variable names for all QE widgets. In some widgets are are also used for other purposes.

### 9.87.3.13 bool QESpinBox::visible [read, write]

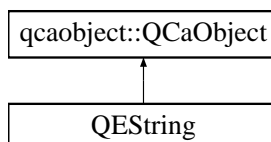
Display the widget. Default is true. Setting this property false is usefull if widget is only used to provide a signal - for example, when supplying data to a [QELink](#) widget. Note, when false the widget will still be visible in Qt Designer.

The documentation for this class was generated from the following files:

- /home/rhydera/epicsqt/framework/widgets/QESpinBox/QESpinBox.h
- /home/rhydera/epicsqt/framework/widgets/QESpinBox/QESpinBox.cpp

## 9.88 QString Class Reference

Inheritance diagram for QString:



### Public Slots

- void **writeString** (const QString &data)

### Signals

- void **stringConnectionChanged** ([QCaConnectionInfo](#) &connectionInfo, const unsigned int &variableIndex)
- void **stringChanged** (const QString &value, [QCaAlarmInfo](#) &alarmInfo, [QCaDateTime](#) &timeStamp, const unsigned int &variableIndex)

### Public Member Functions

- **QString** (QString recordName, QObject \*eventObject, [QStringFormatting](#) \*stringFormattingIn, unsigned int variableIndexIn)

- **QString** (QString recordName, QObject \*eventObject, [QStringFormatting](#) \*stringFormattingIn, unsigned int variableIndexIn, [UserMessage](#) \*userMessageIn)
- bool **writeString** (const QString &data, QString &message)

The documentation for this class was generated from the following files:

- /home/rhydera/epicsqt/framework/data/include/QString.h
- /home/rhydera/epicsqt/framework/data/src/QString.cpp

## 9.89 QStringFormatting Class Reference

### Public Types

- enum [formats](#) {  
[FORMAT\\_DEFAULT](#), [FORMAT\\_FLOATING](#), [FORMAT\\_INTEGER](#), [FORMAT\\_UNSIGNEDINTEGER](#),  
[FORMAT\\_TIME](#), [FORMAT\\_LOCAL\\_ENUMERATE](#), [FORMAT\\_STRING](#) }
- enum [notations](#) { [NOTATION\\_FIXED](#) = QTextStream::FixedNotation, [NOTATION\\_SCIENTIFIC](#) = QTextStream::ScientificNotation, [NOTATION\\_AUTOMATIC](#) = QTextStream::SmartNotation }
- enum [arrayActions](#) { [APPEND](#), [ASCII](#), [INDEX](#) }

### Public Member Functions

- QString **formatString** (const QVariant &value)
- QVariant **formatValue** (const QString &text, bool &ok)
- void **setDbEgu** (QString egu)
- void **setDbEnumerations** (QStringList enumerations)
- void **setDbPrecision** (unsigned int dbPrecisionIn)
- void **setDbVariableIsStatField** (bool isStatField)
- void **setPrecision** (int precision)
- void **setUseDbPrecision** (bool useDbPrecision)
- void **setLeadingZero** (bool leadingZero)
- void **setTrailingZeros** (bool trailingZeros)
- void **setFormat** ([formats](#) format)
- void **setRadix** (unsigned int radix)
- void **setNotation** ([notations](#) notation)
- void **setArrayAction** ([arrayActions](#) arrayActionIn)
- void **setArrayIndex** (unsigned int arrayIndexIn)
- void **setAddUnits** (bool addUnits)
- void **setLocalEnumeration** (QString localEnumerationIn)
- int **getPrecision** ()
- bool **getUseDbPrecision** ()
- bool **getLeadingZero** ()
- bool **getTrailingZeros** ()

- [formats](#) `getFormat ()`
- unsigned int `getRadix ()`
- [notations](#) `getNotation ()`
- [arrayActions](#) `getArrayAction ()`
- unsigned int `getArrayIndex ()`
- bool `getAddUnits ()`
- QString `getLocalEnumeration ()`

### 9.89.1 Member Enumeration Documentation

#### 9.89.1.1 enum `QStringFormatting::arrayActions`

What action to take when formatting array data

##### Enumerator:

**APPEND** Interpret each element in the array as an unsigned integer and append string representations of each element from the array with a space in between each.

**ASCII** Interpret each element from the array as a character in a string. Translate all non printing characters to '?' except for trailing zeros (ignore them)

**INDEX** Interpret the element selected by `setArrayIndex()` as an unsigned integer.

#### 9.89.1.2 enum `QStringFormatting::formats`

Formatting options

##### Enumerator:

**FORMAT\_DEFAULT** Format according to the EPICS database record type.

**FORMAT\_FLOATING** Format as a floating point number.

**FORMAT\_INTEGER** Format as an integer.

**FORMAT\_UNSIGNEDINTEGER** Format as an unsigned integer.

**FORMAT\_TIME** Format as a time.

**FORMAT\_LOCAL\_ENUMERATE** Format as a selection from the local enumerations set by `setLocalEnumeration()`

**FORMAT\_STRING** Format as a string.

#### 9.89.1.3 enum `QStringFormatting::notations`

Notations when formatting a floating point number

##### Enumerator:

**NOTATION\_FIXED** Standard floating point 123456.789.



**NOTATION\_SCIENTIFIC** Scientific representation 1.23456789e6.

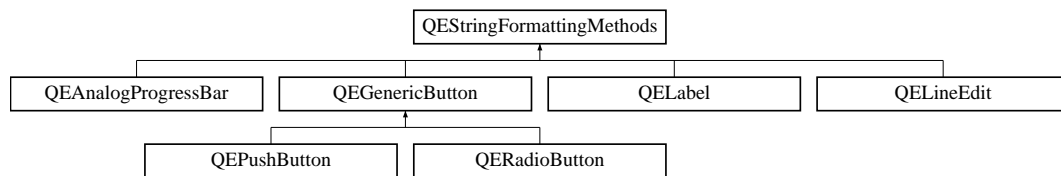
**NOTATION\_AUTOMATIC** Automatic choice of standard or scientific notation.

The documentation for this class was generated from the following files:

- /home/rhydera/epicsqt/framework/data/include/QStringFormatting.h
- /home/rhydera/epicsqt/framework/data/src/QStringFormatting.cpp

## 9.90 QStringFormattingMethods Class Reference

Inheritance diagram for QStringFormattingMethods:



### Public Member Functions

- virtual void **stringFormattingChange** ()=0
- void **setPrecision** (int precision)
- int **getPrecision** ()
- void **setUseDbPrecision** (bool useDbPrecision)
- bool **getUseDbPrecision** ()
- void **setLeadingZero** (bool leadingZero)
- bool **getLeadingZero** ()
- void **setTrailingZeros** (bool trailingZeros)
- bool **getTrailingZeros** ()
- void **setAddUnits** (bool addUnits)
- bool **getAddUnits** ()
- void **setLocalEnumeration** (QString localEnumeration)
- QString **getLocalEnumeration** ()
- void **setFormat** (QStringFormatting::formats format)
- QStringFormatting::formats **getFormat** ()
- void **setRadix** (unsigned int radix)
- unsigned int **getRadix** ()
- void **setNotation** (QStringFormatting::notations notation)
- QStringFormatting::notations **getNotation** ()
- void **setArrayAction** (QStringFormatting::arrayActions arrayAction)
- QStringFormatting::arrayActions **getArrayAction** ()
- void **setArrayIndex** (unsigned int arrayIndex)
- unsigned int **getArrayIndex** ()

- `QStringFormatting` **stringFormatting**

- /home/rhydera/epicsqt/framework/widgets/include/QCStringFormattingMethods.h
- /home/rhydera/epicsqt/framework/widgets/src/QCStringFormattingMethods.cpp

Inheritance diagram for QEStripChart:



- ## Public Types

- ## Public Member Functions

- **QStripChart** (QWidget \*parent=0)
- QSize **sizeHint** () const
- QDateTime **getStartDateTime** ()
- QDateTime **getEndTimeTime** ()
- void **setEndTimeTime** (QDateTime endTimeIn)
- int **getDuration** ()
- void **setDuration** (int durationIn)
- double **getYMinimum** ()
- void **setYMinimum** (double yMinimumIn)
- double **getYMaximum** ()
- void **setYMaximum** (double yMaximumIn)
- void **plotData** ()

### Protected Member Functions

- void **dragEnterEvent** (QDragEnterEvent \*event)
- void **dropEvent** (QDropEvent \*event)
- void **mousePressEvent** (QMouseEvent \*event)
- void **setDrop** (QVariant drop)
- QVariant **getDrop** ()
- QString **copyVariable** ()
- QVariant **copyData** ()
- void **paste** (QVariant s)
- void **setup** ()
- [qcaobject::QCaObject](#) \* **createQcaltem** (unsigned int variableIndex)
- void **establishConnection** (unsigned int variableIndex)

### Properties

- int **duration**
- double **yMinimum**
- double **yMaximum**
- QString **variable1**
- QString **variable2**
- QString **variable3**
- QString **variable4**
- QString **variable5**
- QString **variable6**
- QString **variable7**
- QString **variable8**
- QString **variable9**
- QString **variable10**
- QString **variable11**
- QString **variable12**
- QColor **colour1**
- QColor **colour2**
- QColor **colour3**
- QColor **colour4**
- QColor **colour5**
- QColor **colour6**
- QColor **colour7**
- QColor **colour8**
- QColor **colour9**
- QColor **colour10**
- QColor **colour11**
- QColor **colour12**

### Friends

- class **PrivateData**
- class [QEStripChartItem](#)

The documentation for this class was generated from the following files:

- /home/rhydera/epicsqt/framework/widgets/QEStripChart/QEStripChart.h
- /home/rhydera/epicsqt/framework/widgets/QEStripChart/QEStripChart.cpp

## 9.92 QEStripChartItem Class Reference

### Classes

- class [PrivateData](#)

### Friends

- class [QEStripChart](#)

The documentation for this class was generated from the following files:

- /home/rhydera/epicsqt/framework/widgets/QEStripChart/QEStripChartItem.h
- /home/rhydera/epicsqt/framework/widgets/QEStripChart/QEStripChartItem.cpp

## 9.93 QEStripChartItemDialog Class Reference

### Public Member Functions

- **QEStripChartItemDialog** (QWidget \*parent=0)
- void **setPvName** (QString pvNameIn)
- QString **getPvName** ()
- void **setColour** (QColor colourIn)
- QColor **getColour** ()
- bool **isClear** ()

The documentation for this class was generated from the following files:

- /home/rhydera/epicsqt/framework/widgets/QEStripChart/QEStripChartItemDialog.h
- /home/rhydera/epicsqt/framework/widgets/QEStripChart/QEStripChartItemDialog.cpp

## 9.94 QEStripChartTimeDialog Class Reference

### Public Member Functions

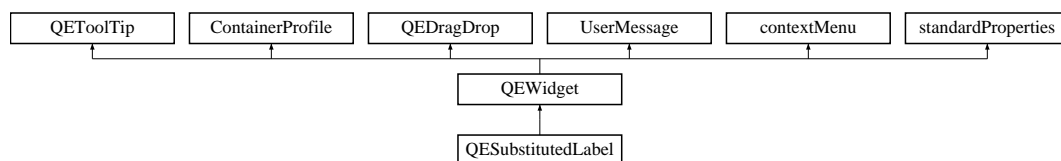
- **QEStripChartTimeDialog** (QWidget \*parent=0)
- void **setMaximumDateTime** (QDateTime datetime)
- void **setStartDateTime** (QDateTime datetime)
- QDateTime **getStartDateTime** ()
- void **setEndDateTime** (QDateTime datetime)
- QDateTime **getEndDateTime** ()

The documentation for this class was generated from the following files:

- /home/rhydera/epicsqt/framework/widgets/QEStripChart/QEStripChartTimeDialog.h
- /home/rhydera/epicsqt/framework/widgets/QEStripChart/QEStripChartTimeDialog.cpp

## 9.95 QESubstitutedLabel Class Reference

Inheritance diagram for QESubstitutedLabel:



### Public Member Functions

- **QESubstitutedLabel** (QWidget \*parent=0)
- void **establishConnection** (unsigned int variableIndex)
- void **setLabelTextProperty** (QString labelTextIn)
- QString **getLabelTextProperty** ()
- QString **getLabelTextPropertyFormat** ()
- void **setLabelTextPropertyFormat** (QString labelTextIn)

### Protected Attributes

- QString [labelText](#)

### Properties

- QString [textSubstitutions](#)

### 9.95.1 Member Data Documentation

9.95.1.1 **QString QESubstitutedLabel::labelText** [read, write, protected]

Label text to be substituted. This text will be copied to the label text after applying any macro substitutions from the textSubstitutions property

### 9.95.2 Property Documentation

9.95.2.1 **QString QESubstitutedLabel::textSubstitutions** [read, write]

Text substitutions. These substitutions are applied to the 'labelText' property prior to copying it to the label text.

The documentation for this class was generated from the following files:

- /home/rhydera/epicsqt/framework/widgets/QESubstitutedLabel/QESubstitutedLabel.h
- /home/rhydera/epicsqt/framework/widgets/QESubstitutedLabel/QESubstitutedLabel.cpp

## 9.96 QEToolTip Class Reference

Inheritance diagram for QEToolTip:



- ## Protected Member Functions

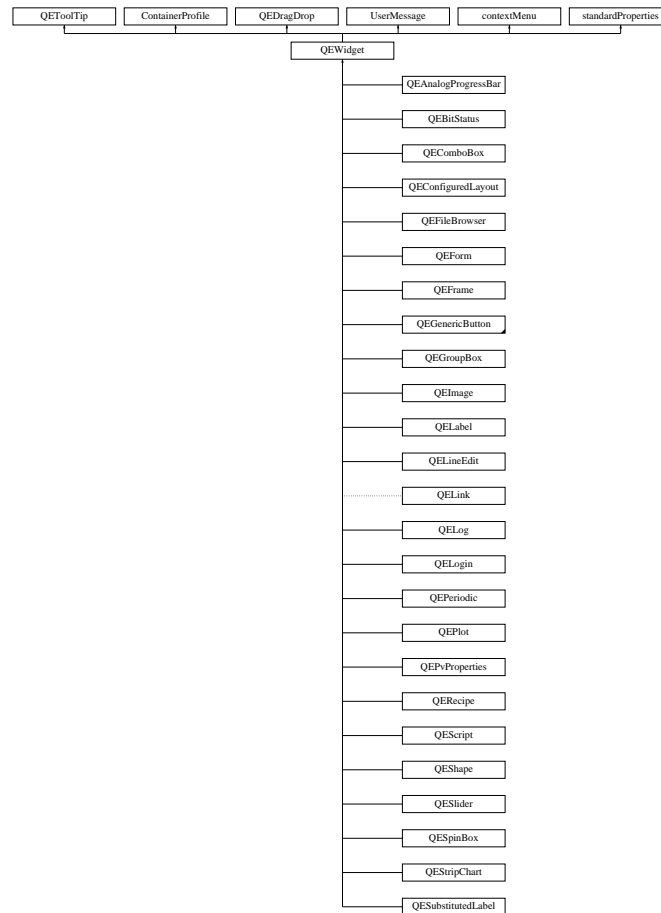
- The documentation for this class was generated from the following files:

- Generated on Wed Jan 23 2013 15:43:31 for EPICS QT Framework by Doxygen

## 9.97 QEWidget Class Reference

```
#include <QEWidget.h>
```

Inheritance diagram for QEWidget:



### Public Member Functions

- [QEWidget](#) (QWidget \*ownerIn)  
*Constructor.*
- virtual [~QEWidget](#) ()  
*Destructor.*
- void [activate](#) ()
- unsigned int [getMessageSourceId](#) ()
- void [setMessageSourceId](#) (unsigned int messageSourceId)
- [qcaobject::QCaObject](#) \* [getQcaItem](#) (unsigned int variableIndex)
- void [setupContextMenu](#) (QWidget \*w)
- QColor [getColor](#) (QCaAlarmInfo &alarmInfo, const int saturation)



- void [readNow](#) ()
- virtual void [writeNow](#) ()
- virtual void [setVariableNameAndSubstitutions](#) (QString variableNameIn, QString variableNameSubstitutionsIn, unsigned int variableIndex)
- QFile \* [openQEFile](#) (QString name, QFile::OpenModeFlag mode)
- QString [defaultFileLocation](#) ()

### Static Public Member Functions

- static bool **inDesigner** ()

### Protected Member Functions

- void **setNumVariables** (unsigned int numVariablesIn)
- [qcaobject::QCaObject](#) \* **createConnection** (unsigned int variableIndex)
- virtual [qcaobject::QCaObject](#) \* **createQcaltem** (unsigned int variableIndex)
- virtual void **establishConnection** (unsigned int variableIndex)

### Protected Attributes

- bool **subscribe**

#### 9.97.1 Detailed Description

This class is used as a base for all CA aware widgets, such as [QELabel](#), [QESpinBox](#), etc. It manages common issues including creating a source of CA data updates, handling error, warning and status messages, and setting tool tips based on variable names.

Note, there is tight integration between the CA aware widget classes, this class, and its base classes, especially [VariableNameManager](#) and [QEToolTip](#).

In particular, this class manages [QCaObject](#) classes that stream updates to the CA aware widget class. But this class, however, doesn't know how to format the data, or how the updates will be used. To resolve this, this class asks its parent class (such as [QELabel](#)) to create the [QCaObject](#) class in what ever flavour it wants, by calling the virtual function [createQcaltem](#). A [QELabel](#), for example, wants string updates so it creates a [QEString](#) which is based on a [QCaObject](#) class and formats all updates as strings.

The CA aware parent class (such as [QELabel](#)) defines a variable by calling [VariableNameManager::setVariableName\(\)](#). The [VariableNamePropertyManager](#) class calls the [establishConnection](#) function of the CA aware parent class, such as [QELabel](#) when it has a new variable name.

This class uses its base [QEToolTip](#) class to format tool tips. that class in turn calls the CA aware parent class (such as [QELabel](#)) directly to make use of a new tool tip.

After construction, a CA aware widget is activated (starts updating) by calling its [establishConnection\(\)](#) function in one of two ways:

1) The variable name or variable name substitutions is changed by calling `setVariableName` or `setVariableNameSubstitutions` respectively. These functions are in the `VariableNameManager` class. The `VariableNamePropertyManager` calls a virtual function `establishConnection()` which is implemented by the CA aware widget. This is how a CA aware widget is activated in 'designer'. It occurs when 'designer' updates the variable name property or variable name substitution property.

2) When an `QForm` widget is created, resulting in a set of CA aware widgets being created by loading a UI file containing plugin definitions. After loading the plugin widgets, code in the `QForm` class calls the `activate()` function in this class (`QForm`). the `activate()` function calls `establishConnection()` in the CA aware widget for each variable. This simulates what the `VariableNamePropertyManager` does as each variable name is entered (see 1, above, for details)

No matter which way a CA aware widget is activated, the `establishConnection()` function in the CA aware widget is called for each variable. The `establishConnection()` function asks this `QForm` base class, by calling the `createConnection()` function, to perform the tasks common to all CA aware widgets for establishing a stream of CA data.

The `createConnection()` function sets up the widget 'tool tip', then immediately calls the CA aware widget back asking it to create an object based on `QCaObject`. This object will supply a stream of CA update signals to the CA aware object in a form that it needs. For example a `QLabel` creates a `QString` object. The `QString` class is based on the `QCaObject` class and converts all update data to a strings which is required for updating a Qt label widget. This class stores the `QCaObject` based class.

After the `establishConnection()` function in the CA aware widget has called `createConnection()`, the remaining task of the `establishConnection()` function is to connect the signals of the newly created `QCaObject` based classes to its own slots so that data updates can be used. For example, a `QLabel` connects the 'stringChanged' signal from the `QString` object to its `setLabelText` slot.

## 9.97.2 Member Function Documentation

### 9.97.2.1 void QForm::activate ( )

Initiate updates. Called after all configuration is complete.

### 9.97.2.2 QString QForm::defaultFileLocation ( )

Returns the default location to create files. Use this to create files in a consistent location

### 9.97.2.3 QColor QForm::getColor ( QCaAlarmInfo & alarmInfo, const int saturation )

Return a colour to update the widget's look to reflect the current alarm state Note, the color is determined by the `alarmInfo` class, but since that class is used in non gui applications, it can't return a `QColor`

9.97.2.4 `unsigned int QEWidget::getMessageSourceId ( ) [inline]`

Get the message source ID. The message source ID is used as part of the system where QE widgets can emit a message and have the right QE widget in the right form catch the message. Refer to the [UserMessage](#) class for further details.

9.97.2.5 `qcaobject::QCaObject * QEWidget::getQcaltem ( unsigned int variableIndex )`

Return a reference to one of the qCaObjects used to stream CA updates

9.97.2.6 `QFile * QEWidget::openQEFile ( QString name, QFile::OpenModeFlag mode )`

Looks for a file in a standard set of locations (and opens the file)

9.97.2.7 `void QEWidget::readNow ( )`

Perform a single shot read on all variables (Usefull when not subscribing by default)

9.97.2.8 `void QEWidget::setMessageSourceId ( unsigned int messageSourceId )  
[inline]`

Set the message source ID. The message source ID is used as part of the system where QE widgets can emit a message and have the right QE widget in the right form catch the message. Refer to the [UserMessage](#) class for further details.

9.97.2.9 `void QEWidget::setupContextMenu ( QWidget * w )`

Take a menu widgt and add it as the context menu for this widget

9.97.2.10 `void QEWidget::setVariableNameAndSubstitutions ( QString variableNameIn, QString variableNameSubstitutionsIn, unsigned int variableIndex ) [virtual]`

Virtual function that may be implimented by users of [QEWidget](#) to update variable names and macro substitutions. A default is provided that is suitable in most cases.

Reimplemented in [QEBitStatus](#), and [QEForm](#).

9.97.2.11 `virtual void QEWidget::writeNow ( ) [inline, virtual]`

(Control widgets only - such as [QELineEdit](#)) Write the value now. Used when writeOn-Change, writeOnEnter, etc are all false

The documentation for this class was generated from the following files:

- `/home/rhydera/epicsqt/framework/widgets/include/QEWidget.h`

- /home/rhydera/epicsqt/framework/widgets/src/QEWidget.cpp

## 9.98 QEWidgets Class Reference

### Public Member Functions

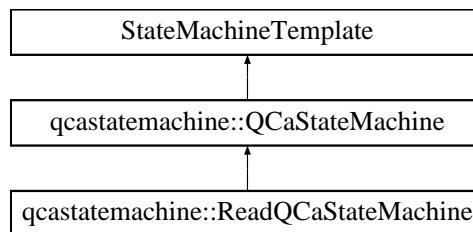
- **QEWidgets** (QObject \*parent=0)
- virtual QList< QDesignerCustomWidgetInterface \* > **customWidgets** () const

The documentation for this class was generated from the following files:

- /home/rhydera/epicsqt/framework/widgets/include/QEDesignerPlugin.h
- /home/rhydera/epicsqt/framework/widgets/src/QEDesignerPlugin.cpp

## 9.99 qcastatemachine::ReadQCaStateMachine Class Reference

Inheritance diagram for qcastatemachine::ReadQCaStateMachine:



### Public Member Functions

- **ReadQCaStateMachine** (void \*parent)
- bool **process** (int requestedState)

The documentation for this class was generated from the following files:

- /home/rhydera/epicsqt/framework/data/include/QCaStateMachine.h
- /home/rhydera/epicsqt/framework/data/src/QCaStateMachine.cpp

## 9.100 RecordSpec Class Reference

### Public Member Functions

- **RecordSpec** (const QString theRecordType)

- QString **getRecordType** ()
- QString **getFieldName** (const int index)

The documentation for this class was generated from the following file:

- /home/rhydera/epicsqt/framework/widgets/QEPvProperties/QEPvProperties.cpp

## 9.101 RecordSpecList Class Reference

### Public Member Functions

- [RecordSpec](#) \* **find** (const QString recordType)
- void **appendOrReplace** ([RecordSpec](#) \*pRecordSpec)

The documentation for this class was generated from the following file:

- /home/rhydera/epicsqt/framework/widgets/QEPvProperties/QEPvProperties.cpp

## 9.102 selectMenu Class Reference

### Public Member Functions

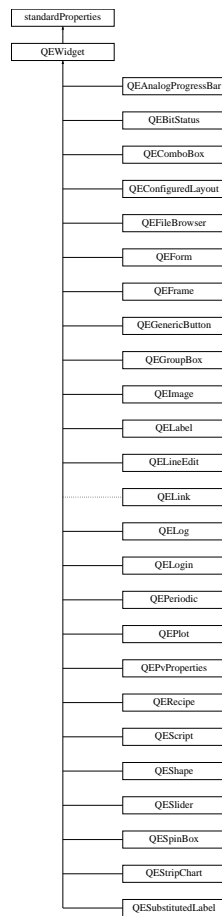
- **selectMenu** (QWidget \*parent=0)
- [imageContextMenu::imageContextMenuOptions](#) **getSelectOption** (const QPoint &pos)
- void **setChecked** (const int mode)
- void **setPanEnabled** (bool enablePan)
- void **setVSliceEnabled** (bool enableVSliceSelection)
- void **setHSliceEnabled** (bool enableHSliceSelection)
- void **setAreaEnabled** (bool enableAreaSelection)
- void **setProfileEnabled** (bool enableProfileSelection)
- void **setTargetEnabled** (bool enableTargetSelection)

The documentation for this class was generated from the following files:

- /home/rhydera/epicsqt/framework/widgets/QEImage/selectMenu.h
- /home/rhydera/epicsqt/framework/widgets/QEImage/selectMenu.cpp

## 9.103 standardProperties Class Reference

Inheritance diagram for standardProperties:



## Public Member Functions

- **standardProperties** (QWidget \*ownerIn)

## Protected Member Functions

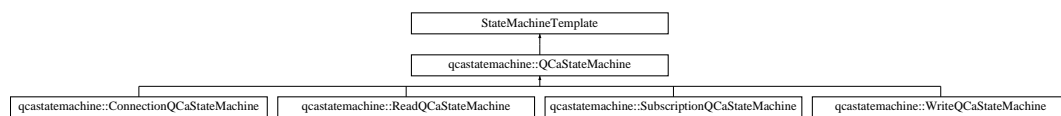
- userLevels **getUserLevelVisibility** ()
- void **setUserLevelVisibility** (userLevels level)
- userLevels **getUserLevelEnabled** ()
- void **setUserLevelEnabled** (userLevels level)
- bool **getApplicationEnabled** () const
- void **setApplicationEnabled** (bool state)
- void **setDataDisabled** (bool disable)
- void **setRunVisible** (bool visibleIn)
- bool **getRunVisible** ()
- void **checkVisibilityEnabledLevel** (userLevels level)

The documentation for this class was generated from the following files:

- /home/rhydera/epicsqt/framework/widgets/include/standardProperties.h
- /home/rhydera/epicsqt/framework/widgets/src/standardProperties.cpp

## 9.104 StateMachineTemplate Class Reference

Inheritance diagram for StateMachineTemplate:



### Public Member Functions

- virtual bool **process** (int requestedState)=0

### Public Attributes

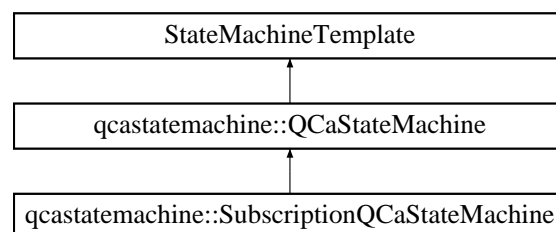
- int **currentState**
- int **requestState**

The documentation for this class was generated from the following file:

- /home/rhydera/epicsqt/framework/data/include/QCaStateMachine.h

## 9.105 qcastatemachine::SubscriptionQCaStateMachine Class Reference

Inheritance diagram for qcastatemachine::SubscriptionQCaStateMachine:



### Public Member Functions

- **SubscriptionQCaStateMachine** (void \*parent)
- bool **process** (int requestedState)

The documentation for this class was generated from the following files:

- /home/rhydera/epicsqt/framework/data/include/QCaStateMachine.h
- /home/rhydera/epicsqt/framework/data/src/QCaStateMachine.cpp

## 9.106 trace Class Reference

### Public Attributes

- QVector< [QCaDateTime](#) > **timeStamps**
- QVector< double > **xdata**
- QVector< double > **ydata**
- QwtPlotCurve \* **curve**
- QColor **color**
- QString **legend**
- bool **waveform**
- QwtPlotCurve::CurveStyle **style**

The documentation for this class was generated from the following file:

- /home/rhydera/epicsqt/framework/widgets/QEPlot/QEPlot.h

## 9.107 TrackRange Class Reference

### Public Member Functions

- void **clear** ()
- void **merge** (const double d)
- void **merge** (const [TrackRange](#) that)
- bool **getMinMax** (double &min, double &max)

The documentation for this class was generated from the following files:

- /home/rhydera/epicsqt/framework/widgets/QEStripChart/QEStripChartItem.h
- /home/rhydera/epicsqt/framework/widgets/QEStripChart/QEStripChartItem.cpp



## 9.108 userInfoStruct Class Reference

### Public Attributes

- bool **enable**
- double **value1**
- double **value2**
- QString **elementText**

The documentation for this class was generated from the following file:

- /home/rhydera/epicsqt/framework/widgets/QEPeriodic/QEPeriodic.h

## 9.109 QEPeriodic::userInfoStructArray Struct Reference

### Public Attributes

- [userInfoStruct](#) **array** [NUM\_ELEMENTS]

The documentation for this struct was generated from the following file:

- /home/rhydera/epicsqt/framework/widgets/QEPeriodic/QEPeriodic.h

## 9.110 userLevelSignal Class Reference

### Signals

- void [userChanged](#) (userLevels level)  
*Internal use only. Send when the user level has changed.*

### Public Member Functions

- void **setLevel** (userLevels levelIn)
- userLevels **getLevel** ()

The documentation for this class was generated from the following files:

- /home/rhydera/epicsqt/framework/widgets/include/ContainerProfile.h
- /home/rhydera/epicsqt/framework/widgets/src/ContainerProfile.cpp

## 9.111 userLevelSlot Class Reference

### Public Slots

- void **userChanged** (userLevels level)

### Public Member Functions

- void **setOwner** ([ContainerProfile](#) \*ownerIn)

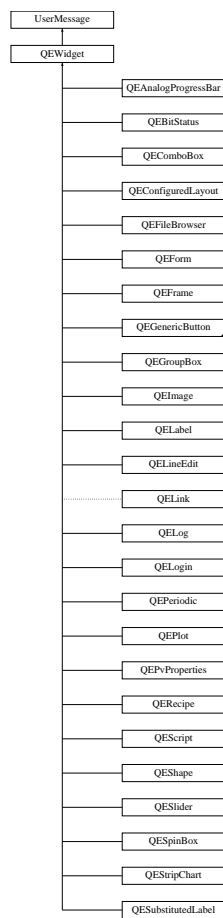
The documentation for this class was generated from the following files:

- /home/rhydera/epicsqt/framework/widgets/include/ContainerProfile.h
- /home/rhydera/epicsqt/framework/widgets/src/ContainerProfile.cpp

## 9.112 UserMessage Class Reference

```
#include <UserMessage.h>
```

Inheritance diagram for UserMessage:



## Public Types

- enum **message\_filter\_options** { MESSAGE\_FILTER\_ANY, MESSAGE\_FILTER\_MATCH, MESSAGE\_FILTER\_NONE }

## Public Member Functions

- void **setSourceId** (unsigned int sourceId)  
Set the source ID (the ID set up by the GUI designer, usually matched to the source ID of logging widgets)
- void **setFormId** (unsigned int formId)  
Set the form ID (the the same ID for all sibling widgets within an **QEForm** widget)
- void **setFormFilter** (message\_filter\_options formFilterIn)  
Set the message filtering applied to the form ID.
- void **setSourceFilter** (message\_filter\_options sourceFilterIn)  
Set the message filtering applied to the source ID.

- unsigned int [getSourceId](#) ()  
*Get the source ID (the ID set up by the GUI designer, usually matched to the source ID of logging widgets).*
- unsigned int [getFormId](#) ()  
*Get the form ID (the the same ID for all sibling widgets within an [QEForm](#) widget)*
- message\_filter\_options [getFormFilter](#) ()  
*Get the message filtering applied to the form ID.*
- message\_filter\_options [getSourceFilter](#) ()  
*Get the message filtering applied to the source ID.*
- void [setChildFormId](#) (unsigned int)  
*Set the for ID of all widgets that are children of this widget.*
- unsigned int [getChildFormId](#) ()  
*Get the for ID of all widgets that are children of this widget.*
- unsigned int [getNextMessageFormId](#) ()  
*Generate a new form ID for all widgets in a new form.*
- void [sendMessage](#) (QString message, message\_types type=MESSAGE\_TYPE\_INFO)  
*Send a message to the user.*
- void [sendMessage](#) (QString message, QString source, message\_types type=MESSAGE\_TYPE\_INFO)  
*Send a message to the user with a source reference.*
- QString [getMessageTypeName](#) (message\_types type)  
*Convenience function to provide string names for each message type.*
- virtual void [newMessage](#) (QString, message\_types)  
*Virtual function to pass messages to derived classes (typically logging widgets or application windows)*

## Friends

- class [UserMessageSlot](#)
- class [UserMessageSignal](#)

### 9.112.1 Detailed Description

A class to manage user messages.

This class passes messages between widgets and application code

This class is used as a base class.

Messages are sent by calling [sendMessage\(\)](#) Messages are received by implementing [newMessage\(\)](#) in the derived class.

Messages can be filtered based on a source ID or a form ID

The derived widget is free to set the source ID to any value

Derived form widgets ([QForm](#)) get a unique form ID using [getNextMessageFormId\(\)](#) (as well as being able to set a source ID like any other QE widget) and pass this unique form ID to all widgets within the form using the [ContainerProfile](#) class.

Messages sent by a QE widget are received by all QE widgets and can filter the messages required by form ID and source ID. The form ID is under the management of the [QForm](#) widget, the source ID is under the control of the GUI designer.

The [QForm](#) widget does not display messages, but re-send them using its own form ID. Read on to see how this can be used.

Widgets that generate messages, and widgets (or application code) that use messages can be set up as follows:

- Application wide logging: An application with a single log window can base a class on the [UserMessage](#) class and set up filtering to receive all messages. An application with log messages for separate windows containing [QForm](#) widgets (such as [QEGui](#)) can base each window class on the [UserMessage](#) class, then set up filtering for the appropriate form ID.
- Logging within a [QForm](#). A logging widget can be set to filter matching on the current form and so will pick up messages from any sibling widget. This includes messages from a sibling widget which is a nested [QForm](#). Whatever messages that nested form is set to receive, it will resend to its siblings. For example, if it is set to receive messages from the widgets it contains, these are resent up one level to the main form. If messages are dealt with within the nested [QForm](#) (for example, it may have its own logging QE widget) then the nested [QForm](#) could be set up not to filter and resend any messages.

The documentation for this class was generated from the following files:

- `/home/rhydera/epicsqt/framework/widgets/include/UserMessage.h`
- `/home/rhydera/epicsqt/framework/widgets/src/UserMessage.cpp`

## 9.113 UserMessageSignal Class Reference

```
#include <UserMessage.h>
```

### Signals

- void [message](#) (QString msg, message\_types type, unsigned int formId, unsigned int sourceId, [UserMessage](#) \*originator)

*Emit a message signal. Any widget based on the [UserMessage](#) class can receive these messages, filtered on formId and sourceId.*

## Public Member Functions

- void [sendMessage](#) (QString msg, message\_types type, unsigned int formId, unsigned int sourceId, [UserMessage](#) \*originator)

*Send a message to all widgets based on the [UserMessage](#) class.*

### 9.113.1 Detailed Description

Class used to send message signals. Used only within UserMessage.cpp A single instance of this class is shared by all instances of the [UserMessage](#) class. This allows every [UserMessage](#) class instance to connect to a single source of messages

The documentation for this class was generated from the following files:

- /home/rhydera/epicsqt/framework/widgets/include/UserMessage.h
- /home/rhydera/epicsqt/framework/widgets/src/UserMessage.cpp

## 9.114 UserMessageSlot Class Reference

```
#include <UserMessage.h>
```

## Public Slots

- void [message](#) (QString msg, message\_types type, unsigned int formId, unsigned int sourceId, [UserMessage](#) \*originator)

*A message has been received.*

## Public Member Functions

- void [setOwner](#) ([UserMessage](#) \*ownerIn)  
*Set the [UserMessage](#) class this is a part of.*

### 9.114.1 Detailed Description

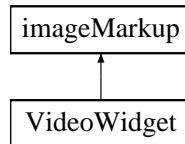
Class used to receive message signals. Used only within UserMessage.cpp An instance of this class is created by all instances of the [UserMessage](#) class. The [UserMessage](#) class uses an instance of this class to receive messages so it does not have to be based on QObject itself. This is required as derived classes generally need to be also based on another object derived from QObject (and QObject can only be the base of a single base class)

The documentation for this class was generated from the following files:

- /home/rhydera/epicsqt/framework/widgets/include/UserMessage.h
- /home/rhydera/epicsqt/framework/widgets/src/UserMessage.cpp

## 9.115 VideoWidget Class Reference

Inheritance diagram for VideoWidget:



### Signals

- void **userSelection** (imageMarkup::markupIds mode, bool complete, bool clearing, QPoint point1, QPoint point2)
- void **zoomInOut** (int zoomAmount)
- void **currentPixelInfo** (QPoint pos)
- void **pan** (QPoint pos)

### Public Member Functions

- **VideoWidget** (QWidget \*parent=0)
- void **setNewImage** (const QImage image, [QCaDateTime](#) &time)
- void **setPanning** (bool panningIn)
- bool **getPanning** ()
- QPoint **scalePoint** (QPoint pnt)
- int **scaleOrdinate** (int ord)
- QImage **getImage** ()

### Protected Member Functions

- void **paintEvent** (QPaintEvent \*)
- void **mousePressEvent** (QMouseEvent \*event)
- void **mouseReleaseEvent** (QMouseEvent \*event)
- void **mouseMoveEvent** (QMouseEvent \*event)
- void **wheelEvent** (QWheelEvent \*event)
- void **markupChange** (QImage &markups, QVector< QRect > &changedAreas)
- void **resizeEvent** (QResizeEvent \*event)
- void **markupSetCursor** (QCursor cursor)
- void **markupAction** (markupIds mode, bool complete, bool clearing, QPoint point1, QPoint point2)

The documentation for this class was generated from the following files:

- /home/rhydera/epicsqt/framework/widgets/QEImage/videowidget.h
- /home/rhydera/epicsqt/framework/widgets/QEImage/videowidget.cpp

## 9.116 WidgetRef Class Reference

### Public Member Functions

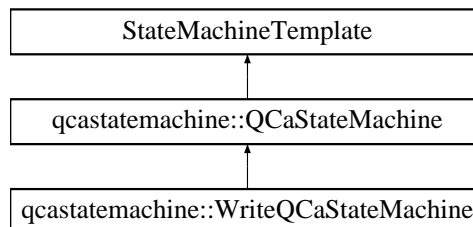
- **WidgetRef** ([QEWidget](#) \*refIn)
- [QEWidget](#) \* **getRef** ()

The documentation for this class was generated from the following file:

- /home/rhydera/epicsqt/framework/widgets/include/ContainerProfile.h

## 9.117 qcastatemachine::WriteQCaStateMachine Class Reference

Inheritance diagram for qcastatemachine::WriteQCaStateMachine:



### Public Member Functions

- **WriteQCaStateMachine** (void \*parent)
- bool **process** (int requestedState)

The documentation for this class was generated from the following files:

- /home/rhydera/epicsqt/framework/data/include/QCaStateMachine.h
- /home/rhydera/epicsqt/framework/data/src/QCaStateMachine.cpp

## 9.118 zoomMenu Class Reference

### Public Member Functions

- **zoomMenu** (QWidget \*parent=0)
- void **enableAreaSelected** (bool enable)
- imageContextMenu::imageContextMenuOptions **getZoom** (const QPoint &pos)

The documentation for this class was generated from the following files:

- /home/rhydera/epicsqt/framework/widgets/QEImage/zoomMenu.h
- /home/rhydera/epicsqt/framework/widgets/QEImage/zoomMenu.cpp



# Index

- [\\_Field](#), [27](#)
- [\\_Item](#), [28](#)
- [\\_QDialogItem](#), [28](#)
- [\\_QDialogLogin](#), [28](#)
- [\\_QPushButtonGroup](#), [29](#)
- [\\_QTableWidgetFileBrowser](#), [29](#)
- [\\_QTableWidgetLog](#), [30](#)
- [\\_QTableWidgetScript](#), [30](#)
- activate
  - [QEObject](#), [232](#)
- addUnits
  - [QEObjectProgressbar](#), [66](#)
  - [QEObjectLabel](#), [124](#)
  - [QEObjectLineEdit](#), [134](#)
  - [QEObjectPushButton](#), [166](#)
  - [QEObjectRadioButton](#), [185](#)
- alarmSeverityDisplayMode
  - [QEObjectProgressbar](#), [66](#)
- alignment
  - [QEObjectPushButton](#), [166](#)
  - [QEObjectRadioButton](#), [185](#)
- allowDrop
  - [QEObjectProgressbar](#), [66](#)
  - [QEObjectBitStatus](#), [73](#)
  - [QEObjectComboBox](#), [78](#)
  - [QEObjectFrame](#), [91](#)
  - [QEObjectGroupBox](#), [96](#)
  - [QEObjectImage](#), [109](#)
  - [QEObjectLabel](#), [124](#)
  - [QEObjectLineEdit](#), [134](#)
  - [QEObjectPeriodic](#), [149](#)
  - [QEObjectPlot](#), [158](#)
  - [QEObjectPushButton](#), [167](#)
  - [QEObjectPvProperties](#), [177](#)
  - [QEObjectRadioButton](#), [185](#)
  - [QEObjectShape](#), [203](#)
  - [QEObjectSlider](#), [213](#)
  - [QEObjectSpinBox](#), [218](#)
- altReadbackVariable
  - [QEObjectPushButton](#), [167](#)
- animation1
  - [QEObjectShape](#), [203](#)
- animation2
  - [QEObjectShape](#), [203](#)
- animation3
  - [QEObjectShape](#), [203](#)
- animation4
  - [QEObjectShape](#), [203](#)
- animation5
  - [QEObjectShape](#), [203](#)
- animation6
  - [QEObjectShape](#), [203](#)
- animationOptions
  - [QEObjectShape](#), [201](#)
- APPEND
  - [QEObjectStringFormatting](#), [222](#)
- Append
  - [QEObjectAnalogProgressbar](#), [64](#)
  - [QEObjectLabel](#), [122](#)
  - [QEObjectLineEdit](#), [131](#)
  - [QEObjectPushButton](#), [164](#)
  - [QEObjectRadioButton](#), [182](#)
- areaColor
  - [QEObjectImage](#), [109](#)
- arguments
  - [QEObjectPushButton](#), [167](#)
  - [QEObjectRadioButton](#), [185](#)
- arrayAction
  - [QEObjectAnalogProgressbar](#), [66](#)
  - [QEObjectLabel](#), [124](#)
  - [QEObjectLineEdit](#), [134](#)
  - [QEObjectPushButton](#), [167](#)
  - [QEObjectRadioButton](#), [186](#)
- ArrayActions
  - [QEObjectAnalogProgressbar](#), [64](#)
  - [QEObjectLabel](#), [122](#)
  - [QEObjectLineEdit](#), [131](#)
  - [QEObjectPushButton](#), [163](#)
  - [QEObjectRadioButton](#), [182](#)
- arrayActions
  - [QEObjectStringFormatting](#), [222](#)

- ASCII
  - QStringFormatting, [222](#)
- Ascii
  - QEAAnalogProgressBar, [64](#)
  - QELabel, [122](#)
  - QELineEdit, [131](#)
  - QEPushButton, [164](#)
  - QERadioButton, [182](#)
- Automatic
  - QEAAnalogProgressBar, [65](#)
  - QELabel, [122](#)
  - QELineEdit, [132](#)
  - QEPushButton, [164](#)
  - QERadioButton, [183](#)
- backgroundColour
  - QEAAnalogIndicator, [60](#)
- Bar
  - QEAAnalogIndicator, [59](#)
- beamColor
  - QEImage, [109](#)
- beamXVariable
  - QEImage, [109](#)
- beamYVariable
  - QEImage, [109](#)
- borderColour
  - QEAAnalogIndicator, [60](#)
- Bottom\_To\_Top
  - QEAAnalogIndicator, [59](#)
- centreAngle
  - QEAAnalogIndicator, [60](#)
- clickCheckedText
  - QEPushButton, [167](#)
  - QERadioButton, [186](#)
- clicked
  - QEPushButton, [165](#)
  - QERadioButton, [184](#)
- clickText
  - QEPushButton, [168](#)
  - QERadioButton, [186](#)
- clippingHighVariable
  - QEImage, [109](#)
- clippingLowVariable
  - QEImage, [109](#)
- clippingOnOffVariable
  - QEImage, [110](#)
- color1
  - QEShape, [204](#)
- color10
  - QEShape, [204](#)
- color2
  - QEShape, [204](#)
- color3
  - QEShape, [204](#)
- color4
  - QEShape, [204](#)
- color5
  - QEShape, [204](#)
- color6
  - QEShape, [204](#)
- color7
  - QEShape, [204](#)
- color8
  - QEShape, [204](#)
- color9
  - QEShape, [205](#)
- confirmAction
  - QEPushButton, [168](#)
  - QERadioButton, [186](#)
- confirmWrite
  - QELineEdit, [135](#)
- ContainerProfile, [31](#)
- contextMenu, [33](#)
- contextMenuObject, [35](#)
- creationOption
  - QEPushButton, [168](#)
  - QERadioButton, [186](#)
- CreationOptionNames
  - QEPushButton, [164](#)
  - QERadioButton, [182](#)
- dbElementChanged
  - QEPeriodic, [148](#)
- dbValueChanged
  - QEAAnalogProgressBar, [66](#)
  - QEBitStatus, [72](#)
  - QEComboBox, [78](#)
  - QEImage, [108](#)
  - QELabel, [123](#)
  - QELineEdit, [133](#)
  - QEPeriodic, [148](#)
  - QEPlot, [157](#)
  - QEPushButton, [165](#)
  - QERadioButton, [184](#)
  - QESlider, [212](#)
  - QESpinBox, [217](#)
- dbValueChanged1
  - QEShape, [202](#)
- dbValueChanged2
  - QEShape, [202](#)

- QEShape, [202](#)
- dbValueChanged3
  - QEShape, [202](#)
- dbValueChanged4
  - QEShape, [202](#)
- dbValueChanged5
  - QEShape, [202](#)
- dbValueChanged6
  - QEShape, [202](#)
- Default
  - QEAAnalogProgressBar, [65](#)
  - QELabel, [122](#)
  - QELineEdit, [132](#)
  - QEPushButton, [164](#)
  - QERadioButton, [183](#)
- defaultFileLocation
  - QEWidget, [232](#)
- displayButtonBar
  - QEImage, [109](#)
- enabled
  - QEAAnalogProgressBar, [67](#)
  - QEBitStatus, [73](#)
  - QECombobox, [79](#)
  - QEFrame, [91](#)
  - QEGroupBox, [96](#)
  - QEImage, [110](#)
  - QELabel, [124](#)
  - QELineEdit, [135](#)
  - QEPeriodic, [149](#)
  - QEPlot, [158](#)
  - QEPushButton, [168](#)
  - QEPvProperties, [177](#)
  - QERadioButton, [187](#)
  - QEShape, [205](#)
  - QESlider, [213](#)
  - QESpinBox, [218](#)
- enableHozSliceSelection
  - QEImage, [110](#)
- enableVertSliceSelection
  - QEImage, [110](#)
- Engineer
  - QEAAnalogProgressBar, [65](#)
  - QEBitStatus, [72](#)
  - QECombobox, [78](#)
  - QEFrame, [91](#)
  - QEGroupBox, [96](#)
  - QEImage, [108](#)
  - QELabel, [123](#)
  - QELineEdit, [132](#)
  - QEPeriodic, [148](#)
  - QEPlot, [157](#)
  - QEPushButton, [165](#)
  - QEPvProperties, [176](#)
  - QERadioButton, [184](#)
  - QEShape, [201](#)
  - QESlider, [212](#)
  - QESpinBox, [217](#)
- Fit
  - QEImage, [106](#)
- Fixed
  - QEAAnalogProgressBar, [65](#)
  - QELabel, [122](#)
  - QELineEdit, [132](#)
  - QEPushButton, [164](#)
  - QERadioButton, [183](#)
- flipRotateMenu, [36](#)
- Floating
  - QEAAnalogProgressBar, [65](#)
  - QELabel, [122](#)
  - QELineEdit, [132](#)
  - QEPushButton, [164](#)
  - QERadioButton, [183](#)
- fontColour
  - QEAAnalogIndicator, [60](#)
- foregroundColour
  - QEAAnalogIndicator, [60](#)
- format
  - QEAAnalogProgressBar, [67](#)
  - QELabel, [124](#)
  - QELineEdit, [135](#)
  - QEPushButton, [168](#)
  - QERadioButton, [187](#)
- FORMAT\_DEFAULT
  - QEStringFormatting, [222](#)
- FORMAT\_FLOATING
  - QEStringFormatting, [222](#)
- FORMAT\_INTEGER
  - QEStringFormatting, [222](#)
- FORMAT\_LOCAL\_ENUMERATE
  - QEStringFormatting, [222](#)
- FORMAT\_STRING
  - QEStringFormatting, [222](#)
- FORMAT\_TIME
  - QEStringFormatting, [222](#)
- FORMAT\_UNSIGNEDINTEGER
  - QEStringFormatting, [222](#)
- formatInteger
  - QEIntegerFormatting, [117](#)

- formatIntegerArray
  - QEIntegerFormatting, 117
- formatOption
  - QEImage, 110
- FormatOptions
  - QEImage, 106
- formatOptions
  - QEImage, 106
- Formats
  - QEAnalogProgressBar, 64
  - QELabel, 122
  - QELineEdit, 131
  - QEPushButton, 164
  - QERadioButton, 183
- formats
  - QEStringFormatting, 222
- formatValue
  - QEIntegerFormatting, 117
- getColor
  - QEWidget, 232
- getConfirmWrite
  - QELineEdit, 133
- getMessageSourceId
  - QEWidget, 232
- getQcaltem
  - QEWidget, 233
- getSubscribe
  - QELineEdit, 133
- getWriteOnEnter
  - QELineEdit, 133
- getWriteOnFinish
  - QELineEdit, 133
- getWriteOnLoseFocus
  - QELineEdit, 133
- GREY12
  - QEImage, 106
- GREY16
  - QEImage, 106
- GREY8
  - QEImage, 106
- Grey\_12
  - QEImage, 106
- Grey\_16
  - QEImage, 106
- Grey\_8
  - QEImage, 106
- guiFile
  - QEPushButton, 168
  - QERadioButton, 187
- heightVariable
  - QEImage, 110
- horizontalFlip
  - QEImage, 110
- hozSliceColor
  - QEImage, 110
- Icon
  - QEPushButton, 165
  - QERadioButton, 183
- imageContextMenu, 36
- imageMarkup, 37
- imageVariable
  - QEImage, 111
- INDEX
  - QEStringFormatting, 222
- Index
  - QEAnalogProgressBar, 64
  - QELabel, 122
  - QELineEdit, 131
  - QEPushButton, 164
  - QERadioButton, 182
- initialHosScrollPos
  - QEImage, 111
- initialVertScrollPos
  - QEImage, 109
- int
  - QEAnalogProgressBar, 67
  - QEBitStatus, 73
  - QEComboBox, 79
  - QEFrame, 91
  - QEGroupBox, 97
  - QEImage, 111
  - QELabel, 125
  - QELineEdit, 135
  - QEPeriodic, 149
  - QEPlot, 158
  - QEPushButton, 169
  - QEPvProperties, 177
  - QERadioButton, 187
  - QEShape, 205
  - QESlider, 213
  - QESpinBox, 218
- Integer
  - QEAnalogProgressBar, 65
  - QELabel, 122
  - QELineEdit, 132
  - QEPushButton, 164
  - QERadioButton, 183

- labelText
  - QEPushButton, [169](#)
  - QERadioButton, [187](#)
  - QESubstitutedLabel, [228](#)
- launchGui
  - QEPushButton, [166](#)
  - QERadioButton, [184](#)
- leadingZero
  - QEAnalogProgressBar, [67](#)
  - QELabel, [125](#)
  - QELineEdit, [135](#)
  - QEPushButton, [169](#)
  - QERadioButton, [188](#)
- Left\_To\_Right
  - QEAnalogIndicator, [59](#)
- LocalEnumeration
  - QEAnalogProgressBar, [65](#)
  - QELabel, [122](#)
  - QELineEdit, [132](#)
  - QEPushButton, [164](#)
  - QERadioButton, [183](#)
- localEnumeration
  - QEAnalogProgressBar, [67](#)
  - QELabel, [125](#)
  - QELineEdit, [135](#)
  - QEPushButton, [169](#)
  - QERadioButton, [188](#)
- localEnumerationItem, [38](#)
- logScale
  - QEAnalogIndicator, [60](#)
- logScaleInterval
  - QEAnalogIndicator, [60](#)
- majorInterval
  - QEAnalogIndicator, [60](#)
- managePixmap, [39](#)
- markupBeam, [39](#)
- markupHLine, [40](#)
- markupItem, [41](#)
- markupLine, [42](#)
- markupRegion, [43](#)
- markupTarget, [44](#)
- markupText, [44](#)
- markupVLine, [45](#)
- maximum
  - QEAnalogIndicator, [60](#)
- Meter
  - QEAnalogIndicator, [59](#)
- minimum
  - QEAnalogIndicator, [60](#)
- minorInterval
  - QEAnalogIndicator, [60](#)
- mode
  - QEAnalogIndicator, [61](#)
- Modes
  - QEAnalogIndicator, [59](#)
- NewTab
  - QEPushButton, [164](#)
  - QERadioButton, [183](#)
- NewWindow
  - QEPushButton, [164](#)
  - QERadioButton, [183](#)
- NoRotation
  - QEImage, [107](#)
- notation
  - QEAnalogProgressBar, [68](#)
  - QELabel, [126](#)
  - QELineEdit, [136](#)
  - QEPushButton, [170](#)
  - QERadioButton, [188](#)
- NOTATION\_AUTOMATIC
  - QStringFormatting, [223](#)
- NOTATION\_FIXED
  - QStringFormatting, [222](#)
- NOTATION\_SCIENTIFIC
  - QStringFormatting, [222](#)
- Notations
  - QEAnalogProgressBar, [65](#)
  - QELabel, [122](#)
  - QELineEdit, [132](#)
  - QEPushButton, [164](#)
  - QERadioButton, [183](#)
- notations
  - QStringFormatting, [222](#)
- offset1
  - QEShape, [205](#)
- offset2
  - QEShape, [205](#)
- offset3
  - QEShape, [205](#)
- offset4
  - QEShape, [205](#)
- offset5
  - QEShape, [206](#)
- offset6
  - QEShape, [206](#)
- Open
  - QEPushButton, [164](#)

- QERadioButton, 183
- openQEFile
  - QEWWidget, 233
- orientation
  - QEAnalogIndicator, 61
- Orientations
  - QEAnalogIndicator, 59
- password
  - QEPushButton, 170
  - QERadioButton, 189
- PeriodicDialog, 46
- PeriodicElementSetupForm, 46
- PeriodicSetupDialog, 46
- Picture
  - QELabel, 122
- pixmap0
  - QELabel, 126
  - QEPushButton, 170
  - QERadioButton, 189
- pixmap1
  - QELabel, 126
  - QEPushButton, 170
  - QERadioButton, 189
- pixmap2
  - QELabel, 126
  - QEPushButton, 170
  - QERadioButton, 189
- pixmap3
  - QELabel, 126
  - QEPushButton, 170
  - QERadioButton, 189
- pixmap4
  - QELabel, 126
  - QEPushButton, 171
  - QERadioButton, 189
- pixmap5
  - QELabel, 126
  - QEPushButton, 171
  - QERadioButton, 189
- pixmap6
  - QELabel, 126
  - QEPushButton, 171
  - QERadioButton, 189
- pixmap7
  - QELabel, 126
  - QEPushButton, 171
  - QERadioButton, 189
- point1
  - QEShape, 206
- point10
  - QEShape, 206
- point2
  - QEShape, 206
- point3
  - QEShape, 206
- point4
  - QEShape, 206
- point5
  - QEShape, 206
- point6
  - QEShape, 206
- point7
  - QEShape, 207
- point8
  - QEShape, 207
- point9
  - QEShape, 207
- precision
  - QEAnalogProgressBar, 68
  - QELabel, 127
  - QELineEdit, 136
  - QEPushButton, 171
  - QERadioButton, 190
- pressed
  - QEPushButton, 166
  - QERadioButton, 185
- pressText
  - QEPushButton, 171
  - QERadioButton, 190
- profileColor
  - QELImage, 111
- profilePlot, 48
- program
  - QEPushButton, 171
  - QERadioButton, 190
- PushButtonSpecifications, 48
- QBitStatus, 48
- QCaAlarmInfo, 50
- QCaConnectionInfo, 51
- QCaDataPoint, 51
- QCaDataPointList, 51
- QCaDateTime, 51
- QCaEventFilter, 52
- QCaEventItem, 52
- QCaEventUpdate, 52
- QCaInstalledFiltersListItem, 53
- qcaobject::QCaObject, 53

- qcastatemachine::ConnectionQCaStateMachine, enabled, [67](#)
- [31](#)
- qcastatemachine::QCaStateMachine, [55](#)
- qcastatemachine::ReadQCaStateMachine, [234](#)
- qcastatemachine::SubscriptionQCaStateMachine, [237](#)
- qcastatemachine::WriteQCaStateMachine, [246](#)
- QCaVariableNamePropertyManager, [55](#)
- QEAnalogIndicator, [56](#)
  - backgroundColour, [60](#)
  - Bar, [59](#)
  - borderColour, [60](#)
  - Bottom\_To\_Top, [59](#)
  - centreAngle, [60](#)
  - fontColour, [60](#)
  - foregroundColour, [60](#)
  - Left\_To\_Right, [59](#)
  - logScale, [60](#)
  - logScaleInterval, [60](#)
  - majorInterval, [60](#)
  - maximum, [60](#)
  - Meter, [59](#)
  - minimum, [60](#)
  - minorInterval, [60](#)
  - mode, [61](#)
  - Modes, [59](#)
  - orientation, [61](#)
  - Orientations, [59](#)
  - Right\_To\_Left, [59](#)
  - Scale, [59](#)
  - showScale, [61](#)
  - showText, [61](#)
  - spanAngle, [61](#)
  - Top\_To\_Bottom, [59](#)
  - value, [61](#)
- QEAnalogIndicator::Band, [30](#)
- QEAnalogIndicator::BandList, [31](#)
- QEAnalogProgressBar, [61](#)
  - addUnits, [66](#)
  - alarmSeverityDisplayMode, [66](#)
  - allowDrop, [66](#)
  - Append, [64](#)
  - arrayAction, [66](#)
  - ArrayActions, [64](#)
  - Ascii, [64](#)
  - Automatic, [65](#)
  - dbValueChanged, [66](#)
  - Default, [65](#)
  - enabled, [67](#)
  - Engineer, [65](#)
  - Fixed, [65](#)
  - Floating, [65](#)
  - format, [67](#)
  - Formats, [64](#)
  - Index, [64](#)
  - int, [67](#)
  - Integer, [65](#)
  - leadingZero, [67](#)
  - LocalEnumeration, [65](#)
  - localEnumeration, [67](#)
  - notation, [68](#)
  - Notations, [65](#)
  - precision, [68](#)
  - QEAnalogProgressBar, [65](#)
  - requestEnabled, [66](#)
  - Scientific, [65](#)
  - Scientist, [65](#)
  - Time, [65](#)
  - trailingZeros, [68](#)
  - UnsignedInteger, [65](#)
  - useDbDisplayLimits, [68](#)
  - useDbPrecision, [68](#)
  - User, [65](#)
  - userLevelEnabled, [69](#)
  - userLevelEngineerStyle, [69](#)
  - UserLevels, [65](#)
  - userLevelScientistStyle, [69](#)
  - userLevelUserStyle, [69](#)
  - userLevelVisibility, [69](#)
  - variable, [70](#)
  - variableAsToolTip, [70](#)
  - variableSubstitutions, [70](#)
  - visible, [70](#)
- QEBitStatus, [70](#)
  - allowDrop, [73](#)
  - dbValueChanged, [72](#)
  - enabled, [73](#)
  - Engineer, [72](#)
  - int, [73](#)
  - requestEnabled, [72](#)
  - Scientist, [72](#)
  - setVariableNameAndSubstitutions, [72](#)
  - User, [72](#)
  - userLevelEnabled, [73](#)
  - userLevelEngineerStyle, [73](#)
  - UserLevels, [72](#)
  - userLevelScientistStyle, [74](#)
  - userLevelUserStyle, [74](#)

- userLevelVisibility, 74
  - variable, 74
  - variableAsToolTip, 74
  - variableSubstitutions, 74
  - visible, 75
- QEByteArray, 75
- QEComboBox, 76
  - allowDrop, 78
  - dbValueChanged, 78
  - enabled, 79
  - Engineer, 78
  - int, 79
  - requestEnabled, 78
  - Scientist, 78
  - subscribe, 79
  - User, 78
  - userLevelEnabled, 79
  - userLevelEngineerStyle, 79
  - UserLevels, 78
  - userLevelScientistStyle, 79
  - userLevelUserStyle, 80
  - userLevelVisibility, 80
  - variable, 80
  - variableAsToolTip, 80
  - variableSubstitutions, 80
  - visible, 80
  - writeOnChange, 78
- QEConfiguredLayout, 81
- QEConfiguredLayoutManager, 83
- QEDragDrop, 83
- QEFileBrowser, 85
- QEFloating, 86
- QEFloatingFormatting, 87
- QEForm, 88
  - setVariableNameAndSubstitutions, 89
- QEFrame, 89
  - allowDrop, 91
  - enabled, 91
  - Engineer, 91
  - int, 91
  - requestEnabled, 91
  - Scientist, 91
  - User, 91
  - userLevelEnabled, 91
  - userLevelEngineerStyle, 92
  - UserLevels, 91
  - userLevelScientistStyle, 92
  - userLevelUserStyle, 92
  - userLevelVisibility, 92
  - variableAsToolTip, 92
  - visible, 93
- QEGenericButton, 93
- QEGroupBox, 95
  - allowDrop, 96
  - enabled, 96
  - Engineer, 96
  - int, 97
  - requestEnabled, 96
  - Scientist, 96
  - User, 96
  - userLevelEnabled, 97
  - userLevelEngineerStyle, 97
  - UserLevels, 96
  - userLevelScientistStyle, 97
  - userLevelUserStyle, 97
  - userLevelVisibility, 98
  - variableAsToolTip, 98
  - visible, 98
- QEImage, 98
  - allowDrop, 109
  - areaColor, 109
  - beamColor, 109
  - beamXVariable, 109
  - beamYVariable, 109
  - clippingHighVariable, 109
  - clippingLowVariable, 109
  - clippingOnOffVariable, 110
  - dbValueChanged, 108
  - displayButtonBar, 109
  - enabled, 110
  - enableHozSliceSelection, 110
  - enableVertSliceSelection, 110
  - Engineer, 108
  - Fit, 106
  - formatOption, 110
  - FormatOptions, 106
  - formatOptions, 106
  - GREY12, 106
  - GREY16, 106
  - GREY8, 106
  - Grey\_12, 106
  - Grey\_16, 106
  - Grey\_8, 106
  - heightVariable, 110
  - horizontalFlip, 110
  - hozSliceColor, 110
  - imageVariable, 111
  - initialHosScrollPos, 111
  - initialVertScrollPos, 109
  - int, 111



NoRotation, [107](#)  
profileColor, [111](#)  
QEImage, [108](#)  
regionOfInterest1HVariable, [111](#)  
regionOfInterest1WVariable, [111](#)  
regionOfInterest1XVariable, [111](#)  
regionOfInterest1YVariable, [111](#)  
regionOfInterest2HVariable, [111](#)  
regionOfInterest2WVariable, [112](#)  
regionOfInterest2XVariable, [112](#)  
regionOfInterest2YVariable, [112](#)  
regionOfInterest3HVariable, [112](#)  
regionOfInterest3WVariable, [112](#)  
regionOfInterest3XVariable, [112](#)  
regionOfInterest3YVariable, [112](#)  
regionOfInterest4HVariable, [112](#)  
regionOfInterest4WVariable, [112](#)  
regionOfInterest4XVariable, [113](#)  
regionOfInterest4YVariable, [113](#)  
requestEnabled, [108](#)  
RESIZE\_OPTION\_FIT, [107](#)  
RESIZE\_OPTION\_ZOOM, [107](#)  
resizeOption, [113](#)  
ResizeOptions, [106](#)  
resizeOptions, [106](#)  
RGB\_888, [106](#)  
Rotate180, [107](#)  
Rotate90Left, [107](#)  
Rotate90Right, [107](#)  
rotation, [113](#)  
ROTATION\_0, [107](#)  
ROTATION\_180, [107](#)  
ROTATION\_90\_LEFT, [107](#)  
ROTATION\_90\_RIGHT, [107](#)  
RotationOptions, [107](#)  
rotationOptions, [107](#)  
Scientist, [108](#)  
selectOptions, [107](#)  
showTime, [113](#)  
SO\_AREA4, [107](#)  
SO\_BEAM, [107](#)  
SO\_HSLICE, [107](#)  
SO\_NONE, [107](#)  
SO\_PANNING, [107](#)  
SO\_PROFILE, [107](#)  
SO\_TARGET, [107](#)  
SO\_VSLICE, [107](#)  
targetColor, [113](#)  
targetTriggerVariable, [113](#)  
targetXVariable, [113](#)  
targetYVariable, [113](#)  
timeColor, [114](#)  
User, [108](#)  
userLevelEnabled, [114](#)  
userLevelEngineerStyle, [114](#)  
UserLevels, [107](#)  
userLevelScientistStyle, [114](#)  
userLevelUserStyle, [114](#)  
userLevelVisibility, [114](#)  
variableAsToolTip, [115](#)  
variableSubstitutions, [115](#)  
verticalFlip, [115](#)  
vertSliceColor, [115](#)  
visible, [115](#)  
widthVariable, [115](#)  
Zoom, [106](#)  
QEInteger, [116](#)  
QEIntegerFormatting, [116](#)  
    formatInteger, [117](#)  
    formatIntegerArray, [117](#)  
    formatValue, [117](#)  
QELabel, [118](#)  
    addUnits, [124](#)  
    allowDrop, [124](#)  
    Append, [122](#)  
    arrayAction, [124](#)  
    ArrayActions, [122](#)  
    Ascii, [122](#)  
    Automatic, [122](#)  
    dbValueChanged, [123](#)  
    Default, [122](#)  
    enabled, [124](#)  
    Engineer, [123](#)  
    Fixed, [122](#)  
    Floating, [122](#)  
    format, [124](#)  
    Formats, [122](#)  
    Index, [122](#)  
    int, [125](#)  
    Integer, [122](#)  
    leadingZero, [125](#)  
    LocalEnumeration, [122](#)  
    localEnumeration, [125](#)  
    notation, [126](#)  
    Notations, [122](#)  
    Picture, [122](#)  
    pixmap0, [126](#)  
    pixmap1, [126](#)  
    pixmap2, [126](#)  
    pixmap3, [126](#)

- pixmap4, 126
  - pixmap5, 126
  - pixmap6, 126
  - pixmap7, 126
  - precision, 127
  - QELabel, 123
  - requestEnabled, 123
  - Scientific, 122
  - Scientist, 123
  - Text, 122
  - Time, 122
  - trailingZeros, 127
  - UnsignedInteger, 122
  - UPDATE\_PIXMAP, 123
  - UPDATE\_TEXT, 123
  - updateOption, 127
  - UpdateOptions, 122
  - updateOptions, 122
  - useDbPrecision, 127
  - User, 123
  - userLevelEnabled, 127
  - userLevelEngineerStyle, 127
  - UserLevels, 123
  - userLevelScientistStyle, 127
  - userLevelUserStyle, 128
  - userLevelVisibility, 128
  - variable, 128
  - variableAsToolTip, 128
  - variableSubstitutions, 128
  - visible, 128
- QELineEdit, 129
  - addUnits, 134
  - allowDrop, 134
  - Append, 131
  - arrayAction, 134
  - ArrayActions, 131
  - Ascii, 131
  - Automatic, 132
  - confirmWrite, 135
  - dbValueChanged, 133
  - Default, 132
  - enabled, 135
  - Engineer, 132
  - Fixed, 132
  - Floating, 132
  - format, 135
  - Formats, 131
  - getConfirmWrite, 133
  - getSubscribe, 133
  - getWriteOnEnter, 133
  - getWriteOnFinish, 133
  - getWriteOnLoseFocus, 133
  - Index, 131
  - int, 135
  - Integer, 132
  - leadingZero, 135
  - LocalEnumeration, 132
  - localEnumeration, 135
  - notation, 136
  - Notations, 132
  - precision, 136
  - QELineEdit, 132
  - requestEnabled, 133
  - Scientific, 132
  - Scientist, 132
  - setConfirmWrite, 133
  - setSubscribe, 134
  - setWriteOnEnter, 134
  - setWriteOnFinish, 134
  - setWriteOnLoseFocus, 134
  - subscribe, 136
  - Time, 132
  - trailingZeros, 136
  - UnsignedInteger, 132
  - useDbPrecision, 137
  - User, 132
  - userLevelEnabled, 137
  - userLevelEngineerStyle, 137
  - UserLevels, 132
  - userLevelScientistStyle, 137
  - userLevelUserStyle, 137
  - userLevelVisibility, 137
  - variable, 138
  - variableAsToolTip, 138
  - variableSubstitutions, 138
  - visible, 138
  - writeOnEnter, 138
  - writeOnFinish, 138
  - writeOnLoseFocus, 138
- QELineEditManager, 139
- QELink, 139
- QELog, 141
- QELogin, 143
- QEPeriodic, 145
  - allowDrop, 149
  - dbElementChanged, 148
  - dbValueChanged, 148
  - enabled, 149
  - Engineer, 148
  - int, 149

- readbackLabelVariable1, 149
- readbackLabelVariable2, 149
- requestEnabled, 148
- Scientist, 148
- subscribe, 149
- User, 148
- userLevelEnabled, 149
- userLevelEngineerStyle, 150
- UserLevels, 148
- userLevelScientistStyle, 150
- userLevelUserStyle, 150
- userLevelVisibility, 150
- variableAsToolTip, 150
- variableSubstitutions, 151
- visible, 151
- writeButtonVariable1, 151
- writeButtonVariable2, 151
- QEPeiodic::elementInfoStruct, 35
- QEPeiodic::userInfoStructArray, 239
- QEPeiodicComponentData, 151
- QEPeiodicTaskMenu, 152
- QEPeiodicTaskMenuFactory, 152
- QEpicsPV, 152
- QEPlot, 153
  - allowDrop, 158
  - dbValueChanged, 157
  - enabled, 158
  - Engineer, 157
  - int, 158
  - requestEnabled, 157
  - Scientist, 157
  - User, 157
  - userLevelEnabled, 158
  - userLevelEngineerStyle, 158
  - UserLevels, 157
  - userLevelScientistStyle, 159
  - userLevelUserStyle, 159
  - userLevelVisibility, 159
  - variable1, 159
  - variable2, 159
  - variable3, 159
  - variable4, 160
  - variableAsToolTip, 160
  - variableSubstitutions, 160
  - visible, 160
- QEPushButton, 160
  - addUnits, 166
  - alignment, 166
  - allowDrop, 167
  - altReadbackVariable, 167
  - Append, 164
  - arguments, 167
  - arrayAction, 167
  - ArrayActions, 163
  - Ascii, 164
  - Automatic, 164
  - clickCheckedText, 167
  - clicked, 165
  - clickText, 168
  - confirmAction, 168
  - creationOption, 168
  - CreationOptionNames, 164
  - dbValueChanged, 165
  - Default, 164
  - enabled, 168
  - Engineer, 165
  - Fixed, 164
  - Floating, 164
  - format, 168
  - Formats, 164
  - guiFile, 168
  - Icon, 165
  - Index, 164
  - int, 169
  - Integer, 164
  - labelText, 169
  - launchGui, 166
  - leadingZero, 169
  - LocalEnumeration, 164
  - localEnumeration, 169
  - NewTab, 164
  - NewWindow, 164
  - notation, 170
  - Notations, 164
  - Open, 164
  - password, 170
  - pixmap0, 170
  - pixmap1, 170
  - pixmap2, 170
  - pixmap3, 170
  - pixmap4, 171
  - pixmap5, 171
  - pixmap6, 171
  - pixmap7, 171
  - precision, 171
  - pressed, 166
  - pressText, 171
  - program, 171
  - QEPushButton, 165
  - released, 166

- releaseText, 171
- requestEnabled, 166
- Scientific, 164
- Scientist, 165
- State, 165
- subscribe, 172
- Text, 165
- TextAndIcon, 165
- Time, 164
- trailingZeros, 172
- UnsignedInteger, 164
- updateOption, 172
- UpdateOptions, 164
- useDbPrecision, 172
- User, 165
- userLevelEnabled, 172
- userLevelEngineerStyle, 172
- UserLevels, 165
- userLevelScientistStyle, 172
- userLevelUserStyle, 173
- userLevelVisibility, 173
- variable, 173
- variableAsToolTip, 173
- variableSubstitutions, 173
- visible, 173
- writeOnClick, 174
- writeOnPress, 174
- writeOnRelease, 174
- QEPvProperties, 174
  - allowDrop, 177
  - enabled, 177
  - Engineer, 176
  - int, 177
  - requestEnabled, 176
  - Scientist, 176
  - User, 176
  - userLevelEnabled, 177
  - userLevelEngineerStyle, 177
  - UserLevels, 176
  - userLevelScientistStyle, 177
  - userLevelUserStyle, 178
  - userLevelVisibility, 178
  - variable, 178
  - variableAsToolTip, 178
  - variableSubstitutions, 178
  - visible, 178
- QEPvPropertiesManager, 179
- QERadioButton, 179
  - addUnits, 185
  - alignment, 185
  - allowDrop, 185
  - Append, 182
  - arguments, 185
  - arrayAction, 186
  - ArrayActions, 182
  - Ascii, 182
  - Automatic, 183
  - clickCheckedText, 186
  - clicked, 184
  - clickText, 186
  - confirmAction, 186
  - creationOption, 186
  - CreationOptionNames, 182
  - dbValueChanged, 184
  - Default, 183
  - enabled, 187
  - Engineer, 184
  - Fixed, 183
  - Floating, 183
  - format, 187
  - Formats, 183
  - guiFile, 187
  - Icon, 183
  - Index, 182
  - int, 187
  - Integer, 183
  - labelText, 187
  - launchGui, 184
  - leadingZero, 188
  - LocalEnumeration, 183
  - localEnumeration, 188
  - NewTab, 183
  - NewWindow, 183
  - notation, 188
  - Notations, 183
  - Open, 183
  - password, 189
  - pixmap0, 189
  - pixmap1, 189
  - pixmap2, 189
  - pixmap3, 189
  - pixmap4, 189
  - pixmap5, 189
  - pixmap6, 189
  - pixmap7, 189
  - precision, 190
  - pressed, 185
  - pressText, 190
  - program, 190
  - QERadioButton, 184

- released, [185](#)
- releaseText, [190](#)
- requestEnabled, [185](#)
- Scientific, [183](#)
- Scientist, [184](#)
- State, [183](#)
- subscribe, [190](#)
- Text, [183](#)
- TextAndIcon, [183](#)
- Time, [183](#)
- trailingZeros, [190](#)
- UnsignedInteger, [183](#)
- updateOption, [190](#)
- UpdateOptions, [183](#)
- useDbPrecision, [190](#)
- User, [184](#)
- userLevelEnabled, [191](#)
- userLevelEngineerStyle, [191](#)
- UserLevels, [183](#)
- userLevelScientistStyle, [191](#)
- userLevelUserStyle, [191](#)
- userLevelVisibility, [191](#)
- variable, [192](#)
- variableAsToolTip, [192](#)
- variableSubstitutions, [192](#)
- visible, [192](#)
- writeOnClick, [192](#)
- writeOnPress, [192](#)
- writeOnRelease, [192](#)
- QERecipe, [193](#)
- QEScript, [195](#)
- QEShape, [197](#)
  - allowDrop, [203](#)
  - animation1, [203](#)
  - animation2, [203](#)
  - animation3, [203](#)
  - animation4, [203](#)
  - animation5, [203](#)
  - animation6, [203](#)
  - animationOptions, [201](#)
  - color1, [204](#)
  - color10, [204](#)
  - color2, [204](#)
  - color3, [204](#)
  - color4, [204](#)
  - color5, [204](#)
  - color6, [204](#)
  - color7, [204](#)
  - color8, [204](#)
  - color9, [205](#)
  - dbValueChanged1, [202](#)
  - dbValueChanged2, [202](#)
  - dbValueChanged3, [202](#)
  - dbValueChanged4, [202](#)
  - dbValueChanged5, [202](#)
  - dbValueChanged6, [202](#)
  - enabled, [205](#)
  - Engineer, [201](#)
  - int, [205](#)
  - offset1, [205](#)
  - offset2, [205](#)
  - offset3, [205](#)
  - offset4, [205](#)
  - offset5, [206](#)
  - offset6, [206](#)
  - point1, [206](#)
  - point10, [206](#)
  - point2, [206](#)
  - point3, [206](#)
  - point4, [206](#)
  - point5, [206](#)
  - point6, [206](#)
  - point7, [207](#)
  - point8, [207](#)
  - point9, [207](#)
  - QEShape, [202](#)
  - requestEnabled, [203](#)
  - scale2, [207](#)
  - scale3, [207](#)
  - scale4, [207](#)
  - scale5, [207](#)
  - scale6, [207](#)
  - Scientist, [201](#)
  - shapeOptions, [201](#)
  - User, [201](#)
  - userLevelEnabled, [207](#)
  - userLevelEngineerStyle, [208](#)
  - UserLevels, [201](#)
  - userLevelScientistStyle, [208](#)
  - userLevelUserStyle, [208](#)
  - userLevelVisibility, [208](#)
  - variable1, [208](#)
  - variable2, [209](#)
  - variable3, [209](#)
  - variable4, [209](#)
  - variable5, [209](#)
  - variable6, [209](#)
  - variableAsToolTip, [209](#)
  - variableSubstitutions, [209](#)
  - visible, [210](#)

- QESlider, 210
  - allowDrop, 213
  - dbValueChanged, 212
  - enabled, 213
  - Engineer, 212
  - int, 213
  - requestEnabled, 212
  - Scientist, 212
  - subscribe, 213
  - User, 212
  - userLevelEnabled, 213
  - userLevelEngineerStyle, 214
  - UserLevels, 212
  - userLevelScientistStyle, 214
  - userLevelUserStyle, 214
  - userLevelVisibility, 214
  - variable, 214
  - variableAsToolTip, 214
  - variableSubstitutions, 215
  - visible, 215
  - writeOnChange, 213
- QESpinBox, 215
  - allowDrop, 218
  - dbValueChanged, 217
  - enabled, 218
  - Engineer, 217
  - int, 218
  - requestEnabled, 217
  - Scientist, 217
  - subscribe, 218
  - User, 217
  - userLevelEnabled, 218
  - userLevelEngineerStyle, 218
  - UserLevels, 217
  - userLevelScientistStyle, 219
  - userLevelUserStyle, 219
  - userLevelVisibility, 219
  - variable, 219
  - variableAsToolTip, 219
  - variableSubstitutions, 219
  - visible, 220
- QString, 220
- QStringFormatting, 221
  - APPEND, 222
  - arrayActions, 222
  - ASCII, 222
  - FORMAT\_DEFAULT, 222
  - FORMAT\_FLOATING, 222
  - FORMAT\_INTEGER, 222
  - FORMAT\_LOCAL\_ENUMERATE, 222
  - FORMAT\_STRING, 222
  - FORMAT\_TIME, 222
  - FORMAT\_UNSIGNEDINTEGER, 222
  - formats, 222
  - INDEX, 222
  - NOTATION\_AUTOMATIC, 223
  - NOTATION\_FIXED, 222
  - NOTATION\_SCIENTIFIC, 222
  - notations, 222
- QStringFormattingMethods, 223
- QStripChart, 224
- QStripChart::PrivateData, 47
- QStripChartItem, 226
- QStripChartItem::PrivateData, 47
- QStripChartItemDialog, 226
- QStripChartTimeDialog, 227
- QSubstitutedLabel, 227
  - labelText, 228
  - textSubstitutions, 228
- QEToolTip, 228
- QEWWidget, 230
  - activate, 232
  - defaultFileLocation, 232
  - getColor, 232
  - getMessageSourceId, 232
  - getQcItem, 233
  - openQEFile, 233
  - readNow, 233
  - setMessageSourceId, 233
  - setupContextMenu, 233
  - setVariableNameAndSubstitutions, 233
  - writeNow, 233
- QEWWidgets, 234
- readbackLabelVariable1
  - QEPeiodic, 149
- readbackLabelVariable2
  - QEPeiodic, 149
- readNow
  - QEWWidget, 233
- RecordSpec, 234
- RecordSpecList, 235
- regionOfInterest1HVariable
  - QEImage, 111
- regionOfInterest1WVariable
  - QEImage, 111
- regionOfInterest1XVariable
  - QEImage, 111
- regionOfInterest1YVariable
  - QEImage, 111

- regionOfInterest2HVariable
  - QEImage, [111](#)
- regionOfInterest2WVVariable
  - QEImage, [112](#)
- regionOfInterest2XVVariable
  - QEImage, [112](#)
- regionOfInterest2YVVariable
  - QEImage, [112](#)
- regionOfInterest3HVariable
  - QEImage, [112](#)
- regionOfInterest3WVVariable
  - QEImage, [112](#)
- regionOfInterest3XVVariable
  - QEImage, [112](#)
- regionOfInterest3YVVariable
  - QEImage, [112](#)
- regionOfInterest4HVariable
  - QEImage, [112](#)
- regionOfInterest4WVVariable
  - QEImage, [112](#)
- regionOfInterest4XVVariable
  - QEImage, [113](#)
- regionOfInterest4YVVariable
  - QEImage, [113](#)
- released
  - QEPushButton, [166](#)
  - QERadioButton, [185](#)
- releaseText
  - QEPushButton, [171](#)
  - QERadioButton, [190](#)
- requestEnabled
  - QEAnalogProgressBar, [66](#)
  - QEBitStatus, [72](#)
  - QECombobox, [78](#)
  - QEFrame, [91](#)
  - QEGroupBox, [96](#)
  - QEImage, [108](#)
  - QELabel, [123](#)
  - QELineEdit, [133](#)
  - QEPeriodic, [148](#)
  - QEPlot, [157](#)
  - QEPushButton, [166](#)
  - QEPvProperties, [176](#)
  - QERadioButton, [185](#)
  - QEShape, [203](#)
  - QESlider, [212](#)
  - QESpinBox, [217](#)
- RESIZE\_OPTION\_FIT
  - QEImage, [107](#)
- RESIZE\_OPTION\_ZOOM
  - QEImage, [107](#)
- resizeOption
  - QEImage, [113](#)
- ResizeOptions
  - QEImage, [106](#)
- resizeOptions
  - QEImage, [106](#)
- RGB\_888
  - QEImage, [106](#)
- Right\_To\_Left
  - QEAnalogIndicator, [59](#)
- Rotate180
  - QEImage, [107](#)
- Rotate90Left
  - QEImage, [107](#)
- Rotate90Right
  - QEImage, [107](#)
- rotation
  - QEImage, [113](#)
- ROTATION\_0
  - QEImage, [107](#)
- ROTATION\_180
  - QEImage, [107](#)
- ROTATION\_90\_LEFT
  - QEImage, [107](#)
- ROTATION\_90\_RIGHT
  - QEImage, [107](#)
- RotationOptions
  - QEImage, [107](#)
- rotationOptions
  - QEImage, [107](#)
- Scale
  - QEAnalogIndicator, [59](#)
- scale2
  - QEShape, [207](#)
- scale3
  - QEShape, [207](#)
- scale4
  - QEShape, [207](#)
- scale5
  - QEShape, [207](#)
- scale6
  - QEShape, [207](#)
- Scientific
  - QEAnalogProgressBar, [65](#)
  - QELabel, [122](#)
  - QELineEdit, [132](#)
  - QEPushButton, [164](#)
  - QERadioButton, [183](#)

- Scientist
  - QEAAnalogProgressBar, 65
  - QEBitStatus, 72
  - QEComboBox, 78
  - QEFrame, 91
  - QEGroupBox, 96
  - QEImage, 108
  - QELabel, 123
  - QELineEdit, 132
  - QEPeriodic, 148
  - QEPlot, 157
  - QEPushButton, 165
  - QEPvProperties, 176
  - QERadioButton, 184
  - QEShape, 201
  - QESlider, 212
  - QESpinBox, 217
- selectMenu, 235
- selectOptions
  - QEImage, 107
- setConfirmWrite
  - QELineEdit, 133
- setMessageSourceId
  - QEWidget, 233
- setSubscribe
  - QELineEdit, 134
- setupContextMenu
  - QEWidget, 233
- setVariableNameAndSubstitutions
  - QEBitStatus, 72
  - QEForm, 89
  - QEWidget, 233
- setWriteOnEnter
  - QELineEdit, 134
- setWriteOnFinish
  - QELineEdit, 134
- setWriteOnLoseFocus
  - QELineEdit, 134
- shapeOptions
  - QEShape, 201
- showScale
  - QEAAnalogIndicator, 61
- showText
  - QEAAnalogIndicator, 61
- showTime
  - QEImage, 113
- SO\_AREA4
  - QEImage, 107
- SO\_BEAM
  - QEImage, 107
- SO\_HSLICE
  - QEImage, 107
- SO\_NONE
  - QEImage, 107
- SO\_PANNING
  - QEImage, 107
- SO\_PROFILE
  - QEImage, 107
- SO\_TARGET
  - QEImage, 107
- SO\_VSLICE
  - QEImage, 107
- spanAngle
  - QEAAnalogIndicator, 61
- standardProperties, 235
- State
  - QEPushButton, 165
  - QERadioButton, 183
- StateMachineTemplate, 237
- subscribe
  - QEComboBox, 79
  - QELineEdit, 136
  - QEPeriodic, 149
  - QEPushButton, 172
  - QERadioButton, 190
  - QESlider, 213
  - QESpinBox, 218
- targetColor
  - QEImage, 113
- targetTriggerVariable
  - QEImage, 113
- targetXVariable
  - QEImage, 113
- targetYVariable
  - QEImage, 113
- Text
  - QELabel, 122
  - QEPushButton, 165
  - QERadioButton, 183
- TextAndIcon
  - QEPushButton, 165
  - QERadioButton, 183
- textSubstitutions
  - QESubstitutedLabel, 228
- Time
  - QEAAnalogProgressBar, 65
  - QELabel, 122
  - QELineEdit, 132
  - QEPushButton, 164



- QERadioButton, 183
- timeColor
  - QEImage, 114
- Top\_To\_Bottom
  - QEAnalogIndicator, 59
- trace, 238
- TrackRange, 238
- trailingZeros
  - QEAnalogProgressBar, 68
  - QELabel, 127
  - QELineEdit, 136
  - QEPushButton, 172
  - QERadioButton, 190
- UnsignedInteger
  - QEAnalogProgressBar, 65
  - QELabel, 122
  - QELineEdit, 132
  - QEPushButton, 164
  - QERadioButton, 183
- UPDATE\_PIXMAP
  - QELabel, 123
- UPDATE\_TEXT
  - QELabel, 123
- updateOption
  - QELabel, 127
  - QEPushButton, 172
  - QERadioButton, 190
- UpdateOptions
  - QELabel, 122
  - QEPushButton, 164
  - QERadioButton, 183
- updateOptions
  - QELabel, 122
- useDbDisplayLimits
  - QEAnalogProgressBar, 68
- useDbPrecision
  - QEAnalogProgressBar, 68
  - QELabel, 127
  - QELineEdit, 137
  - QEPushButton, 172
  - QERadioButton, 190
- User
  - QEAnalogProgressBar, 65
  - QEBitStatus, 72
  - QEComboBox, 78
  - QEFrmae, 91
  - QEGroupBox, 96
  - QEImage, 108
  - QELabel, 123
  - QELineEdit, 132
  - QEPeriodic, 148
  - QEPlot, 157
  - QEPushButton, 165
  - QEPvProperties, 176
  - QERadioButton, 184
  - QEShape, 201
  - QESlider, 212
  - QESpinBox, 217
- userInfoStruct, 239
- userLevelEnabled
  - QEAnalogProgressBar, 69
  - QEBitStatus, 73
  - QEComboBox, 79
  - QEFrmae, 91
  - QEGroupBox, 97
  - QEImage, 114
  - QELabel, 127
  - QELineEdit, 137
  - QEPeriodic, 149
  - QEPlot, 158
  - QEPushButton, 172
  - QEPvProperties, 177
  - QERadioButton, 191
  - QEShape, 207
  - QESlider, 213
  - QESpinBox, 218
- userLevelEngineerStyle
  - QEAnalogProgressBar, 69
  - QEBitStatus, 73
  - QEComboBox, 79
  - QEFrmae, 92
  - QEGroupBox, 97
  - QEImage, 114
  - QELabel, 127
  - QELineEdit, 137
  - QEPeriodic, 150
  - QEPlot, 158
  - QEPushButton, 172
  - QEPvProperties, 177
  - QERadioButton, 191
  - QEShape, 208
  - QESlider, 214
  - QESpinBox, 218
- UserLevels
  - QEAnalogProgressBar, 65
  - QEBitStatus, 72
  - QEComboBox, 78
  - QEFrmae, 91
  - QEGroupBox, 96

- QEImage, [107](#)
- QELabel, [123](#)
- QELineEdit, [132](#)
- QEPeriodic, [148](#)
- QEPlot, [157](#)
- QEPushButton, [165](#)
- QEPvProperties, [176](#)
- QERadioButton, [183](#)
- QEShape, [201](#)
- QESlider, [212](#)
- QESpinBox, [217](#)
- userLevelScientistStyle
  - QEAnalogProgressBar, [69](#)
  - QEBitStatus, [74](#)
  - QEComboBox, [79](#)
  - QEFrame, [92](#)
  - QEGroupBox, [97](#)
  - QEImage, [114](#)
  - QELabel, [127](#)
  - QELineEdit, [137](#)
  - QEPeriodic, [150](#)
  - QEPlot, [159](#)
  - QEPushButton, [172](#)
  - QEPvProperties, [177](#)
  - QERadioButton, [191](#)
  - QEShape, [208](#)
  - QESlider, [214](#)
  - QESpinBox, [219](#)
- userLevelSignal, [239](#)
- userLevelSlot, [240](#)
- userLevelUserStyle
  - QEAnalogProgressBar, [69](#)
  - QEBitStatus, [74](#)
  - QEComboBox, [80](#)
  - QEFrame, [92](#)
  - QEGroupBox, [97](#)
  - QEImage, [114](#)
  - QELabel, [128](#)
  - QELineEdit, [137](#)
  - QEPeriodic, [150](#)
  - QEPlot, [159](#)
  - QEPushButton, [173](#)
  - QEPvProperties, [178](#)
  - QERadioButton, [191](#)
  - QEShape, [208](#)
  - QESlider, [214](#)
  - QESpinBox, [219](#)
- userLevelVisibility
  - QEAnalogProgressBar, [69](#)
  - QEBitStatus, [74](#)
- QEComboBox, [80](#)
- QEFrame, [92](#)
- QEGroupBox, [98](#)
- QEImage, [114](#)
- QELabel, [128](#)
- QELineEdit, [137](#)
- QEPeriodic, [150](#)
- QEPlot, [159](#)
- QEPushButton, [173](#)
- QEPvProperties, [178](#)
- QERadioButton, [191](#)
- QEShape, [208](#)
- QESlider, [214](#)
- QESpinBox, [219](#)
- UserMessage, [240](#)
- UserMessageSignal, [243](#)
- UserMessageSlot, [244](#)
- value
  - QEAnalogIndicator, [61](#)
- variable
  - QEAnalogProgressBar, [70](#)
  - QEBitStatus, [74](#)
  - QEComboBox, [80](#)
  - QELabel, [128](#)
  - QELineEdit, [138](#)
  - QEPushButton, [173](#)
  - QEPvProperties, [178](#)
  - QERadioButton, [192](#)
  - QESlider, [214](#)
  - QESpinBox, [219](#)
- variable1
  - QEPlot, [159](#)
  - QEShape, [208](#)
- variable2
  - QEPlot, [159](#)
  - QEShape, [209](#)
- variable3
  - QEPlot, [159](#)
  - QEShape, [209](#)
- variable4
  - QEPlot, [160](#)
  - QEShape, [209](#)
- variable5
  - QEShape, [209](#)
- variable6
  - QEShape, [209](#)
- variableAsToolTip
  - QEAnalogProgressBar, [70](#)
  - QEBitStatus, [74](#)

- QComboBox, 80
- QFrame, 92
- QGroupBox, 98
- QImage, 115
- QLabel, 128
- QLineEdit, 138
- QPeriodic, 150
- QPlot, 160
- QPushButton, 173
- QEPvProperties, 178
- QERadioButton, 192
- QEShape, 209
- QESlider, 214
- QESpinBox, 219
- variableSubstitutions
  - QEAAnalogProgressBar, 70
  - QEBitStatus, 74
  - QComboBox, 80
  - QImage, 115
  - QLabel, 128
  - QLineEdit, 138
  - QPeriodic, 151
  - QPlot, 160
  - QPushButton, 173
  - QEPvProperties, 178
  - QERadioButton, 192
  - QEShape, 209
  - QESlider, 215
  - QESpinBox, 219
- verticalFlip
  - QImage, 115
- vertSliceColor
  - QImage, 115
- VideoWidget, 245
- visible
  - QEAAnalogProgressBar, 70
  - QEBitStatus, 75
  - QComboBox, 80
  - QFrame, 93
  - QGroupBox, 98
  - QImage, 115
  - QLabel, 128
  - QLineEdit, 138
  - QPeriodic, 151
  - QPlot, 160
  - QPushButton, 173
  - QEPvProperties, 178
  - QERadioButton, 192
  - QEShape, 210
  - QESlider, 215
  - QESpinBox, 220
- WidgetRef, 246
- widthVariable
  - QImage, 115
- writeButtonVariable1
  - QPeriodic, 151
- writeButtonVariable2
  - QPeriodic, 151
- writeNow
  - QEWWidget, 233
- writeOnChange
  - QComboBox, 78
  - QESlider, 213
- writeOnClick
  - QPushButton, 174
  - QERadioButton, 192
- writeOnEnter
  - QLineEdit, 138
- writeOnFinish
  - QLineEdit, 138
- writeOnLoseFocus
  - QLineEdit, 138
- writeOnPress
  - QPushButton, 174
  - QERadioButton, 192
- writeOnRelease
  - QPushButton, 174
  - QERadioButton, 192
- Zoom
  - QImage, 106
- zoomMenu, 246