

## EPICS QT Framework

3.2.4

Generated by Doxygen 1.7.4

Tue Dec 22 2015 17:06:23



# Contents

<b>1 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 GNU General Public License</b>	<b>5</b>
<b>3 ASgui screen shots</b>	<b>7</b>
<b>4 other applications using epicsqt widgets</b>	<b>13</b>
<b>5 Qt Designer</b>	<b>15</b>
<b>6 Qt Creator</b>	<b>17</b>
<b>7 Class Index</b>	<b>19</b>
7.1 Class Hierarchy . . . . .	19
<b>8 Class Index</b>	<b>23</b>
8.1 Class List . . . . .	23
<b>9 Class Documentation</b>	<b>27</b>
9.1 _CopyPaste Class Reference . . . . .	27

9.2	_Field Class Reference . . . . .	27
9.3	_Item Class Reference . . . . .	28
9.4	_QDialogItem Class Reference . . . . .	29
9.5	_QPushButtonGroup Class Reference . . . . .	29
9.6	_QTableWidgetFileBrowser Class Reference . . . . .	29
9.7	_QTableWidgetLog Class Reference . . . . .	30
9.8	_QTableWidgetScript Class Reference . . . . .	30
9.9	arealInfo Class Reference . . . . .	30
9.10	QEAnalogIndicator::Band Struct Reference . . . . .	31
9.11	QEAnalogIndicator::BandList Class Reference . . . . .	31
9.12	QEPeriodic::elementInfoStruct Struct Reference . . . . .	31
9.13	FFBuffer Class Reference . . . . .	32
9.14	FFThread Class Reference . . . . .	32
9.15	flipRotateMenu Class Reference . . . . .	33
9.16	fullScreenWindow Class Reference . . . . .	33
9.17	histogram Class Reference . . . . .	34
9.18	histogramScroll Class Reference . . . . .	34
9.19	historicImage Class Reference . . . . .	34
9.20	imageContextMenu Class Reference . . . . .	35
9.21	imageDisplayProperties Class Reference . . . . .	36
9.22	imageInfo Class Reference . . . . .	37
9.23	imageMarkup Class Reference . . . . .	38
9.24	imageMarkupLegendSetText Class Reference . . . . .	40
9.25	imageProcessor Class Reference . . . . .	41
9.25.1	Detailed Description . . . . .	43
9.25.2	Member Function Documentation . . . . .	44
9.25.2.1	getPixelValueFromData . . . . .	44
9.26	imageProperties Class Reference . . . . .	44
9.26.1	Detailed Description . . . . .	46
9.26.2	Member Enumeration Documentation . . . . .	46
9.26.2.1	rotationOptions . . . . .	46
9.26.3	Constructor & Destructor Documentation . . . . .	46
9.26.3.1	imageProperties . . . . .	46
9.27	imagePropertiesCore Class Reference . . . . .	46

9.27.1 Member Function Documentation . . . . .	47
9.27.1.1 buildImageCore . . . . .	47
9.28 imageUpdateIndicator Class Reference . . . . .	47
9.29 loginWidget Class Reference . . . . .	47
9.30 markupCrosshair1 Class Reference . . . . .	48
9.31 markupCrosshair2 Class Reference . . . . .	48
9.32 markupDisplayMenu Class Reference . . . . .	49
9.33 markupEllipse Class Reference . . . . .	49
9.34 markupHLine Class Reference . . . . .	50
9.34.1 Member Function Documentation . . . . .	51
9.34.1.1 drawMarkup . . . . .	51
9.35 markupItem Class Reference . . . . .	51
9.36 markupLine Class Reference . . . . .	54
9.37 markupRegion Class Reference . . . . .	54
9.38 markupText Class Reference . . . . .	55
9.39 markupVLine Class Reference . . . . .	56
9.39.1 Member Function Documentation . . . . .	56
9.39.1.1 drawMarkup . . . . .	56
9.40 mpegSource Class Reference . . . . .	57
9.40.1 Member Function Documentation . . . . .	57
9.40.1.1 updatelImage . . . . .	57
9.41 mpegSourceObject Class Reference . . . . .	57
9.42 QEStripChartToolBar::OwnWidgets Class Reference . . . . .	58
9.43 PeriodicDialog Class Reference . . . . .	58
9.44 PeriodicElementSetupForm Class Reference . . . . .	59
9.45 PeriodicSetupDialog Class Reference . . . . .	59
9.46 playbackTimer Class Reference . . . . .	59
9.47 pointInfo Class Reference . . . . .	60
9.48 profilePlot Class Reference . . . . .	60
9.49 QBitStatus Class Reference . . . . .	61
9.50 QEAnalogIndicator Class Reference . . . . .	63
9.50.1 Detailed Description . . . . .	66
9.50.2 Member Enumeration Documentation . . . . .	66
9.50.2.1 Modes . . . . .	66

9.50.2.2	Orientations . . . . .	66
9.50.3	Property Documentation . . . . .	66
9.50.3.1	backgroundColour . . . . .	66
9.50.3.2	borderColour . . . . .	66
9.50.3.3	centreAngle . . . . .	66
9.50.3.4	fontColour . . . . .	67
9.50.3.5	foregroundColour . . . . .	67
9.50.3.6	logScale . . . . .	67
9.50.3.7	logScaleInterval . . . . .	67
9.50.3.8	majorInterval . . . . .	67
9.50.3.9	maximum . . . . .	67
9.50.3.10	minimum . . . . .	67
9.50.3.11	minorInterval . . . . .	67
9.50.3.12	mode . . . . .	67
9.50.3.13	orientation . . . . .	67
9.50.3.14	showScale . . . . .	68
9.50.3.15	showText . . . . .	68
9.50.3.16	spanAngle . . . . .	68
9.50.3.17	value . . . . .	68
9.51	QEAnalogProgressBar Class Reference . . . . .	68
9.51.1	Member Enumeration Documentation . . . . .	72
9.51.1.1	ArrayActions . . . . .	72
9.51.1.2	DisplayAlarmStateOptions . . . . .	72
9.51.1.3	Formats . . . . .	72
9.51.1.4	Notations . . . . .	73
9.51.1.5	Separators . . . . .	73
9.51.1.6	UserLevels . . . . .	73
9.51.2	Constructor & Destructor Documentation . . . . .	73
9.51.2.1	QEAnalogProgressBar . . . . .	73
9.51.2.2	QEAnalogProgressBar . . . . .	74
9.51.3	Member Function Documentation . . . . .	74
9.51.3.1	dbConnectionChanged . . . . .	74
9.51.3.2	dbValueChanged . . . . .	74
9.51.3.3	setManagedVisible . . . . .	74

9.51.4 Property Documentation . . . . .	74
9.51.4.1 addUnits . . . . .	74
9.51.4.2 alarmSeverityDisplayMode . . . . .	74
9.51.4.3 allowDrop . . . . .	74
9.51.4.4 arrayAction . . . . .	75
9.51.4.5 defaultStyle . . . . .	75
9.51.4.6 displayAlarmState . . . . .	75
9.51.4.7 displayAlarmStateOption . . . . .	75
9.51.4.8 format . . . . .	75
9.51.4.9 int . . . . .	76
9.51.4.10 leadingZero . . . . .	76
9.51.4.11 localEnumeration . . . . .	76
9.51.4.12 notation . . . . .	77
9.51.4.13 precision . . . . .	77
9.51.4.14 radix . . . . .	77
9.51.4.15 separator . . . . .	77
9.51.4.16 styleSheet . . . . .	77
9.51.4.17 trailingZeros . . . . .	77
9.51.4.18 useDbDisplayLimits . . . . .	77
9.51.4.19 useDbPrecision . . . . .	77
9.51.4.20 userLevelEnabled . . . . .	77
9.51.4.21 userLevelEngineerStyle . . . . .	78
9.51.4.22 userLevelScientistStyle . . . . .	78
9.51.4.23 userLevelUserStyle . . . . .	78
9.51.4.24 userLevelVisibility . . . . .	78
9.51.4.25 value . . . . .	78
9.51.4.26 variable . . . . .	78
9.51.4.27 variableAsToolTip . . . . .	79
9.51.4.28 variableSubstitutions . . . . .	79
9.51.4.29 visible . . . . .	79
9.52 QEBitStatus Class Reference . . . . .	79
9.52.1 Member Enumeration Documentation . . . . .	81
9.52.1.1 DisplayAlarmStateOptions . . . . .	81
9.52.1.2 UserLevels . . . . .	82

---

9.52.2 Member Function Documentation . . . . .	82
9.52.2.1 dbConnectionChanged . . . . .	82
9.52.2.2 dbValueChanged . . . . .	82
9.52.2.3 setManagedVisible . . . . .	82
9.52.3 Property Documentation . . . . .	82
9.52.3.1 allowDrop . . . . .	82
9.52.3.2 arrayIndex . . . . .	82
9.52.3.3 defaultStyle . . . . .	83
9.52.3.4 displayAlarmState . . . . .	83
9.52.3.5 displayAlarmStateOption . . . . .	83
9.52.3.6 int . . . . .	83
9.52.3.7 styleSheet . . . . .	83
9.52.3.8 userLevelEnabled . . . . .	83
9.52.3.9 userLevelEngineerStyle . . . . .	84
9.52.3.10 userLevelScientistStyle . . . . .	84
9.52.3.11 userLevelUserStyle . . . . .	84
9.52.3.12 userLevelVisibility . . . . .	84
9.52.3.13 variable . . . . .	84
9.52.3.14 variableAsToolTip . . . . .	84
9.52.3.15 variableSubstitutions . . . . .	85
9.52.3.16 visible . . . . .	85
9.53 QECheckBox Class Reference . . . . .	85
9.53.1 Member Enumeration Documentation . . . . .	89
9.53.1.1 ArrayActions . . . . .	89
9.53.1.2 CreationOptionNames . . . . .	89
9.53.1.3 DisplayAlarmStateOptions . . . . .	90
9.53.1.4 Formats . . . . .	90
9.53.1.5 Notations . . . . .	91
9.53.1.6 ProgramStartupOptionNames . . . . .	91
9.53.1.7 Separators . . . . .	91
9.53.1.8 UpdateOptions . . . . .	91
9.53.1.9 UserLevels . . . . .	92
9.53.2 Constructor & Destructor Documentation . . . . .	92
9.53.2.1 QECheckBox . . . . .	92

9.53.2.2	QECheckBox . . . . .	92
9.53.3	Member Function Documentation . . . . .	92
9.53.3.1	clicked . . . . .	92
9.53.3.2	dbValueChanged . . . . .	92
9.53.3.3	pressed . . . . .	92
9.53.3.4	released . . . . .	93
9.53.3.5	requestAction . . . . .	93
9.53.3.6	setManagedVisible . . . . .	93
9.53.4	Property Documentation . . . . .	93
9.53.4.1	addUnits . . . . .	93
9.53.4.2	alignment . . . . .	93
9.53.4.3	allowDrop . . . . .	93
9.53.4.4	arguments . . . . .	93
9.53.4.5	arrayAction . . . . .	94
9.53.4.6	clickCheckedText . . . . .	94
9.53.4.7	clickText . . . . .	94
9.53.4.8	confirmAction . . . . .	94
9.53.4.9	confirmText . . . . .	94
9.53.4.10	creationOption . . . . .	95
9.53.4.11	customisationName . . . . .	95
9.53.4.12	defaultStyle . . . . .	95
9.53.4.13	displayAlarmState . . . . .	95
9.53.4.14	displayAlarmStateOption . . . . .	95
9.53.4.15	format . . . . .	96
9.53.4.16	guiFile . . . . .	96
9.53.4.17	int . . . . .	96
9.53.4.18	labelText . . . . .	96
9.53.4.19	leadingZero . . . . .	96
9.53.4.20	localEnumeration . . . . .	96
9.53.4.21	notation . . . . .	97
9.53.4.22	password . . . . .	97
9.53.4.23	pixmap0 . . . . .	97
9.53.4.24	pixmap1 . . . . .	97
9.53.4.25	pixmap2 . . . . .	98

9.53.4.26 pixmap3 . . . . .	98
9.53.4.27 pixmap4 . . . . .	98
9.53.4.28 pixmap5 . . . . .	98
9.53.4.29 pixmap6 . . . . .	98
9.53.4.30 pixmap7 . . . . .	98
9.53.4.31 precision . . . . .	98
9.53.4.32 pressText . . . . .	98
9.53.4.33 prioritySubstitutions . . . . .	99
9.53.4.34 program . . . . .	99
9.53.4.35 programStartupOption . . . . .	99
9.53.4.36 radix . . . . .	99
9.53.4.37 releaseText . . . . .	99
9.53.4.38 separator . . . . .	99
9.53.4.39 styleSheet . . . . .	99
9.53.4.40 subscribe . . . . .	100
9.53.4.41 trailingZeros . . . . .	100
9.53.4.42 updateOption . . . . .	100
9.53.4.43 useDbPrecision . . . . .	100
9.53.4.44 userLevelEnabled . . . . .	100
9.53.4.45 userLevelEngineerStyle . . . . .	100
9.53.4.46 userLevelScientistStyle . . . . .	100
9.53.4.47 userLevelUserStyle . . . . .	101
9.53.4.48 userLevelVisibility . . . . .	101
9.53.4.49 variable . . . . .	101
9.53.4.50 variableAsToolTip . . . . .	101
9.53.4.51 variableSubstitutions . . . . .	101
9.53.4.52 visible . . . . .	101
9.53.4.53 writeOnClick . . . . .	102
9.53.4.54 writeOnPress . . . . .	102
9.53.4.55 writeOnRelease . . . . .	102
9.54 QECheckBoxManager Class Reference . . . . .	102
9.55 QEComboBox Class Reference . . . . .	103
9.55.1 Member Enumeration Documentation . . . . .	105
9.55.1.1 DisplayAlarmStateOptions . . . . .	105

9.55.1.2	UserLevels . . . . .	106
9.55.2	Member Function Documentation . . . . .	106
9.55.2.1	dbValueChanged . . . . .	106
9.55.2.2	setManagedVisible . . . . .	106
9.55.3	Member Data Documentation . . . . .	106
9.55.3.1	useDbEnumerations . . . . .	106
9.55.3.2	writeOnChange . . . . .	106
9.55.4	Property Documentation . . . . .	106
9.55.4.1	allowDrop . . . . .	106
9.55.4.2	defaultStyle . . . . .	107
9.55.4.3	displayAlarmState . . . . .	107
9.55.4.4	displayAlarmStateOption . . . . .	107
9.55.4.5	int . . . . .	107
9.55.4.6	localEnumeration . . . . .	107
9.55.4.7	styleSheet . . . . .	107
9.55.4.8	subscribe . . . . .	107
9.55.4.9	userLevelEnabled . . . . .	108
9.55.4.10	userLevelEngineerStyle . . . . .	108
9.55.4.11	userLevelScientistStyle . . . . .	108
9.55.4.12	userLevelUserStyle . . . . .	108
9.55.4.13	userLevelVisibility . . . . .	108
9.55.4.14	variable . . . . .	109
9.55.4.15	variableAsToolTip . . . . .	109
9.55.4.16	variableSubstitutions . . . . .	109
9.55.4.17	visible . . . . .	109
9.56	QEConfiguredLayout Class Reference . . . . .	109
9.56.1	Member Enumeration Documentation . . . . .	111
9.56.1.1	DisplayAlarmStateOptions . . . . .	111
9.56.1.2	UserLevels . . . . .	112
9.56.2	Member Function Documentation . . . . .	112
9.56.2.1	setManagedVisible . . . . .	112
9.56.3	Property Documentation . . . . .	112
9.56.3.1	allowDrop . . . . .	112
9.56.3.2	defaultStyle . . . . .	112

9.56.3.3	displayAlarmState . . . . .	112
9.56.3.4	displayAlarmStateOption . . . . .	112
9.56.3.5	int . . . . .	113
9.56.3.6	styleSheet . . . . .	113
9.56.3.7	userLevelEnabled . . . . .	113
9.56.3.8	userLevelEngineerStyle . . . . .	113
9.56.3.9	userLevelScientistStyle . . . . .	113
9.56.3.10	userLevelUserStyle . . . . .	114
9.56.3.11	userLevelVisibility . . . . .	114
9.56.3.12	variableAsToolTip . . . . .	114
9.56.3.13	visible . . . . .	114
9.57	QEConfiguredLayoutManager Class Reference . . . . .	114
9.58	QEFileBrowser Class Reference . . . . .	115
9.58.1	Detailed Description . . . . .	118
9.58.2	Member Enumeration Documentation . . . . .	118
9.58.2.1	DisplayAlarmStateOptions . . . . .	118
9.58.2.2	UserLevels . . . . .	118
9.58.3	Member Function Documentation . . . . .	118
9.58.3.1	selected . . . . .	118
9.58.3.2	setManagedVisible . . . . .	119
9.58.4	Property Documentation . . . . .	119
9.58.4.1	allowDrop . . . . .	119
9.58.4.2	defaultStyle . . . . .	119
9.58.4.3	displayAlarmState . . . . .	119
9.58.4.4	displayAlarmStateOption . . . . .	119
9.58.4.5	int . . . . .	119
9.58.4.6	styleSheet . . . . .	120
9.58.4.7	userLevelEnabled . . . . .	120
9.58.4.8	userLevelEngineerStyle . . . . .	120
9.58.4.9	userLevelScientistStyle . . . . .	120
9.58.4.10	userLevelUserStyle . . . . .	120
9.58.4.11	userLevelVisibility . . . . .	120
9.58.4.12	variable . . . . .	121
9.58.4.13	variableAsToolTip . . . . .	121

9.58.4.14	variableSubstitutions	121
9.58.4.15	visible	121
9.59	QEForm Class Reference	121
9.59.1	Member Data Documentation	123
9.59.1.1	handleGuiLaunchRequests	123
9.59.1.2	resizeContents	124
9.59.2	Property Documentation	124
9.59.2.1	allowDrop	124
9.59.2.2	displayAlarmStateOption	124
9.59.2.3	int	124
9.59.2.4	messageFormFilter	124
9.59.2.5	messageSourceFilter	124
9.59.2.6	uiFile	125
9.59.2.7	variableAsToolTip	125
9.59.2.8	variableSubstitutions	125
9.60	QEFrame Class Reference	125
9.60.1	Member Enumeration Documentation	127
9.60.1.1	DisplayAlarmStateOptions	127
9.60.1.2	UserLevels	127
9.60.2	Member Function Documentation	127
9.60.2.1	setManagedVisible	127
9.60.3	Property Documentation	128
9.60.3.1	allowDrop	128
9.60.3.2	defaultStyle	128
9.60.3.3	displayAlarmState	128
9.60.3.4	displayAlarmStateOption	128
9.60.3.5	int	128
9.60.3.6	styleSheet	128
9.60.3.7	userLevelEnabled	129
9.60.3.8	userLevelEngineerStyle	129
9.60.3.9	userLevelScientistStyle	129
9.60.3.10	userLevelUserStyle	129
9.60.3.11	userLevelVisibility	129
9.60.3.12	variableAsToolTip	130

9.60.3.13	visible	130
9.61	QEGenericButton Class Reference	130
9.62	QEGenericEdit Class Reference	132
9.62.1	Member Enumeration Documentation	135
9.62.1.1	DisplayAlarmStateOptions	135
9.62.1.2	UserLevels	135
9.62.2	Constructor & Destructor Documentation	135
9.62.2.1	QEGenericEdit	135
9.62.2.2	QEGenericEdit	135
9.62.3	Member Function Documentation	135
9.62.3.1	getConfirmWrite	135
9.62.3.2	getSubscribe	136
9.62.3.3	getWriteOnEnter	136
9.62.3.4	getWriteOnFinish	136
9.62.3.5	getWriteOnLoseFocus	136
9.62.3.6	setConfirmWrite	136
9.62.3.7	setManagedVisible	136
9.62.3.8	setSubscribe	136
9.62.3.9	setWriteOnEnter	136
9.62.3.10	setWriteOnFinish	136
9.62.3.11	setWriteOnLoseFocus	137
9.62.4	Property Documentation	137
9.62.4.1	allowDrop	137
9.62.4.2	confirmWrite	137
9.62.4.3	defaultStyle	137
9.62.4.4	displayAlarmState	137
9.62.4.5	displayAlarmStateOption	137
9.62.4.6	int	138
9.62.4.7	styleSheet	138
9.62.4.8	subscribe	138
9.62.4.9	userLevelEnabled	138
9.62.4.10	userLevelEngineerStyle	138
9.62.4.11	userLevelScientistStyle	138
9.62.4.12	userLevelUserStyle	139

9.62.4.13 userLevelVisibility . . . . .	139
9.62.4.14 variable . . . . .	139
9.62.4.15 variableAsToolTip . . . . .	139
9.62.4.16 variableSubstitutions . . . . .	139
9.62.4.17 visible . . . . .	139
9.62.4.18 writeOnEnter . . . . .	139
9.62.4.19 writeOnFinish . . . . .	140
9.62.4.20 writeOnLoseFocus . . . . .	140
9.63 QEGroupBox Class Reference . . . . .	140
9.63.1 Member Enumeration Documentation . . . . .	141
9.63.1.1 DisplayAlarmStateOptions . . . . .	141
9.63.1.2 UserLevels . . . . .	142
9.63.2 Member Function Documentation . . . . .	142
9.63.2.1 setManagedVisible . . . . .	142
9.63.3 Property Documentation . . . . .	142
9.63.3.1 allowDrop . . . . .	142
9.63.3.2 defaultStyle . . . . .	142
9.63.3.3 displayAlarmState . . . . .	142
9.63.3.4 displayAlarmStateOption . . . . .	142
9.63.3.5 int . . . . .	143
9.63.3.6 styleSheet . . . . .	143
9.63.3.7 substitutedTitle . . . . .	143
9.63.3.8 textSubstitutions . . . . .	143
9.63.3.9 userLevelEnabled . . . . .	143
9.63.3.10 userLevelEngineerStyle . . . . .	143
9.63.3.11 userLevelScientistStyle . . . . .	144
9.63.3.12 userLevelUserStyle . . . . .	144
9.63.3.13 userLevelVisibility . . . . .	144
9.63.3.14 variableAsToolTip . . . . .	144
9.63.3.15 visible . . . . .	144
9.64 QEImage Class Reference . . . . .	145
9.64.1 Detailed Description . . . . .	168
9.64.2 Member Enumeration Documentation . . . . .	168
9.64.2.1 DisplayAlarmStateOptions . . . . .	168

9.64.2.2	EllipseVariableDefinitions	168
9.64.2.3	ellipseVariableDefinitions	169
9.64.2.4	FormatOptions	169
9.64.2.5	ProgramStartupOptionNames	169
9.64.2.6	ResizeOptions	170
9.64.2.7	resizeOptions	170
9.64.2.8	RotationOptions	170
9.64.2.9	selectOptions	170
9.64.2.10	TargetOptions	171
9.64.2.11	UserLevels	171
9.64.3	Constructor & Destructor Documentation	171
9.64.3.1	QEImage	171
9.64.3.2	QEImage	172
9.64.4	Member Function Documentation	172
9.64.4.1	dbValueChanged	172
9.64.4.2	setImageFile	172
9.64.4.3	setManagedVisible	172
9.64.5	Member Data Documentation	172
9.64.5.1	displayButtonBar	172
9.64.5.2	initialVertScrollPos	172
9.64.6	Property Documentation	172
9.64.6.1	allowDrop	172
9.64.6.2	areaColor	173
9.64.6.3	arguments1	173
9.64.6.4	arguments2	173
9.64.6.5	autoBrightnessContrast	173
9.64.6.6	beamColor	173
9.64.6.7	beamXVariable	173
9.64.6.8	beamYVariable	173
9.64.6.9	bitDepthVariable	173
9.64.6.10	briefInfoArea	173
9.64.6.11	clippingHighVariable	174
9.64.6.12	clippingLowVariable	174
9.64.6.13	clippingOnOffVariable	174

9.64.6.14 contrastReversal . . . . .	174
9.64.6.15 defaultStyle . . . . .	174
9.64.6.16 dimension1Variable . . . . .	174
9.64.6.17 dimension2Variable . . . . .	174
9.64.6.18 dimension3Variable . . . . .	174
9.64.6.19 dimensionsVariable . . . . .	175
9.64.6.20 displayAlarmState . . . . .	175
9.64.6.21 displayAlarmStateOption . . . . .	175
9.64.6.22 displayArea1Selection . . . . .	175
9.64.6.23 displayArea2Selection . . . . .	175
9.64.6.24 displayArea3Selection . . . . .	175
9.64.6.25 displayArea4Selection . . . . .	175
9.64.6.26 displayBeamSelection . . . . .	176
9.64.6.27 displayCursorPixelInfo . . . . .	176
9.64.6.28 displayEllipse . . . . .	176
9.64.6.29 displayHozSlice1Selection . . . . .	176
9.64.6.30 displayHozSlice2Selection . . . . .	176
9.64.6.31 displayHozSlice3Selection . . . . .	176
9.64.6.32 displayHozSlice4Selection . . . . .	176
9.64.6.33 displayHozSlice5Selection . . . . .	176
9.64.6.34 displayProfileSelection . . . . .	176
9.64.6.35 displayTargetSelection . . . . .	177
9.64.6.36 displayVertSliceSelection . . . . .	177
9.64.6.37 ellipseColor . . . . .	177
9.64.6.38 ellipseHVariable . . . . .	177
9.64.6.39 ellipseWVariable . . . . .	177
9.64.6.40 ellipseXVariable . . . . .	177
9.64.6.41 ellipseYVariable . . . . .	177
9.64.6.42 enableArea1Selection . . . . .	178
9.64.6.43 enableArea2Selection . . . . .	178
9.64.6.44 enableArea3Selection . . . . .	178
9.64.6.45 enableArea4Selection . . . . .	178
9.64.6.46 enableBeamSelection . . . . .	178
9.64.6.47 enableHozSlice1Selection . . . . .	178

9.64.6.48 enableHozSlice2Selection . . . . .	178
9.64.6.49 enableHozSlice3Selection . . . . .	178
9.64.6.50 enableHozSlice4Selection . . . . .	179
9.64.6.51 enableHozSlice5Selection . . . . .	179
9.64.6.52 enableProfileSelection . . . . .	179
9.64.6.53 enableTargetSelection . . . . .	179
9.64.6.54 enableVertSlice1Selection . . . . .	179
9.64.6.55 enableVertSlice2Selection . . . . .	179
9.64.6.56 enableVertSlice3Selection . . . . .	179
9.64.6.57 enableVertSlice4Selection . . . . .	180
9.64.6.58 enableVertSlice5Selection . . . . .	180
9.64.6.59 externalControls . . . . .	180
9.64.6.60 formatOption . . . . .	180
9.64.6.61 formatVariable . . . . .	180
9.64.6.62 heightVariable . . . . .	180
9.64.6.63 horizontalFlip . . . . .	180
9.64.6.64 hozSlice1Color . . . . .	180
9.64.6.65 hozSlice2Color . . . . .	180
9.64.6.66 hozSlice3Color . . . . .	181
9.64.6.67 hozSlice4Color . . . . .	181
9.64.6.68 hozSlice5Color . . . . .	181
9.64.6.69 imageVariable . . . . .	181
9.64.6.70 initialHosScrollPos . . . . .	181
9.64.6.71 int . . . . .	181
9.64.6.72 lineProfileArrayVariable . . . . .	181
9.64.6.73 lineProfileThicknessVariable . . . . .	181
9.64.6.74 lineProfileX1Variable . . . . .	182
9.64.6.75 lineProfileX2Variable . . . . .	182
9.64.6.76 lineProfileY1Variable . . . . .	182
9.64.6.77 lineProfileY2Variable . . . . .	182
9.64.6.78 logBrightness . . . . .	182
9.64.6.79 profileColor . . . . .	182
9.64.6.80 profileHoz1ThicknessVariable . . . . .	182
9.64.6.81 profileHoz1Variable . . . . .	182

9.64.6.82 profileHoz2ThicknessVariable . . . . .	182
9.64.6.83 profileHoz2Variable . . . . .	183
9.64.6.84 profileHoz3ThicknessVariable . . . . .	183
9.64.6.85 profileHoz3Variable . . . . .	183
9.64.6.86 profileHoz4ThicknessVariable . . . . .	183
9.64.6.87 profileHoz4Variable . . . . .	183
9.64.6.88 profileHoz5ThicknessVariable . . . . .	183
9.64.6.89 profileHoz5Variable . . . . .	183
9.64.6.90 profileHozArrayVariable . . . . .	183
9.64.6.91 profileVert1ThicknessVariable . . . . .	183
9.64.6.92 profileVert1Variable . . . . .	184
9.64.6.93 profileVert2ThicknessVariable . . . . .	184
9.64.6.94 profileVert2Variable . . . . .	184
9.64.6.95 profileVert3ThicknessVariable . . . . .	184
9.64.6.96 profileVert3Variable . . . . .	184
9.64.6.97 profileVert4ThicknessVariable . . . . .	184
9.64.6.98 profileVert4Variable . . . . .	184
9.64.6.99 profileVert5ThicknessVariable . . . . .	184
9.64.6.100profileVert5Variable . . . . .	184
9.64.6.101profileVertArrayVariable . . . . .	185
9.64.6.102program1 . . . . .	185
9.64.6.103program2 . . . . .	185
9.64.6.104programStartupOption1 . . . . .	185
9.64.6.105programStartupOption2 . . . . .	185
9.64.6.106regionOfInterest1HVariable . . . . .	185
9.64.6.107regionOfInterest1WVariable . . . . .	185
9.64.6.108regionOfInterest1XVariable . . . . .	186
9.64.6.109regionOfInterest1YVariable . . . . .	186
9.64.6.110regionOfInterest2HVariable . . . . .	186
9.64.6.111regionOfInterest2WVariable . . . . .	186
9.64.6.112regionOfInterest2XVariable . . . . .	186
9.64.6.113regionOfInterest2YVariable . . . . .	186
9.64.6.114regionOfInterest3HVariable . . . . .	186
9.64.6.115regionOfInterest3WVariable . . . . .	186

9.64.6.116regionOfInterest3XVariable . . . . .	186
9.64.6.117regionOfInterest3YVariable . . . . .	187
9.64.6.118regionOfInterest4HVariable . . . . .	187
9.64.6.119regionOfInterest4WVariable . . . . .	187
9.64.6.120regionOfInterest4XVariable . . . . .	187
9.64.6.121regionOfInterest4YVariable . . . . .	187
9.64.6.122resizeOption . . . . .	187
9.64.6.123rotation . . . . .	187
9.64.6.124showTime . . . . .	187
9.64.6.125styleSheet . . . . .	187
9.64.6.126targetColor . . . . .	188
9.64.6.127targetTriggerVariable . . . . .	188
9.64.6.128targetXVariable . . . . .	188
9.64.6.129targetYVariable . . . . .	188
9.64.6.130timeColor . . . . .	188
9.64.6.131URL . . . . .	188
9.64.6.132useFalseColour . . . . .	188
9.64.6.133userLevelEnabled . . . . .	188
9.64.6.134userLevelEngineerStyle . . . . .	189
9.64.6.135userLevelScientistStyle . . . . .	189
9.64.6.136userLevelUserStyle . . . . .	189
9.64.6.137userLevelVisibility . . . . .	189
9.64.6.138variableAsToolTip . . . . .	189
9.64.6.139variableSubstitutions . . . . .	189
9.64.6.140verticalFlip . . . . .	190
9.64.6.141vertSlice1Color . . . . .	190
9.64.6.142vertSlice2Color . . . . .	190
9.64.6.143vertSlice3Color . . . . .	190
9.64.6.144vertSlice4Color . . . . .	190
9.64.6.145vertSlice5Color . . . . .	190
9.64.6.146visible . . . . .	190
9.64.6.147widthVariable . . . . .	190
9.65 QEImageMarkupThickness Class Reference . . . . .	190
9.66 QEImageOptionsDialog Class Reference . . . . .	191

9.67 QELabel Class Reference . . . . .	191
9.67.1 Detailed Description . . . . .	195
9.67.2 Member Enumeration Documentation . . . . .	196
9.67.2.1 ArrayActions . . . . .	196
9.67.2.2 DisplayAlarmStateOptions . . . . .	196
9.67.2.3 Formats . . . . .	196
9.67.2.4 Notations . . . . .	196
9.67.2.5 Separators . . . . .	197
9.67.2.6 UpdateOptions . . . . .	197
9.67.2.7 updateOptions . . . . .	197
9.67.2.8 UserLevels . . . . .	197
9.67.3 Constructor & Destructor Documentation . . . . .	198
9.67.3.1 QELabel . . . . .	198
9.67.3.2 QELabel . . . . .	198
9.67.4 Member Function Documentation . . . . .	198
9.67.4.1 dbValueChanged . . . . .	198
9.67.4.2 setManagedVisible . . . . .	198
9.67.5 Property Documentation . . . . .	198
9.67.5.1 addUnits . . . . .	198
9.67.5.2 allowDrop . . . . .	198
9.67.5.3 arrayAction . . . . .	198
9.67.5.4 defaultStyle . . . . .	199
9.67.5.5 displayAlarmState . . . . .	199
9.67.5.6 displayAlarmStateOption . . . . .	199
9.67.5.7 format . . . . .	199
9.67.5.8 int . . . . .	199
9.67.5.9 leadingZero . . . . .	200
9.67.5.10 localEnumeration . . . . .	200
9.67.5.11 notation . . . . .	200
9.67.5.12 pixmap0 . . . . .	200
9.67.5.13 pixmap1 . . . . .	201
9.67.5.14 pixmap2 . . . . .	201
9.67.5.15 pixmap3 . . . . .	201
9.67.5.16 pixmap4 . . . . .	201

9.67.5.17 pixmap5 . . . . .	201
9.67.5.18 pixmap6 . . . . .	201
9.67.5.19 pixmap7 . . . . .	201
9.67.5.20 precision . . . . .	201
9.67.5.21 radix . . . . .	201
9.67.5.22 separator . . . . .	202
9.67.5.23 styleSheet . . . . .	202
9.67.5.24 trailingZeros . . . . .	202
9.67.5.25 updateOption . . . . .	202
9.67.5.26 useDbPrecision . . . . .	202
9.67.5.27 userLevelEnabled . . . . .	202
9.67.5.28 userLevelEngineerStyle . . . . .	202
9.67.5.29 userLevelScientistStyle . . . . .	203
9.67.5.30 userLevelUserStyle . . . . .	203
9.67.5.31 userLevelVisibility . . . . .	203
9.67.5.32 variable . . . . .	203
9.67.5.33 variableAsToolTip . . . . .	203
9.67.5.34 variableSubstitutions . . . . .	203
9.67.5.35 visible . . . . .	204
9.68 QELlineEdit Class Reference . . . . .	204
9.68.1 Member Enumeration Documentation . . . . .	206
9.68.1.1 ArrayActions . . . . .	206
9.68.1.2 Formats . . . . .	206
9.68.1.3 Notations . . . . .	206
9.68.1.4 Separators . . . . .	206
9.68.2 Constructor & Destructor Documentation . . . . .	207
9.68.2.1 QELlineEdit . . . . .	207
9.68.2.2 QELlineEdit . . . . .	207
9.68.3 Member Function Documentation . . . . .	207
9.68.3.1 dbValueChanged . . . . .	207
9.68.4 Property Documentation . . . . .	207
9.68.4.1 addUnits . . . . .	207
9.68.4.2 arrayAction . . . . .	207
9.68.4.3 format . . . . .	208

9.68.4.4 int . . . . .	208
9.68.4.5 leadingZero . . . . .	208
9.68.4.6 localEnumeration . . . . .	208
9.68.4.7 notation . . . . .	209
9.68.4.8 precision . . . . .	209
9.68.4.9 radix . . . . .	209
9.68.4.10 separator . . . . .	209
9.68.4.11 trailingZeros . . . . .	209
9.68.4.12 useDbPrecision . . . . .	209
9.69 QELineEditManager Class Reference . . . . .	209
9.70 QELink Class Reference . . . . .	210
9.71 QELog Class Reference . . . . .	212
9.71.1 Member Enumeration Documentation . . . . .	214
9.71.1.1 DisplayAlarmStateOptions . . . . .	214
9.71.1.2 UserLevels . . . . .	215
9.71.2 Member Function Documentation . . . . .	215
9.71.2.1 setManagedVisible . . . . .	215
9.71.3 Property Documentation . . . . .	215
9.71.3.1 allowDrop . . . . .	215
9.71.3.2 defaultStyle . . . . .	215
9.71.3.3 displayAlarmState . . . . .	215
9.71.3.4 displayAlarmStateOption . . . . .	215
9.71.3.5 int . . . . .	216
9.71.3.6 styleSheet . . . . .	216
9.71.3.7 userLevelEnabled . . . . .	216
9.71.3.8 userLevelEngineerStyle . . . . .	216
9.71.3.9 userLevelScientistStyle . . . . .	216
9.71.3.10 userLevelUserStyle . . . . .	217
9.71.3.11 userLevelVisibility . . . . .	217
9.71.3.12 variableAsToolTip . . . . .	217
9.71.3.13 visible . . . . .	217
9.72 QELogin Class Reference . . . . .	217
9.73 QELoginDialog Class Reference . . . . .	218
9.74 QENumericEdit Class Reference . . . . .	218

9.74.1	Detailed Description . . . . .	220
9.74.2	Constructor & Destructor Documentation . . . . .	221
9.74.2.1	QENumericEdit . . . . .	221
9.74.2.2	QENumericEdit . . . . .	221
9.74.3	Member Function Documentation . . . . .	221
9.74.3.1	dbConnectionChanged . . . . .	221
9.74.3.2	dbValueChanged . . . . .	221
9.74.4	Property Documentation . . . . .	221
9.74.4.1	addUnits . . . . .	221
9.74.4.2	arrayIndex . . . . .	221
9.74.4.3	autoScale . . . . .	221
9.74.4.4	leadingZeros . . . . .	222
9.74.4.5	maximum . . . . .	222
9.74.4.6	minimum . . . . .	222
9.74.4.7	precision . . . . .	222
9.75	QENumericEditManager Class Reference . . . . .	222
9.76	QEPeriodic Class Reference . . . . .	223
9.76.1	Member Enumeration Documentation . . . . .	226
9.76.1.1	DisplayAlarmStateOptions . . . . .	226
9.76.1.2	UserLevels . . . . .	227
9.76.2	Member Function Documentation . . . . .	227
9.76.2.1	dbElementChanged . . . . .	227
9.76.2.2	dbValueChanged . . . . .	227
9.76.3	Member Data Documentation . . . . .	227
9.76.3.1	allowDrop . . . . .	227
9.76.4	Property Documentation . . . . .	227
9.76.4.1	displayAlarmState . . . . .	227
9.76.4.2	displayAlarmStateOption . . . . .	228
9.76.4.3	int . . . . .	228
9.76.4.4	readbackLabelVariable1 . . . . .	228
9.76.4.5	readbackLabelVariable2 . . . . .	228
9.76.4.6	subscribe . . . . .	228
9.76.4.7	userLevelEnabled . . . . .	228
9.76.4.8	userLevelEngineerStyle . . . . .	229

9.76.4.9 userLevelScientistStyle . . . . .	229
9.76.4.10 userLevelUserStyle . . . . .	229
9.76.4.11 userLevelVisibility . . . . .	229
9.76.4.12 variableAsToolTip . . . . .	229
9.76.4.13 variableSubstitutions . . . . .	229
9.76.4.14 visible . . . . .	230
9.76.4.15 writeButtonVariable1 . . . . .	230
9.76.4.16 writeButtonVariable2 . . . . .	230
9.77 QEPeriodicComponentData Class Reference . . . . .	230
9.78 QEPeriodicTaskMenu Class Reference . . . . .	230
9.79 QEPeriodicTaskMenuFactory Class Reference . . . . .	231
9.80 QEPlot Class Reference . . . . .	231
9.80.1 Member Enumeration Documentation . . . . .	235
9.80.1.1 DisplayAlarmStateOptions . . . . .	235
9.80.1.2 UserLevels . . . . .	235
9.80.2 Member Function Documentation . . . . .	235
9.80.2.1 dbValueChanged . . . . .	235
9.80.2.2 dbValueChanged . . . . .	235
9.80.2.3 setManagedVisible . . . . .	236
9.80.3 Member Data Documentation . . . . .	236
9.80.3.1 allowDrop . . . . .	236
9.80.4 Property Documentation . . . . .	236
9.80.4.1 defaultStyle . . . . .	236
9.80.4.2 displayAlarmState . . . . .	236
9.80.4.3 displayAlarmStateOption . . . . .	236
9.80.4.4 int . . . . .	236
9.80.4.5 styleSheet . . . . .	237
9.80.4.6 userLevelEnabled . . . . .	237
9.80.4.7 userLevelEngineerStyle . . . . .	237
9.80.4.8 userLevelScientistStyle . . . . .	237
9.80.4.9 userLevelUserStyle . . . . .	237
9.80.4.10 userLevelVisibility . . . . .	238
9.80.4.11 variable1 . . . . .	238
9.80.4.12 variable2 . . . . .	238

9.80.4.13 variable3 . . . . .	238
9.80.4.14 variable4 . . . . .	238
9.80.4.15 variableAsToolTip . . . . .	238
9.80.4.16 variableSubstitutions . . . . .	238
9.80.4.17 visible . . . . .	238
9.81 QEPushButton Class Reference . . . . .	239
9.81.1 Member Enumeration Documentation . . . . .	243
9.81.1.1 ArrayActions . . . . .	243
9.81.1.2 CreationOptionNames . . . . .	243
9.81.1.3 DisplayAlarmStateOptions . . . . .	243
9.81.1.4 Formats . . . . .	244
9.81.1.5 Notations . . . . .	244
9.81.1.6 ProgramStartupOptionNames . . . . .	244
9.81.1.7 UpdateOptions . . . . .	244
9.81.1.8 UserLevels . . . . .	245
9.81.2 Constructor & Destructor Documentation . . . . .	245
9.81.2.1 QEPushButton . . . . .	245
9.81.2.2 QEPushButton . . . . .	245
9.81.3 Member Function Documentation . . . . .	245
9.81.3.1 clicked . . . . .	245
9.81.3.2 dbValueChanged . . . . .	245
9.81.3.3 pressed . . . . .	246
9.81.3.4 released . . . . .	246
9.81.3.5 requestAction . . . . .	246
9.81.3.6 setManagedVisible . . . . .	246
9.81.4 Property Documentation . . . . .	246
9.81.4.1 addUnits . . . . .	246
9.81.4.2 alignment . . . . .	246
9.81.4.3 allowDrop . . . . .	246
9.81.4.4 altReadbackVariable . . . . .	247
9.81.4.5 arguments . . . . .	247
9.81.4.6 arrayAction . . . . .	247
9.81.4.7 clickCheckedText . . . . .	247
9.81.4.8 clickText . . . . .	247

9.81.4.9 confirmAction . . . . .	248
9.81.4.10 confirmText . . . . .	248
9.81.4.11 creationOption . . . . .	248
9.81.4.12 customisationName . . . . .	248
9.81.4.13 defaultStyle . . . . .	248
9.81.4.14 displayAlarmState . . . . .	248
9.81.4.15 displayAlarmStateOption . . . . .	249
9.81.4.16 format . . . . .	249
9.81.4.17 guiFile . . . . .	249
9.81.4.18 int . . . . .	249
9.81.4.19 labelText . . . . .	249
9.81.4.20 leadingZero . . . . .	250
9.81.4.21 localEnumeration . . . . .	250
9.81.4.22 notation . . . . .	250
9.81.4.23 password . . . . .	250
9.81.4.24 pixmap0 . . . . .	251
9.81.4.25 pixmap1 . . . . .	251
9.81.4.26 pixmap2 . . . . .	251
9.81.4.27 pixmap3 . . . . .	251
9.81.4.28 pixmap4 . . . . .	251
9.81.4.29 pixmap5 . . . . .	251
9.81.4.30 pixmap6 . . . . .	251
9.81.4.31 pixmap7 . . . . .	251
9.81.4.32 precision . . . . .	251
9.81.4.33 pressText . . . . .	252
9.81.4.34 prioritySubstitutions . . . . .	252
9.81.4.35 program . . . . .	252
9.81.4.36 programStartupOption . . . . .	252
9.81.4.37 releaseText . . . . .	252
9.81.4.38 styleSheet . . . . .	252
9.81.4.39 subscribe . . . . .	252
9.81.4.40 trailingZeros . . . . .	253
9.81.4.41 updateOption . . . . .	253
9.81.4.42 useDbPrecision . . . . .	253

9.81.4.43 userLevelEnabled . . . . .	253
9.81.4.44 userLevelEngineerStyle . . . . .	253
9.81.4.45 userLevelScientistStyle . . . . .	253
9.81.4.46 userLevelUserStyle . . . . .	254
9.81.4.47 userLevelVisibility . . . . .	254
9.81.4.48 variable . . . . .	254
9.81.4.49 variableAsToolTip . . . . .	254
9.81.4.50 variableSubstitutions . . . . .	254
9.81.4.51 visible . . . . .	254
9.81.4.52 writeOnClick . . . . .	254
9.81.4.53 writeOnPress . . . . .	255
9.81.4.54 writeOnRelease . . . . .	255
9.82 QEPVNameLists Class Reference . . . . .	255
9.83 QEPvProperties Class Reference . . . . .	255
9.83.1 Property Documentation . . . . .	256
9.83.1.1 variable . . . . .	256
9.83.1.2 variableSubstitutions . . . . .	257
9.84 QEPvPropertiesManager Class Reference . . . . .	257
9.85 QERadioButton Class Reference . . . . .	257
9.85.1 Member Enumeration Documentation . . . . .	262
9.85.1.1 ArrayActions . . . . .	262
9.85.1.2 CreationOptionNames . . . . .	262
9.85.1.3 DisplayAlarmStateOptions . . . . .	262
9.85.1.4 Formats . . . . .	263
9.85.1.5 Notations . . . . .	263
9.85.1.6 ProgramStartupOptionNames . . . . .	263
9.85.1.7 Separators . . . . .	263
9.85.1.8 UpdateOptions . . . . .	264
9.85.1.9 UserLevels . . . . .	264
9.85.2 Constructor & Destructor Documentation . . . . .	264
9.85.2.1 QERadioButton . . . . .	264
9.85.2.2 QERadioButton . . . . .	264
9.85.3 Member Function Documentation . . . . .	265
9.85.3.1 clicked . . . . .	265

9.85.3.2 dbValueChanged . . . . .	265
9.85.3.3 pressed . . . . .	265
9.85.3.4 released . . . . .	265
9.85.3.5 requestAction . . . . .	265
9.85.3.6 setManagedVisible . . . . .	265
9.85.4 Property Documentation . . . . .	265
9.85.4.1 addUnits . . . . .	266
9.85.4.2 alignment . . . . .	266
9.85.4.3 allowDrop . . . . .	266
9.85.4.4 arguments . . . . .	266
9.85.4.5 arrayAction . . . . .	266
9.85.4.6 clickCheckedText . . . . .	266
9.85.4.7 clickText . . . . .	267
9.85.4.8 confirmAction . . . . .	267
9.85.4.9 confirmText . . . . .	267
9.85.4.10 creationOption . . . . .	267
9.85.4.11 customisationName . . . . .	267
9.85.4.12 defaultStyle . . . . .	267
9.85.4.13 displayAlarmState . . . . .	268
9.85.4.14 displayAlarmStateOption . . . . .	268
9.85.4.15 format . . . . .	268
9.85.4.16 guiFile . . . . .	268
9.85.4.17 int . . . . .	268
9.85.4.18 labelText . . . . .	269
9.85.4.19 leadingZero . . . . .	269
9.85.4.20 localEnumeration . . . . .	269
9.85.4.21 notation . . . . .	270
9.85.4.22 password . . . . .	270
9.85.4.23 pixmap0 . . . . .	270
9.85.4.24 pixmap1 . . . . .	270
9.85.4.25 pixmap2 . . . . .	270
9.85.4.26 pixmap3 . . . . .	270
9.85.4.27 pixmap4 . . . . .	270
9.85.4.28 pixmap5 . . . . .	270

9.85.4.29 pixmap6 . . . . .	270
9.85.4.30 pixmap7 . . . . .	271
9.85.4.31 precision . . . . .	271
9.85.4.32 pressText . . . . .	271
9.85.4.33 prioritySubstitutions . . . . .	271
9.85.4.34 program . . . . .	271
9.85.4.35 programStartupOption . . . . .	271
9.85.4.36 radix . . . . .	271
9.85.4.37 releaseText . . . . .	272
9.85.4.38 separator . . . . .	272
9.85.4.39 styleSheet . . . . .	272
9.85.4.40 subscribe . . . . .	272
9.85.4.41 trailingZeros . . . . .	272
9.85.4.42 updateOption . . . . .	272
9.85.4.43 useDbPrecision . . . . .	272
9.85.4.44 userLevelEnabled . . . . .	272
9.85.4.45 userLevelEngineerStyle . . . . .	273
9.85.4.46 userLevelScientistStyle . . . . .	273
9.85.4.47 userLevelUserStyle . . . . .	273
9.85.4.48 userLevelVisibility . . . . .	273
9.85.4.49 variable . . . . .	273
9.85.4.50 variableAsToolTip . . . . .	273
9.85.4.51 variableSubstitutions . . . . .	274
9.85.4.52 visible . . . . .	274
9.85.4.53 writeOnClick . . . . .	274
9.85.4.54 writeOnPress . . . . .	274
9.85.4.55 writeOnRelease . . . . .	274
9.86 QERecipe Class Reference . . . . .	274
9.87 QERecordFieldName Class Reference . . . . .	276
9.88 QERecordSpec Class Reference . . . . .	277
9.89 QERecordSpecList Class Reference . . . . .	277
9.90 QEScript Class Reference . . . . .	277
9.90.1 Detailed Description . . . . .	282
9.90.2 Member Enumeration Documentation . . . . .	282

9.90.2.1	DisplayAlarmStateOptions . . . . .	282
9.90.2.2	UserLevels . . . . .	282
9.90.3	Member Function Documentation . . . . .	283
9.90.3.1	setManagedVisible . . . . .	283
9.90.4	Property Documentation . . . . .	283
9.90.4.1	allowDrop . . . . .	283
9.90.4.2	defaultStyle . . . . .	283
9.90.4.3	displayAlarmState . . . . .	283
9.90.4.4	displayAlarmStateOption . . . . .	283
9.90.4.5	int . . . . .	283
9.90.4.6	styleSheet . . . . .	284
9.90.4.7	userLevelEnabled . . . . .	284
9.90.4.8	userLevelEngineerStyle . . . . .	284
9.90.4.9	userLevelScientistStyle . . . . .	284
9.90.4.10	userLevelUserStyle . . . . .	284
9.90.4.11	userLevelVisibility . . . . .	285
9.90.4.12	variableAsToolTip . . . . .	285
9.90.4.13	visible . . . . .	285
9.91	QEShape Class Reference . . . . .	285
9.91.1	Detailed Description . . . . .	289
9.91.2	Member Enumeration Documentation . . . . .	290
9.91.2.1	animationOptions . . . . .	290
9.91.2.2	DisplayAlarmStateOptions . . . . .	290
9.91.2.3	shapeOptions . . . . .	290
9.91.2.4	UserLevels . . . . .	290
9.91.3	Constructor & Destructor Documentation . . . . .	290
9.91.3.1	QEShape . . . . .	290
9.91.3.2	QEShape . . . . .	290
9.91.4	Member Function Documentation . . . . .	291
9.91.4.1	dbValueChanged1 . . . . .	291
9.91.4.2	dbValueChanged2 . . . . .	291
9.91.4.3	dbValueChanged3 . . . . .	291
9.91.4.4	dbValueChanged4 . . . . .	291
9.91.4.5	dbValueChanged5 . . . . .	291

9.91.4.6 dbValueChanged . . . . .	291
9.91.4.7 setManagedVisible . . . . .	291
9.91.5 Property Documentation . . . . .	292
9.91.5.1 allowDrop . . . . .	292
9.91.5.2 animation1 . . . . .	292
9.91.5.3 animation2 . . . . .	292
9.91.5.4 animation3 . . . . .	292
9.91.5.5 animation4 . . . . .	292
9.91.5.6 animation5 . . . . .	292
9.91.5.7 animation6 . . . . .	292
9.91.5.8 color1 . . . . .	292
9.91.5.9 color10 . . . . .	293
9.91.5.10 color2 . . . . .	293
9.91.5.11 color3 . . . . .	293
9.91.5.12 color4 . . . . .	293
9.91.5.13 color5 . . . . .	293
9.91.5.14 color6 . . . . .	293
9.91.5.15 color7 . . . . .	293
9.91.5.16 color8 . . . . .	293
9.91.5.17 color9 . . . . .	293
9.91.5.18 defaultStyle . . . . .	294
9.91.5.19 displayAlarmState . . . . .	294
9.91.5.20 displayAlarmStateOption . . . . .	294
9.91.5.21 int . . . . .	294
9.91.5.22 offset1 . . . . .	294
9.91.5.23 offset2 . . . . .	294
9.91.5.24 offset3 . . . . .	295
9.91.5.25 offset4 . . . . .	295
9.91.5.26 offset5 . . . . .	295
9.91.5.27 offset6 . . . . .	295
9.91.5.28 point1 . . . . .	295
9.91.5.29 point10 . . . . .	295
9.91.5.30 point2 . . . . .	295
9.91.5.31 point3 . . . . .	295

9.91.5.32 point4 . . . . .	295
9.91.5.33 point5 . . . . .	296
9.91.5.34 point6 . . . . .	296
9.91.5.35 point7 . . . . .	296
9.91.5.36 point8 . . . . .	296
9.91.5.37 point9 . . . . .	296
9.91.5.38 scale2 . . . . .	296
9.91.5.39 scale3 . . . . .	296
9.91.5.40 scale4 . . . . .	296
9.91.5.41 scale5 . . . . .	296
9.91.5.42 scale6 . . . . .	297
9.91.5.43 styleSheet . . . . .	297
9.91.5.44 userLevelEnabled . . . . .	297
9.91.5.45 userLevelEngineerStyle . . . . .	297
9.91.5.46 userLevelScientistStyle . . . . .	297
9.91.5.47 userLevelUserStyle . . . . .	297
9.91.5.48 userLevelVisibility . . . . .	298
9.91.5.49 variable1 . . . . .	298
9.91.5.50 variable2 . . . . .	298
9.91.5.51 variable3 . . . . .	298
9.91.5.52 variable4 . . . . .	298
9.91.5.53 variable5 . . . . .	298
9.91.5.54 variable6 . . . . .	298
9.91.5.55 variableAsToolTip . . . . .	299
9.91.5.56 variableSubstitutions . . . . .	299
9.91.5.57 visible . . . . .	299
<b>9.92 QESlider Class Reference . . . . .</b>	<b>299</b>
<b>9.92.1 Member Enumeration Documentation . . . . .</b>	<b>301</b>
<b>9.92.1.1 DisplayAlarmStateOptions . . . . .</b>	<b>301</b>
<b>9.92.1.2 UserLevels . . . . .</b>	<b>302</b>
<b>9.92.2 Member Function Documentation . . . . .</b>	<b>302</b>
<b>9.92.2.1 dbValueChanged . . . . .</b>	<b>302</b>
<b>9.92.2.2 setManagedVisible . . . . .</b>	<b>302</b>
<b>9.92.3 Member Data Documentation . . . . .</b>	<b>302</b>

9.92.3.1	writeOnChange . . . . .	302
9.92.4	Property Documentation . . . . .	302
9.92.4.1	allowDrop . . . . .	302
9.92.4.2	defaultStyle . . . . .	303
9.92.4.3	displayAlarmState . . . . .	303
9.92.4.4	displayAlarmStateOption . . . . .	303
9.92.4.5	int . . . . .	303
9.92.4.6	styleSheet . . . . .	303
9.92.4.7	subscribe . . . . .	303
9.92.4.8	userLevelEnabled . . . . .	304
9.92.4.9	userLevelEngineerStyle . . . . .	304
9.92.4.10	userLevelScientistStyle . . . . .	304
9.92.4.11	userLevelUserStyle . . . . .	304
9.92.4.12	userLevelVisibility . . . . .	304
9.92.4.13	variable . . . . .	305
9.92.4.14	variableAsToolTip . . . . .	305
9.92.4.15	variableSubstitutions . . . . .	305
9.92.4.16	visible . . . . .	305
9.93	QESpinBox Class Reference . . . . .	305
9.93.1	Member Enumeration Documentation . . . . .	308
9.93.1.1	DisplayAlarmStateOptions . . . . .	308
9.93.1.2	UserLevels . . . . .	308
9.93.2	Member Function Documentation . . . . .	308
9.93.2.1	dbValueChanged . . . . .	308
9.93.2.2	setManagedVisible . . . . .	308
9.93.3	Property Documentation . . . . .	308
9.93.3.1	allowDrop . . . . .	308
9.93.3.2	defaultStyle . . . . .	309
9.93.3.3	displayAlarmState . . . . .	309
9.93.3.4	displayAlarmStateOption . . . . .	309
9.93.3.5	int . . . . .	309
9.93.3.6	styleSheet . . . . .	309
9.93.3.7	subscribe . . . . .	309
9.93.3.8	userLevelEnabled . . . . .	310

9.93.3.9 userLevelEngineerStyle . . . . .	310
9.93.3.10 userLevelScientistStyle . . . . .	310
9.93.3.11 userLevelUserStyle . . . . .	310
9.93.3.12 userLevelVisibility . . . . .	310
9.93.3.13 variable . . . . .	311
9.93.3.14 variableAsToolTip . . . . .	311
9.93.3.15 variableSubstitutions . . . . .	311
9.93.3.16 visible . . . . .	311
9.94 QEStripChart Class Reference . . . . .	311
9.94.1 Property Documentation . . . . .	313
9.94.1.1 variableSubstitutions . . . . .	313
9.95 QEStripChartAdjustPVDialog Class Reference . . . . .	314
9.96 QEStripChartContextMenu Class Reference . . . . .	314
9.96.1 Constructor & Destructor Documentation . . . . .	314
9.96.1.1 QEStripChartContextMenu . . . . .	314
9.97 QEStripChartDurationDialog Class Reference . . . . .	315
9.98 QEStripChartItem Class Reference . . . . .	315
9.99 QEStripChartNames Class Reference . . . . .	316
9.100QEStripChartPushButtonSpecifications Struct Reference . . . . .	317
9.101QEStripChartRangeDialog Class Reference . . . . .	318
9.102QEStripChartState Class Reference . . . . .	318
9.103QEStripChartStateList Class Reference . . . . .	318
9.104QEStripChartStatistics Class Reference . . . . .	319
9.105QEStripChartTimeDialog Class Reference . . . . .	319
9.106QEStripChartToolBar Class Reference . . . . .	319
9.106.1 Detailed Description . . . . .	320
9.107QESubstitutedLabel Class Reference . . . . .	321
9.107.1 Member Data Documentation . . . . .	321
9.107.1.1 labelText . . . . .	321
9.107.2 Property Documentation . . . . .	321
9.107.2.1 textSubstitutions . . . . .	321
9.108recording Class Reference . . . . .	321
9.109imageDisplayProperties::rgbPixel Struct Reference . . . . .	322
9.110screenSelectDialog Class Reference . . . . .	322

9.111selectMenu Class Reference . . . . .	323
9.112trace Class Reference . . . . .	323
9.113userInfoStruct Class Reference . . . . .	324
9.114QEPeriodic::userInfoStructArray Struct Reference . . . . .	324
9.115ValueScaling Class Reference . . . . .	324
9.116VideoWidget Class Reference . . . . .	325
9.117zoomMenu Class Reference . . . . .	326

# 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
```

Alternativly, 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 enviroment for building or using the epicsqt framework.

To build the framework, open epicsqt.pro in QtCreator, ensure shaddow 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

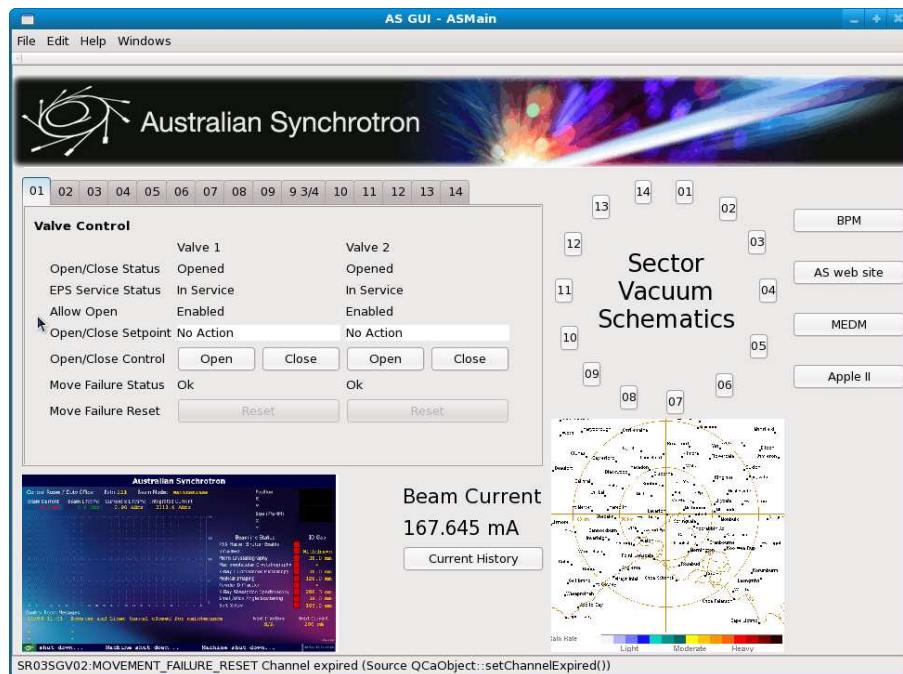


Figure 3.1: Australian Synchrotron mock up

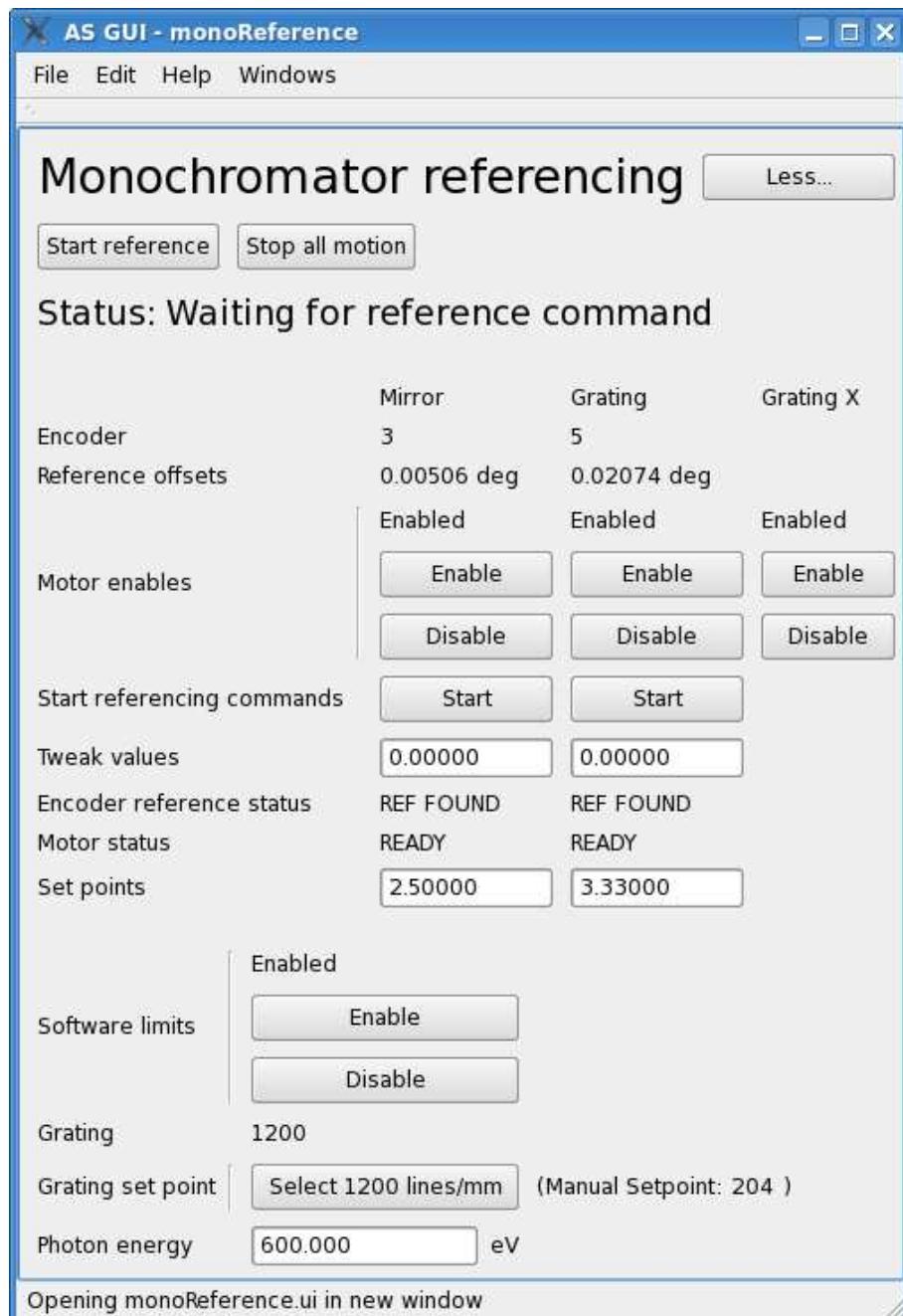


Figure 3.2: Monochromator referencing

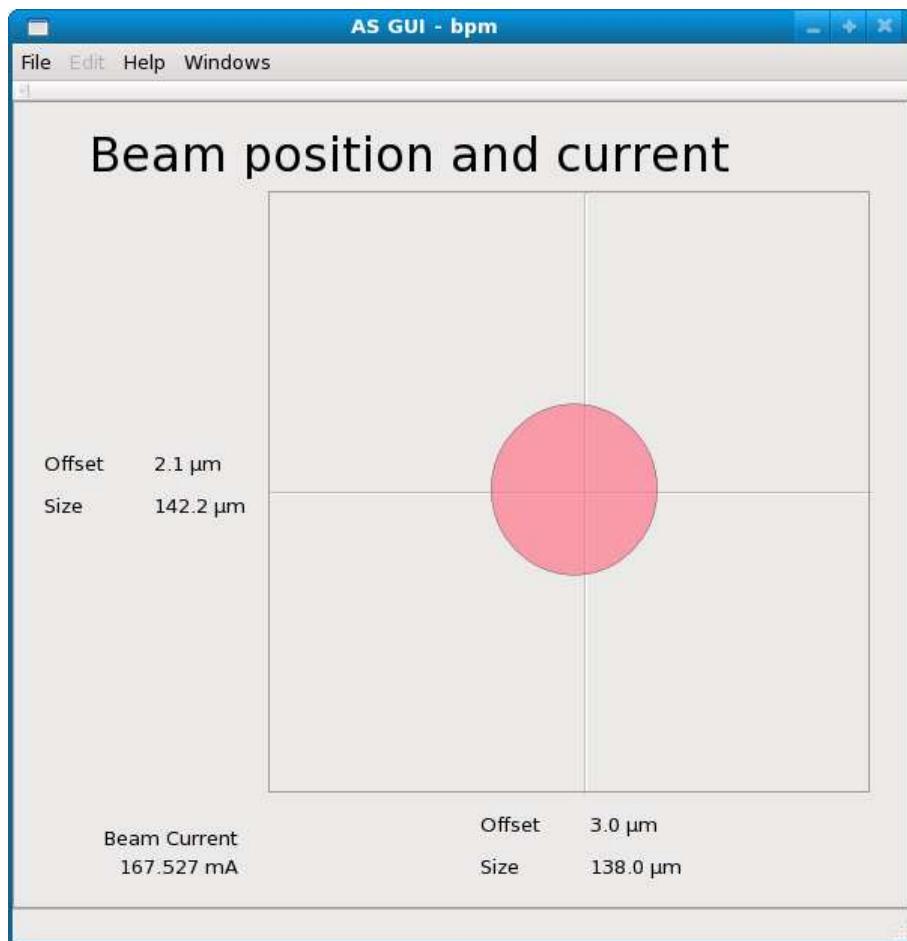


Figure 3.3: Beam position monitor

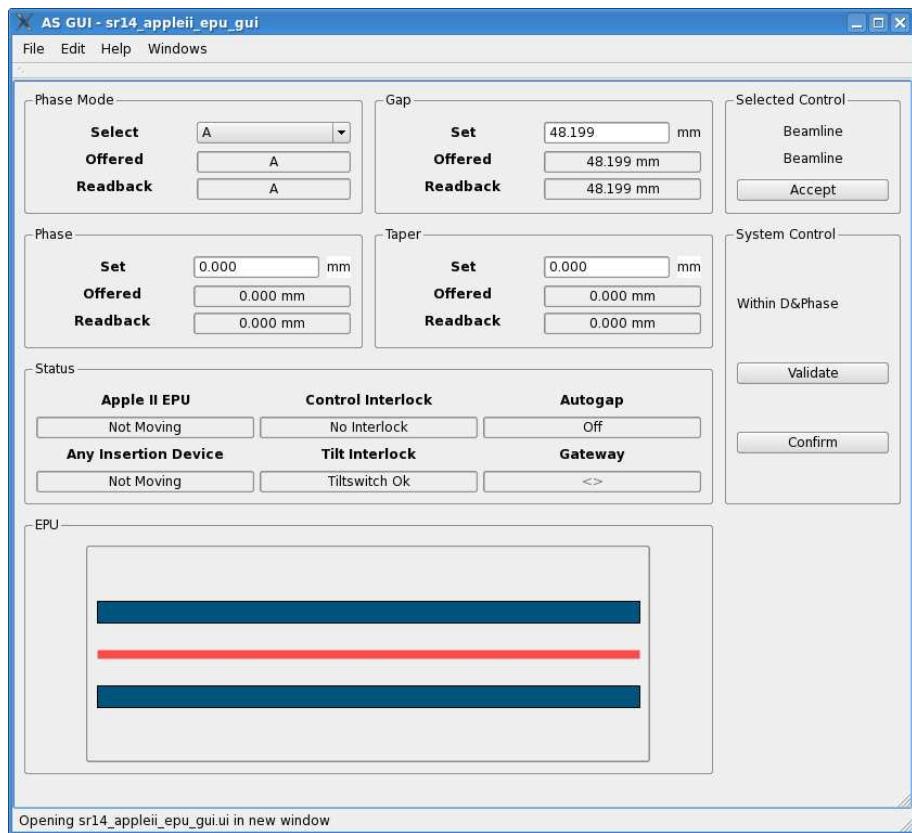


Figure 3.4: Insertion device



Figure 3.5: Injection efficiency monitor



## Chapter 4

# other applications using epicsqt widgets

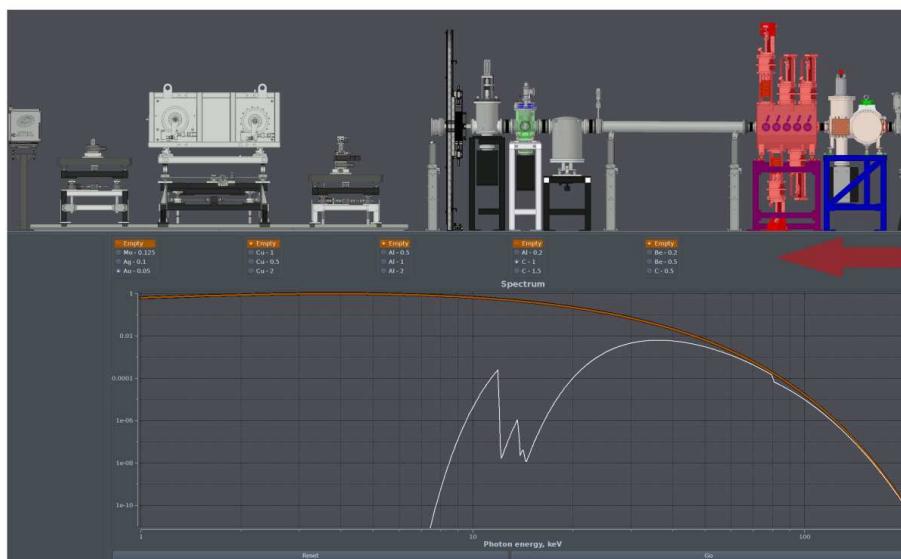


Figure 4.1: Medical Imaging beamline

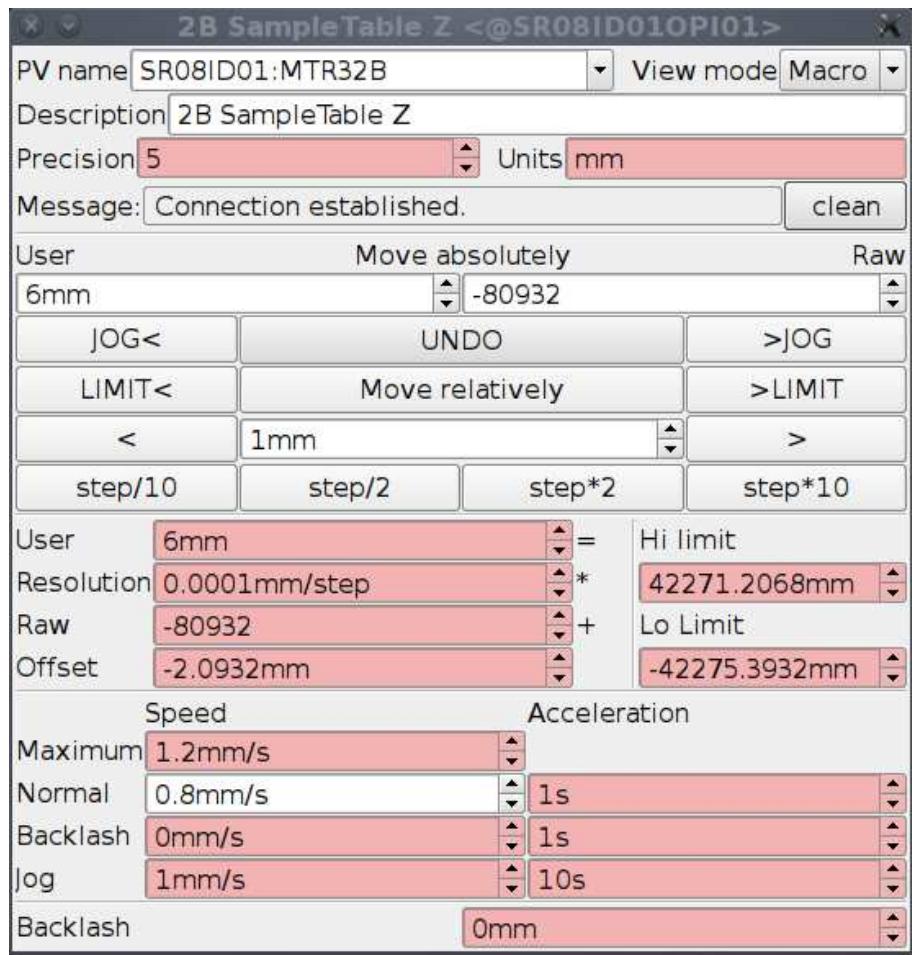


Figure 4.2: Motor controller

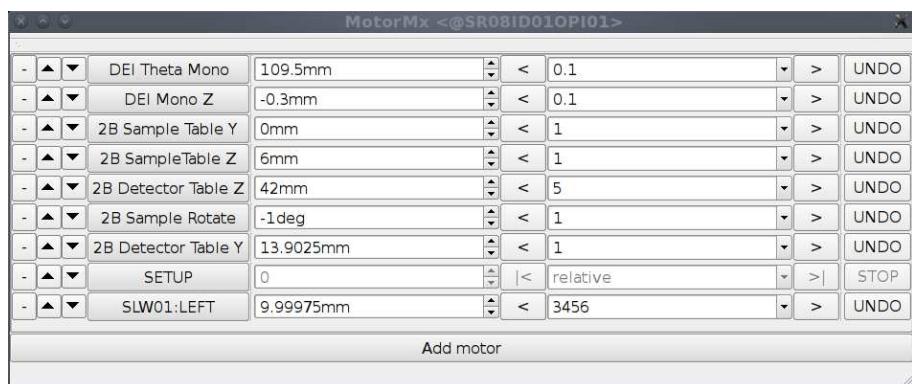


Figure 4.3: Motor controller

# Chapter 5

## Qt Designer

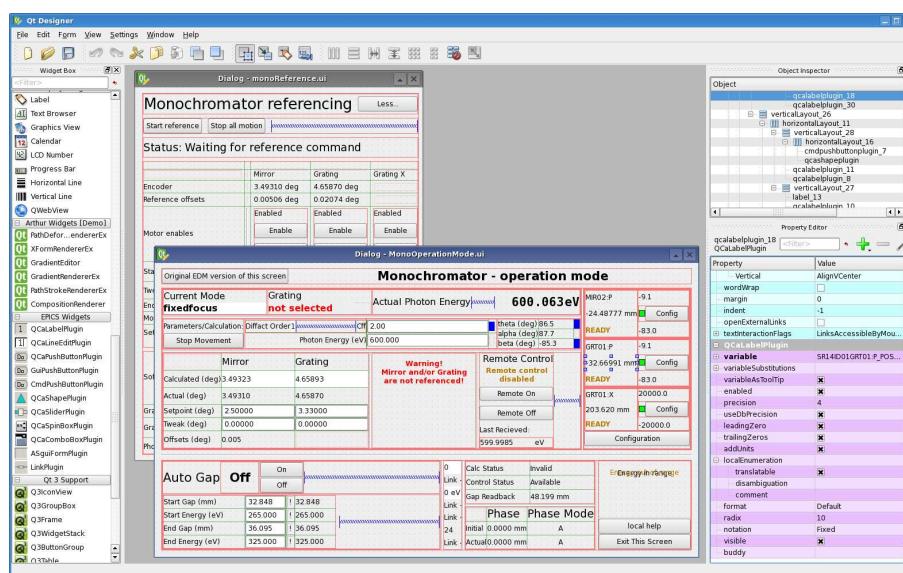


Figure 5.1: Editing multiple GUIs

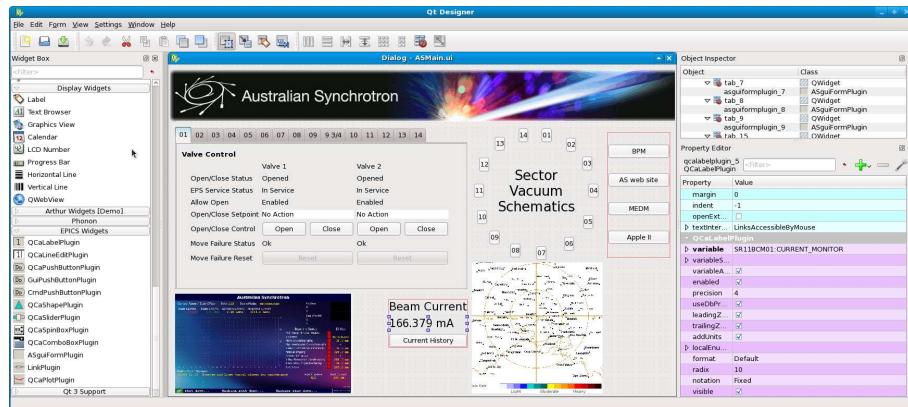


Figure 5.2: Editing a GUI

## Chapter 6

# Qt Creator

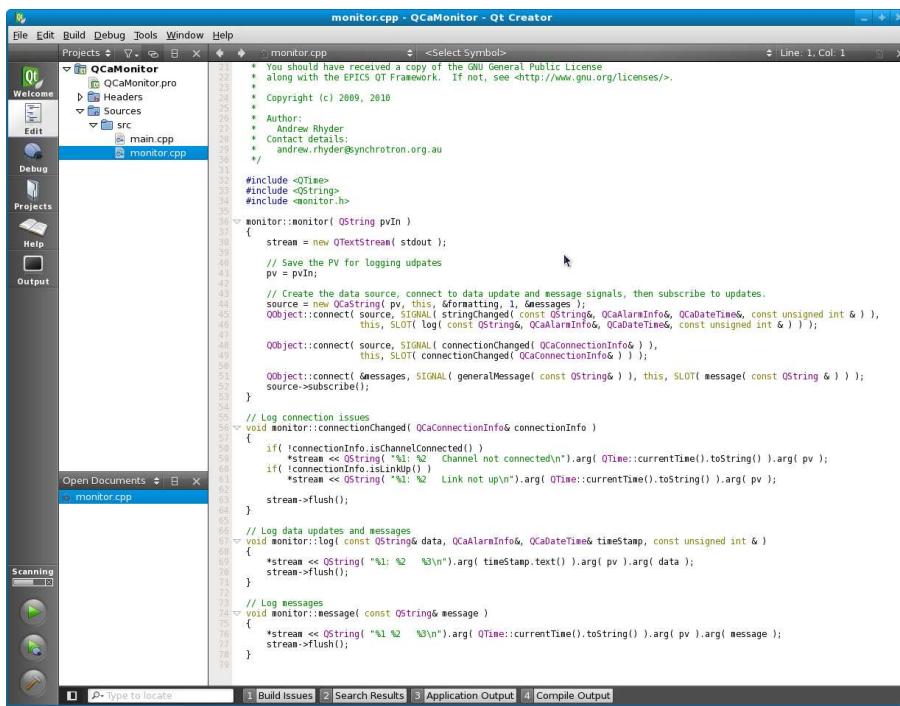


Figure 6.1: Application using epicsqt data source classes



# Chapter 7

## Class Index

### 7.1 Class Hierarchy

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

_CopyPaste . . . . .	27
_Field . . . . .	27
_Item . . . . .	28
_QDialogItem . . . . .	29
_QPushButtonGroup . . . . .	29
_QTableWidgetFileBrowser . . . . .	29
_QTableWidgetLog . . . . .	30
_QTableWidgetScript . . . . .	30
areaInfo . . . . .	30
QEAnalogIndicator::Band . . . . .	31
QEAnalogIndicator::BandList . . . . .	31
QEPeriodic::elementInfoStruct . . . . .	31
FFBuffer . . . . .	32
FFThread . . . . .	32
flipRotateMenu . . . . .	33
fullScreenWindow . . . . .	33
histogram . . . . .	34
histogramScroll . . . . .	34
historicImage . . . . .	34
imageContextMenu . . . . .	35
imageDisplayProperties . . . . .	36
imageInfo . . . . .	37
QEImage . . . . .	145
imageMarkup . . . . .	38
VideoWidget . . . . .	325
imageMarkupLegendSetText . . . . .	40
imageProperties . . . . .	44
imageProcessor . . . . .	41

imagePropertiesCore . . . . .	46
imageUpdateIndicator . . . . .	47
loginWidget . . . . .	47
markupDisplayMenu . . . . .	49
markupItem . . . . .	51
markupCrosshair1 . . . . .	48
markupCrosshair2 . . . . .	48
markupEllipse . . . . .	49
markupHLine . . . . .	50
markupLine . . . . .	54
markupRegion . . . . .	54
markupText . . . . .	55
markupVLine . . . . .	56
mpegSource . . . . .	57
QEImage . . . . .	145
mpegSourceObject . . . . .	57
QEStripChartToolBar::OwnWidgets . . . . .	58
PeriodicDialog . . . . .	58
PeriodicElementSetupForm . . . . .	59
PeriodicSetupDialog . . . . .	59
playbackTimer . . . . .	59
pointInfo . . . . .	60
profilePlot . . . . .	60
QBitStatus . . . . .	61
QEBitStatus . . . . .	79
QEAnalogIndicator . . . . .	63
QEAnalogProgressBar . . . . .	68
QECheckBoxManager . . . . .	102
QEComboBox . . . . .	103
QEConfiguredLayout . . . . .	109
QEConfiguredLayoutManager . . . . .	114
QEFileBrowser . . . . .	115
QEForm . . . . .	121
QEFrame . . . . .	125
QEPvProperties . . . . .	255
QEStripChart . . . . .	311
QEGenericButton . . . . .	130
QECheckBox . . . . .	85
QEPushButton . . . . .	239
QERadioButton . . . . .	257
QEGenericEdit . . . . .	132
QELineEdit . . . . .	204
QENumericEdit . . . . .	218
QEGroupBox . . . . .	140
QEImageMarkupThickness . . . . .	190
QEImageOptionsDialog . . . . .	191
QELabel . . . . .	191
QELineEditManager . . . . .	209

QELink . . . . .	210
QELog . . . . .	212
QELogin . . . . .	217
QELoginDialog . . . . .	218
QENumericEditManager . . . . .	222
QEPeriodic . . . . .	223
QEPeriodicComponentData . . . . .	230
QEPeriodicTaskMenu . . . . .	230
QEPeriodicTaskMenuFactory . . . . .	231
QEPlot . . . . .	231
QEPVNameLists . . . . .	255
QEPvPropertiesManager . . . . .	257
QERecipe . . . . .	274
QERecordFieldName . . . . .	276
QERecordSpec . . . . .	277
QERecordSpecList . . . . .	277
QEScript . . . . .	277
QEShape . . . . .	285
QESlider . . . . .	299
QESpinBox . . . . .	305
QEStripChartAdjustPVDialog . . . . .	314
QEStripChartContextMenu . . . . .	314
QEStripChartDurationDialog . . . . .	315
QEStripChartItem . . . . .	315
QEStripChartNames . . . . .	316
QEStripChartPushButtonSpecifications . . . . .	317
QEStripChartRangeDialog . . . . .	318
QEStripChartState . . . . .	318
QEStripChartStateList . . . . .	318
QEStripChartStatistics . . . . .	319
QEStripChartTimeDialog . . . . .	319
QEStripChartToolBar . . . . .	319
QESubstitutedLabel . . . . .	321
recording . . . . .	321
imageDisplayProperties::rgbPixel . . . . .	322
screenSelectDialog . . . . .	322
selectMenu . . . . .	323
trace . . . . .	323
userInfoStruct . . . . .	324
QEPeriodic::userInfoStructArray . . . . .	324
ValueScaling . . . . .	324
zoomMenu . . . . .	326



# Chapter 8

## Class Index

### 8.1 Class List

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

_CopyPaste . . . . .	27
_Field . . . . .	27
_Item . . . . .	28
_QDialogItem . . . . .	29
_QPushButtonGroup . . . . .	29
_QTableWidgetFileBrowser . . . . .	29
_QTableWidgetLog . . . . .	30
_QTableWidgetScript . . . . .	30
areaInfo . . . . .	30
QEAnalogIndicator::Band . . . . .	31
QEAnalogIndicator::BandList . . . . .	31
QEPeriodic::elementInfoStruct . . . . .	31
FFBuffer . . . . .	32
FFTthread . . . . .	32
flipRotateMenu . . . . .	33
fullScreenWindow . . . . .	33
histogram . . . . .	34
histogramScroll . . . . .	34
historicImage . . . . .	34
imageContextMenu . . . . .	35
imageDisplayProperties . . . . .	36
imageInfo . . . . .	37
imageMarkup . . . . .	38
imageMarkupLegendSetText . . . . .	40
imageProcessor . . . . .	41
imageProperties . . . . .	44
imagePropertiesCore . . . . .	46
imageUpdateIndicator . . . . .	47
loginWidget . . . . .	47

markupCrosshair1	48
markupCrosshair2	48
markupDisplayMenu	49
markupEllipse	49
markupHLine	50
markupItem	51
markupLine	54
markupRegion	54
markupText	55
markupVLine	56
mpegSource	57
mpegSourceObject	57
QEStripChartToolBar::OwnWidgets	58
PeriodicDialog	58
PeriodicElementSetupForm	59
PeriodicSetupDialog	59
playbackTimer	59
pointInfo	60
profilePlot	60
QBitStatus	61
QEAnalogIndicator	63
QEAnalogProgressBar	68
QEBitStatus	79
QECheckBox	85
QECheckBoxManager	102
QEComboBox	103
QEConfiguredLayout	109
QEConfiguredLayoutManager	114
QEFileBrowser	115
QEForm	121
QEFrame	125
QEGenericButton	130
QEGenericEdit	132
QEGroupBox	140
QEImage	145
QEImageMarkupThickness	190
QEImageOptionsDialog	191
QELabel	191
QELineEdit	204
QELineEditManager	209
QELink	210
QELog	212
QELogin	217
QELoginDialog	218
QENumericEdit (The QENumericEdit class This class is similar to QELineEdit (both of which are derived from QLineEdit). However this class is tailored specifically for editing numerical values )	218
QENumericEditManager	222
QEPeriodic	223
QEPeriodicComponentData	230

QEPeriodicTaskMenu . . . . .	230
QEPeriodicTaskMenuFactory . . . . .	231
QEPlot . . . . .	231
QEPushButton . . . . .	239
QEPVNameLists . . . . .	255
QEPvProperties . . . . .	255
QEPvPropertiesManager . . . . .	257
QERadioButton . . . . .	257
QERecipe . . . . .	274
QERecordFieldName . . . . .	276
QERecordSpec . . . . .	277
QERecordSpecList . . . . .	277
QEScript . . . . .	277
QEShape . . . . .	285
QESlider . . . . .	299
QESpinBox . . . . .	305
QEStripChart . . . . .	311
QEStripChartAdjustPVDialo	314
QEStripChartContextMenu . . . . .	314
QEStripChartDurationDialog . . . . .	315
QEStripChartItem . . . . .	315
QEStripChartNames . . . . .	316
QEStripChartPushButtonSpecifications . . . . .	317
QEStripChartRangeDialog . . . . .	318
QEStripChartState . . . . .	318
QEStripChartStateList . . . . .	318
QEStripChartStatistics . . . . .	319
QEStripChartTimeDialog . . . . .	319
QEStripChartToolBar (This class holds all the StripChart tool bar widgets ) . . . . .	319
QESubstitutedLabel . . . . .	321
recording . . . . .	321
imageDisplayProperties::rgbPixel . . . . .	322
screenSelectDialog . . . . .	322
selectMenu . . . . .	323
trace . . . . .	323
userInfoStruct . . . . .	324
QEPeriodic::userInfoStructArray . . . . .	324
ValueScaling . . . . .	324
VideoWidget . . . . .	325
zoomMenu . . . . .	326



# Chapter 9

## Class Documentation

### 9.1 \_CopyPaste Class Reference

#### Public Member Functions

- **\_CopyPaste** (bool pEnable, QString pProgram, QString pParameters, QString pWorkingDirectory, int pTimeOut, bool pStop, bool pLog)
- void **setEnable** (bool pEnable)
- bool **getEnable** ()
- void **setProgram** (QString pProgram)
- QString **getProgram** ()
- void **setParameters** (QString pParameters)
- QString **getParameters** ()
- void **setWorkingDirectory** (QString pWorkingDirectory)
- QString **getWorkingDirectory** ()
- void **setTimeOut** (int pTimeOut)
- int **getTimeOut** ()
- void **setStop** (bool pStop)
- bool **getStop** ()
- void **setLog** (bool pLog)
- bool **getLog** ()

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

- /tmp/epicsqt/trunk/framework/widgets/QEScript/QEScript.h
- /tmp/epicsqt/trunk/framework/widgets/QEScript/QEScript.cpp

### 9.2 \_Field Class Reference

#### Public Member Functions

- QEWidget \* **getWidget** ()

- void **setWidget** (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

- QEWidget \* **qeWidget**

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

- /tmp/epicsqt/trunk/framework/widgets/QEConfiguredLayout/QEConfiguredLayout.h
- /tmp/epicsqt/trunk/framework/widgets/QEConfiguredLayout/QEConfiguredLayout.cpp

## 9.3 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:

- /tmp/epicsqt/trunk/framework/widgets/QEConfiguredLayout/QEConfiguredLayout.h
- /tmp/epicsqt/trunk/framework/widgets/QEConfiguredLayout/QEConfiguredLayout.cpp

## 9.4 \_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:

- /tmp/epicsqt/trunk/framework/widgets/QEConfiguredLayout/QEConfiguredLayout.h
- /tmp/epicsqt/trunk/framework/widgets/QEConfiguredLayout/QEConfiguredLayout.cpp

## 9.5 \_QPushButtonGroup Class Reference

### Public Slots

- void **buttonGroupClicked** ()

### Public Member Functions

- **\_QPushButtonGroup** (QWidget \*pParent=0, QString pItemName="", 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:

- /tmp/epicsqt/trunk/framework/widgets/QEConfiguredLayout/QEConfiguredLayout.h
- /tmp/epicsqt/trunk/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:

- /tmp/epicsqt/trunk/framework/widgets/QEFileBrowser/QEFileBrowser.h
- /tmp/epicsqt/trunk/framework/widgets/QEFileBrowser/QEFileBrowser.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:

- /tmp/epicsqt/trunk/framework/widgets/QELog/QELog.h
- /tmp/epicsqt/trunk/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:

- /tmp/epicsqt/trunk/framework/widgets/QEScript/QEScript.h
- /tmp/epicsqt/trunk/framework/widgets/QEScript/QEScript.cpp

## 9.9 arealInfo Class Reference

### Public Member Functions

- void **setX1** (long x)
- void **setY1** (long y)
- void **setX2** (long x)
- void **setY2** (long y)
- void **setX** (long x)
- void **setY** (long y)
- void **setW** (long w)
- void **setH** (long h)
- void **setPoint1** (QPoint p1In)
- void **setPoint2** (QPoint p2In)
- void **clearX1** ()
- void **clearY1** ()

- void **clearX2** ()
- void **clearY2** ()
- void **clearX** ()
- void **clearY** ()
- void **clearW** ()
- void **clearH** ()
- bool **getStatus** ()
- QRect **getArea** ()
- QPoint **getPoint1** ()
- QPoint **getPoint2** ()

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

- /tmp/epicsqt/trunk/framework/widgets/QEImage/QEImage.h

## 9.10 QEAnalogIndicator::Band Struct Reference

### Public Attributes

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

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

- /tmp/epicsqt/trunk/framework/widgets/QEAnalogIndicator/QEAnalogIndicator.h

## 9.11 QEAnalogIndicator::BandList Class Reference

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

- /tmp/epicsqt/trunk/framework/widgets/QEAnalogIndicator/QEAnalogIndicator.h

## 9.12 QEPeriodic::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:

- /tmp/epicsqt/trunk/framework/widgets/QEPeriodic/QEPeriodic.h

## 9.13 FFBuffer Class Reference

### Public Member Functions

- void **reserve** ()
- void **release** ()
- bool **grabFree** ()

### Public Attributes

- QMutex \* **mutex**
- unsigned char \* **mem**
- AVFrame \* **pFrame**
- PixelFormat **pix\_fmt**
- int **width**
- int **height**
- int **refs**

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

- /tmp/epicsqt/trunk/framework/widgets/QEImage/mpeg.h
- /tmp/epicsqt/trunk/framework/widgets/QEImage/mpeg.cpp

## 9.14 FFThread Class Reference

### Public Slots

- void **stopGracefully** ()

### Signals

- void **updateSignal** ([FFBuffer](#) \*buf)

### Public Member Functions

- **FFThread** (const QString &url, QObject \*parent)
- void **run** ()

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

- /tmp/epicsqt/trunk/framework/widgets/QElImage/mpeg.h
- /tmp/epicsqt/trunk/framework/widgets/QElImage/mpeg.cpp

## 9.15 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:

- /tmp/epicsqt/trunk/framework/widgets/QElImage/flipRotateMenu.h
- /tmp/epicsqt/trunk/framework/widgets/QElImage/flipRotateMenu.cpp

## 9.16 fullScreenWindow Class Reference

### Signals

- void **fullScreenResize** ()

### Public Member Functions

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

### Protected Member Functions

- void **resizeEvent** (QResizeEvent \*event)

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

- /tmp/epicsqt/trunk/framework/widgets/QElImage/fullScreenWindow.h
- /tmp/epicsqt/trunk/framework/widgets/QElImage/fullScreenWindow.cpp

## 9.17 histogram Class Reference

### Public Member Functions

- **histogram** (QWidget \*parent, [imageDisplayProperties](#) \*idp)

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

- /tmp/epicsqt/trunk/framework/widgets/QElImage/brightnessContrast.h
- /tmp/epicsqt/trunk/framework/widgets/QElImage/brightnessContrast.cpp

## 9.18 histogramScroll Class Reference

### Public Member Functions

- **histogramScroll** (QWidget \*parent, [imageDisplayProperties](#) \*idp)

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

- /tmp/epicsqt/trunk/framework/widgets/QElImage/brightnessContrast.h
- /tmp/epicsqt/trunk/framework/widgets/QElImage/brightnessContrast.cpp

## 9.19 historicImage Class Reference

### Public Member Functions

- **historicImage** (QByteArray image, unsigned long dataSize, QCaAlarmInfo &alarmInfo, QCaDateTime &time)

### Public Attributes

- QByteArray **image**
- unsigned long **dataSize**
- QCaAlarmInfo **alarmInfo**
- QCaDateTime **time**

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

- /tmp/epicsqt/trunk/framework/widgets/QElImage/recording.h
- /tmp/epicsqt/trunk/framework/widgets/QElImage/QElImage.cpp

## 9.20 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\_ABOUT\_IMAGE**, **ICM\_ENABLE\_VERT1**,  
    **ICM\_ENABLE\_VERT2**,  
    **ICM\_ENABLE\_VERT3**, **ICM\_ENABLE\_VERT4**, **ICM\_ENABLE\_VERT5**, **ICM\_ENABLE\_HOZ1**,  
    **ICM\_ENABLE\_HOZ2**, **ICM\_ENABLE\_HOZ3**, **ICM\_ENABLE\_HOZ4**, **ICM\_ENABLE\_HOZ5**,  
    **ICM\_ENABLE\_AREA1**, **ICM\_ENABLE\_AREA2**, **ICM\_ENABLE\_AREA3**, **ICM\_ENABLE\_AREA4**,  
    **ICM\_ENABLE\_LINE**, **ICM\_ENABLE\_TARGET**, **ICM\_ENABLE\_BEAM**, **ICM\_DISPLAY\_BUTTON\_BAR**,  
    **ICM\_DISPLAY\_IMAGE\_DISPLAY\_PROPERTIES**, **ICM\_DISPLAY\_RECORDER**,  
    **ICM\_ZOOM\_SELECTED**, **ICM\_ZOOM\_FIT**,  
    **ICM\_ZOOM\_PLUS**, **ICM\_ZOOM\_MINUS**, **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\_HSLICE1**, **ICM\_SELECT\_HSLICE2**,  
    **ICM\_SELECT\_HSLICE3**, **ICM\_SELECT\_HSLICE4**, **ICM\_SELECT\_HSLICE5**, **ICM\_SELECT\_VSLICE1**,  
    **ICM\_SELECT\_VSLICE2**, **ICM\_SELECT\_VSLICE3**, **ICM\_SELECT\_VSLICE4**, **ICM\_SELECT\_VSLICE5**,  
    **ICM\_SELECT\_AREA1**, **ICM\_SELECT\_AREA2**, **ICM\_SELECT\_AREA3**, **ICM\_SELECT\_AREA4**,  
    **ICM\_SELECT\_PROFILE**, **ICM\_SELECT\_TARGET**, **ICM\_SELECT\_BEAM**, **ICM\_CLEAR\_MARKUP**,  
    **ICM\_SET\_LEGEND**, **ICM\_THICKNESS\_ONE\_MARKUP**, **ICM\_THICKNESS\_SELECT\_MARKUP**, **ICM\_COPY\_PLOT\_DATA**,  
    **ICM\_FULL\_SCREEN**, **ICM\_DISPLAY\_HSLICE1**, **ICM\_DISPLAY\_HSLICE2**, **ICM\_DISPLAY\_HSLICE3**,  
    **ICM\_DISPLAY\_HSLICE4**, **ICM\_DISPLAY\_HSLICE5**, **ICM\_DISPLAY\_VSLICE1**,  
    **ICM\_DISPLAY\_VSLICE2**,  
    **ICM\_DISPLAY\_VSLICE3**, **ICM\_DISPLAY\_VSLICE4**, **ICM\_DISPLAY\_VSLICE5**,  
    **ICM\_DISPLAY\_AREA1**,  
    **ICM\_DISPLAY\_AREA2**, **ICM\_DISPLAY\_AREA3**, **ICM\_DISPLAY\_AREA4**, **ICM\_DISPLAY\_PROFILE**,

---

```
ICM_DISPLAY_TARGET, ICM_DISPLAY_BEAM, ICM_DISPLAY_TIMESTAMP,
ICM_DISPLAY_ELLIPSE,
ICM_OPTIONS }
```

### Public Member Functions

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

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

- /tmp/epicsqt/trunk/framework/widgets/QElImage/imageContextMenu.h
- /tmp/epicsqt/trunk/framework/widgets/QElImage/imageContextMenu.cpp

## 9.21 imageDisplayProperties Class Reference

### Classes

- struct [rgbPixel](#)

### Signals

- void **brightnessContrastAutolImage** ()
- void **imageDisplayPropertiesChange** ()

### Public Member Functions

- void **setBrightnessContrast** (const unsigned int max, const unsigned int min)
- void **setAutoBrightnessContrast** (bool autoBrightnessContrast)
- void **setContrastReversal** (bool contrastReversal)
- void **setLog** (bool log)
- void **setFalseColour** (bool falseColour)
- bool **getAutoBrightnessContrast** ()
- bool **getContrastReversal** ()
- bool **getLog** ()
- bool **getFalseColour** ()
- int **getLowPixel** ()
- int **getHighPixel** ()
- void **setStatistics** (unsigned int minPln, unsigned int maxPln, unsigned int bitDepth, unsigned int binsIn[HISTOGRAM\_BINS], [rgbPixel](#) pixelLookup[256])
- void **showStatistics** ()

- void **setHistZoom** (int value)
- int **getHistZoom** ()
- bool **statisticsValid** ()

### Public Attributes

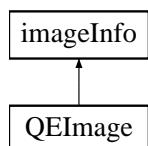
- int **zeroValue**
- int **fullValue**
- bool **defaultFullValue**
- unsigned int **range**
- unsigned int **maxP**
- unsigned int **minP**
- unsigned int **depth**
- unsigned int **bins** [HISTOGRAM\_BINS]
- bool **statisticsSet**
- **rgbPixel** \* **pixelLookup**
- QLabel \* **histXLabel**

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

- /tmp/epicsqt/trunk/framework/widgets/QEImage/brightnessContrast.h
- /tmp/epicsqt/trunk/framework/widgets/QEImage/brightnessContrast.cpp

## 9.22 imageInfo Class Reference

Inheritance diagram for imageInfo:



### Public Member Functions

- void **showInfo** (bool show)
- QLayout \*  **getInfoWidget** ()
- void **infoShow** (const bool show)
- void **infoUpdateTarget** ()
- void **infoUpdateTarget** (const int x, const int y)
- void **infoUpdateBeam** ()
- void **infoUpdateBeam** (const int x, const int y)
- void **infoUpdateVertProfile** ()
- void **infoUpdateVertProfile** (const int x, const unsigned int thickness)

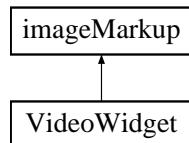
- void **infoUpdateHozProfile** ()
- void **infoUpdateHozProfile** (const int y, const unsigned int thickness)
- void **infoUpdateProfile** ()
- void **infoUpdateProfile** (const QPoint start, const QPoint end, const unsigned int thickness)
- void **infoUpdateRegion** (const unsigned int region)
- void **infoUpdateRegion** (const unsigned int region, const int x1, const int y1, const int x2, const int y2)
- void **infoUpdatePixel** ()
- void **infoUpdatePixel** (const QPoint pos, int value)
- void **infoUpdateZoom** ()
- void **infoUpdateZoom** (int value)
- void **infoUpdatePaused** ()
- void **infoUpdatePaused** (bool paused)
- void **setBriefInfoArea** (const bool briefIn)
- bool **getBriefInfoArea** ()
- void **freshImage** (QDateTime &time)

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

- /tmp/epicsqt/trunk/framework/widgets/QElImage/imageInfo.h
- /tmp/epicsqt/trunk/framework/widgets/QElImage/imageInfo.cpp

## 9.23 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\_H1\_SLICE**, **MARKUP\_ID\_H2\_SLICE**, **MARKUP\_ID\_H3\_SLICE**,
 **MARKUP\_ID\_H4\_SLICE**,
 **MARKUP\_ID\_H5\_SLICE**, **MARKUP\_ID\_V1\_SLICE**, **MARKUP\_ID\_V2\_SLICE**, **MARKUP\_ID\_V3\_SLICE**,
 **MARKUP\_ID\_V4\_SLICE**, **MARKUP\_ID\_V5\_SLICE**, **MARKUP\_ID\_LINE**, **MARKUP\_ID\_TARGET**,
 }

```
MARKUP_ID_BEAM, MARKUP_ID_TIMESTAMP, MARKUP_ID_ELLIPSE, MARKUP_ID_COUNT,
MARKUP_ID_NONE }
• enum beamAndTargetOptions { CROSSHAIR1, CROSSHAIR2 }
```

## 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)
- void **markupRegionValueChange** (int arealIndex, QRect area, bool displayMarkups)
  
- void **markupH1ProfileChange** (int y, bool displayMarkups)
- void **markupH2ProfileChange** (int y, bool displayMarkups)
- void **markupH3ProfileChange** (int y, bool displayMarkups)
- void **markupH4ProfileChange** (int y, bool displayMarkups)
- void **markupH5ProfileChange** (int y, bool displayMarkups)
- void **markupV1ProfileChange** (int x, bool displayMarkups)
- void **markupV2ProfileChange** (int x, bool displayMarkups)
- void **markupV3ProfileChange** (int x, bool displayMarkups)
- void **markupV4ProfileChange** (int x, bool displayMarkups)
- void **markupV5ProfileChange** (int x, bool displayMarkups)
- void **markupLineProfileChange** (QPoint start, QPoint end, bool displayMarkups)
  
- void **markupTargetValueChange** (QPoint point, bool displayMarkups)
- void **markupBeamValueChange** (QPoint point, bool displayMarkups)
- void **markupEllipseValueChange** (QPoint point1, QPoint point2, bool displayMarkups)
- void **markupValueChange** (int markup, bool displayMarkups, QPoint p1, QPoint p2=QPoint())
- QCursor **getCircleCursor** ()
- QCursor **getTargetCursor** ()
- QCursor **getVLineCursor** ()
- QCursor **getHLineCursor** ()
- QCursor **getLineCursor** ()
- QCursor **getRegionCursor** ()
- virtual void **markupSetCursor** (QCursor cursor)=0
- void **setMarkupLegend** (markupIds mode, QString legend)
- QString **getMarkupLegend** (markupIds mode)
- void **clearMarkup** (markupIds markupId)
- void **showMarkup** (markupIds markupId)
- void **displayMarkup** (markupIds markupId, bool state)

- bool **isMarkupVisible** (markupIds mode)
- double **getZoomScale** ()
- QSize **getImageSize** ()
- void **setImageSize** (const QSize &imageSizeIn)
- beamAndTargetOptions **getTargetOption** ()
- void **setTargetOption** (beamAndTargetOptions option)
- beamAndTargetOptions **getBeamOption** ()
- void **setBeamOption** (beamAndTargetOptions option)
- void **setBeamOrTargetOption** (markupIds item, beamAndTargetOptions option)

### Public Attributes

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

### Protected Member Functions

- void **drawMarkups** (QPainter &p, const QRect &rect)
- bool **anyVisibleMarkups** ()
- 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** (const double scale)
- virtual void **markupChange** (QVector< QRect > &changedAreas)=0
- virtual void **markupAction** (markupIds mode, bool complete, bool clearing, QPoint point1, QPoint point2, unsigned int thickness)=0

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

- /tmp/epicsqt/trunk/framework/widgets/QElImage/imageMarkup.h
- /tmp/epicsqt/trunk/framework/widgets/QElImage/imageMarkup.cpp

## 9.24 imageMarkupLegendSetText Class Reference

### Public Member Functions

- **imageMarkupLegendSetText** (QString existingLegend, QWidget \*parent=0)
- QString **getLegend** ()

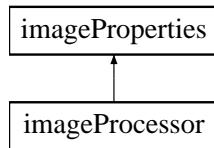
The documentation for this class was generated from the following files:

- /tmp/epicsqt/trunk/framework/widgets/QElImage/imageMarkupLegendSetText.h
- /tmp/epicsqt/trunk/framework/widgets/QElImage/imageMarkupLegendSetText.cpp

## 9.25 imageProcessor Class Reference

```
#include <imageProcessor.h>
```

Inheritance diagram for imageProcessor:



### Signals

- void [imageBuilt](#) (QImage imageData, QString error)  
*An image has been generated from image data and is now ready for presentation.*

### Public Member Functions

- [imageProcessor \(\)](#)  
*Constructor.*
- [~imageProcessor \(\)](#)  
*Destructor.*
- void [setImageBuff \(\)](#)  
*Ensure the image buffer used to process images is appropriately sized. This is called whenever an image attribute changes that may affect the buffer size required.*
- void [setImage](#) (const QByteArray &imageIn, unsigned long dataSize)  
*Save the image data for analysis processing and display.*
- void [buildImage \(\)](#)  
*Generate a new image.*
- bool [setWidth](#) (unsigned long uValue)  
*Set the image width.*
- bool [setHeight](#) (unsigned long uValue)  
*Set the image height.*
- bool [setNumDimensions](#) (unsigned long uValue)  
*Set the number of dimensions.*
- bool [setDimension0](#) (unsigned long uValue)  
*Set the first dimension (width if two dimensions, bytes per element if three dimensions)*

- bool `setDimension1` (unsigned long uValue)  
*Set the second dimension (height if two dimensions, width if three dimensions)*
- bool `setDimension2` (unsigned long uValue)  
*Set the third dimension (unused if two dimensions, height if three dimensions)*
- void `setClippingOn` (bool clippingOnIn)  
*Set clipping flag. If true, `setClippingLow()` and `setClippingHigh()` are used to set clipping values.*
- void `setClippingLow` (unsigned int value)  
*Set pixel value below which low clip colour is displayed.*
- void `setClippingHigh` (unsigned int value)  
*Set pixel value above which high clip colour is displayed.*
- int `getScanOption` ()  
*Determine the way the input pixel data must be scanned to accommodate the required rotate and flip options.*
- void `getPixelTranslation` ()  
*Generate a lookup table to convert raw pixel values to display pixel values.*
- unsigned int `maxPixelValue` ()  
*Determine the maximum pixel value for the current format.*
- unsigned int `rotatedImageBuffWidth` ()  
*Return the image width following any rotation.*
- unsigned int `rotatedImageBuffHeight` ()  
*Return the image height following any rotation.*
- `imageDisplayProperties::rgbPixel getFalseColor` (const unsigned char value)  
*Get a false color representation for an entry from the color lookup table.*
- int `getElementCount` ()  
*Determine the element count expected based on the available dimensions.*
- bool `validateDimensions` ()  
*Determine if the image dimensional information is valid.*
- void `getPixelRange` (const QRect &area, unsigned int \*min, unsigned int \*max)  
*Determine the range of pixel values an area of the image.*
- bool `hasImage` ()  
*Return true if the current image is empty.*
- bool `hasImageBuff` ()  
*Return true if the current image data buffer is empty.*
- const unsigned char \* `getImageDataPtr` (QPoint &pos)  
*Return a pointer to pixel data in the original image data.*
- int `getPixelValueFromData` (const unsigned char \*ptr)  
*Return a number representing a pixel intensity given a pointer into an image data buffer.*
- double `getFloatingPixelValueFromData` (const unsigned char \*ptr)  
*Return a floating point number representing a pixel intensity given a pointer into an image data buffer.*
- QImage `copyImage` ()  
*Return a QImage based on the current image.*

- void [generateVSliceData](#) (QVector< QPointF > &vSliceData, int x, unsigned int thickness)
 

*Generate a series of pixel values from a vertical slice through the current image.*
- void [generateHSliceData](#) (QVector< QPointF > &hSliceData, int y, unsigned int thickness)
 

*Generate a series of pixel values from a horizontal slice through the current image.*
- void [generateProfileData](#) (QVector< QPointF > &profileData, QPoint point1, QPoint point2, unsigned int thickness)
 

*Generate a series of pseudo pixel values from an arbitrary line between two pixels.*
- QRect [rotateFlipToDataRectangle](#) (const QRect &rect)
 

*Transform a rectangle from the image to the original data according to current rotation and flip options.*
- QRect [rotateFlipToDataRectangle](#) (const QPoint &pos1, const QPoint &pos2)
 

*Transform a rectangle from the image to the original data according to current rotation and flip options.*
- QPoint [rotateFlipToDataPoint](#) (const QPoint &pos)
 

*Transform a point from the image to the original data according to current rotation and flip options.*
- QRect [rotateFlipToImageRectangle](#) (const QRect &rect)
 

*Transform a rectangle from the original data to the image according to current rotation and flip options.*
- QRect [rotateFlipToImageRectangle](#) (const QPoint &pos1, const QPoint &pos2)
 

*Transform a rectangle from the original data to the image according to current rotation and flip options.*
- QPoint [rotateFlipToImagePoint](#) (const QPoint &pos)
 

*Transform a point from the original data to the image according to current rotation and flip options.*
- void [run](#) ()
 

*Starts the processing loop.*

## Public Attributes

- QWaitCondition **imageSync**
- QReadWriteLock **imageWait**
- QMutex **imageLock**
- bool **finishNow**
- **imagePropertiesCore** \* **next**

### 9.25.1 Detailed Description

This class generates images for presentation from raw image data and formatting information such as brightness, contrast, flip, rotate, canvas size, etc. The work is performed in a dedicated thread .

### 9.25.2 Member Function Documentation

#### 9.25.2.1 int imageProcessor::getPixelValueFromData ( const unsigned char \* ptr )

Return a number representing a pixel intensity given a pointer into an image data buffer.

```
!! not done - copy of RGB1
```

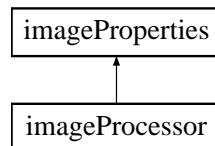
The documentation for this class was generated from the following files:

- /tmp/epicsqt/trunk/framework/widgets/QElImage/imageProcessor.h
- /tmp/epicsqt/trunk/framework/widgets/QElImage/imageProcessor.cpp

## 9.26 imageProperties Class Reference

```
#include <imageProperties.h>
```

Inheritance diagram for imageProperties:



### Public Types

- enum `rotationOptions` { `ROTATION_0`, `ROTATION_90_RIGHT`, `ROTATION_90_LEFT`, `ROTATION_180` }

### Public Member Functions

- `imageProperties ()`  
*Constructor.*
- `void setRotation (rotationOptions rotationIn)`
- `rotationOptions getRotation ()`
- `void setFlipVert (bool flipVertIn)`
- `bool getFlipVert ()`
- `void setFlipHoz (bool flipHozIn)`
- `bool getFlipHoz ()`

- void **setImageBuffWidth** (unsigned long imageBuffWidthIn)
- void **setImageBuffHeight** (unsigned long imageBuffHeightIn)
- unsigned long **getImageBuffWidth** ()
- unsigned long **getImageBuffHeight** ()
- imageDataFormats::formatOptions **getFormat** ()
- void **setFormat** (imageDataFormats::formatOptions formatIn)
- bool **setFormat** (const QString &text)
- void **setBitDepth** (unsigned int bitDepthIn)
- unsigned int **getBitDepth** ()
- void **setElementsPerPixel** (unsigned long elementsPerPixelIn)
- void **setImageDisplayProperties** (imageDisplayProperties \*imageDisplayPropsIn)
  
- void **setWidthHeightFromDimensions** ()  
    *// Update the image dimensions (width and height) from the area detector dimension variables.*
- void **invalidatePixelLookup** ()  
    *recalculate (when next required) pixel summary information*
- QString **getInfoText** ()  
    *Generate textual information regarding the current image.*

### Protected Attributes

- imageDisplayProperties \* **imageDisplayProps**
- imageDataFormats::formatOptions **formatOption**
- unsigned int **bitDepth**
- unsigned long **imageDataSize**
- unsigned long **elementsPerPixel**
- unsigned long **bytesPerPixel**
- QByteArray **imageData**
- unsigned long **receivedImageSize**
- QString **previousMessageText**
- QByteArray **imageBuff**
- QByteArray **lastImageBuff**
- QImage **image**
- unsigned long **imageBuffWidth**
- unsigned long **imageBuffHeight**
- unsigned long **numDimensions**
- unsigned long **imageDimension0**
- unsigned long **imageDimension1**
- unsigned long **imageDimension2**
- bool **pixelLookupValid**
- imageDisplayProperties::rgbPixel **pixelLookup** [256]
- int **pixelLow**
- int **pixelHigh**
- bool **clippingOn**
- unsigned int **clippingLow**

- unsigned int **clippingHigh**
- **rotationOptions rotation**
- bool **flipVert**
- bool **flipHoz**

### 9.26.1 Detailed Description

This class manages the image attributes required for generating a QImage from a QByteArray holding CA image data. It is used as the base class for the [imageProcessor](#) class. Note, while this class holds and manages all the information needed to process an image, a snapshot of all the information required for processing an image in a separate thread is made by the [imagePropertiesCore](#) class.

### 9.26.2 Member Enumeration Documentation

#### 9.26.2.1 enum imageProperties::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.

### 9.26.3 Constructor & Destructor Documentation

#### 9.26.3.1 imageProperties::imageProperties( )

Constructor.

Construction. Set all image attributes to sensible defaults.

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

- /tmp/epicsqt/trunk/framework/widgets/QElImage/imageProperties.h
- /tmp/epicsqt/trunk/framework/widgets/QElImage/imageProperties.cpp

## 9.27 imagePropertiesCore Class Reference

### Public Member Functions

- **imagePropertiesCore** (QByteArray imageDataIn, QByteArray imageBuffIn, unsigned long imageBuffWidthIn, unsigned long imageBuffHeightIn, int scanOptionIn, unsigned long bytesPerPixelIn, int pixelLowIn, int pixelHighIn, unsigned

```
int bitDepthIn, imageDisplayProperties::rgbPixel *pixelLookupIn, imageDataFormats::formatOptions formatOptionIn, unsigned long imageDataSizeIn, imageDisplayProperties *imageDisplayPropsIn, unsigned int rotatedImageBuffWidthIn, unsigned int rotatedImageBuffHeightIn)
• QImage buildImageCore ()
```

### 9.27.1 Member Function Documentation

#### 9.27.1.1 QImage imagePropertiesCore::buildImageCore ( )

```
!! not done yet - just do the same as RGB1 for the time being and hope
!! not done yet - just do the same as RGB1 for the time being and hope
!! not done yet. do the same as for YUV422
!! not done yet. do the same as for YUV422
```

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

- /tmp/epicsqt/trunk/framework/widgets/QEImage/imageProperties.h
- /tmp/epicsqt/trunk/framework/widgets/QEImage/imageProcessor.cpp

## 9.28 imageUpdateIndicator Class Reference

### Public Member Functions

- void **freshImage** ()
- void **paintEvent** (QPaintEvent \*)

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

- /tmp/epicsqt/trunk/framework/widgets/QEImage/imageInfo.h
- /tmp/epicsqt/trunk/framework/widgets/QEImage/imageInfo.cpp

## 9.29 loginWidget Class Reference

### Public Member Functions

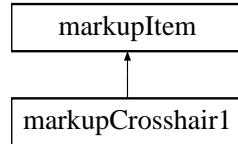
- **loginWidget** (QELogin \*ownerIn)
- userLevelTypes::userLevels **getUserType** ()
- QString **getPassword** ()
- void **clearPassword** ()

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

- /tmp/epicsqt/trunk/framework/widgets/QELogin/QELogin.h
- /tmp/epicsqt/trunk/framework/widgets/QELogin/QELogin.cpp

## 9.30 markupCrosshair1 Class Reference

Inheritance diagram for markupCrosshair1:



### Public Member Functions

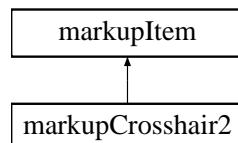
- **markupCrosshair1** ([imageMarkup](#) \*ownerIn, const bool interactiveIn, const bool reportOnMoveIn, const QString legendIn)
- void **startDrawing** (const QPoint pos)
- void **setArea** ()
- void **drawMarkup** ( QPainter &p )
- void **moveTo** (const QPoint pos)
- bool **isOver** (const QPoint point, QCursor \*cursor)
- QPoint **origin** ()
- QCursor **cursorForHandle** (const markupItem::markupHandles handle)
- QPoint **getPoint1** ()
- QPoint **getPoint2** ()
- QCursor **defaultCursor** ()
- void **nonInteractiveUpdate** (QPoint p1, QPoint p2)

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

- /tmp/epicsqt/trunk/framework/widgets/QElImage/markupTarget.h
- /tmp/epicsqt/trunk/framework/widgets/QElImage/markupTarget.cpp

## 9.31 markupCrosshair2 Class Reference

Inheritance diagram for markupCrosshair2:



### Public Member Functions

- **markupCrosshair2** (imageMarkup \*ownerIn, const bool interactiveIn, const bool reportOnMoveIn, const QString legendIn)
- void **startDrawing** (const QPoint pos)
- void **setArea** ()
- void **drawMarkup** (QPainter &p)
- void **moveTo** (const QPoint pos)
- bool **isOver** (const QPoint point, QCursor \*cursor)
- QPoint **origin** ()
- QCursors **cursorForHandle** (const markupItem::markupHandles handle)
- QPoint **getPoint1** ()
- QPoint **getPoint2** ()
- QCursors **defaultCursor** ()
- void **nonInteractiveUpdate** (QPoint p1, QPoint p2)

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

- /tmp/epicsqt/trunk/framework/widgets/QEImage/markupBeam.h
- /tmp/epicsqt/trunk/framework/widgets/QEImage/markupBeam.cpp

## 9.32 markupDisplayMenu Class Reference

### Public Member Functions

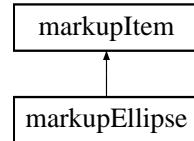
- **markupDisplayMenu** (QWidget \*parent=0)
- void **setDisplay** (imageContextMenu::imageContextMenuOptions option, bool state)
- void **setItemText** (imageContextMenu::imageContextMenuOptions option, QString title)
- bool **isDisplayed** (imageContextMenu::imageContextMenuOptions option)
- void **enable** (imageContextMenu::imageContextMenuOptions option, bool state)

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

- /tmp/epicsqt/trunk/framework/widgets/QEImage/markupDisplayMenu.h
- /tmp/epicsqt/trunk/framework/widgets/QEImage/markupDisplayMenu.cpp

## 9.33 markupEllipse Class Reference

Inheritance diagram for markupEllipse:



### Public Member Functions

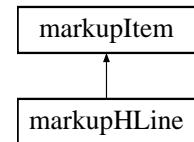
- **markupEllipse** ([imageMarkup](#) \*ownerIn, const bool interactiveIn, const bool reportOnMoveIn, const QString legendIn)
- void **startDrawing** (const QPoint pos)
- void **setArea** ()
- void **drawMarkup** ( QPainter &p )
- void **moveTo** (const QPoint pos)
- bool **isOver** (const QPoint point, QCursor \*cursor)
- QPoint **origin** ()
- QCursor **cursorForHandle** (const markupItem::markupHandles handle)
- QPoint **getPoint1** ()
- QPoint **getPoint2** ()
- QCursor **defaultCursor** ()
- void **nonInteractiveUpdate** (QPoint p1, QPoint p2)

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

- /tmp/epicsqt/trunk/framework/widgets/QElImage/markupEllipse.h
- /tmp/epicsqt/trunk/framework/widgets/QElImage/markupEllipse.cpp

### 9.34 markupHLine Class Reference

Inheritance diagram for markupHLine:



### Public Member Functions

- **markupHLine** ([imageMarkup](#) \*ownerIn, const bool interactiveIn, const bool reportOnMoveIn, const QString legendIn)
- void **startDrawing** (const QPoint pos)
- void **setArea** ()

- void **drawMarkup** (QPainter &p)
- void **moveTo** (const QPoint pos)
- bool **isOver** (const QPoint point, QCursor \*cursor)
- QPoint **origin** ()
- QCursor **cursorForHandle** (const markupItem::markupHandles handle)
- QPoint **getPoint1** ()
- QPoint **getPoint2** ()
- QCursor **defaultCursor** ()
- void **nonInteractiveUpdate** (QPoint p1, QPoint p2)

### 9.34.1 Member Function Documentation

#### 9.34.1.1 void markupHLine::drawMarkup ( QPainter & p ) [virtual]

!! draw the handle in the middle of the existing view, not the entire image

Implements [markupItem](#).

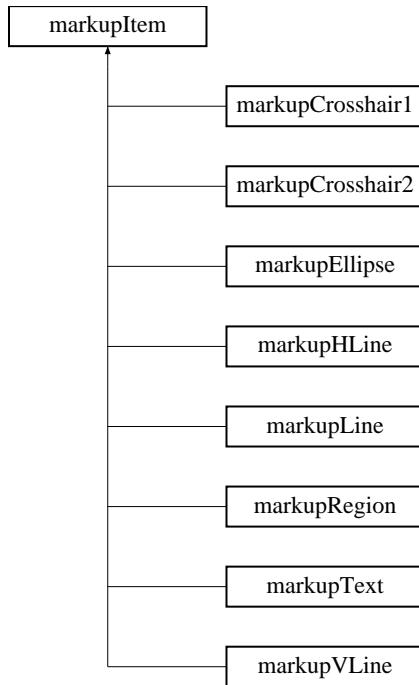
The documentation for this class was generated from the following files:

- /tmp/epicsqt/trunk/framework/widgets/QEImage/markupHLine.h
- /tmp/epicsqt/trunk/framework/widgets/QEImage/markupHLine.cpp

## 9.35 markupItem Class Reference

Inheritance diagram for markupItem:

---



## Public Types

- enum **markupHandles** {
   
    **MARKUP\_HANDLE\_NONE**, **MARKUP\_HANDLE\_START**, **MARKUP\_HANDLE\_END**, **MARKUP\_HANDLE\_CENTER**,
   
    **MARKUP\_HANDLE\_TL**, **MARKUP\_HANDLE\_TR**, **MARKUP\_HANDLE\_BL**, **MARKUP\_HANDLE\_BR**,
   
    **MARKUP\_HANDLE\_T**, **MARKUP\_HANDLE\_B**, **MARKUP\_HANDLE\_L**, **MARKUP\_HANDLE\_R** }

## Public Member Functions

- void **drawMarkupItem** (QPainter &p)
- QSize **getImageSize** ()
- virtual QPoint **origin** ()=0
- virtual void **moveTo** (const QPoint pos)=0
- virtual void **startDrawing** (const QPoint pos)=0
- virtual bool **isOver** (const QPoint point, QCursor \*cursor)=0
- virtual QCursors **cursorForHandle** (const markupItem::markupHandles handle)=0
  
- virtual QPoint **getPoint1** ()=0
- virtual QPoint **getPoint2** ()=0
- virtual QCursors **defaultCursor** ()=0

- virtual void **nonInteractiveUpdate** (QPoint, QPoint)
- void **setThickness** (const unsigned int thicknessIn)
- unsigned int **getThickness** ()
- void **setLegend** (const QString legendIn)
- const QString **getLegend** ()
- void **setColor** (QColor colorIn)
- QColor **getColor** ()

### Public Attributes

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

### Protected Types

- enum **isOverOptions** { **OVER\_LINE**, **OVER\_BORDER**, **OVER\_AREA** }
- enum **legendJustification** { **ABOVE\_RIGHT**, **BELOW\_LEFT**, **BELOW\_RIGHT** }

### Protected Member Functions

- **markupItem** ([imageMarkup](#) \*ownerIn, const isOverOptions over, const bool interactiveIn, const bool reportOnMoveIn, const QString legendIn)
- virtual void **setArea** ()=0
- virtual void **drawMarkup** ( QPainter &p )=0
- bool **pointIsNear** (QPoint p1, QPoint p)
- const QSize **getLegendSize** ()
- void **addLegendArea** ()
- const QPoint **getLegendTextOrigin** (QPoint posScaled)
- void **setLegendOffset** (QPoint offset, legendJustification just)
- const QPoint **getLegendOffset** ()
- void **drawLegend** ( QPainter &p, QPoint posScaled)
- QPoint **limitPointToImage** (const QPoint pos)
- double **getZoomScale** ()

### Protected Attributes

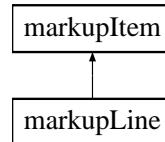
- markupHandles **activeHandle**
- [imageMarkup](#) \* **owner**
- unsigned int **thickness**
- unsigned int **maxThickness**

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

- /tmp/epicsqt/trunk/framework/widgets/QElImage/markupItem.h
- /tmp/epicsqt/trunk/framework/widgets/QElImage/markupItem.cpp

## 9.36 markupLine Class Reference

Inheritance diagram for markupLine:



### Public Member Functions

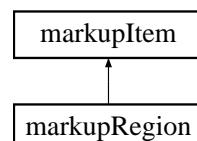
- **markupLine** ([imageMarkup](#) \*ownerIn, const bool interactiveIn, const bool reportOnMoveIn, const QString legendIn)
- void **startDrawing** (const QPoint pos)
- void **setArea** ()
- void **drawMarkup** ( QPainter &p )
- void **moveTo** (const QPoint pos)
- bool **isOver** (const QPoint point, QCursor \*cursor)
- QPoint **origin** ()
- QCursor **cursorForHandle** (const markupItem::markupHandles handle)
- QPoint **getPoint1** ()
- QPoint **getPoint2** ()
- QCursor **defaultCursor** ()
- void **nonInteractiveUpdate** (QPoint p1, QPoint p2)

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

- /tmp/epicsqt/trunk/framework/widgets/QElImage/markupLine.h
- /tmp/epicsqt/trunk/framework/widgets/QElImage/markupLine.cpp

## 9.37 markupRegion Class Reference

Inheritance diagram for markupRegion:



### Public Member Functions

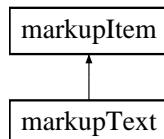
- **markupRegion** (*imageMarkup* \*ownerIn, const bool interactiveIn, const bool reportOnMoveIn, const QString legendIn)
- void **startDrawing** (const QPoint pos)
- void **setArea** ()
- void **drawMarkup** ( QPainter &p )
- void **moveTo** (const QPoint pos)
- bool **isOver** (const QPoint point, QCursor \*cursor)
- QPoint **origin** ()
- QCursors **cursorForHandle** (const markupItem::markupHandles handle)
- QPoint **getPoint1** ()
- QPoint **getPoint2** ()
- QCursors **defaultCursor** ()
- void **nonInteractiveUpdate** (QPoint p1, QPoint p2)

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

- /tmp/epicsqt/trunk/framework/widgets/QElImage/markupRegion.h
- /tmp/epicsqt/trunk/framework/widgets/QElImage/markupRegion.cpp

## 9.38 markupText Class Reference

Inheritance diagram for markupText:



### Public Member Functions

- **markupText** (*imageMarkup* \*ownerIn, const bool interactiveIn, const bool reportOnMoveIn, const QString legendIn)
- void **setText** (QString textIn)
- void **startDrawing** (const QPoint pos)
- void **setArea** ()
- void **drawMarkup** ( QPainter &p )
- void **moveTo** (const QPoint pos)
- bool **isOver** (const QPoint point, QCursors \*cursor)
- QPoint **origin** ()
- QCursors **cursorForHandle** (const markupItem::markupHandles handle)
- QPoint **getPoint1** ()

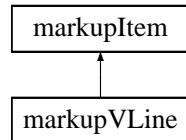
- QPoint **getPoint2 ()**
- QCursor **defaultCursor ()**

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

- /tmp/epicsqt/trunk/framework/widgets/QElImage/markupText.h
- /tmp/epicsqt/trunk/framework/widgets/QElImage/markupText.cpp

## 9.39 markupVLine Class Reference

Inheritance diagram for markupVLine:



### Public Member Functions

- **markupVLine** (*imageMarkup \*ownerIn*, const bool *interactiveIn*, const bool *reportOnMoveIn*, const QString *legendIn*)
- void **startDrawing** (const QPoint *pos*)
- void **setArea** ()
- void **drawMarkup** (QPainter &*p*)
- void **moveTo** (const QPoint *pos*)
- bool **isOver** (const QPoint *point*, QCcursor \**cursor*)
- QPoint **origin** ()
- QCcursor **cursorForHandle** (const markupItem::markupHandles *handle*)
- QPoint **getPoint1 ()**
- QPoint **getPoint2 ()**
- QCcursor **defaultCursor ()**
- void **nonInteractiveUpdate** (QPoint *p1*, QPoint *p2*)

### 9.39.1 Member Function Documentation

#### 9.39.1.1 void markupVLine::drawMarkup ( QPainter & *p* ) [virtual]

!! draw the handle in the middle of the existing view, not the entire image

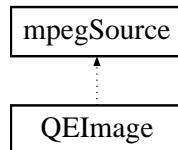
Implements [markupItem](#).

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

- /tmp/epicsqt/trunk/framework/widgets/QElImage/markupVLine.h
- /tmp/epicsqt/trunk/framework/widgets/QElImage/markupVLine.cpp

## 9.40 mpegSource Class Reference

Inheritance diagram for mpegSource:



### Public Member Functions

- void [updateImage \(FFBuffer \\*buf\)](#)
- void [setURL \(QString\)](#)
- void [startStream \(\)](#)
- void [stopStream \(\)](#)

### Protected Member Functions

- QString [getURL \(\)](#)
- void [setURL \(QString urlIn\)](#)
- void [stopStream \(\)](#)
- void [startStream \(\)](#)

#### 9.40.1 Member Function Documentation

##### 9.40.1.1 void mpegSource::updateImage ( FFBuffer \* buf )

!???: \* 3 for color only

!! Since the [QEImage](#) widget handles (or should handle) CA image data in all the formats that are expected in this mpeg stream !! perhaps this formatting here should be simply packaging the data in a QByteArray and delivering it, rather than perform any conversion.

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

- /tmp/epicsqt/trunk/framework/widgets/QEImage/mpeg.h
- /tmp/epicsqt/trunk/framework/widgets/QEImage/QEImage.h
- /tmp/epicsqt/trunk/framework/widgets/QEImage/mpeg.cpp

## 9.41 mpegSourceObject Class Reference

### Public Slots

- void [updateImage \(FFBuffer \\*buf\)](#)

### Signals

- void **aboutToQuit** ()

### Public Member Functions

- **mpegSourceObject** ([mpegSource](#) \*msIn)
- void **sentAboutToQuit** ()

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

- /tmp/epicsqt/trunk/framework/widgets/QElImage/mpeg.h
- /tmp/epicsqt/trunk/framework/widgets/QElImage/mpeg.cpp

## 9.42 QEStripChartToolBar::OwnWidgets Class Reference

### Public Member Functions

- **OwnWidgets** ([QEStripChartToolBar](#) \*parent)

### Public Attributes

- QPushButtons \* **pushButtons** [NUMBER\_OF\_BUTTONS]
- QLabel \* **yScaleStatus**
- QLabel \* **timeStatus**
- QLabel \* **durationStatus**

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

- /tmp/epicsqt/trunk/framework/widgets/QEStripChart/QEStripChartToolBar.cpp

## 9.43 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:

- /tmp/epicsqt/trunk/framework/widgets/QEPeriodic/PeriodicDialog.h
- /tmp/epicsqt/trunk/framework/widgets/QEPeriodic/PeriodicDialog.cpp

## 9.44 PeriodicElementSetupForm Class Reference

### Public Member Functions

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

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

- /tmp/epicsqt/trunk/framework/widgets/QEPeriodic/PeriodicElementSetupForm.h
- /tmp/epicsqt/trunk/framework/widgets/QEPeriodic/PeriodicElementSetupForm.cpp

## 9.45 PeriodicSetupDialog Class Reference

### Public Member Functions

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

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

- /tmp/epicsqt/trunk/framework/widgets/QEPeriodic/PeriodicSetupDialog.h
- /tmp/epicsqt/trunk/framework/widgets/QEPeriodic/PeriodicSetupDialog.cpp

## 9.46 playbackTimer Class Reference

### Public Member Functions

- **playbackTimer** (**recording** \*recorderIn)
- void **timerEvent** (QTimerEvent \*event)

### Public Attributes

- `recording` \* **recorder**

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

- /tmp/epicsqt/trunk/framework/widgets/QElImage/recording.h
- /tmp/epicsqt/trunk/framework/widgets/QElImage/recording.cpp

## 9.47 pointInfo Class Reference

### Public Member Functions

- void **setX** (long x)
- void **setY** (long y)
- void **setPoint** (QPoint pln)
- void **clearX** ()
- void **clearY** ()
- bool **getStatus** ()
- QPoint **getPoint** ()

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

- /tmp/epicsqt/trunk/framework/widgets/QElImage/QElImage.h

## 9.48 profilePlot Class Reference

### Public Types

- enum **plotDirections** { PROFILEPLOT\_LR, PROFILEPLOT\_RL, PROFILEPLOT\_TB, PROFILEPLOT\_BT }

### Public Member Functions

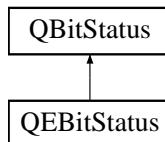
- **profilePlot** (plotDirections plotDirectionIn)
- void **setProfile** ( QVector< QPointF > \*profile, double minX, double maxX, double minY, double maxY, QString title, QPoint start, QPoint end, unsigned int thicknessIn)
- void **clearProfile** ()

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

- /tmp/epicsqt/trunk/framework/widgets/QElImage/profilePlot.h
- /tmp/epicsqt/trunk/framework/widgets/QElImage/profilePlot.cpp

## 9.49 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** ()

### Protected Member Functions

- void **setisActive** (const bool value)
- bool **getisActive** ()

### 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**
- bool **isActive**

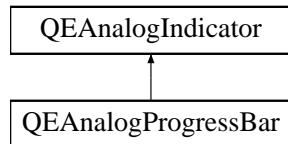
The documentation for this class was generated from the following files:

- /tmp/epicsqt/trunk/framework/widgets/QEBitStatus/QBitStatus.h
- /tmp/epicsqt/trunk/framework/widgets/QEBitStatus/QBitStatus.cpp

## 9.50 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](#) () const  
*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](#) () const  
*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 () const`  
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 () const`  
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 () const`  
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 () const`  
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 () const`  
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 () const`  
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 () const`  
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 () const`  
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 () const`  
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 () const`  
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 () const`  
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** () const  
*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** () const  
*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** () const  
*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** () const  
*Access function for `logScale` - refer to `logScale` property for details.*

## Protected Member Functions

- virtual QString **getTextImage** ()
- virtual BandList **getBandList** ()
- void **setIsActive** (const bool value)
- bool **getIsActive** () const

## 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`
- bool `isActive`

*Alternative to `isEnabled`. Default is true.*

### 9.50.1 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.

### 9.50.2 Member Enumeration Documentation

#### 9.50.2.1 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)

#### 9.50.2.2 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.

### 9.50.3 Property Documentation

#### 9.50.3.1 QColor QEAnalogIndicator::backgroundColour [read, write]

Background colour

#### 9.50.3.2 QColor QEAnalogIndicator::borderColour [read, write]

Border colour

#### 9.50.3.3 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.

9.50.3.4 **QColor QEAnalogIndicator::fontColour** [read, write]

Font colour

9.50.3.5 **QColor QEAnalogIndicator::foregroundColour** [read, write]

Foreground colour

9.50.3.6 **bool QEAnalogIndicator::logScale** [read, write]

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

9.50.3.7 **int QEAnalogIndicator::logScaleInterval** [read, write]

Log scale interval.

9.50.3.8 **double QEAnalogIndicator::majorInterval** [read, write]

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

9.50.3.9 **double QEAnalogIndicator::maximum** [read, write]

Maximum indicated value.

9.50.3.10 **double QEAnalogIndicator::minimum** [read, write]

Minimum indicated value.

9.50.3.11 **double QEAnalogIndicator::minorInterval** [read, write]

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

9.50.3.12 **Modes QEAnalogIndicator::mode** [read, write]

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

9.50.3.13 **Orientations QEAnalogIndicator::orientation** [read, write]

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

---

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

If set, show the scale

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

If set, show textual representation of value on the indicator

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

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

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

Current indicated value.

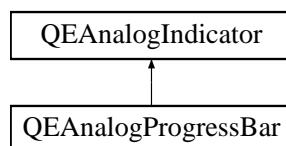
Reimplemented in [QEAnalogProgressBar](#).

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

- /tmp/epicsqt/trunk/framework/widgets/QEAnalogIndicator/QEAnalogIndicator.h
- /tmp/epicsqt/trunk/framework/widgets/QEAnalogIndicator/QEAnalogIndicator.cpp

## 9.51 QEAnalogProgressBar Class Reference

Inheritance diagram for QEAnalogProgressBar:



### Public Types

- enum `UserLevels { User = userLevelTypes::USERLEVEL_USER, Scientist = userLevelTypes::USERLEVEL_SCIENTIST, Engineer = userLevelTypes::USERLEVEL_ENGINEER }`
- enum `DisplayAlarmStateOptions { Never = standardProperties::DISPLAY_ALARM_STATE_NEVER, Always = standardProperties::DISPLAY_ALARM_STATE_ALWAYS, WhenInAlarm = standardProperties::DISPLAY_ALARM_STATE_WHEN_IN_ALARM }`
- enum `AlarmSeverityDisplayModes { foreground, background }`

- 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 `Separators` { `NoSeparator` = `QQStringFormatting::SEPARATOR_NONE`, `Comma` = `QQStringFormatting::SEPARATOR_COMMA`, `Underscore` = `QQStringFormatting::SEPARATOR_UNDERSCORE`, `Space` = `QQStringFormatting::SEPARATOR_SPACE` }
- 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` }

## Public Slots

- void `setManagedVisible` (bool v)

## Signals

- void `dbValueChanged` (const `QString` &out)
- void `dbValueChanged` (const `int` &out)
- void `dbValueChanged` (const `long` &out)
- void `dbValueChanged` (const `qlonglong` &out)
- void `dbValueChanged` (const `double` &out)
- void `dbValueChanged` (const `bool` &out)
- void `dbConnectionChanged` (const `bool` &isConnected)
- 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

- void `setVariableNameProperty` (`QString` variableName)
   
*Property access function for `variable` property. This has special behaviour to work well within designer.*
- `QString getVariableNameProperty` ()
   
*Property access function for `variable` property. This has special behaviour to work well within designer.*
- void `setVariableNameSubstitutionsProperty` (`QString` variableNameSubstitutions)
   
*Property access function for `variableSubstitutions` property. This has special behaviour to work well within designer.*
- `QString getVariableNameSubstitutionsProperty` ()

*Property access function for `variableSubstitutions` property. This has special behaviour to work well within designer.*

- **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.*
- **DisplayAlarmStateOptions getDisplayAlarmStateOptionProperty ()**  
*Access function for `displayAlarmStateOption` property - refer to `displayAlarmStateOption` property for details.*
- **void setDisplayAlarmStateOptionProperty (DisplayAlarmStateOptions option)**  
*Access function for `displayAlarmStateOption` property - refer to `displayAlarmStateOption` 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 setSeparatorProperty (const Separators notation)**  
*Access function for `separator` property - refer to `separator` property for details.*
- **Separators getSeparatorProperty () const**  
*Access function for `separator` property - refer to `separator` 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 setArrayIndex (const unsigned int arrayIndex)**
- **unsigned int getArrayIndex () const**
- **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 `setAlarmSeverityDisplayStyle (AlarmSeverityDisplayModes value)`  
*Access function for `#AlarmSeverityDisplayStyle` property - refer to `#AlarmSeverityDisplayStyle` property for details.*
- AlarmSeverityDisplayModes `getAlarmSeverityDisplayStyle ()`  
*Access function for `#AlarmSeverityDisplayStyle` property - refer to `#AlarmSeverityDisplayStyle` property for details.*

### Protected Member Functions

- 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 ()`
- QString `getTextImage ()`
- BandList `getBandList ()`

### Properties

- QString `variable`
- QString `variableSubstitutions`
- bool `variableAsToolTip`
- bool `allowDrop`
- bool `visible`
- unsigned `int`
- QString `styleSheet`
- QString `defaultStyle`
- QString `userLevelUserStyle`
- QString `userLevelScientistStyle`
- QString `userLevelEngineerStyle`
- UserLevels `userLevelVisibility`
- UserLevels `userLevelEnabled`
- bool `displayAlarmState`
- DisplayAlarmStateOptions `displayAlarmStateOption`
- AlarmSeverityDisplayModes `alarmSeverityDisplayStyle`
- bool `useDbDisplayLimits`
- int `value`
- bool `isActive`

*Alternative to isEnabled. Default is true.*

- int [precision](#)
- bool [useDbPrecision](#)
- bool [leadingZero](#)
- bool [trailingZeros](#)
- bool [addUnits](#)
- QString [localEnumeration](#)
- [Formats format](#)
- int [radix](#)
- [Separators separator](#)
- [Notations notation](#)
- [ArrayActions arrayAction](#)

### 9.51.1 Member Enumeration Documentation

#### 9.51.1.1 enum QEAnalogProgressBar::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.

#### 9.51.1.2 enum QEAnalogProgressBar::DisplayAlarmStateOptions

User friendly enumerations for [displayAlarmStateOption](#) property - refer to [displayAlarm-StateOption](#) property and [displayAlarmStateOptions](#) enumeration for details.

**Enumerator:**

**Never** Refer to [DISPLAY\\_ALARM\\_STATE\\_NEVER](#) for details.

**Always** Refer to [DISPLAY\\_ALARM\\_STATE\\_ALWAYS](#) for details.

**WhenInAlarm** Refer to [DISPLAY\\_ALARM\\_STATE\\_WHEN\\_IN\\_ALARM](#) for details.

#### 9.51.1.3 enum QEAnalogProgressBar::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.

#### 9.51.1.4 enum QEAnalogProgressBar::Notations

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

**Enumerator:**

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

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

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

#### 9.51.1.5 enum QEAnalogProgressBar::Separators

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

**Enumerator:**

***NoSeparator*** Use no separator.

***Comma*** Use ',' as separator.

***Underscore*** Use '\_' as separator.

***Space*** Use ' ' as separator.

#### 9.51.1.6 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.

### 9.51.2 Constructor & Destructor Documentation

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

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

9.51.2.2 `QEAnalogProgressBar::QEAnalogProgressBar ( 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.

### 9.51.3 Member Function Documentation

9.51.3.1 `void QEAnalogProgressBar::dbConnectionChanged ( const bool & isConnected ) [signal]`

Sent when the widget state updated following a channel connection change Applied to provary varible.

9.51.3.2 `void QEAnalogProgressBar::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.

9.51.3.3 `void QEAnalogProgressBar::setManagedVisible ( bool v ) [inline, slot]`

Slot to set the visibility of a QE widget, taking into account the user level. Widget will be hidden if hidden by a call this slot, by will only be made visible by a calll to this slot if the user level allows.

### 9.51.4 Property Documentation

9.51.4.1 `bool QEAnalogProgressBar::addUnits [read, write]`

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

9.51.4.2 `AlarmSeverityDisplayModes QEAnalogProgressBar::alarmSeverityDisplayStyle [read, write]`

Visualise the EPICS alarm severity

9.51.4.3 `bool QEAnalogProgressBar::allowDrop [read, write]`

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

**9.51.4.4 ArrayActions QEAnalogProgressBar::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'.

**9.51.4.5 QString QEAnalogProgressBar::defaultStyle [read, write]**

Style Sheet string to be applied before, i.e. lower priority than, any other style, e.g. alarm style and/or user level style. Default is an empty string.

**9.51.4.6 bool QEAnalogProgressBar::displayAlarmState [read, write]**

DEPRECATED. USE displayAlarmStateOption INSTEAD. If set (default) widget will indicate the alarm state of any variable data it is displaying. If clear widget will never indicate the alarm state of any variable data it is displaying. Typically the background colour is set to indicate the alarm state. Note, this property is included in the set of standard properties as it applies to most widgets. It will do nothing for widgets that don't display data.

**9.51.4.7 DisplayAlarmStateOptions QEAnalogProgressBar::displayAlarmStateOption [read, write]**

If 'Always' (default) widget will indicate the alarm state of any variable data it is displaying, including 'No Alarm'. If 'Never' widget will never indicate the alarm state of any variable data it is displaying. If 'WhenInAlarm' widget only indicate the alarm state of any variable data it is displaying if it is 'in alarm'. Typically the background colour is set to indicate the alarm state. Note, this property is included in the set of standard properties as it applies to most widgets. It will do nothing for widgets that don't display data.

**9.51.4.8 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).

---

**9.51.4.9 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.

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.

**9.51.4.10 bool QEAnalogProgressBar::leadingZero [read, write]**

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

**9.51.4.11 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)  
>= Greater 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"`

**9.51.4.12 Notations QEAnalogProgressBar::notation [read, write]**

Notation used for numerical formatting. Default is fixed.

**9.51.4.13 int QEAnalogProgressBar::precision [read, write]**

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

**9.51.4.14 int QEAnalogProgressBar::radix [read, write]**

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

**9.51.4.15 Separators QEAnalogProgressBar::separator [read, write]**

Separators used for integer and fixed point formatting. Default is None.

**9.51.4.16 QString QEAnalogProgressBar::styleSheet [read, write]**

Hide style sheet from designer as style calculation by the styleManager and not directly setable per se. This also stops transient styles being saved to the ui file.

**9.51.4.17 bool QEAnalogProgressBar::trailingZeros [read, write]**

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

**9.51.4.18 bool QEAnalogProgressBar::useDbDisplayLimits [read, write]**

Use the EPICS database display limits

**9.51.4.19 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.

**9.51.4.20 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 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'.

**9.51.4.21 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.

**9.51.4.22 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.

**9.51.4.23 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.

**9.51.4.24 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'.

**9.51.4.25 int QEAnalogProgressBar::value [read, write]**

Current indicated value.

Reimplemented from [QEAnalogIndicator](#).

**9.51.4.26 QString QEAnalogProgressBar::variable [read, write]**

EPICS variable name (CA PV)

#### 9.51.4.27 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.

#### 9.51.4.28 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 also used for other purposes.

#### 9.51.4.29 bool QEAnalogProgressBar::visible [read, write]

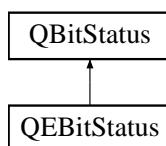
Display the widget. Default is true. Setting this property false is useful if widget is only used to provide a signal - for example, when supplying data to a [QELink](#) widget. Note, when false the widget will still be visible in Qt Designer.

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

- /tmp/epicsqt/trunk/framework/widgets/QEAnalogProgressBar/QEAnalogProgressBar.h
- /tmp/epicsqt/trunk/framework/widgets/QEAnalogProgressBar/QEAnalogProgressBar.cpp

## 9.52 QEBitStatus Class Reference

Inheritance diagram for QEBitStatus:



### Public Types

- enum [UserLevels](#) { [User](#) = userLevelTypes::USERLEVEL\_USER, [Scientist](#) = userLevelTypes::USERLEVEL\_SCIENTIST, [Engineer](#) = userLevelTypes::USERLEVEL\_ENGINEER }
- enum [DisplayAlarmStateOptions](#) { [Never](#) = standardProperties::DISPLAY\_ALARM\_STATE\_NEVER, [Always](#) = standardProperties::DISPLAY\_ALARM\_STATE\_ALWAYS, [WhenInAlarm](#) = standardProperties::DISPLAY\_ALARM\_STATE\_WHEN\_IN\_ALARM }

### Public Slots

- void [setManagedVisible](#) (bool v)

## Signals

- void **dbValueChanged** (const QString &out)
- void **dbValueChanged** (const int &out)
- void **dbValueChanged** (const long &out)
- void **dbValueChanged** (const qlonglong &out)
- void **dbValueChanged** (const double &out)
- void **dbValueChanged** (const bool &out)
- void **dbConnectionChanged** (const bool &isConnected)

## Public Member Functions

- void **setVariableNameProperty** (QString variableName)  
*Property access function for `variable` property. This has special behaviour to work well within designer.*
- QString **getVariableNameProperty** ()  
*Property access function for `variable` property. This has special behaviour to work well within designer.*
- void **setVariableNameSubstitutionsProperty** (QString variableNameSubstitutions)  
*Property access function for `variableSubstitutions` property. This has special behaviour to work well within designer.*
- QString **getVariableNameSubstitutionsProperty** ()  
*Property access function for `variableSubstitutions` property. This has special behaviour to work well within designer.*
- 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.*
- DisplayAlarmStateOptions **getDisplayAlarmStateOptionProperty** ()  
*Access function for `displayAlarmStateOption` property - refer to `displayAlarmStateOption` property for details.*
- void **setDisplayAlarmStateOptionProperty** (DisplayAlarmStateOptions option)  
*Access function for `displayAlarmStateOption` property - refer to `displayAlarmStateOption` property for details.*
- **QEBitStatus** (QWidget \*parent=0)
- **QEBitStatus** (const QString &variableName, QWidget \*parent=0)
- void **setArrayIndex** (const int arrayIndex)
- int **getArrayIndex** () const

## Protected Member Functions

- qcaobject::QCaObject \* **createQcalItem** (unsigned int variableIndex)
- void **establishConnection** (unsigned int variableIndex)
- void **dragEnterEvent** (QDragEnterEvent \*event)
- void **dropEvent** (QDropEvent \*event)
- void **mousePressEvent** (QMouseEvent \*event)
- QString **copyVariable** ()
- QVariant **copyData** ()

## Properties

- QString **variable**
- QString **variableSubstitutions**
- bool **variableAsToolTip**
- bool **allowDrop**
- bool **visible**
- unsigned **int**
- QString **styleSheet**
- QString **defaultStyle**
- QString **userLevelUserStyle**
- QString **userLevelScientistStyle**
- QString **userLevelEngineerStyle**
- UserLevels **userLevelVisibility**
- UserLevels **userLevelEnabled**
- bool **displayAlarmState**
- DisplayAlarmStateOptions **displayAlarmStateOption**
- int **arrayIndex**
- double **value**
- bool **isActive**
- bool **isValid**

### 9.52.1 Member Enumeration Documentation

#### 9.52.1.1 enum QEBitStatus::DisplayAlarmStateOptions

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

##### Enumerator:

**Never** Refer to DISPLAY\_ALARM\_STATE\_NEVER for details.

**Always** Refer to DISPLAY\_ALARM\_STATE\_ALWAYS for details.

**WhenInAlarm** Refer to DISPLAY\_ALARM\_STATE\_WHEN\_IN\_ALARM for details.

### 9.52.1.2 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.

### 9.52.2 Member Function Documentation

#### 9.52.2.1 void QEBitStatus::dbConnectionChanged ( const bool & isConnected ) [signal]

Sent when the widget state updated following a channel connection change Applied to provary varible.

#### 9.52.2.2 void QEBitStatus::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.

#### 9.52.2.3 void QEBitStatus::setManagedVisible ( bool v ) [inline, slot]

Slot to set the visibility of a QE widget, taking into account the user level. Widget will be hidden if hidden by a call this slot, by will only be made visible by a call to this slot if the user level allows.

### 9.52.3 Property Documentation

#### 9.52.3.1 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.

#### 9.52.3.2 int QEBitStatus::arrayIndex [read, write]

Array Index element to display if main (non-edge) variable is a waveform. Defaults to 0.

**9.52.3.3 QString QEBitStatus::defaultStyle [read, write]**

Style Sheet string to be applied before, i.e. lower priority than, any other style, e.g. alarm style and/or user level style. Default is an empty string.

**9.52.3.4 bool QEBitStatus::displayAlarmState [read, write]**

DEPRECATED. USE displayAlarmStateOption INSTEAD. If set (default) widget will indicate the alarm state of any variable data it is displaying. If clear widget will never indicate the alarm state of any variable data it is displaying. Typically the background colour is set to indicate the alarm state. Note, this property is included in the set of standard properties as it applies to most widgets. It will do nothing for widgets that don't display data.

**9.52.3.5 DisplayAlarmStateOptions QEBitStatus::displayAlarmStateOption [read, write]**

If 'Always' (default) widget will indicate the alarm state of any variable data it is displaying, including 'No Alarm'. If 'Never' widget will never indicate the alarm state of any variable data it is displaying. If 'WhenInAlarm' widget only indicate the alarm state of any variable data it is displaying if it is 'in alarm'. Typically the background colour is set to indicate the alarm state. Note, this property is included in the set of standard properties as it applies to most widgets. It will do nothing for widgets that don't display data.

**9.52.3.6 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.

**9.52.3.7 QString QEBitStatus::styleSheet [read, write]**

Hide style sheet from designer as style calculation by the styleManager and not directly setable per se. This also stops transient styles being saved to the ui file.

**9.52.3.8 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 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'.

**9.52.3.9 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.

**9.52.3.10 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.

**9.52.3.11 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.

**9.52.3.12 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 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'.

**9.52.3.13 QString QEBitStatus::variable [read, write]**

EPICS variable name (CA PV)

**9.52.3.14 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.

### 9.52.3.15 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 also used for other purposes.

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

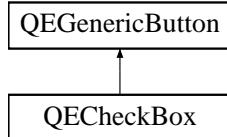
Display the widget. Default is true. Setting this property false is useful if widget is only used to provide a signal - for example, when supplying data to a [QELink](#) widget. Note, when false the widget will still be visible in Qt Designer.

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

- /tmp/epicsqt/trunk/framework/widgets/QEBitStatus/QEBitStatus.h
- /tmp/epicsqt/trunk/framework/widgets/QEBitStatus/QEBitStatus.cpp

## 9.53 QECheckBox Class Reference

Inheritance diagram for QECheckBox:



### Public Types

- enum [UserLevels](#) { [User](#) = userLevelTypes::USERLEVEL\_USER, [Scientist](#) = userLevelTypes::USERLEVEL\_SCIENTIST, [Engineer](#) = userLevelTypes::USERLEVEL\_ENGINEER }
- enum [DisplayAlarmStateOptions](#) { [Never](#) = standardProperties::DISPLAY\_ALARM\_STATE\_NEVER, [Always](#) = standardProperties::DISPLAY\_ALARM\_STATE\_ALWAYS, [WhenInAlarm](#) = standardProperties::DISPLAY\_ALARM\_STATE\_WHEN\_IN\_ALARM }
- 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 [Separators](#) { [NoSeparator](#) = QEStringFormatting::SEPARATOR\_NONE, [Comma](#) = QEStringFormatting::SEPARATOR\_COMMA, [Underscore](#) = QEStringFormatting::SEPARATOR\_UNDERSCORE, [Space](#) = QEStringFormatting::SEPARATOR\_SPACE }

- 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` = 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 `ProgramStartupOptionNames` { `None` = applicationLauncher::PSO\_NONE, `Terminal` = applicationLauncher::PSO\_TERMINAL, `LogOutput` = applicationLauncher::PSO\_LOGOUTPUT, `StdOutput` = applicationLauncher::PSO\_STDOUTPUT }
- enum `CreationOptionNames` {
   
`Open` = QEActionRequests::OptionOpen, `NewTab` = QEActionRequests::OptionNewTab,
   
`NewWindow` = QEActionRequests::OptionNewWindow, `DockTop` = QEActionRequests::OptionTopDockWindow,
   
`DockBottom` = QEActionRequests::OptionBottomDockWindow, `DockLeft` = QEActionRequests::OptionLeftDockWindow, `DockRight` = QEActionRequests::OptionRightDockWindow,
   
`DockTopTabbed` = QEActionRequests::OptionTopDockWindowTabbed,
   
`DockBottomTabbed` = QEActionRequests::OptionBottomDockWindowTabbed, `DockLeftTabbed` = QEActionRequests::OptionLeftDockWindowTabbed, `DockRightTabbed` = QEActionRequests::OptionRightDockWindowTabbed, `DockFloating` = QEActionRequests::OptionFloatingDockWindow
 }

*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 `requestAction` (const QEActionRequests &request)
- void `setDefaultStyle` (const QString &style)
   
*Update the default style applied to this widget.*
- void `setManagedVisible` (bool v)

## Signals

- void `dbValueChanged` (const QString &out)
   
Internal use only. Used when changing a property value to force a re-display to reflect the new property value.
- void `requestResend` ()
   
Internal use only. Used when changing a property value to force a re-display to reflect the new property value.
- void `newGui` (const QEActionRequests &request)
   
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)
- void `programCompleted` ()

*Program started by button has completed.*

## Public Member Functions

- `QECheckBox` (QWidget \*parent=0)
- `QECheckBox` (const QString &variableName, QWidget \*parent=0)
- void `writeNow` ()
- void `setVariableNameProperty` (QString variableName)  
*Property access function for `variable` property. This has special behaviour to work well within designer.*
- QString `getVariableNameProperty` ()  
*Property access function for `variable` property. This has special behaviour to work well within designer.*
- void `setVariableNameSubstitutionsProperty` (QString variableNameSubstitutions)  
*Property access function for `variableSubstitutions` property. This has special behaviour to work well within designer.*
- QString `getVariableNameSubstitutionsProperty` ()  
*Property access function for `variableSubstitutions` property. This has special behaviour to work well within designer.*
- `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.*
- `DisplayAlarmStateOptions getDisplayAlarmStateOptionProperty` ()  
*Access function for `displayAlarmStateOption` property - refer to `displayAlarmStateOption` property for details.*
- void `setDisplayAlarmStateOptionProperty` (`DisplayAlarmStateOptions` option)  
*Access function for `displayAlarmStateOption` property - refer to `displayAlarmStateOption` 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 `setSeparatorProperty` (const `Separators` notation)

*Access function for `separator` property - refer to `separator` property for details.*

- `Separators getSeparatorProperty () const`

*Access function for `separator` property - refer to `separator` 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 allowDrop`
- `bool visible`
- `unsigned int`
- `QString styleSheet`
- `QString defaultStyle`
- `QString userLevelUserStyle`
- `QString userLevelScientistStyle`
- `QString userLevelEngineerStyle`
- `UserLevels userLevelVisibility`
- `UserLevels userLevelEnabled`
- `bool displayAlarmState`
- `DisplayAlarmStateOptions displayAlarmStateOption`
- `int precision`
- `bool useDbPrecision`
- `bool leadingZero`
- `bool trailingZeros`
- `bool addUnits`
- `QString localEnumeration`
- `Formats format`
- `int radix`
- `Separators separator`
- `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](#)
- QString  [confirmText](#)
- bool  [writeOnPress](#)
- bool  [writeOnRelease](#)
- bool  [writeOnClick](#)
- QString  [pressText](#)
- QString  [releaseText](#)
- QString  [clickText](#)
- QString  [clickCheckedText](#)
- QString  [labelText](#)
- QString  [program](#)
- QStringList  [arguments](#)
- ProgramStartupOptionNames  [programStartupOption](#)
- QString  [guiFile](#)
- CreationOptionNames  [creationOption](#)
- QString  [prioritySubstitutions](#)
- QString  [customisationName](#)

### 9.53.1 Member Enumeration Documentation

#### 9.53.1.1 enum QECheckBox::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.

#### 9.53.1.2 enum QECheckBox::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.

**DockTop** Open new GUI in a top dock window.

**DockBottom** Open new GUI in a bottom dock window.

**DockLeft** Open new GUI in a left dock window.

**DockRight** Open new GUI in a right dock window.

**DockTopTabbed** Open new GUI in a top dock window (tabbed with any existing dock in that area)

**DockBottomTabbed** Open new GUI in a bottom dock window (tabbed with any existing dock in that area)

**DockLeftTabbed** Open new GUI in a left dock window (tabbed with any existing dock in that area)

**DockRightTabbed** Open new GUI in a right dock window (tabbed with any existing dock in that area)

**DockFloating** Open new GUI in a floating dock window.

#### 9.53.1.3 enum QECheckBox::DisplayAlarmStateOptions

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

##### Enumerator:

**Never** Refer to DISPLAY\_ALARM\_STATE\_NEVER for details.

**Always** Refer to DISPLAY\_ALARM\_STATE\_ALWAYS for details.

**WhenInAlarm** Refer to DISPLAY\_ALARM\_STATE\_WHEN\_IN\_ALARM for details.

#### 9.53.1.4 enum QECheckBox::Formats

User friendly enumerations for format property - refer to [QEStringFormatting::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.

### 9.53.1.5 enum QECheckBox::Notations

User friendly enumerations for notation property - refer to QEStringFormatting::notations for details.

#### Enumerator:

**Fixed** Refer to QEStringFormatting::NOTATION\_FIXED for details.

**Scientific** Refer to QEStringFormatting::NOTATION\_SCIENTIFIC for details.

**Automatic** Refer to QEStringFormatting::NOTATION\_AUTOMATIC for details.

### 9.53.1.6 enum QECheckBox::ProgramStartupOptionNames

Startup options. Just run the command, run the command within a terminal, or display the output in QE message system.

#### Enumerator:

**None** Just run the program.

**Terminal** Run the program in a terminal (in Windows a command interpreter will also be started, so the program may be a built-in command like 'dir')

**LogOutput** Run the program, and log the output in the QE message system.

**StdOutput** Run the program, and send output to standard output and standard error.

### 9.53.1.7 enum QECheckBox::Separators

User friendly enumerations for separator property - refer to QEStringFormatting::formats for details.

#### Enumerator:

**NoSeparator** Use no separator.

**Comma** Use ',' as separator.

**Underscore** Use '\_' as separator.

**Space** Use ' ' as separator.

### 9.53.1.8 enum QECheckBox::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)

### 9.53.1.9 enum QECheckBox::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 Constructor & Destructor Documentation

#### 9.53.2.1 QECheckBox::QECheckBox ( QWidget \* *parent* = 0 )

Create without a variable. Use `setVariableNameProperty()` and `setSubstitutionsProperty()` to define a variable and, optionally, macro substitutions later.

#### 9.53.2.2 QECheckBox::QECheckBox ( 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.

### 9.53.3 Member Function Documentation

#### 9.53.3.1 void QECheckBox::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)

#### 9.53.3.2 void QECheckBox::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.

#### 9.53.3.3 void QECheckBox::pressed ( int *value* ) [signal]

Button has been Pressed. The value emitted is the integer interpretation of the pressText property

**9.53.3.4 void QECheckBox::released ( int *value* ) [signal]**

Button has been Released The value emitted is the integer interpretation of the release-Text property

**9.53.3.5 void QECheckBox::requestAction ( const QEActionRequests & *request* )  
[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.

**9.53.3.6 void QECheckBox::setManagedVisible ( bool *v* ) [inline, slot]**

Slot to set the visibility of a QE widget, taking into account the user level. Widget will be hidden if hidden by a call this slot, by will only be made visible by a call to this slot if the user level allows.

## 9.53.4 Property Documentation

**9.53.4.1 bool QECheckBox::addUnits [read, write]**

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

**9.53.4.2 Qt::Alignment QECheckBox::alignment [read, write]**

Set the buttons text alignment. Left justification is particularly useful when displaying quickly changing numeric data updates.

**9.53.4.3 bool QECheckBox::allowDrop [read, write]**

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

**9.53.4.4 QStringList QECheckBox::arguments [read, write]**

Arguments for program specified in the 'program' property.

**9.53.4.5 ArrayActions QECheckBox::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'.

**9.53.4.6 QString QECheckBox::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](#).

**9.53.4.7 QString QECheckBox::clickText [read, write]**

Value written when user clicks button if 'writeOnClick' property is true

Reimplemented from [QEGenericButton](#).

**9.53.4.8 bool QECheckBox::confirmAction [read, write]**

If true, a dialog will be presented asking the user to confirm if the button action should be carried out

**9.53.4.9 QString QECheckBox::confirmText [read, write]**

Text used to confirm action if confirmation dialog is presented

Reimplemented from [QEGenericButton](#).

**9.53.4.10 CreationOptionNames QECheckBox::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](#).

**9.53.4.11 QString QECheckBox::customisationName [read, write]**

Window customisation name. This name will be used to select a set of window customisations including menu items and tool bar buttons. Applications such as QEGui can load .xml files containing named sets of window customisations. This property is used to select a set loaded from these files. The selected set of customisations will be applied to the main window containing the new GUI.

Reimplemented from [QEGenericButton](#).

**9.53.4.12 QString QECheckBox::defaultStyle [read, write]**

Style Sheet string to be applied before, i.e. lower priority than, any other style, e.g. alarm style and/or user level style. Default is an empty string.

**9.53.4.13 bool QECheckBox::displayAlarmState [read, write]**

DEPRECATED. USE displayAlarmStateOption INSTEAD. If set (default) widget will indicate the alarm state of any variable data it is displaying. If clear widget will never indicate the alarm state of any variable data it is displaying. Typically the background colour is set to indicate the alarm state. Note, this property is included in the set of standard properties as it applies to most widgets. It will do nothing for widgets that don't display data.

**9.53.4.14 DisplayAlarmStateOptions QECheckBox::displayAlarmStateOption [read, write]**

If 'Always' (default) widget will indicate the alarm state of any variable data it is displaying, including 'No Alarm'. If 'Never' widget will never indicate the alarm state of any variable data it is displaying. If 'WhenInAlarm' widget only indicate the alarm state of any variable data it is displaying if it is 'in alarm'. Typically the background colour is set to indicate the alarm state. Note, this property is included in the set of standard properties as it applies to most widgets. It will do nothing for widgets that don't display data.

**9.53.4.15 Formats QECheckBox::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).

**9.53.4.16 QString QECheckBox::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.

**9.53.4.17 unsigned QECheckBox::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.

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.

**9.53.4.18 QString QECheckBox::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. IF NOT LEFT EMPTY, THIS TEXT WILL TAKE PRIORITY OVER THE PUSH BUTTON 'text' PROPERTY! 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](#).

**9.53.4.19 bool QECheckBox::leadingZero [read, write]**

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

**9.53.4.20 QString QECheckBox::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)  
>= Greater 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!", "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"

#### 9.53.4.21 Notations QECheckBox::notation [read, write]

Notation used for numerical formatting. Default is fixed.

#### 9.53.4.22 QString QECheckBox::password [read, write]

Password user will need to enter before any action is taken

Reimplemented from [QEGenericButton](#).

#### 9.53.4.23 QPixmap QECheckBox::pixmap0 [read, write]

Pixmap to display if updateOption is Icon or TextAndIcon and data value translates to an index of 0

#### 9.53.4.24 QPixmap QECheckBox::pixmap1 [read, write]

Pixmap to display if updateOption is Icon or TextAndIcon and data value translates to an index of 1

**9.53.4.25 QPixmap QECheckBox:: pixmap2 [read, write]**

Pixmap to display if updateOption is Icon or TextAndIcon and data value translates to an index of 2

**9.53.4.26 QPixmap QECheckBox:: pixmap3 [read, write]**

Pixmap to display if updateOption is Icon or TextAndIcon and data value translates to an index of 3

**9.53.4.27 QPixmap QECheckBox:: pixmap4 [read, write]**

Pixmap to display if updateOption is Icon or TextAndIcon and data value translates to an index of 4

**9.53.4.28 QPixmap QECheckBox:: pixmap5 [read, write]**

Pixmap to display if updateOption is Icon or TextAndIcon and data value translates to an index of 5

**9.53.4.29 QPixmap QECheckBox:: pixmap6 [read, write]**

Pixmap to display if updateOption is Icon or TextAndIcon and data value translates to an index of 6

**9.53.4.30 QPixmap QECheckBox:: pixmap7 [read, write]**

Pixmap to display if updateOption is Icon or TextAndIcon and data value translates to an index of 7

**9.53.4.31 int QECheckBox:: precision [read, write]**

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

**9.53.4.32 QString QECheckBox:: pressText [read, write]**

Value written when user presses button if 'writeOnPress' property is true

Reimplemented from [QEGenericButton](#).

**9.53.4.33 QString QECheckBox::prioritySubstitutions [read, write]**

Overriding macro substitutions. These macro substitions take precedence over any existing macro substitutions defined by the variableSubstitutions property, any parent forms, or the application containing the button. These macro substitutions are particularly usefull when the button's function is to reload the same form but with different macro substitutions. The variableSubstitutions property cannot be used for this since, although they are added to the list of macro substitions applied to the new form, they are appended to the list and the existing macro substitutions take precedence.

Reimplemented from [QEGenericButton](#).

**9.53.4.34 QString QECheckBox::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

**9.53.4.35 ProgramStartupOptionNames QECheckBox::programStartupOption [read, write]**

Startup options. Just run the command, run the command within a terminal, or display the output in QE message system.

**9.53.4.36 int QECheckBox::radix [read, write]**

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

**9.53.4.37 QString QECheckBox::releaseText [read, write]**

Value written when user releases button if 'writeOnRelease' property is true

Reimplemented from [QEGenericButton](#).

**9.53.4.38 Separators QECheckBox::separator [read, write]**

Separators used for interger and fixed point formatting. Default is None.

**9.53.4.39 QString QECheckBox::styleSheet [read, write]**

Hide style sheet from designer as style calculation by the styleManager and not directly setable per se. This also stops transient styles being saved to the ui file.

**9.53.4.40 bool QECheckBox::subscribe [read, write]**

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

**9.53.4.41 bool QECheckBox::trailingZeros [read, write]**

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

**9.53.4.42 UpdateOptions QECheckBox::updateOption [read, write]**

Update options (text, pixmap, both, or state (checked or unchecked)

Reimplemented from [QEGenericButton](#).

**9.53.4.43 bool QECheckBox::useDbPrecision [read, write]**

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

**9.53.4.44 UserLevels QECheckBox::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'.

**9.53.4.45 QString QECheckBox::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.

**9.53.4.46 QString QECheckBox::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.

#### 9.53.4.47 `QString QECheckBox::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.

#### 9.53.4.48 `UserLevels QECheckBox::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'.

#### 9.53.4.49 `QString QECheckBox::variable [read, write]`

EPICS variable name (CA PV)

#### 9.53.4.50 `bool QECheckBox::variableAsToolTip [read, write]`

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

#### 9.53.4.51 `QString QECheckBox::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.

#### 9.53.4.52 `bool QECheckBox::visible [read, write]`

Display the widget. Default is true. Setting this property false is useful 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.

**9.53.4.53 bool QECheckBox::writeOnClick [read, write]**

If true, the 'clickText' property is written when the button is clicked. Default is true

Reimplemented from [QEGenericButton](#).

**9.53.4.54 bool QECheckBox::writeOnPress [read, write]**

If true, the 'pressText' property is written when the button is pressed. Default is false

Reimplemented from [QEGenericButton](#).

**9.53.4.55 bool QECheckBox::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:

- /tmp/epicsqt/trunk/framework/widgets/QEButton/QECheckBox.h
- /tmp/epicsqt/trunk/framework/widgets/QEButton/QECheckBox.cpp

## 9.54 QECheckBoxManager Class Reference

### Public Member Functions

- **QECheckBoxManager** (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:

- /tmp/epicsqt/trunk/framework/widgets/QEButton/QECheckBoxManager.h
- /tmp/epicsqt/trunk/framework/widgets/QEButton/QECheckBoxManager.cpp

## 9.55 QEComboBox Class Reference

### Public Types

- enum `UserLevels` { `User` = userLevelTypes::USERLEVEL\_USER, `Scientist` = userLevelTypes::USERLEVEL\_SCIENTIST, `Engineer` = userLevelTypes::USERLEVEL\_ENGINEER }
- enum `DisplayAlarmStateOptions` { `Never` = standardProperties::DISPLAY\_ALARM\_STATE\_NEVER, `Always` = standardProperties::DISPLAY\_ALARM\_STATE\_ALWAYS, `WhenInAlarm` = standardProperties::DISPLAY\_ALARM\_STATE\_WHEN\_IN\_ALARM }

### Public Slots

- void `setDefaultStyle` (const QString &style)  
*Update the default style applied to this widget.*
- void `setManagedVisible` (bool v)

### Signals

- void `dbValueChanged` (const QString &out)
- void `dbValueChanged` (const int &out)
- void `dbValueChanged` (const long &out)
- void `dbValueChanged` (const qlonglong &out)
- void `dbValueChanged` (const double &out)
- void `dbValueChanged` (const bool &out)
- void `dbConnectionChanged` (const bool &isConnected)  
*Sent when the widget state updated following a channel connection change.*
- 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` ()
- void `setLocalEnumerations` (const QString &localEnumerations)
- QString `getLocalEnumerations` ()
- void `setVariableNameProperty` (QString variableName)

*Property access function for `variable` property. This has special behaviour to work well within designer.*

- `QString getVariableNameProperty ()`

*Property access function for `variable` property. This has special behaviour to work well within designer.*

- `void setVariableNameSubstitutionsProperty (QString variableNameSubstitutions)`

*Property access function for `variableSubstitutions` property. This has special behaviour to work well within designer.*

- `QString getVariableNameSubstitutionsProperty ()`

*Property access function for `variableSubstitutions` property. This has special behaviour to work well within designer.*

- `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.*

- `DisplayAlarmStateOptions getDisplayAlarmStateOptionProperty ()`

*Access function for `displayAlarmStateOption` property - refer to `displayAlarmStateOption` property for details.*

- `void setDisplayAlarmStateOptionProperty (DisplayAlarmStateOptions option)`

*Access function for `displayAlarmStateOption` property - refer to `displayAlarmStateOption` 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 ()`
- `QString copyVariable ()`
- `QVariant copyData ()`
- `void paste (QVariant s)`

### Protected Attributes

- QEIntegerFormatting **integerFormatting**
- QELocalEnumeration **localEnumerations**
- bool **useDbEnumerations**
- bool **writeOnChange**

### Properties

- QString **variable**
- QString **variableSubstitutions**
- bool **subscribe**
- bool **variableAsToolTip**
- bool **allowDrop**
- bool **visible**
- unsigned **int**
- QString **styleSheet**
- QString **defaultStyle**
- QString **userLevelUserStyle**
- QString **userLevelScientistStyle**
- QString **userLevelEngineerStyle**
- **UserLevels userLevelVisibility**
- **UserLevels userLevelEnabled**
- bool **displayAlarmState**
- **DisplayAlarmStateOptions displayAlarmStateOption**
- QString **localEnumeration**

#### 9.55.1 Member Enumeration Documentation

##### 9.55.1.1 enum QEComboBox::DisplayAlarmStateOptions

User friendly enumerations for **displayAlarmStateOption** property - refer to **displayAlarmStateOption** property and **displayAlarmStateOptions** enumeration for details.

###### Enumerator:

**Never** Refer to DISPLAY\_ALARM\_STATE\_NEVER for details.

**Always** Refer to DISPLAY\_ALARM\_STATE\_ALWAYS for details.

**WhenInAlarm** Refer to DISPLAY\_ALARM\_STATE\_WHEN\_IN\_ALARM for details.

### 9.55.1.2 enum QEComboBox::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.

### 9.55.2 Member Function Documentation

#### 9.55.2.1 void QEComboBox::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.

#### 9.55.2.2 void QEComboBox::setManagedVisible ( bool v ) [inline, slot]

Slot to set the visibility of a QE widget, taking into account the user level. Widget will be hidden if hidden by a call this slot, by will only be made visible by a call to this slot if the user level allows.

### 9.55.3 Member Data Documentation

#### 9.55.3.1 bool QEComboBox::useDbEnumerations [read, write, protected]

Use database enumerations - defaults to true

#### 9.55.3.2 bool QEComboBox::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).

### 9.55.4 Property Documentation

#### 9.55.4.1 bool QEComboBox::allowDrop [read, write]

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

**9.55.4.2 QString QEComboBox::defaultStyle [read, write]**

Style Sheet string to be applied before, i.e. lower priority than, any other style, e.g. alarm style and/or user level style. Default is an empty string.

**9.55.4.3 bool QEComboBox::displayAlarmState [read, write]**

DEPRECATED. USE displayAlarmStateOption INSTEAD. If set (default) widget will indicate the alarm state of any variable data it is displaying. If clear widget will never indicate the alarm state of any variable data it is displaying. Typically the background colour is set to indicate the alarm state. Note, this property is included in the set of standard properties as it applies to most widgets. It will do nothing for widgets that don't display data.

**9.55.4.4 DisplayAlarmStateOptions QEComboBox::displayAlarmStateOption [read, write]**

If 'Always' (default) widget will indicate the alarm state of any variable data it is displaying, including 'No Alarm' If 'Never' widget will never indicate the alarm state of any variable data it is displaying. If 'WhenInAlarm' widget only indicate the alarm state of any variable data it is displaying if it is 'in alarm'. Typically the background colour is set to indicate the alarm state. Note, this property is included in the set of standard properties as it applies to most widgets. It will do nothing for widgets that don't display data.

**9.55.4.5 unsigned QEComboBox::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.

**9.55.4.6 QString QEComboBox::localEnumeration [read, write]**

Enumerations values used when useDbEnumerations is false.

**9.55.4.7 QString QEComboBox::styleSheet [read, write]**

Hide style sheet from designer as style calculation by the styleManager and not directly setable per se. This also stops transient styles being saved to the ui file.

**9.55.4.8 bool QEComboBox::subscribe [read, write]**

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

**9.55.4.9 UserLevels QEComboBox::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'.

**9.55.4.10 QString QEComboBox::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.

**9.55.4.11 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.

**9.55.4.12 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.

**9.55.4.13 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 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'.

**9.55.4.14** `QString QEComboBox::variable [read, write]`

EPICS variable name (CA PV)

**9.55.4.15** `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.

**9.55.4.16** `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.

**9.55.4.17** `bool QEComboBox::visible [read, write]`

Display the widget. Default is true. Setting this property false is useful 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:

- /tmp/epicsqt/trunk/framework/widgets/QEComboBox/QEComboBox.h
- /tmp/epicsqt/trunk/framework/widgets/QEComboBox/QEComboBox.cpp

## 9.56 QEConfiguredLayout Class Reference

### Public Types

- enum **configurationTypesProperty** { **File** = FROM\_FILE, **Text** = FROM\_TEXT }
- enum **optionsLayoutProperty** { **Top** = TOP, **Bottom** = BOTTOM, **Left** = LEFT, **Right** = RIGHT }
- enum **UserLevels** { **User** = userLevelTypes::USERLEVEL\_USER, **Scientist** = userLevelTypes::USERLEVEL\_SCIENTIST, **Engineer** = userLevelTypes::USERLEVEL\_ENGINEER }
- enum **DisplayAlarmStateOptions** { **Never** = standardProperties::DISPLAY\_ALARM\_STATE\_NEVER, **Always** = standardProperties::DISPLAY\_ALARM\_STATE\_ALWAYS, **WhenInAlarm** = standardProperties::DISPLAY\_ALARM\_STATE\_WHEN\_IN\_ALARM }

### Public Slots

- void [setManagedVisible](#) (bool v)

## 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 **setOptionsLayout** (int pValue)
- int **getOptionsLayout** ()
- void **setCurrentUserType** (int pValue)
- int **getCurrentUserType** ()
- void **refreshFields** ()
- void **userLevelChanged** (userLevelTypes::userLevels pValue)
- void **setConfigurationTypeProperty** (configurationTypesProperty pConfigurationType)
  
- configurationTypesProperty **getConfigurationTypeProperty** ()
- void **setOptionsLayoutProperty** (optionsLayoutProperty pOptionsLayout)
- optionsLayoutProperty **getOptionsLayoutProperty** ()
- 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.
- DisplayAlarmStateOptions **getDisplayAlarmStateOptionProperty** ()
 

Access function for *displayAlarmStateOption* property - refer to *displayAlarmStateOption* property for details.
- void **setDisplayAlarmStateOptionProperty** (DisplayAlarmStateOptions option)
 

Access function for *displayAlarmStateOption* property - refer to *displayAlarmStateOption* property for details.

## 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 **optionsLayout**
- int **currentUserType**
- bool **subscription**

### Properties

- QString **itemDescription**
- bool **showItemList**
- configurationTypesProperty **configurationType**
- optionsLayoutProperty **optionsLayout**

*Change the order of the widgets. Valid orders are: TOP, BOTTOM, LEFT and RIG.*

- bool **variableAsToolTip**
- bool **allowDrop**
- bool **visible**
- unsigned **int**
- QString **styleSheet**
- QString **defaultStyle**
- QString **userLevelUserStyle**
- QString **userLevelScientistStyle**
- QString **userLevelEngineerStyle**
- UserLevels **userLevelVisibility**
- UserLevels **userLevelEnabled**
- bool **displayAlarmState**
- DisplayAlarmStateOptions **displayAlarmStateOption**

#### 9.56.1 Member Enumeration Documentation

##### 9.56.1.1 enum QEConfiguredLayout::DisplayAlarmStateOptions

User friendly enumerations for **displayAlarmStateOption** property - refer to **displayAlarmStateOption** property and **displayAlarmStateOptions** enumeration for details.

###### Enumerator:

- Never** Refer to DISPLAY\_ALARM\_STATE\_NEVER for details.  
**Always** Refer to DISPLAY\_ALARM\_STATE\_ALWAYS for details.  
**WhenInAlarm** Refer to DISPLAY\_ALARM\_STATE\_WHEN\_IN\_ALARM for details.

### 9.56.1.2 enum QEConfiguredLayout::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.

## 9.56.2 Member Function Documentation

### 9.56.2.1 void QEConfiguredLayout::setManagedVisible( bool v ) [inline, slot]

Slot to set the visibility of a QE widget, taking into account the user level. Widget will be hidden if hidden by a call this slot, by will only be made visible by a call to this slot if the user level allows.

## 9.56.3 Property Documentation

### 9.56.3.1 bool QEConfiguredLayout::allowDrop [read, write]

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

### 9.56.3.2 QString QEConfiguredLayout::defaultStyle [read, write]

Style Sheet string to be applied before, i.e. lower priority than, any other style, e.g. alarm style and/or user level style. Default is an empty string.

### 9.56.3.3 bool QEConfiguredLayout::displayAlarmState [read, write]

DEPRECATED. USE `displayAlarmStateOption` INSTEAD. If set (default) widget will indicate the alarm state of any variable data it is displaying. If clear widget will never indicate the alarm state of any variable data it is displaying. Typically the background colour is set to indicate the alarm state. Note, this property is included in the set of standard properties as it applies to most widgets. It will do nothing for widgets that don't display data.

### 9.56.3.4 DisplayAlarmStateOptions QEConfiguredLayout::displayAlarmStateOption [read, write]

If 'Always' (default) widget will indicate the alarm state of any variable data it is displaying, including 'No Alarm'. If 'Never' widget will never indicate the alarm state of any

variable data it is displaying. If 'WhenInAlarm' widget only indicate the alarm state of any variable data it is displaying if it is 'in alarm'. Typically the background colour is set to indicate the alarm state. Note, this property is included in the set of standard properties as it applies to most widgets. It will do nothing for widgets that don't display data.

#### 9.56.3.5 unsigned QEConfiguredLayout::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.

#### 9.56.3.6 QString QEConfiguredLayout::styleSheet [read, write]

Hide style sheet from designer as style calculation by the styleManager and not directly setable per se. This also stops transient styles being saved to the ui file.

#### 9.56.3.7 UserLevels QEConfiguredLayout::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'.

#### 9.56.3.8 QString QEConfiguredLayout::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.

#### 9.56.3.9 QString QEConfiguredLayout::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.

---

**9.56.3.10 QString QEConfiguredLayout::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.

**9.56.3.11 UserLevels QEConfiguredLayout::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'.

**9.56.3.12 bool QEConfiguredLayout::variableAsToolTip [read, write]**

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

**9.56.3.13 bool QEConfiguredLayout::visible [read, write]**

Display the widget. Default is true. Setting this property false is useful 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:

- /tmp/epicsqt/trunk/framework/widgets/QEConfiguredLayout/QEConfiguredLayout.h
- /tmp/epicsqt/trunk/framework/widgets/QEConfiguredLayout/QEConfiguredLayout.cpp

## 9.57 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:

- `/tmp/epicsqt/trunk/framework/widgets/QEConfiguredLayout/QEConfiguredLayoutManager.h`
- `/tmp/epicsqt/trunk/framework/widgets/QEConfiguredLayout/QEConfiguredLayoutManager.cpp`

## 9.58 QEFileBrowser Class Reference

```
#include <QEFileBrowser.h>
```

### Public Types

- enum `optionsLayoutProperty` { `Top` = TOP, `Bottom` = BOTTOM, `Left` = LEFT, `Right` = RIGHT }
- enum `UserLevels` { `User` = userLevelTypes::USERLEVEL\_USER, `Scientist` = userLevelTypes::USERLEVEL\_SCIENTIST, `Engineer` = userLevelTypes::USERLEVEL\_ENGINEER }
- enum `DisplayAlarmStateOptions` { `Never` = standardProperties::DISPLAY\_ALARM\_STATE\_NEVER, `Always` = standardProperties::DISPLAY\_ALARM\_STATE\_ALWAYS, `WhenInAlarm` = standardProperties::DISPLAY\_ALARM\_STATE\_WHEN\_IN\_ALARM }

### Public Slots

- void `setManagedVisible` (bool v)

### Signals

- void `selected` (QString pFilename)

### Public Member Functions

- `QEFileBrowser` (QWidget \*pParent=0)
- void `setVariableName` (QString pValue)
- `QString getVariableName ()`
- void `setVariableNameSubstitutions` (QString pValue)
- `QString getVariableNameSubstitutions ()`
- 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 **setShowTable** (bool pValue)
- bool **getShowTable** ()
- 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 **setFileDialogDirectoriesOnly** (bool pValue)
- bool **getFileDialogDirectoriesOnly** ()
- void **setOptionsLayout** (int pValue)
- int **getOptionsLayout** ()
- void **updateTable** ()
- void **setOptionsLayoutProperty** (optionsLayoutProperty pOptionsLayout)
- optionsLayoutProperty **getOptionsLayoutProperty** ()
- 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.*
- DisplayAlarmStateOptions **getDisplayAlarmStateOptionProperty** ()  
*Access function for displayAlarmStateOption property - refer to displayAlarmStateOption property for details.*
- void **setDisplayAlarmStateOptionProperty** (DisplayAlarmStateOptions option)  
*Access function for displayAlarmStateOption property - refer to displayAlarmStateOption property for details.*

## Protected Attributes

- `QELineEdit * qeLineEditDirectoryPath`
- `QPushButton * qPushButtonDirectoryBrowser`
- `QPushButton * qPushButtonRefresh`
- `_QTableWidgetFileBrowser * qTableWidgetFileBrowser`
- `QString fileFilter`

*Specify which files to browse. To specify more than one filter, please separate them with a ;. Example: \*.py;\*.ui (this will only display files with an extension .py or .ui).*
- `bool showFileExtension`

*Show/hide the extension of files.*
- `bool fileDialogDirectoriesOnly`

*Enable/disable the browsing of directories-only when opening the dialog window.*
- `int optionsLayout`

## Properties

- `QString variable`
- `QString variableSubstitutions`
- `QString directoryPath`

*Default directory where to browse files when `QEFileBrowser` is launched for the first time.*
- `bool showDirectoryPath`

*Show/hide directory path line edit where the user can specify the directory to browse files.*
- `bool showDirectoryBrowser`

*Show/hide button to open the dialog window to browse for directories and files.*
- `bool showRefresh`

*Show/hide button to refresh the table containing the list of files being browsed.*
- `bool showTable`

*Show/hide table containing the list of files being browsed.*
- `bool showColumnTime`

*Show/hide column containing the time of creation of files.*
- `bool showColumnSize`

*Show/hide column containing the size (in bytes) of files.*
- `bool showColumnFilename`

*Show/hide column containing the name of files.*
- `optionsLayoutProperty optionsLayout`

*Change the order of the widgets. Valid orders are: TOP, BOTTOM, LEFT and RIGHT.*
- `bool variableAsToolTip`
- `bool allowDrop`
- `bool visible`
- `unsigned int`
- `QString styleSheet`
- `QString defaultStyle`

- `QString userLevelUserStyle`
- `QString userLevelScientistStyle`
- `QString userLevelEngineerStyle`
- `UserLevels userLevelVisibility`
- `UserLevels userLevelEnabled`
- `bool displayAlarmState`
- `DisplayAlarmStateOptions displayAlarmStateOption`

### 9.58.1 Detailed Description

This class is a EPICS aware widget. The [QEFileBrowser](#) widget allows the user to browse existing files from a certain directory.

### 9.58.2 Member Enumeration Documentation

#### 9.58.2.1 enum QEFileBrowser::DisplayAlarmStateOptions

User friendly enumerations for `displayAlarmStateOption` property - refer to [displayAlarmStateOption](#) property and `displayAlarmStateOptions` enumeration for details.

##### Enumerator:

**Never** Refer to `DISPLAY_ALARM_STATE_NEVER` for details.

**Always** Refer to `DISPLAY_ALARM_STATE_ALWAYS` for details.

**WhenInAlarm** Refer to `DISPLAY_ALARM_STATE_WHEN_IN_ALARM` for details.

#### 9.58.2.2 enum QEFileBrowser::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.

### 9.58.3 Member Function Documentation

#### 9.58.3.1 void QEFileBrowser::selected ( `QString pFilename` ) [signal]

Signal that is generated every time the user double-clicks a certain file. This signal emits a string that contains the full path and the name of the selected file. This signal may be captured by other widgets that perform further operations (for instance, the [QEImage](#) displays the content of this file if it is a graphical one).

**9.58.3.2 void QEFileBrowser::setManagedVisible( bool v ) [inline, slot]**

Slot to set the visibility of a QE widget, taking into account the user level. Widget will be hidden if hidden by a call this slot, by will only be made visible by a call to this slot if the user level allows.

**9.58.4 Property Documentation****9.58.4.1 bool QEFileBrowser::allowDrop [read, write]**

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

**9.58.4.2 QString QEFileBrowser::defaultStyle [read, write]**

Style Sheet string to be applied before, i.e. lower priority than, any other style, e.g. alarm style and/or user level style. Default is an empty string.

**9.58.4.3 bool QEFileBrowser::displayAlarmState [read, write]**

DEPRECATED. USE displayAlarmStateOption INSTEAD. If set (default) widget will indicate the alarm state of any variable data it is displaying. If clear widget will never indicate the alarm state of any variable data it is displaying. Typically the background colour is set to indicate the alarm state. Note, this property is included in the set of standard properties as it applies to most widgets. It will do nothing for widgets that don't display data.

**9.58.4.4 DisplayAlarmStateOptions QEFileBrowser::displayAlarmStateOption [read, write]**

If 'Always' (default) widget will indicate the alarm state of any variable data it is displaying, including 'No Alarm'. If 'Never' widget will never indicate the alarm state of any variable data it is displaying. If 'WhenInAlarm' widget only indicate the alarm state of any variable data it is displaying if it is 'in alarm'. Typically the background colour is set to indicate the alarm state. Note, this property is included in the set of standard properties as it applies to most widgets. It will do nothing for widgets that don't display data.

**9.58.4.5 unsigned QEFileBrowser::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.

**9.58.4.6 QString QEFileBrowser::styleSheet [read, write]**

Hide style sheet from designer as style calculation by the styleManager and not directly setable per se. This also stops transient styles being saved to the ui file.

**9.58.4.7 UserLevels QEFileBrowser::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'.

**9.58.4.8 QString QEFileBrowser::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.

**9.58.4.9 QString QEFileBrowser::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.

**9.58.4.10 QString QEFileBrowser::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.

**9.58.4.11 UserLevels QEFileBrowser::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'.

#### 9.58.4.12 `QString QEFileBrowser::variable [read, write]`

EPICS variable name (CA PV). This variable is used for both writing and reading the directory to be used by the widget.

#### 9.58.4.13 `bool QEFileBrowser::variableAsToolTip [read, write]`

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

#### 9.58.4.14 `QString QEFileBrowser::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.

#### 9.58.4.15 `bool QEFileBrowser::visible [read, write]`

Display the widget. Default is true. Setting this property false is useful 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:

- /tmp/epicsqt/trunk/framework/widgets/QEFileBrowser/QEFileBrowser.h
- /tmp/epicsqt/trunk/framework/widgets/QEFileBrowser/QEFileBrowser.cpp

## 9.59 QEForm Class Reference

### Public Types

- enum **MessageFilterOptions** { **Match** = UserMessage::MESSAGE\_FILTER\_MATCH, **None** = UserMessage::MESSAGE\_FILTER\_NONE }

### Public Slots

- bool [readUiFile \(\)](#)

*Find a widget within the ui loaded by the [QEForm](#). Returns NULL if no UI is loaded yet or if the named widget can't be found.*

- void [requestAction](#) (const QEActionRequests &request)

*Read a .ui file and present it within this [QEForm](#).*

## Signals

- void **formLoaded** (bool fileLoaded)

## Public Member Functions

- **QEForm** (QWidget \*parent=0)
- **QEForm** (const QString &uifileNameIn, QWidget \*parent=0)
- void **commonInit** (const bool alertIfUINotFoundIn, const bool loadManuallyIn)
- void **setQEGuiTitle** (const QString titleIn)
- QString [getQEGuiTitle](#) ()
 

*Set the title to be used as the window or form title. (note, also set when reading a .ui file)*
- QString [getFullName](#) ()
 

*Get the title to be used as the window or form title.*
- QString [getUiFileName](#) ()
 

*Get the standard, absolute UI file name.*
- void [setFileMonitoringEnabled](#) (bool fileMonitoringEnabled)
 

*Get the fully substituted file name (Not the uiFile property)*
- bool [getFileMonitoringEnabled](#) ()
 

*Set flag indicating if form should take account of file monitoring.*
- void [setHandleGuiLaunchRequests](#) (bool handleGuiLaunchRequests)
 

*Get flag indicating if form should take account of file monitoring.*
- bool [getHandleGuiLaunchRequests](#) ()
 

*Set flag indicating form should handle gui form launch requests.*
- void [setResizeContents](#) (bool resizeContentsIn)
 

*Get flag indicating form should handle gui form launch requests.*
- bool [getResizeContents](#) ()
 

*Set flag indicating form should resize contents to match form size (otherwise resize form to match contents)*
- QString [getContainedFrameworkVersion](#) ()
 

*Get flag indicating form should resize contents to match form size (otherwise resize form to match contents)*
- QString [getUniqueId](#) ()
 

*Get the version of the first QE widget (if any) of QE widgets by QUILoader.*
- void [setUniqueId](#) (QString name)
 

*Get a unique identifier string for this form. This identifier should be persistant across application runs as it is based on the QEForm's position in the widget hierarchy. The same widget will generate the same identifier when opened within the same GUI.*

- int **getDisconnectedCount** ()  
*Set a unique identifier string for this form. This identifier should be persistant across application runs as it is based on the QEForm's position in the widget hierarchy. The same widget will generate the same identifier when opened within the same GUI.*
- int **getConnectedCount** ()  
*Return the count of disconnected variables.*
- QWidget \* **getChild** (QString name)  
*Return the count of connected variables.*
- void **setUiFileNameProperty** (QString uiFileName)
- QString **getUiFileNameProperty** ()
- void **setVariableNameSubstitutionsProperty** (QString variableNameSubstitutions)
- QString **getVariableNameSubstitutionsProperty** ()
- MessageFilterOptions **getMessageFormFilter** ()
- void **setMessageFormFilter** (MessageFilterOptions messageFormFilter)
- MessageFilterOptions **getMessageSourceFilter** ()
- void **setMessageSourceFilter** (MessageFilterOptions messageSourceFilter)

## Protected Attributes

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

## Properties

- QString **uiFile**
- QString **variableSubstitutions**
- unsigned **int**
- MessageFilterOptions **messageFormFilter**
- MessageFilterOptions **messageSourceFilter**
- bool **variableAsToolTip**
- bool **allowDrop**
- DisplayAlarmStateOptions **displayAlarmStateOption**

### 9.59.1 Member Data Documentation

9.59.1.1 **bool QEForm::handleGuiLaunchRequests** [read, write, protected]

If true, the **QEForm** widget publishes its own slot for launching new GUIs so all QE widgets within it will use the QEForm's mechanism for launching new GUIs, rather than any mechanism the application may provide (through the ContainerProfile mechanism)

**9.59.1.2 bool QEForm::resizeContents [read, write, protected]**

If set, the [QEForm](#) will resize the top level widget of the .ui file it opens (and set other size and border related properties) to match itself. This is useful if the [QEForm](#) is used as a sub form within a main form (possibly another [QEForm](#)) and you want to control the size of the [QEForm](#) being used as a sub form. If clear, the [QEForm](#) will resize itself (and set other size and border related properties) to match the top level widget of the .ui file it opens. This is useful if the [QEForm](#) is used as a sub form within a main form (possibly another [QEForm](#)) and you want the main form to resize to match the size of the [QEForm](#) being used as a sub form, or you want the sub form border decorations (such as frame shape and shadow) to be displayed.

**9.59.2 Property Documentation****9.59.2.1 bool QEForm::allowDrop [read, write]**

allowDrop is added as a non-designable property here only to hide the implementation present in QEAbstractWidget

**9.59.2.2 DisplayAlarmStateOptions QEForm::displayAlarmStateOption [read, write]**

displayAlarmStateOption is added as a non-designable property here only to hide the implementation present in QEAbstractWidget

**9.59.2.3 unsigned QEForm::int [read, write]**

Widgets or applications that use messages from the framework have the option of filtering on this ID. Messages that the [QEForm](#) widget catches with its message filters will be regenerated using this ID

**9.59.2.4 MessageFilterOptions QEForm::messageFormFilter [read, write]**

Message filter that attempts to match messages sent through the QE message logging system based on the automatically generated message form ID. This filter will match form ID of the message to the form ID of this QEform as follows:

Any - A message will always be accepted. Match - A message will be accepted if it comes from a QE widget within this form. None - The message will not be matched based on the form the message comes from. (It may still be accepted based on the message source ID.) Matched messages will be resent with the messageSourceld of this [QEForm](#)

**9.59.2.5 MessageFilterOptions QEForm::messageSourceFilter [read, write]**

!!!! is this a valid property. Resending messages based on the source ID is unnecessary as they will be sent on with the same source ID? Message filter that attempts

to match messages sent through the QE message logging system based on the messageSourceId of the widget that generatedd the messge. This filter will match message message source ID of the message to the message source ID of this QEform as follows:

Any - A message will always be accepted. Match - A message will be accepted if the message source ID matches this [QEForm](#). None - The message will not be matched based of message source ID (It may still be accepted based on the message form ID.) Matched messages will be resend with the messageSourceId of this [QEForm](#).

#### 9.59.2.6 QString QEForm::uiFile [read, write]

File name of the .ui file being presented within the [QEForm](#) widget.

#### 9.59.2.7 bool QEForm::variableAsToolTip [read, write]

variableAsToolTip is added as a non-designable property here only to hide the implementation present in QEAbstractWidget

#### 9.59.2.8 QString QEForm::variableSubstitutions [read, write]

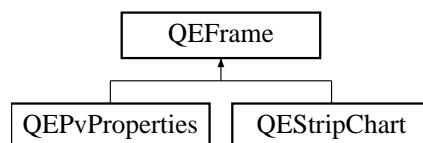
Macro substitutions to be applied to this widget, and all QE widgets that are opened when the .ui file is presented. Note, despite the name, the macro substitutions are general macro substitutions, and do not just apply to a variable name (in fact a [QEForm](#) widget does not even have a variable name property.

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

- /tmp/epicsqt/trunk/framework/widgets/QEForm/QEForm.h
- /tmp/epicsqt/trunk/framework/widgets/QEForm/QEForm.cpp

## 9.60 QEFrame Class Reference

Inheritance diagram for QEFrame:



### Public Types

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

- enum **DisplayAlarmStateOptions** { **Never** = standardProperties::DISPLAY\_ALARM\_STATE\_NEVER, **Always** = standardProperties::DISPLAY\_ALARM\_STATE\_ALWAYS, **WhenInAlarm** = standardProperties::DISPLAY\_ALARM\_STATE\_WHEN\_IN\_ALARM }

## Public Slots

- void **setManagedVisible** (bool v)

## Public Member Functions

- void **setPixmap** (QPixmap pixmapIn)
- QPixmap **getPixmap** () const
- void **setScaledContents** (bool scaledContentsIn)
- bool **getScaledContents** ()
- 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.*
- DisplayAlarmStateOptions **getDisplayAlarmStateOptionProperty** ()
 

*Access function for displayAlarmStateOption property - refer to displayAlarmStateOption property for details.*
- void **setDisplayAlarmStateOptionProperty** (DisplayAlarmStateOptions option)
 

*Access function for displayAlarmStateOption property - refer to displayAlarmStateOption property for details.*
- QEFrame (**QWidget** \*parent=0)
- QSize **sizeHint** () const

## Protected Member Functions

- void **paintEvent** (QPaintEvent \*event)

## Properties

- bool `variableAsToolTip`
- bool `allowDrop`
- bool `visible`
- unsigned `int`
- QString `styleSheet`
- QString `defaultStyle`
- QString `userLevelUserStyle`
- QString `userLevelScientistStyle`
- QString `userLevelEngineerStyle`
- `UserLevels userLevelVisibility`
- `UserLevels userLevelEnabled`
- bool `displayAlarmState`
- `DisplayAlarmStateOptions displayAlarmStateOption`

### 9.60.1 Member Enumeration Documentation

#### 9.60.1.1 enum QEFrame::DisplayAlarmStateOptions

User friendly enumerations for `displayAlarmStateOption` property - refer to `displayAlarmStateOption` property and `displayAlarmStateOptions` enumeration for details.

##### Enumerator:

- Never** Refer to `DISPLAY_ALARM_STATE_NEVER` for details.  
**Always** Refer to `DISPLAY_ALARM_STATE_ALWAYS` for details.  
**WhenInAlarm** Refer to `DISPLAY_ALARM_STATE_WHEN_IN_ALARM` for details.

#### 9.60.1.2 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.

### 9.60.2 Member Function Documentation

#### 9.60.2.1 void QEFrame::setManagedVisible ( bool v ) [inline, slot]

Slot to set the visibility of a QE widget, taking into account the user level. Widget will be hidden if hidden by a call this slot, by will only be made visible by a call to this slot if the user level allows.

### 9.60.3 Property Documentation

#### 9.60.3.1 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.

#### 9.60.3.2 QString QEFrame::defaultStyle [read, write]

Style Sheet string to be applied before, i.e. lower priority than, any other style, e.g. alarm style and/or user level style. Default is an empty string.

#### 9.60.3.3 bool QEFrame::displayAlarmState [read, write]

DEPRECATED. USE `displayAlarmStateOption` INSTEAD. If set (default) widget will indicate the alarm state of any variable data it is displaying. If clear widget will never indicate the alarm state of any variable data it is displaying. Typically the background colour is set to indicate the alarm state. Note, this property is included in the set of standard properties as it applies to most widgets. It will do nothing for widgets that don't display data.

#### 9.60.3.4 DisplayAlarmStateOptions QEFrame::displayAlarmStateOption [read, write]

If 'Always' (default) widget will indicate the alarm state of any variable data it is displaying, including 'No Alarm'. If 'Never' widget will never indicate the alarm state of any variable data it is displaying. If 'WhenInAlarm' widget only indicate the alarm state of any variable data it is displaying if it is 'in alarm'. Typically the background colour is set to indicate the alarm state. Note, this property is included in the set of standard properties as it applies to most widgets. It will do nothing for widgets that don't display data.

#### 9.60.3.5 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.

#### 9.60.3.6 QString QEFrame::styleSheet [read, write]

Hide style sheet from designer as style calculation by the styleManager and not directly setable per se. This also stops transient styles being saved to the ui file.

**9.60.3.7 UserLevels QEFrame::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'.

**9.60.3.8 QString QEFrame::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.

**9.60.3.9 QString QEFrame::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.

**9.60.3.10 QString QEFrame::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.

**9.60.3.11 UserLevels QEFrame::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'.

### 9.60.3.12 bool QEFrame::variableAsToolTip [read, write]

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

### 9.60.3.13 bool QEFrame::visible [read, write]

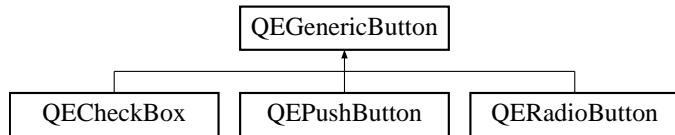
Display the widget. Default is true. Setting this property false is useful if widget is only used to provide a signal - for example, when supplying data to a [QELink](#) widget. Note, when false the widget will still be visible in Qt Designer.

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

- /tmp/epicsqt/trunk/framework/widgets/QEFrame/QEFrame.h
- /tmp/epicsqt/trunk/framework/widgets/QEFrame/QEFrame.cpp

## 9.61 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 **setUpdOption** (updateOptions updateOptionIn)
- updateOptions **getUpdOption** ()
- void **setTextAlignment** (Qt::Alignment alignment)
- Qt::Alignment **getTextAlignment** ()
- void **setPassword** (QString password)
- QString **getPassword** ()
- void **setConfirmAction** (bool confirmRequiredIn)
- bool **getConfirmAction** ()

- void **setConfirmText** (QString confirmTextIn)
- QString **getConfirmText** ()
- 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 **setProgramStartupOption** (applicationLauncher::programStartupOptions programStartupOptionIn)
- applicationLauncher::programStartupOptions **getProgramStartupOption** ()
- void **setGuiName** (QString guiName)
- QString **getGuiName** ()
- void **setPrioritySubstitutions** (QString prioritySubstitutionsIn)
- QString **getPrioritySubstitutions** ()
- void **setCustomisationName** (QString customisationNameIn)
- QString **getCustomisationName** ()
- void **setCreationOption** (QEActionRequests::Options creationOption)
- QEActionRequests::Options **getCreationOption** ()
- void **setLabelTextProperty** (QString labelTextIn)
- QString **getLabelTextProperty** ()

### Protected Member Functions

- void **connectionChanged** (QCaConnectionInfo &connectionInfo, const unsigned int &variableIndex)
- void **setGenericButtonText** (const QString &text, QCaAlarmInfo &alarmInfo, QCaDateTime &, const unsigned int &variableIndex)
- void **userPressed** ()
- void **userReleased** ()
- void **userClicked** (bool checked)
- void **processWriteNow** (const bool checked)
- virtual updateOptions **getDefaultUpdateOption** ()=0
- void **setup** ()
- void **establishConnection** (unsigned int variableIndex)
- void **calcStyleOption** ()

## Protected Attributes

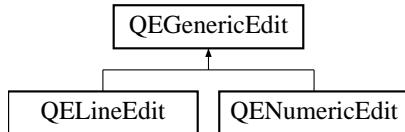
- applicationLauncher **programLauncher**

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

- /tmp/epicsqt/trunk/framework/widgets/QEButton/QEGenericButton.h
- /tmp/epicsqt/trunk/framework/widgets/QEButton/QEGenericButton.cpp

## 9.62 QEGenericEdit Class Reference

Inheritance diagram for QEGenericEdit:



## Public Types

- enum **UserLevels** { **User** = userLevelTypes::USERLEVEL\_USER, **Scientist** = userLevelTypes::USERLEVEL\_SCIENTIST, **Engineer** = userLevelTypes::USERLEVEL\_ENGINEER }
- enum **DisplayAlarmStateOptions** { **Never** = standardProperties::DISPLAY\_ALARM\_STATE\_NEVER, **Always** = standardProperties::DISPLAY\_ALARM\_STATE\_ALWAYS, **WhenInAlarm** = standardProperties::DISPLAY\_ALARM\_STATE\_WHEN\_IN\_ALARM }

## Public Slots

- void **setManagedVisible** (bool v)
- void **setDefaultStyle** (const QString &style)

*Update the default style applied to this widget.*

## Signals

- void **userChange** (const QVariant &oldValue, const QVariant &newValue, const QVariant &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

- void [setVariableNameProperty](#) (QString variableName)  
*Property access function for `variable` property. This has special behaviour to work well within designer.*
- QString [getVariableNameProperty](#) ()  
*Property access function for `variable` property. This has special behaviour to work well within designer.*
- void [setVariableNameSubstitutionsProperty](#) (QString variableNameSubstitutions)  
*Property access function for `variableSubstitutions` property. This has special behaviour to work well within designer.*
- QString [getVariableNameSubstitutionsProperty](#) ()  
*Property access function for `variableSubstitutions` property. This has special behaviour to work well within designer.*
- 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.*
- DisplayAlarmStateOptions [getDisplayAlarmStateOptionProperty](#) ()  
*Access function for `displayAlarmStateOption` property - refer to `displayAlarmStateOption` property for details.*
- void [setDisplayAlarmStateOptionProperty](#) (DisplayAlarmStateOptions option)  
*Access function for `displayAlarmStateOption` property - refer to `displayAlarmStateOption` property for details.*
- QEGenericEdit (QWidget \*parent=0)
- QEGenericEdit (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](#) ()
- void [writeValue](#) (qcaobject::QCaObject \*qca, QVariant newValue)
- void [writeNow](#) ()

### Protected Member Functions

- void **setDataIfNoFocus** (const QVariant &value, QCaAlarmInfo &alarmInfo, QCaDateTime &dateTime)
- bool **getIsConnected** () const
- bool **getIsFirstUpdate** () const
- virtual void **setValue** (const QVariant &value)=0
- virtual QVariant **getValue** ()=0
- virtual bool **writeData** (const QVariant &value, QString &message)=0

### Protected Attributes

- QVariant **lastValue**
- QVariant **lastUserValue**
- bool **messageDialogPresent**
- bool **writeFailMessageDialogPresent**
- bool **isConnected**

### Properties

- QString **text**
- QString **variable**
- QString **variableSubstitutions**
- bool **subscribe**
- bool **writeOnLoseFocus**
- bool **writeOnEnter**
- bool **writeOnFinish**
- bool **confirmWrite**
- bool **variableAsToolTip**
- bool **allowDrop**
- bool **visible**
- unsigned **int**
- QString **styleSheet**
- QString **defaultStyle**
- QString **userLevelUserStyle**
- QString **userLevelScientistStyle**
- QString **userLevelEngineerStyle**
- **UserLevels userLevelVisibility**
- **UserLevels userLevelEnabled**
- bool **displayAlarmState**
- **DisplayAlarmStateOptions displayAlarmStateOption**

### 9.62.1 Member Enumeration Documentation

#### 9.62.1.1 enum QEGenericEdit::DisplayAlarmStateOptions

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

##### Enumerator:

**Never** Refer to DISPLAY\_ALARM\_STATE\_NEVER for details.

**Always** Refer to DISPLAY\_ALARM\_STATE\_ALWAYS for details.

**WhenInAlarm** Refer to DISPLAY\_ALARM\_STATE\_WHEN\_IN\_ALARM for details.

#### 9.62.1.2 enum QEGenericEdit::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.

### 9.62.2 Constructor & Destructor Documentation

#### 9.62.2.1 QEGenericEdit::QEGenericEdit ( QWidget \* *parent* = 0 )

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

#### 9.62.2.2 QEGenericEdit::QEGenericEdit ( 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.

### 9.62.3 Member Function Documentation

#### 9.62.3.1 bool QEGenericEdit::getConfirmWrite ( )

Returns 'true' if this widget will ask for confirmation (using a dialog box) prior to writing data.

**9.62.3.2 bool QEGenericEdit::getSubscribe ( )**

Returns 'true' if this widget subscribes for data updates and displays current data.

**9.62.3.3 bool QEGenericEdit::getWriteOnEnter ( )**

Returns 'true' if this widget writes any changes when the user presses 'enter'.

**9.62.3.4 bool QEGenericEdit::getWriteOnFinish ( )**

Returns 'true' if this widget writes any changes when the user finished editing (the QLineEdit 'editingFinished' signal is emitted).

**9.62.3.5 bool QEGenericEdit::getWriteOnLoseFocus ( )**

Returns 'true' if this widget automatically writes any changes when it loses focus.

**9.62.3.6 void QEGenericEdit::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).

**9.62.3.7 void QEGenericEdit::setManagedVisible ( bool *v* ) [inline, slot]**

Slot to set the visibility of a QE widget, taking into account the user level. Widget will be hidden if hidden by a call this slot, by will only be made visible by a call to this slot if the user level allows.

**9.62.3.8 void QEGenericEdit::setSubscribe ( bool *subscribe* )**

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

**9.62.3.9 void QEGenericEdit::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').

**9.62.3.10 void QEGenericEdit::setWriteOnFinish ( bool *writeOnFinish* )**

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).

**9.62.3.11 void QEGenericEdit::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).

#### 9.62.4 Property Documentation

**9.62.4.1 bool QEGenericEdit::allowDrop [read, write]**

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

**9.62.4.2 bool QEGenericEdit::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).

**9.62.4.3 QString QEGenericEdit::defaultStyle [read, write]**

Style Sheet string to be applied before, i.e. lower priority than, any other style, e.g. alarm style and/or user level style. Default is an empty string.

**9.62.4.4 bool QEGenericEdit::displayAlarmState [read, write]**

DEPRECATED. USE displayAlarmStateOption INSTEAD. If set (default) widget will indicate the alarm state of any variable data it is displaying. If clear widget will never indicate the alarm state of any variable data it is displaying. Typically the background colour is set to indicate the alarm state. Note, this property is included in the set of standard properties as it applies to most widgets. It will do nothing for widgets that don't display data.

**9.62.4.5 DisplayAlarmStateOptions QEGenericEdit::displayAlarmStateOption [read, write]**

If 'Always' (default) widget will indicate the alarm state of any variable data it is displaying, including 'No Alarm'. If 'Never' widget will never indicate the alarm state of any variable data it is displaying. If 'WhenInAlarm' widget only indicate the alarm state of any variable data it is displaying if it is 'in alarm'. Typically the background colour is set to indicate the alarm state. Note, this property is included in the set of standard properties as it applies to most widgets. It will do nothing for widgets that don't display data.

**9.62.4.6 unsigned QEGenericEdit::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.

Reimplemented in [QELineEdit](#).

**9.62.4.7 QString QEGenericEdit::styleSheet [read, write]**

Hide style sheet from designer as style calculation by the styleManager and not directly setable per se. This also stops transient styles being saved to the ui file.

**9.62.4.8 bool QEGenericEdit::subscribe [read, write]**

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

**9.62.4.9 UserLevels QEGenericEdit::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'.

**9.62.4.10 QString QEGenericEdit::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.

**9.62.4.11 QString QEGenericEdit::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.

**9.62.4.12 QString QEGenericEdit::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.

**9.62.4.13 UserLevels QEGenericEdit::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'.

**9.62.4.14 QString QEGenericEdit::variable [read, write]**

EPICS variable name (CA PV)

**9.62.4.15 bool QEGenericEdit::variableAsToolTip [read, write]**

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

**9.62.4.16 QString QEGenericEdit::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.

**9.62.4.17 bool QEGenericEdit::visible [read, write]**

Display the widget. Default is true. Setting this property false is useful 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.

**9.62.4.18 bool QEGenericEdit::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').

---

**9.62.4.19 bool QEGenericEdit::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).

**9.62.4.20 bool QEGenericEdit::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:

- /tmp/epicsqt/trunk/framework/widgets/QELineEdit/QEGenericEdit.h
- /tmp/epicsqt/trunk/framework/widgets/QELineEdit/QEGenericEdit.cpp

## 9.63 QEGroupBox Class Reference

### Public Types

- enum [UserLevels](#) { [User](#) = userLevelTypes::USERLEVEL\_USER, [Scientist](#) = userLevelTypes::USERLEVEL\_SCIENTIST, [Engineer](#) = userLevelTypes::USERLEVEL\_ENGINEER }
- enum [DisplayAlarmStateOptions](#) { [Never](#) = standardProperties::DISPLAY\_ALARM\_STATE\_NEVER, [Always](#) = standardProperties::DISPLAY\_ALARM\_STATE\_ALWAYS, [WhenInAlarm](#) = standardProperties::DISPLAY\_ALARM\_STATE\_WHEN\_IN\_ALARM }

### Public Slots

- void [setManagedVisible](#) (bool v)

### Public Member Functions

- [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.*

- `DisplayAlarmStateOptions getDisplayAlarmStateOptionProperty ()`  
*Access function for `displayAlarmStateOption` property - refer to `displayAlarmStateOption` property for details.*
- `void setDisplayAlarmStateOptionProperty (DisplayAlarmStateOptions option)`  
*Access function for `displayAlarmStateOption` property - refer to `displayAlarmStateOption` property for details.*
- `QEGroupBox (QWidget *parent=0)`
- `QEGroupBox (const QString &title, QWidget *parent=0)`
- `QSize sizeHint () const`

### Protected Member Functions

- `virtual void setSubstitutionsProperty (QString macroSubstitutionsIn)`
- `QString getSubstitutionsProperty ()`

### Properties

- `bool variableAsToolTip`
- `bool allowDrop`
- `bool visible`
- `unsigned int`
- `QString styleSheet`
- `QString defaultStyle`
- `QString userLevelUserStyle`
- `QString userLevelScientistStyle`
- `QString userLevelEngineerStyle`
- `UserLevels userLevelVisibility`
- `UserLevels userLevelEnabled`
- `bool displayAlarmState`
- `DisplayAlarmStateOptions displayAlarmStateOption`
- `QString substitutedTitle`
- `QString textSubstitutions`

#### 9.63.1 Member Enumeration Documentation

##### 9.63.1.1 enum QEGroupBox::DisplayAlarmStateOptions

User friendly enumerations for `displayAlarmStateOption` property - refer to `displayAlarmStateOption` property and `displayAlarmStateOptions` enumeration for details.

###### Enumerator:

- Never** Refer to `DISPLAY_ALARM_STATE_NEVER` for details.
- Always** Refer to `DISPLAY_ALARM_STATE_ALWAYS` for details.
- WhenInAlarm** Refer to `DISPLAY_ALARM_STATE_WHEN_IN_ALARM` for details.

### 9.63.1.2 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::setManagedVisible ( bool v ) [inline, slot]

Slot to set the visibility of a QE widget, taking into account the user level. Widget will be hidden if hidden by a call this slot, by will only be made visible by a call to this slot if the user level allows.

## 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.

### 9.63.3.2 QString QEGroupBox::defaultStyle [read, write]

Style Sheet string to be applied before, i.e. lower priority than, any other style, e.g. alarm style and/or user level style. Default is an empty string.

### 9.63.3.3 bool QEGroupBox::displayAlarmState [read, write]

DEPRECATED. USE `displayAlarmStateOption` INSTEAD. If set (default) widget will indicate the alarm state of any variable data it is displaying. If clear widget will never indicate the alarm state of any variable data it is displaying. Typically the background colour is set to indicate the alarm state. Note, this property is included in the set of standard properties as it applies to most widgets. It will do nothing for widgets that don't display data.

### 9.63.3.4 DisplayAlarmStateOptions QEGroupBox::displayAlarmStateOption [read, write]

If 'Always' (default) widget will indicate the alarm state of any variable data it is displaying, including 'No Alarm' If 'Never' widget will never indicate the alarm state of any

variable data it is displaying. If 'WhenInAlarm' widget only indicate the alarm state of any variable data it is displaying if it is 'in alarm'. Typically the background colour is set to indicate the alarm state. Note, this property is included in the set of standard properties as it applies to most widgets. It will do nothing for widgets that don't display data.

#### 9.63.3.5 `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.

#### 9.63.3.6 `QString QEGroupBox::styleSheet [read, write]`

Hide style sheet from designer as style calculation by the styleManager and not directly setable per se. This also stops transient styles being saved to the ui file.

#### 9.63.3.7 `QString QEGroupBox::substitutedTitle [read, write]`

Group box title text to be substituted. This text will be copied to the group box title text after applying any macro substitutions from the `textSubstitutions` property

#### 9.63.3.8 `QString QEGroupBox::textSubstitutions [read, write]`

Text substitutions. These substitutions are applied to the 'substitutedTitle' property prior to copying it to the label text.

#### 9.63.3.9 `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 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'.

#### 9.63.3.10 `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.

#### 9.63.3.11 `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.

#### 9.63.3.12 `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.

#### 9.63.3.13 `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'.

#### 9.63.3.14 `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.

#### 9.63.3.15 `bool QEGroupBox::visible` [read, write]

Display the widget. Default is true. Setting this property false is useful 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.

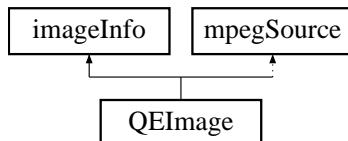
The documentation for this class was generated from the following files:

- /tmp/epicsqt/trunk/framework/widgets/QEGroupBox/QEGroupBox.h
- /tmp/epicsqt/trunk/framework/widgets/QEGroupBox/QEGroupBox.cpp

## 9.64 QEImage Class Reference

```
#include <QEImage.h>
```

Inheritance diagram for QEImage:



### Public Types

- enum `selectOptions` {
 `SO_NONE, SO_PANNING, SO_VSLICE1, SO_VSLICE2,`  
`SO_VSLICE3, SO_VSLICE4, SO_VSLICE5, SO_HSLICE1,`  
`SO_HSLICE2, SO_HSLICE3, SO_HSLICE4, SO_HSLICE5,`  
`SO_AREA1, SO_AREA2, SO_AREA3, SO_AREA4,`  
`SO_PROFILE, SO_TARGET, SO_BEAM }`
- enum `imageUses` { `IMAGE_USE_DISPLAY, IMAGE_USE_SAVE, IMAGE_USE_-`  
`DISPLAY_AND_SAVE }`
- enum `resizeOptions` { `RESIZE_OPTION_ZOOM, RESIZE_OPTION_FIT }`
- enum `ellipseVariableDefinitions` { `BOUNDRING_RECTANGLE, CENTRE_AND_-`  
`SIZE }`
- enum `UserLevels` { `User = userLevelTypes::USERLEVEL_USER, Scientist = userLevelTypes::USERLEVEL_-`  
`SCIENTIST, Engineer = userLevelTypes::USERLEVEL_ENGINEER }`
- enum `DisplayAlarmStateOptions` { `Never = standardProperties::DISPLAY_ALARM_-`  
`STATE_NEVER, Always = standardProperties::DISPLAY_ALARM_STATE_ALWAYS,`  
`WhenInAlarm = standardProperties::DISPLAY_ALARM_STATE_WHEN_IN_ALARM`  
`}`
- enum `FormatOptions` {
 `Mono = imageDataFormats::MONO, Bayer = imageDataFormats::BAYERRG, Bay-`  
`erGB = imageDataFormats::BAYERGB, BayerBG = imageDataFormats::BAYERBG,`  
`BayerGR = imageDataFormats::BAYERGR, BayerRG = imageDataFormats::BAYERRG,`  
`rgb1 = imageDataFormats::RGB1, rgb2 = imageDataFormats::RGB2,`  
`rgb3 = imageDataFormats::RGB3, yuv444 = imageDataFormats::YUV444, yuv422`  
`= imageDataFormats::YUV422, yuv421 = imageDataFormats::YUV421 }`
- enum `EllipseVariableDefinitions` { `BoundingRectangle = BOUNDING_RECTANGLE,`  
`CenterAndSize = CENTRE_AND_SIZE }`
- enum `TargetOptions` { `DottedFullCrosshair = VideoWidget::CROSSHAIR1, SolidS-`  
`mallCrosshair = VideoWidget::CROSSHAIR2 }`
- enum `ResizeOptions` { `Zoom = QEImage::RESIZE_OPTION_ZOOM, Fit = QEImage::RESIZE_-`  
`OPTION_FIT }`

- enum `RotationOptions` { `NoRotation` = imageProperties::ROTATION\_0, `Rotate90Right` = imageProperties::ROTATION\_90\_RIGHT, `Rotate90Left` = imageProperties::ROTATION\_90\_LEFT, `Rotate180` = imageProperties::ROTATION\_180 }
- enum `ProgramStartupOptionNames` { `None` = applicationLauncher::PSO\_NONE, `Terminal` = applicationLauncher::PSO\_TERMINAL, `LogOutput` = applicationLauncher::PSO\_LOGOUTPUT, `StdOutput` = applicationLauncher::PSO\_STDOUPUT }

## Public Slots

- void `setImageFile` (QString name)  
  
*Framework use only. Slot to allow external setting of selection menu options.*
- 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 `setSelectVSlice1Mode` ()  
  
*Framework use only. Slot to allow external setting of selection menu options.*
- void `setSelectVSlice2Mode` ()  
  
*Framework use only. Slot to allow external setting of selection menu options.*
- void `setSelectVSlice3Mode` ()  
  
*Framework use only. Slot to allow external setting of selection menu options.*
- void `setSelectVSlice4Mode` ()  
  
*Framework use only. Slot to allow external setting of selection menu options.*
- void `setSelectVSlice5Mode` ()  
  
*Framework use only. Slot to allow external setting of selection menu options.*
- void `setSelectHSlice1Mode` ()  
  
*Framework use only. Slot to allow external setting of selection menu options.*
- void `setSelectHSlice2Mode` ()  
  
*Framework use only. Slot to allow external setting of selection menu options.*

- void `setSelectHSlice3Mode ()`  
Framework use only. Slot to allow external setting of selection menu options.
- void `setSelectHSlice4Mode ()`  
Framework use only. Slot to allow external setting of selection menu options.
- void `setSelectHSlice5Mode ()`  
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 `targetClicked ()`  
Framework use only. Slot to allow external setting of selection menu options.
- void `imageDisplayPropsDestroyed (QObject *)`  
Framework use only. Slot to catch deletion of components (such as profile plots) that have been passed to the application for presentation.
- void `vSliceDisplayDestroyed (QObject *)`  
Framework use only. Slot to catch deletion of components (such as profile plots) that have been passed to the application for presentation.
- void `hSliceDisplayDestroyed (QObject *)`  
Framework use only. Slot to catch deletion of components (such as profile plots) that have been passed to the application for presentation.
- void `profileDisplayDestroyed (QObject *)`  
Framework use only. Slot to catch deletion of components (such as profile plots) that have been passed to the application for presentation.
- void `recorderDestroyed (QObject *)`  
Framework use only. Slot to catch deletion of components (such as profile plots) that have been passed to the application for presentation.
- void `showProfile ()`  
Show the arbitrary line (profile) markup. --refer to refer to `enableProfileSelection` property and `#displayMarkups` property for details.
- void `hideProfile ()`  
Hide the arbitrary line (profile) markup but note that if its PV changes it will reshown unless `DisplayMarkups` has been set to off - refer to `enableProfileSelection` property and `#displayMarkups` property for details.
- void `showArea1 ()`  
Show the area1 markup - refer to `enableArea1Selection` property and `#displayMarkups` property for details.
- void `hideArea1 ()`  
Hide the area1 markup but note that if its PV changes it will reshown unless `DisplayMarkups` has been set to off - refer to `enableArea1Selection` property and `#displayMarkups` property for details.
- void `setDisplayMarkupsOn ()`  
Set markup display to on to show all markups that change either due to user or PV activity, even if their `setDisplay????Selection` is off - refer to `#displayMarkups` property for details.

- void **setDisplayMarkupsOff** ()
 

*Set markup display to off to stop PV controlled pvs from showing even if they change, unless their setDisplay????Selection is on - refer to #displayMarkups property for details.*
- void **setManagedVisible** (bool v)

### 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 **componentHostRequest** (const QEActionRequests &request)

### Public Member Functions

- **QEImage** (QWidget \*parent=0)
- **QEImage** (const QString &variableName, QWidget \*parent=0)
- **~QEImage** ()
 

*Destructor.*
- **selectOptions getSelectionOption** ()
  - void **setBitDepth** (unsigned int bitDepthIn)
 

*Access function for #bitDepth property - refer to #bitDepth property for details.*
  - unsigned int **getBitDepth** ()
 

*Access function for #bitDepth property - refer to #bitDepth property for details.*
- void **setFormatOption** (imageDataFormats::formatOptions formatOption)
 

*Access function for formatOption property - refer to formatOption property for details.*
- imageDataFormats::formatOptions **getFormatOption** ()
 

*Access function for formatOption property - refer to formatOption property for details.*
- void **setResizeOption** (resizeOptions resizeOptionIn)
 

*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** (imageProperties::rotationOptions rotationIn)
 

*Access function for rotation property - refer to rotation property for details.*
- **imageProperties::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.*

- 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 `setUseFalseColour` (bool pValue)  
Access function for `useFalseColour` property - refer to `useFalseColour` property for details.
  - bool `getUseFalseColour` ()  
Access function for `useFalseColour` property - refer to `useFalseColour` property for details.
  - void `setVertSlice1MarkupColor` (QColor pValue)  
Access function for `#vertSliceColor` property - refer to `#vertSliceColor` property for details.
  - QColor `getVertSlice1MarkupColor` ()  
Access function for `#vertSliceColor` property - refer to `#vertSliceColor` property for details.
  - void `setVertSlice2MarkupColor` (QColor pValue)  
Access function for `vertSlice2Color` property - refer to `vertSlice2Color` property for details.
  - QColor `getVertSlice2MarkupColor` ()  
Access function for `vertSlice2Color` property - refer to `vertSlice2Color` property for details.
  - void `setVertSlice3MarkupColor` (QColor pValue)

- Access function for `vertSlice3Color` property - refer to `vertSlice3Color` property for details.
  - QColor `getVertSlice3MarkupColor ()`  
Access function for `vertSlice3Color` property - refer to `vertSlice3Color` property for details.
  - void `setVertSlice4MarkupColor (QColor pValue)`  
Access function for `vertSlice4Color` property - refer to `vertSlice4Color` property for details.
  - QColor `getVertSlice4MarkupColor ()`  
Access function for `vertSlice4Color` property - refer to `vertSlice4Color` property for details.
  - void `setVertSlice5MarkupColor (QColor pValue)`  
Access function for `vertSlice5Color` property - refer to `vertSlice5Color` property for details.
  - QColor `getVertSlice5MarkupColor ()`  
Access function for `vertSlice5Color` property - refer to `vertSlice5Color` property for details.
  - void `setHozSlice1MarkupColor (QColor pValue)`  
Access function for `#hozSliceColor` property - refer to `#hozSliceColor` property for details.
  - QColor `getHozSlice1MarkupColor ()`  
Access function for `#hozSliceColor` property - refer to `#hozSliceColor` property for details.
  - void `setHozSlice2MarkupColor (QColor pValue)`  
Access function for `hozSlice2Color` property - refer to `hozSlice2Color` property for details.
  - QColor `getHozSlice2MarkupColor ()`  
Access function for `hozSlice2Color` property - refer to `hozSlice2Color` property for details.
  - void `setHozSlice3MarkupColor (QColor pValue)`  
Access function for `hozSlice3Color` property - refer to `hozSlice3Color` property for details.
  - QColor `getHozSlice3MarkupColor ()`  
Access function for `hozSlice3Color` property - refer to `hozSlice3Color` property for details.
  - void `setHozSlice4MarkupColor (QColor pValue)`  
Access function for `hozSlice4Color` property - refer to `hozSlice4Color` property for details.
  - QColor `getHozSlice4MarkupColor ()`  
Access function for `hozSlice4Color` property - refer to `hozSlice4Color` property for details.
  - void `setHozSlice5MarkupColor (QColor pValue)`  
Access function for `hozSlice5Color` property - refer to `hozSlice5Color` property for details.
  - QColor `getHozSlice5MarkupColor ()`  
Access function for `hozSlice5Color` property - refer to `hozSlice5Color` 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 `setEllipseMarkupColor` (QColor markupColor)  
*Access function for ellipseColor property - refer to [ellipseColor](#) property for details.*
- QColor `getEllipseMarkupColor` ()  
*Access function for ellipseColor property - refer to [ellipseColor](#) property for details.*
- void `setDisplayCursorPixelInfo` (bool displayCursorPixelInfo)  
*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 `setLog` (bool log)  
*Access function for logBrightness property - refer to [logBrightness](#) property for details.*
- bool `getLog` ()  
*Access function for logBrightness property - refer to [logBrightness](#) property for details.*
- void `setEnableVertSlice1Selection` (bool enableVSliceSelection)  
*Access function for enableVertSlice1Selection property - refer to [enableVertSlice1Selection](#) property for details.*
- bool `getEnableVertSlice1Selection` ()

- Access function for `enableVertSlice1Selection` property - refer to `enableVertSlice1Selection` property for details.
  - void `setEnableVertSlice2Selection` (bool enableVSliceSelection)  
Access function for `enableVertSlice2Selection` property - refer to `enableVertSlice2Selection` property for details.
  - bool `getEnableVertSlice2Selection` ()  
Access function for `enableVertSlice2Selection` property - refer to `enableVertSlice2Selection` property for details.
- void `setEnableVertSlice3Selection` (bool enableVSliceSelection)  
Access function for `enableVertSlice3Selection` property - refer to `enableVertSlice3Selection` property for details.
- bool `getEnableVertSlice3Selection` ()  
Access function for `enableVertSlice3Selection` property - refer to `enableVertSlice3Selection` property for details.
- void `setEnableVertSlice4Selection` (bool enableVSliceSelection)  
Access function for `enableVertSlice4Selection` property - refer to `enableVertSlice4Selection` property for details.
- bool `getEnableVertSlice4Selection` ()  
Access function for `enableVertSlice4Selection` property - refer to `enableVertSlice4Selection` property for details.
- void `setEnableVertSlice5Selection` (bool enableVSliceSelection)  
Access function for `enableVertSlice5Selection` property - refer to `enableVertSlice5Selection` property for details.
- bool `getEnableVertSlice5Selection` ()  
Access function for `enableVertSlice5Selection` property - refer to `enableVertSlice5Selection` property for details.
- void `setEnableHozSlice1Selection` (bool enableHSliceSelection)  
Access function for `enableHozSlice1Selection` property - refer to `enableHozSlice1Selection` property for details.
- bool `getEnableHozSlice1Selection` ()  
Access function for `enableHozSlice1Selection` property - refer to `enableHozSlice1Selection` property for details.
- void `setEnableHozSlice2Selection` (bool enableHSliceSelection)  
Access function for `enableHozSlice2Selection` property - refer to `enableHozSlice2Selection` property for details.
- bool `getEnableHozSlice2Selection` ()  
Access function for `enableHozSlice2Selection` property - refer to `enableHozSlice2Selection` property for details.
- void `setEnableHozSlice3Selection` (bool enableHSliceSelection)  
Access function for `enableHozSlice3Selection` property - refer to `enableHozSlice3Selection` property for details.
- bool `getEnableHozSlice3Selection` ()  
Access function for `enableHozSlice3Selection` property - refer to `enableHozSlice3Selection` property for details.
- void `setEnableHozSlice4Selection` (bool enableHSliceSelection)  
Access function for `enableHozSlice4Selection` property - refer to `enableHozSlice4Selection` property for details.

- bool [getEnableHozSlice4Selection \(\)](#)  
*Access function for `enableHozSlice4Selection` property - refer to `enableHozSlice4Selection` property for details.*
- void [setEnableHozSlice5Selection \(bool enableHSliceSelection\)](#)  
*Access function for `enableHozSlice5Selection` property - refer to `enableHozSlice5Selection` property for details.*
- bool [getEnableHozSlice5Selection \(\)](#)  
*Access function for `enableHozSlice5Selection` property - refer to `enableHozSlice5Selection` property for details.*
- void [setEnableArea1Selection \(bool enableAreaSelectionIn\)](#)  
*Access function for `enableArea1Selection` property - refer to `enableArea1Selection` property for details.*
- bool [getEnableArea1Selection \(\)](#)  
*Access function for `enableArea1Selection` property - refer to `enableArea1Selection` property for details.*
- void [setEnableArea2Selection \(bool enableAreaSelectionIn\)](#)  
*Access function for `enableArea2Selection` property - refer to `enableArea2Selection` property for details.*
- bool [getEnableArea2Selection \(\)](#)  
*Access function for `enableArea2Selection` property - refer to `enableArea2Selection` property for details.*
- void [setEnableArea3Selection \(bool enableAreaSelectionIn\)](#)  
*Access function for `enableArea3Selection` property - refer to `enableArea3Selection` property for details.*
- bool [getEnableArea3Selection \(\)](#)  
*Access function for `enableArea3Selection` property - refer to `enableArea3Selection` property for details.*
- void [setEnableArea4Selection \(bool enableAreaSelectionIn\)](#)  
*Access function for `enableArea4Selection` property - refer to `enableArea4Selection` property for details.*
- bool [getEnableArea4Selection \(\)](#)  
*Access function for `enableArea4Selection` property - refer to `enableArea4Selection` 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.*
- void [setEnableBeamSelection \(bool enableBeamSelectionIn\)](#)

- Access function for `enableBeamSelection` property - refer to `enableBeamSelection` property for details.
- `bool getEnableBeamSelection ()`  
Access function for `enableBeamSelection` property - refer to `enableBeamSelection` property for details.
  - `void setEnableImageDisplayProperties (bool enableImageDisplayPropertiesIn)`  
Access function for `enableImageDisplayProperties` property - refer to `enableImageDisplayProperties` property for details.
  - `bool getEnableImageDisplayProperties ()`  
Access function for `enableImageDisplayProperties` property - refer to `enableImageDisplayProperties` property for details.
  - `void setEnableRecording (bool enableRecordingIn)`  
Access function for `enableRecording` property - refer to `enableRecording` property for details.
  - `bool getEnableRecording ()`  
Access function for `enableRecording` property - refer to `enableRecording` property for details.
  - `void setAutoBrightnessContrast (bool autoBrightnessContrastIn)`  
Access function for `autoBrightnessContrast` property - refer to `autoBrightnessContrast` property for details.
  - `bool getAutoBrightnessContrast ()`  
Access function for `autoBrightnessContrast` property - refer to `autoBrightnessContrast` property for details.
  - `void setExternalControls (bool externalControlsIn)`  
Access function for `externalControls` property - refer to `externalControls` property for details.
  - `bool getExternalControls ()`  
Access function for `externalControls` property - refer to `externalControls` property for details.
  - `void setFullContextMenu (bool fullContextMenuIn)`  
Access function for `#fullContextMenu` property - refer to `#fullContextMenu` property for details.
  - `bool getFullContextMenu ()`  
Access function for `#fullContextMenu` property - refer to `#fullContextMenu` property for details.
  - `void setEnableProfilePresentation (bool enableProfilePresentationIn)`  
Access function for `#enableProfilePresentation` property - refer to `#enableProfilePresentation` property for details.
  - `bool getEnableProfilePresentation ()`  
Access function for `#enableProfilePresentation` property - refer to `#enableProfilePresentation` property for details.
  - `void setEnableHozSlicePresentation (bool enableHozSlicePresentationIn)`  
Access function for `#enableHozSlicePresentation` property - refer to `#enableHozSlicePresentation` property for details.
  - `bool getEnableHozSlicePresentation ()`  
Access function for `#enableHozSlicePresentation` property - refer to `#enableHozSlicePresentation` property for details.

- void [setEnableVertSlicePresentation](#) (bool enableVertSlicePresentationIn)  
*Access function for #enableVertSlicePresentation property - refer to #enableVertSlicePresentation property for details.*
- bool [getEnableVertSlicePresentation](#) ()  
*Access function for #enableVertSlicePresentation property - refer to #enableVertSlicePresentation property for details.*
- void [setDisplayVertSlice1Selection](#) (bool displayVSliceSelection)  
*Access function for #displayVertSlice1Selection property - refer to #displayVertSlice1Selection property for details.*
- bool [getDisplayVertSlice1Selection](#) ()  
*Access function for #displayVertSlice1Selection property - refer to #displayVertSlice1Selection property for details.*
- void [setDisplayVertSlice2Selection](#) (bool displayVSliceSelection)  
*Access function for #displayVertSlice2Selection property - refer to #displayVertSlice2Selection property for details.*
- bool [getDisplayVertSlice2Selection](#) ()  
*Access function for #displayVertSlice2Selection property - refer to #displayVertSlice2Selection property for details.*
- void [setDisplayVertSlice3Selection](#) (bool displayVSliceSelection)  
*Access function for #displayVertSlice3Selection property - refer to #displayVertSlice3Selection property for details.*
- bool [getDisplayVertSlice3Selection](#) ()  
*Access function for #displayVertSlice3Selection property - refer to #displayVertSlice3Selection property for details.*
- void [setDisplayVertSlice4Selection](#) (bool displayVSliceSelection)  
*Access function for #displayVertSlice4Selection property - refer to #displayVertSlice4Selection property for details.*
- bool [getDisplayVertSlice4Selection](#) ()  
*Access function for #displayVertSlice4Selection property - refer to #displayVertSlice4Selection property for details.*
- void [setDisplayVertSlice5Selection](#) (bool displayVSliceSelection)  
*Access function for #displayVertSlice5Selection property - refer to #displayVertSlice5Selection property for details.*
- bool [getDisplayVertSlice5Selection](#) ()  
*Access function for #displayVertSlice5Selection property - refer to #displayVertSlice5Selection property for details.*
- void [setDisplayHozSlice1Selection](#) (bool displayHSliceSelection)  
*Access function for displayHozSlice1Selection property - refer to displayHozSlice1Selection property for details.*
- bool [getDisplayHozSlice1Selection](#) ()  
*Access function for displayHozSlice1Selection property - refer to displayHozSlice1Selection property for details.*
- void [setDisplayHozSlice2Selection](#) (bool displayHSliceSelection)  
*Access function for displayHozSlice2Selection property - refer to displayHozSlice2Selection property for details.*
- bool [getDisplayHozSlice2Selection](#) ()  
*Access function for displayHozSlice2Selection property - refer to displayHozSlice2Selection property for details.*

- Access function for `displayHozSlice2Selection` property - refer to `displayHozSlice2Selection` property for details.
- void `setDisplayHozSlice3Selection` (bool displayHSliceSelection)  
Access function for `displayHozSlice3Selection` property - refer to `displayHozSlice3Selection` property for details.
- bool `getDisplayHozSlice3Selection` ()  
Access function for `displayHozSlice3Selection` property - refer to `displayHozSlice3Selection` property for details.
- void `setDisplayHozSlice4Selection` (bool displayHSliceSelection)  
Access function for `displayHozSlice4Selection` property - refer to `displayHozSlice4Selection` property for details.
- bool `getDisplayHozSlice4Selection` ()  
Access function for `displayHozSlice4Selection` property - refer to `displayHozSlice4Selection` property for details.
- void `setDisplayHozSlice5Selection` (bool displayHSliceSelection)  
Access function for `displayHozSlice5Selection` property - refer to `displayHozSlice5Selection` property for details.
- bool `getDisplayHozSlice5Selection` ()  
Access function for `displayHozSlice5Selection` property - refer to `displayHozSlice5Selection` property for details.
- void `setDisplayArea1Selection` (bool displayAreaSelection)  
Access function for `displayArea1Selection` property - refer to `displayArea1Selection` property for details.
- bool `getDisplayArea1Selection` ()  
Access function for `displayArea1Selection` property - refer to `displayArea1Selection` property for details.
- void `setDisplayArea2Selection` (bool displayAreaSelection)  
Access function for `displayArea2Selection` property - refer to `displayArea2Selection` property for details.
- bool `getDisplayArea2Selection` ()  
Access function for `displayArea2Selection` property - refer to `displayArea2Selection` property for details.
- void `setDisplayArea3Selection` (bool displayAreaSelection)  
Access function for `displayArea3Selection` property - refer to `displayArea3Selection` property for details.
- bool `getDisplayArea3Selection` ()  
Access function for `displayArea3Selection` property - refer to `displayArea3Selection` property for details.
- void `setDisplayArea4Selection` (bool displayAreaSelection)  
Access function for `displayArea4Selection` property - refer to `displayArea4Selection` property for details.
- bool `getDisplayArea4Selection` ()  
Access function for `displayArea4Selection` property - refer to `displayArea4Selection` property for details.
- void `setDisplayProfileSelection` (bool displayProfileSelection)  
Access function for `displayProfileSelection` property - refer to `displayProfileSelection` property for details.

- `bool getDisplayProfileSelection ()`  
*Access function for `displayProfileSelection` property - refer to `displayProfileSelection` property for details.*
- `void setDisplayTargetSelection (bool displayTargetSelection)`  
*Access function for `displayTargetSelection` property - refer to `displayTargetSelection` property for details.*
- `bool getDisplayTargetSelection ()`  
*Access function for `displayTargetSelection` property - refer to `displayTargetSelection` property for details.*
- `void setDisplayBeamSelection (bool displayBeamSelection)`  
*Access function for `displayBeamSelection` property - refer to `displayBeamSelection` property for details.*
- `bool getDisplayBeamSelection ()`  
*Access function for `displayBeamSelection` property - refer to `displayBeamSelection` property for details.*
- `void setDisplayEllipse (bool displayEllipse)`  
*Access function for `displayEllipse` property - refer to `displayEllipse` property for details.*
- `bool getDisplayEllipse ()`  
*Access function for `displayEllipse` property - refer to `displayEllipse` property for details.*
- `ellipseVariableDefinitions getEllipseVariableDefinition ()`  
*Access function for `ellipseVariableDefinition` property - refer to `ellipseVariableDefinition` property for details.*
- `void setEllipseVariableDefinition (ellipseVariableDefinitions def)`  
*Access function for `ellipseVariableDefinition` property - refer to `ellipseVariableDefinition` property for details.*
- `void setDisplayMarkups (bool displayMarkupsIn)`  
*Access function for `#displayMarkups` property - refer to `#displayMarkups` property for details.*
- `bool getDisplayMarkups ()`  
*Access function for `#displayMarkups` property - refer to `#displayMarkups` property for details.*
- `void setName (QString nameIn)`  
*Access function for `name` property - refer to `#name` property for details.*
- `QString getName ()`  
*Access function for `name` property - refer to `#name` property for details.*
- `void setProgram1 (QString program)`  
*Access function for `program1` property - refer to `program1` property for details.*
- `QString getProgram1 ()`  
*Access function for `program1` property - refer to `program1` property for details.*
- `void setProgram2 (QString program)`  
*Access function for `program2` property - refer to `program2` property for details.*
- `QString getProgram2 ()`  
*Access function for `program2` property - refer to `program2` property for details.*
- `void setArguments1 (QStringList arguments)`  
*Access function for `arguments1` property - refer to `arguments1` property for details.*

- `QStringList getArguments1 ()`  
*Access function for `arguments1` property - refer to `arguments1` property for details.*
- `void setArguments2 (QStringList arguments)`  
*Access function for `arguments2` property - refer to `arguments2` property for details.*
- `QStringList getArguments2 ()`  
*Access function for `arguments2` property - refer to `arguments2` property for details.*
- `void setProgramStartupOption1 (applicationLauncher::programStartupOptions programStartupOption)`  
*Access function for `programStartupOption1` property - refer to `programStartupOption1` property for details.*
- `applicationLauncher::programStartupOptions getProgramStartupOption1 ()`  
*Access function for `programStartupOption1` property - refer to `programStartupOption1` property for details.*
- `void setProgramStartupOption2 (applicationLauncher::programStartupOptions programStartupOption)`  
*Access function for `programStartupOption2` property - refer to `programStartupOption2` property for details.*
- `applicationLauncher::programStartupOptions getProgramStartupOption2 ()`  
*Access function for `programStartupOption2` property - refer to `programStartupOption2` property for details.*
- `QString getHozSlice1Legend ()`  
*Access function for `hozSlice1Legend` property - refer to `hozSlice1Legend` property for details.*
- `void setHozSlice1Legend (QString legend)`  
*Access function for `hozSlice1Legend` property - refer to `hozSlice1Legend` property for details.*
- `QString getHozSlice2Legend ()`  
*Access function for `hozSlice2Legend` property - refer to `hozSlice2Legend` property for details.*
- `void setHozSlice2Legend (QString legend)`  
*Access function for `hozSlice2Legend` property - refer to `hozSlice2Legend` property for details.*
- `QString getHozSlice3Legend ()`  
*Access function for `hozSlice3Legend` property - refer to `hozSlice3Legend` property for details.*
- `void setHozSlice3Legend (QString legend)`  
*Access function for `hozSlice3Legend` property - refer to `hozSlice3Legend` property for details.*
- `QString getHozSlice4Legend ()`  
*Access function for `hozSlice4Legend` property - refer to `hozSlice4Legend` property for details.*
- `void setHozSlice4Legend (QString legend)`  
*Access function for `hozSlice4Legend` property - refer to `hozSlice4Legend` property for details.*
- `QString getHozSlice5Legend ()`

- `void setHozSlice5Legend (QString legend)`  
*Access function for `hozSlice5Legend` property - refer to `hozSlice5Legend` property for details.*
- `QString getVertSlice1Legend ()`  
*Access function for `vertSlice1Legend` property - refer to `vertSlice1Legend` property for details.*
- `void setVertSlice1Legend (QString legend)`  
*Access function for `vertSlice1Legend` property - refer to `vertSlice1Legend` property for details.*
- `QString getVertSlice2Legend ()`  
*Access function for `vertSlice2Legend` property - refer to `vertSlice2Legend` property for details.*
- `void setVertSlice2Legend (QString legend)`  
*Access function for `vertSlice2Legend` property - refer to `vertSlice2Legend` property for details.*
- `QString getVertSlice3Legend ()`  
*Access function for `vertSlice3Legend` property - refer to `vertSlice3Legend` property for details.*
- `void setVertSlice3Legend (QString legend)`  
*Access function for `vertSlice3Legend` property - refer to `vertSlice3Legend` property for details.*
- `QString getVertSlice4Legend ()`  
*Access function for `vertSlice4Legend` property - refer to `vertSlice4Legend` property for details.*
- `void setVertSlice4Legend (QString legend)`  
*Access function for `vertSlice4Legend` property - refer to `vertSlice4Legend` property for details.*
- `QString getVertSlice5Legend ()`  
*Access function for `vertSlice5Legend` property - refer to `vertSlice5Legend` property for details.*
- `void setVertSlice5Legend (QString legend)`  
*Access function for `vertSlice5Legend` property - refer to `vertSlice5Legend` property for details.*
- `QString getprofileLegend ()`  
*Access function for `profileLegend` property - refer to `profileLegend` property for details.*
- `void setProfileLegend (QString legend)`  
*Access function for `profileLegend` property - refer to `profileLegend` property for details.*
- `QString getAreaSelection1Legend ()`  
*Access function for `areaSelection1Legend` property - refer to `areaSelection1Legend` property for details.*
- `void setAreaSelection1Legend (QString legend)`  
*Access function for `areaSelection1Legend` property - refer to `areaSelection1Legend` property for details.*
- `QString getAreaSelection2Legend ()`

- Access function for `areaSelection2Legend` property - refer to `areaSelection2Legend` property for details.
- void `setAreaSelection2Legend` (QString legend)
  - Access function for `areaSelection2Legend` property - refer to `areaSelection2Legend` property for details.
- QString `getAreaSelection3Legend` ()
  - Access function for `areaSelection3Legend` property - refer to `areaSelection3Legend` property for details.
- void `setAreaSelection3Legend` (QString legend)
  - Access function for `areaSelection3Legend` property - refer to `areaSelection3Legend` property for details.
- QString `getAreaSelection4Legend` ()
  - Access function for `areaSelection4Legend` property - refer to `areaSelection4Legend` property for details.
- void `setAreaSelection4Legend` (QString legend)
  - Access function for `areaSelection4Legend` property - refer to `areaSelection4Legend` property for details.
- QString `getTargetLegend` ()
  - Access function for `targetLegend` property - refer to `targetLegend` property for details.
- void `setTargetLegend` (QString legend)
  - Access function for `targetLegend` property - refer to `targetLegend` property for details.
- QString `getBeamLegend` ()
  - Access function for `beamLegend` property - refer to `beamLegend` property for details.
- void `setBeamLegend` (QString legend)
  - Access function for `beamLegend` property - refer to `beamLegend` property for details.
- QString `getEllipseLegend` ()
  - Access function for `ellipseLegend` property - refer to `ellipseLegend` property for details.
- void `setEllipseLegend` (QString legend)
  - Access function for `ellipseLegend` property - refer to `ellipseLegend` property for details.
- bool `getFullScreen` ()
  - Access function for `#fullScreen` property - refer to `#fullScreen` property for details.
- void `setFullScreen` (bool fullScreenIn)
  - Access function for `#fullScreen` property - refer to `#fullScreen` property for details.
- void `setSubstitutedUrl` (QString urlIn)
  - Access function for `URL` property - refer to `URL` property for data.
- QString `getSubstitutedUrl` ()
  - Access function for `URL` property - refer to `URL` property for data.
- void `setVariableNameSubstitutionsProperty` (QString variableNameSubstitutions)
  - Property access function for `variableSubstitutions` property. This has special behaviour to work well within designer.*
- QString `getVariableNameSubstitutionsProperty` ()
  - Property access function for `variableSubstitutions` property. This has special behaviour to work well within designer.*
- UserLevels `getUserLevelVisibilityProperty` ()

- `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.
- `DisplayAlarmStateOptions getDisplayAlarmStateOptionProperty ()`  
Access function for `displayAlarmStateOption` property - refer to `displayAlarmStateOption` property for details.
- `void setDisplayAlarmStateOptionProperty (DisplayAlarmStateOptions option)`  
Access function for `displayAlarmStateOption` property - refer to `displayAlarmStateOption` 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 setBitDepthProperty (unsigned int bitDepth)`  
Access function for `#bitDepth` property - refer to `#bitDepth` property for details.
- `unsigned int getBitDepthProperty ()`  
Access function for `#bitDepth` property - refer to `#bitDepth` property for details.
- `EllipseVariableDefinitions getEllipseVariableDefinitionProperty ()`  
Access function for `#EllipseVariableDefinition` property - refer to `#EllipseVariableDefinition` property for details.
- `void setEllipseVariableDefinitionProperty (EllipseVariableDefinitions variableUsage)`  
Access function for `EllipseVariableDefinitions` property - refer to `EllipseVariableDefinitions` property for details.
- `TargetOptions getTargetOptionProperty ()`  
Access function for `targetOption` property - refer to `targetOption` property for details.
- `void setTargetOptionProperty (TargetOptions option)`  
Access function for `targetOption` property - refer to `targetOption` property for details.
- `TargetOptions getBeamOptionProperty ()`  
Access function for `beamOption` property - refer to `beamOption` property for details.
- `void setBeamOptionProperty (TargetOptions option)`  
Access function for `beamOption` property - refer to `beamOption` 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.
- `void setProgramStartupOptionProperty1 (ProgramStartupOptionNames programStartupOption)`  
Access function for `#ProgramStartupOptionNames1` property - refer to `#ProgramStartupOptionNames1` property for details.
- `ProgramStartupOptionNames getProgramStartupOptionProperty1 ()`  
Access function for `#ProgramStartupOptionNames1` property - refer to `#ProgramStartupOptionNames1` property for details.
- `void setProgramStartupOptionProperty2 (ProgramStartupOptionNames programStartupOption)`  
Access function for `#ProgramStartupOptionNames2` property - refer to `#ProgramStartupOptionNames2` property for details.
- `ProgramStartupOptionNames getProgramStartupOptionProperty2 ()`  
Access function for `#ProgramStartupOptionNames2` property - refer to `#ProgramStartupOptionNames2` property for details.

## Protected Types

- enum `variableIndexes` {
   
`IMAGE_VARIABLE, FORMAT_VARIABLE, BIT_DEPTH_VARIABLE, WIDTH_VARIABLE,`
  
`HEIGHT_VARIABLE, NUM_DIMENSIONS_VARIABLE, DIMENSION_0_VARIABLE,`
  
`DIMENSION_1_VARIABLE,`
  
`DIMENSION_2_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, PROFILE_H1_VARIABLE, PROFILE_H1_THICKNESS_VARIABLE, PROFILE_H2_VARIABLE,`
  
`PROFILE_H2_THICKNESS_VARIABLE, PROFILE_H3_VARIABLE, PROFILE_H3_THICKNESS_VARIABLE, PROFILE_H4_VARIABLE,`
  
`PROFILE_H4_THICKNESS_VARIABLE, PROFILE_H5_VARIABLE, PROFILE_H5_THICKNESS_VARIABLE, PROFILE_V1_VARIABLE,`
  
`PROFILE_V1_THICKNESS_VARIABLE, PROFILE_V2_VARIABLE, PROFILE_V2_THICKNESS_VARIABLE, PROFILE_V3_VARIABLE,`
  
`PROFILE_V3_THICKNESS_VARIABLE, PROFILE_V4_VARIABLE, PROFILE_V4_THICKNESS_VARIABLE, PROFILE_V5_VARIABLE,`

```
PROFILE_V5_THICKNESS_VARIABLE, LINE_PROFILE_X1_VARIABLE, LINE_-
PROFILE_Y1_VARIABLE, LINE_PROFILE_X2_VARIABLE,
LINE_PROFILE_Y2_VARIABLE, LINE_PROFILE_THICKNESS_VARIABLE, PROFILE_-
H_ARRAY, PROFILE_V_ARRAY,
PROFILE_LINE_ARRAY, ELLIPSE_X_VARIABLE, ELLIPSE_Y_VARIABLE, ELLIPSE_-
W_VARIABLE,
ELLIPSE_H_VARIABLE, QEIMAGE_NUM_VARIABLES }
```

### Protected Member Functions

- void **establishConnection** (unsigned int variableIndex)
- void **redisplayAllMarkups** ()
- 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

- QEStringFormatting **stringFormatting**
- QEIntegerFormatting **integerFormatting**
- QEFloatingFormatting **floatingFormatting**
- **resizeOptions** **resizeOption**
- int **zoom**

*Zoom percentage. Used when #resizeOption is Zoom.*
- int **initialHozScrollPos**
- int **initialVertScrollPos**
- bool **displayButtonBar**

### Properties

- QString **imageVariable**
- QString **formatVariable**
- QString **bitDepthVariable**
- QString **widthVariable**
- QString **heightVariable**
- QString **dimensionsVariable**
- QString **dimension1Variable**
- QString **dimension2Variable**
- QString **dimension3Variable**

- 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 **profileHozVariable**
- QString profileHoz1Variable
- QString **profileHozThicknessVariable**
- QString profileHoz1ThicknessVariable
- QString profileHoz2Variable
- QString profileHoz2ThicknessVariable
- QString profileHoz3Variable
- QString profileHoz3ThicknessVariable
- QString profileHoz4Variable
- QString profileHoz4ThicknessVariable
- QString profileHoz5Variable
- QString profileHoz5ThicknessVariable
- QString **profileVertVariable**
- QString profileVert1Variable
- QString **profileVertThicknessVariable**
- QString profileVert1ThicknessVariable
- QString profileVert2Variable
- QString profileVert2ThicknessVariable
- QString profileVert3Variable
- QString profileVert3ThicknessVariable
- QString profileVert4Variable
- QString profileVert4ThicknessVariable

- QString `profileVert5Variable`
- QString `profileVert5ThicknessVariable`
- QString `lineProfileX1Variable`
- QString `lineProfileY1Variable`
- QString `lineProfileX2Variable`
- QString `lineProfileY2Variable`
- QString `lineProfileThicknessVariable`
- QString `profileHozArrayVariable`
- QString `profileVertArrayVariable`
- QString `lineProfileArrayVariable`
- QString `ellipseXVariable`
- QString `ellipseYVariable`
- QString `ellipseWVariable`
- QString `ellipseHVariable`
- QString `variableSubstitutions`
- bool `variableAsToolTip`
- bool `allowDrop`
- bool `visible`
- unsigned `int`
- QString `styleSheet`
- QString `defaultStyle`
- QString `userLevelUserStyle`
- QString `userLevelScientistStyle`
- QString `userLevelEngineerStyle`
- UserLevels `userLevelVisibility`
- UserLevels `userLevelEnabled`
- bool `displayAlarmState`
- DisplayAlarmStateOptions `displayAlarmStateOption`
- FormatOptions `formatOption`
- bool **`enableVertSliceSelection`**
- bool `enableVertSlice1Selection`
- bool `enableVertSlice2Selection`
- bool `enableVertSlice3Selection`
- bool `enableVertSlice4Selection`
- bool `enableVertSlice5Selection`
- bool **`enableHozSliceSelection`**
- bool `enableHozSlice1Selection`
- bool `enableHozSlice2Selection`
- bool `enableHozSlice3Selection`
- bool `enableHozSlice4Selection`
- bool `enableHozSlice5Selection`
- bool `enableProfileSelection`
- bool `enableArea1Selection`
- bool `enableArea2Selection`
- bool `enableArea3Selection`
- bool `enableArea4Selection`

- bool `enableTargetSelection`
- bool `enableBeamSelection`
- QString **hozSliceLegend**
  - QString `hozSlice1Legend`  
*Name of horizontal slice 1 markup.*
  - QString `hozSlice2Legend`  
*Name of horizontal slice 2 markup.*
  - QString `hozSlice3Legend`  
*Name of horizontal slice 3 markup.*
  - QString `hozSlice4Legend`  
*Name of horizontal slice 4 markup.*
  - QString `hozSlice5Legend`  
*Name of horizontal slice 5 markup.*
- QString **vertSliceLegend**
  - QString `vertSlice1Legend`  
*Name of vertical slice 1 markup.*
  - QString `vertSlice2Legend`  
*Name of vertical slice 2 markup.*
  - QString `vertSlice3Legend`  
*Name of vertical slice 3 markup.*
  - QString `vertSlice4Legend`  
*Name of vertical slice 4 markup.*
  - QString `vertSlice5Legend`  
*Name of vertical slice 5 markup.*
- QString **profileLegend**
  - QString `areaSelection1Legend`  
*Name of area selection 1 markup.*
  - QString `areaSelection2Legend`  
*Name of area selection 2 markup.*
  - QString `areaSelection3Legend`  
*Name of area selection 3 markup.*
  - QString `areaSelection4Legend`  
*Name of area selection 4 markup.*
- QString **targetLegend**
  - QString `beamLegend`  
*Name of beam markup.*
  - QString `ellipseLegend`  
*Name of ellipse markup.*
- bool `displayVertSliceSelection`
- bool `displayHozSliceSelection`
- bool `displayHozSlice1Selection`
- bool `displayHozSlice2Selection`

- bool `displayHozSlice3Selection`
- bool `displayHozSlice4Selection`
- bool `displayHozSlice5Selection`
- bool `displayProfileSelection`
- bool `displayArea1Selection`
- bool `displayArea2Selection`
- bool `displayArea3Selection`
- bool `displayArea4Selection`
- bool `displayTargetSelection`
- bool `displayBeamSelection`
- bool `displayEllipse`
- `EllipseVariableDefinitions ellipseVariableDefinition`  
*Definition of how ellipse variables are to be used.*
- `TargetOptions targetOption`  
*Definition of target markup options.*
- `TargetOptions beamOption`  
*Definition of beam markup options.*
- bool `displayCursorPixelInfo`
- bool `contrastReversal`
- bool `logBrightness`
- bool `showTime`
- bool `useFalseColour`
- QColor `vertSliceColor`
- QColor `vertSlice1Color`
- QColor `vertSlice2Color`
- QColor `vertSlice3Color`
- QColor `vertSlice4Color`
- QColor `vertSlice5Color`
- QColor `hozSliceColor`
- QColor `hozSlice1Color`
- QColor `hozSlice2Color`
- QColor `hozSlice3Color`
- QColor `hozSlice4Color`
- QColor `hozSlice5Color`
- QColor `profileColor`
- QColor `areaColor`
- QColor `beamColor`
- QColor `targetColor`
- QColor `timeColor`
- QColor `ellipseColor`
- `ResizeOptions resizeOption`
- `RotationOptions rotation`
- bool `verticalFlip`
- bool `horizontalFlip`
- int `initialHosScrollPos`
- bool `enableImageDisplayProperties`

*If true, the local Image Display Properties controls are displayed.*

- bool [enableRecording](#)  
*If true, the recording controls are displayed.*
- bool [autoBrightnessContrast](#)
- bool [externalControls](#)
- bool [briefInfoArea](#)
- QString [program1](#)
- QStringList [arguments1](#)
- [ProgramStartupOptionNames](#) [programStartupOption1](#)
- QString [program2](#)
- QStringList [arguments2](#)
- [ProgramStartupOptionNames](#) [programStartupOption2](#)
- QString [URL](#)

#### 9.64.1 Detailed Description

This class is a EPICS aware image widget. When image related variables are defined the image will be displayed. Many PVs may be defined to allow user interaction, such as selecting regions of interest. It is tightly integrated with the base class QEWidget which provides generic support such as macro substitutions, drag/drop, and standard properties.

#### 9.64.2 Member Enumeration Documentation

##### 9.64.2.1 enum QEImage::DisplayAlarmStateOptions

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

###### Enumerator:

**Never** Refer to DISPLAY\_ALARM\_STATE\_NEVER for details.

**Always** Refer to DISPLAY\_ALARM\_STATE\_ALWAYS for details.

**WhenInAlarm** Refer to DISPLAY\_ALARM\_STATE\_WHEN\_IN\_ALARM for details.

##### 9.64.2.2 enum QEImage::EllipseVariableDefinitions

User friendly enumerations for [ellipseVariableDefinition](#) property - refer to [ellipseVariableDefinition](#) property for details.

###### Enumerator:

**BoundingRectangle** Refer to BOUNDING\_RECTANGLE for details.

**CenterAndSize** Refer to CENTRE\_AND\_SIZE for details.

### 9.64.2.3 enum QEImage::ellipseVariableDefinitions

Options for the use of ellipse markup variables.

**Enumerator:**

**BOUNDING\_RECTANGLE** Variables define bounding rectangle of ellipse.

### 9.64.2.4 enum QEImage::FormatOptions

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

**Enumerator:**

**Mono** Grey scale.

**Bayer** Colour (Bayer Red Green)

**BayerGB** Colour (Bayer Green Blue)

**BayerBG** Colour (Bayer Blue Green)

**BayerGR** Colour (Bayer Green Red)

**BayerRG** Colour (Bayer Red Green)

**rgb1** Colour (24 bit RGB)

**rgb2** Colour (??? bit RGB)

**rgb3** Colour (??? bit RGB)

**yuv444** Colour (???)

**yuv422** Colour (???)

### 9.64.2.5 enum QEImage::ProgramStartupOptionNames

Startup options. Just run the command, run the command within a terminal, or display the output in QE message system.

**Enumerator:**

**None** Just run the program.

**Terminal** Run the program in a terminal (in Windows a command interpreter will also be started, so the program may be a built-in command like 'dir')

**LogOutput** Run the program, and log the output in the QE message system.

**StdOutput** Run the program, and send output to standard output and standard error.

#### 9.64.2.6 enum QEImage::ResizeOptions

User friendly enumerations for #resizeOption property

**Enumerator:**

**Zoom** Zoom to selected percentage.

**Fit** Zoom to fit the current window size.

#### 9.64.2.7 enum QEImage::resizeOptions

Image resize options

**Enumerator:**

**RESIZE\_OPTION\_ZOOM** Zoom to selected percentage.

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

#### 9.64.2.8 enum QEImage::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.

#### 9.64.2.9 enum QEImage::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\_VSLICE1** Select the vertical slice 1 point.

**SO\_VSLICE2** Select the vertical slice 2 point.

**SO\_VSLICE3** Select the vertical slice 3 point.

**SO\_VSLICE4** Select the vertical slice 4 point.

**SO\_VSLICE5** Select the vertical slice 5 point.

**SO\_HSLICE1** Select the horizontal slice 1 point.

***SO\_HSLICE2*** Select the horizontal slice 2 point.  
***SO\_HSLICE3*** Select the horizontal slice 3 point.  
***SO\_HSLICE4*** Select the horizontal slice 4 point.  
***SO\_HSLICE5*** Select the horizontal slice 5 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.

#### 9.64.2.10 enum QEImage::TargetOptions

User friendly enumerations for #targetOptions property - refer to #targetOptions property for details.

**Enumerator:**

***DottedFullCrosshair*** Refer to CROSSHAIR1 for details.  
***SolidSmallCrosshair*** Refer to CROSSHAIR2 for details.

#### 9.64.2.11 enum QEImage::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.

### 9.64.3 Constructor & Destructor Documentation

#### 9.64.3.1 QEImage::QEImage ( QWidget \* parent = 0 )

Create without a variable. Use setVariableName'n'Property() - where 'n' is a number from 0 to 40 - 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.

**9.64.3.2 QEImage::QEImage ( 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 consistant with other widgets, but will not result in an updating image as the width and height variables are required as a minimum.

**9.64.4 Member Function Documentation****9.64.4.1 void QEImage::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.

**9.64.4.2 void QEImage::setImageFile ( QString *name* ) [slot]**

!! memcpy will be more efficient.

**9.64.4.3 void QEImage::setManagedVisible ( bool *v* ) [inline, slot]**

Slot to set the visibility of a QE widget, taking into account the user level. Widget will be hidden if hidden by a call this slot, by will only be made visible by a call to this slot if the user level allows.

**9.64.5 Member Data Documentation****9.64.5.1 bool QEImage::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.

**9.64.5.2 int QEImage::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.

**9.64.6 Property Documentation****9.64.6.1 bool QEImage::allowDrop [read, write]**

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

**9.64.6.2 QColor QEImage::areaColor [read, write]**

Used to select the color of the area selection markups.

**9.64.6.3 QStringList QEImage::arguments1 [read, write]**

Arguments for program specified in the 'program1' property.

**9.64.6.4 QStringList QEImage::arguments2 [read, write]**

Arguments for program specified in the 'program2' property.

**9.64.6.5 bool QEImage::autoBrightnessContrast [read, write]**

If true, auto set local brightness and contrast when any area is selected. The brightness and contrast is set to use the full range of pixels in the selected area.

**9.64.6.6 QColor QEImage::beamColor [read, write]**

Used to select the color of the beam marker.

**9.64.6.7 QString QEImage::beamXVariable [read, write]**

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

**9.64.6.8 QString QEImage::beamYVariable [read, write]**

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

**9.64.6.9 QString QEImage::bitDepthVariable [read, write]**

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

**9.64.6.10 bool QEImage::briefInfoArea [read, write]**

If true, the information area will be brief (one row)

**9.64.6.11 QString QEImage::clippingHighVariable [read, write]**

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

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

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

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

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

**9.64.6.14 bool QEImage::contrastReversal [read, write]**

If true, the image will undergo contrast reversal.

**9.64.6.15 QString QEImage::defaultStyle [read, write]**

Style Sheet string to be applied before, i.e. lower priority than, any other style, e.g. alarm style and/or user level style. Default is an empty string.

**9.64.6.16 QString QEImage::dimension1Variable [read, write]**

EPICS variable name (CA PV). This variable is used to read the first area detector dimension of the image. If there are 2 dimensions, this will be the image width. If there are 3 dimensions, this will be the number of elements per pixel.

**9.64.6.17 QString QEImage::dimension2Variable [read, write]**

EPICS variable name (CA PV). This variable is used to read the second area detector dimension of the image. If there are 2 dimensions, this will be the image height. If there are 3 dimensions, this will be the image width.

**9.64.6.18 QString QEImage::dimension3Variable [read, write]**

EPICS variable name (CA PV). This variable is used to read the third area detector dimension of the image. If there are 3 dimensions, this will be the image height.

**9.64.6.19 QString QEImage::dimensionsVariable [read, write]**

EPICS variable name (CA PV). This variable is used to read the number of area detector dimensions of the image. If used, this will be 2 (one element per pixel arranged by width and height) or 3 (multiple elements per pixel arranged by pixel, width and height)

**9.64.6.20 bool QEImage::displayAlarmState [read, write]**

DEPRECATED. USE `displayAlarmStateOption` INSTEAD. If set (default) widget will indicate the alarm state of any variable data it is displaying. If clear widget will never indicate the alarm state of any variable data it is displaying. Typically the background colour is set to indicate the alarm state. Note, this property is included in the set of standard properties as it applies to most widgets. It will do nothing for widgets that don't display data.

**9.64.6.21 DisplayAlarmStateOptions QEImage::displayAlarmStateOption [read, write]**

If 'Always' (default) widget will indicate the alarm state of any variable data it is displaying, including 'No Alarm'. If 'Never' widget will never indicate the alarm state of any variable data it is displaying. If 'WhenInAlarm' widget only indicate the alarm state of any variable data it is displaying if it is 'in alarm'. Typically the background colour is set to indicate the alarm state. Note, this property is included in the set of standard properties as it applies to most widgets. It will do nothing for widgets that don't display data.

**9.64.6.22 bool QEImage::displayArea1Selection [read, write]**

If true, selected area 1 will be displayed on the image. Note, this property is ignored unless the `enableArea1Selection` property is true.

**9.64.6.23 bool QEImage::displayArea2Selection [read, write]**

If true, selected area 2 will be displayed on the image. Note, this property is ignored unless the `enableArea2Selection` property is true.

**9.64.6.24 bool QEImage::displayArea3Selection [read, write]**

If true, selected area 3 will be displayed on the image. Note, this property is ignored unless the `enableArea3Selection` property is true.

**9.64.6.25 bool QEImage::displayArea4Selection [read, write]**

If true, selected area 4 will be displayed on the image. Note, this property is ignored unless the `enableArea4Selection` property is true.

**9.64.6.26 bool QEImage::displayBeamSelection [read, write]**

If true, beam selection will be displayed on the image. Note, this property is ignored unless the [enableBeamSelection](#) property is true.

**9.64.6.27 bool QEImage::displayCursorPixelInfo [read, write]**

If true, an area will be presented under the image with textual information about the pixel under the cursor, and for other selections such as selected areas.

**9.64.6.28 bool QEImage::displayEllipse [read, write]**

If true, the ellipse markup will be displayed on the image.

**9.64.6.29 bool QEImage::displayHozSlice1Selection [read, write]**

If true, the selected horizontal slice will be displayed on the image. Note, this property is ignored unless the [enableHozSlice1Selection](#) property is true.

**9.64.6.30 bool QEImage::displayHozSlice2Selection [read, write]**

If true, the selected horizontal slice will be displayed on the image. Note, this property is ignored unless the [enableHozSlice2Selection](#) property is true.

**9.64.6.31 bool QEImage::displayHozSlice3Selection [read, write]**

If true, the selected horizontal slice will be displayed on the image. Note, this property is ignored unless the [enableHozSlice3Selection](#) property is true.

**9.64.6.32 bool QEImage::displayHozSlice4Selection [read, write]**

If true, the selected horizontal slice will be displayed on the image. Note, this property is ignored unless the [enableHozSlice4Selection](#) property is true.

**9.64.6.33 bool QEImage::displayHozSlice5Selection [read, write]**

If true, the selected horizontal slice will be displayed on the image. Note, this property is ignored unless the [enableHozSlice5Selection](#) property is true.

**9.64.6.34 bool QEImage::displayProfileSelection [read, write]**

If true, the selected arbitrary line will be displayed on the image. Note, this property is ignored unless the [enableProfileSelection](#) property is true.

**9.64.6.35 bool QEImage::displayTargetSelection [read, write]**

If true, target selection will be displayed on the image. Note, this property is ignored unless the [enableTargetSelection](#) property is true.

**9.64.6.36 bool QEImage::displayVertSliceSelection [read, write]**

If true, the selected vertical slice 1 will be displayed on the image. Note, this property is ignored unless the [enableVertSlice1Selection](#) property is true.

If true, the selected vertical slice 2 will be displayed on the image. Note, this property is ignored unless the [enableVertSlice2Selection](#) property is true.

If true, the selected vertical slice 3 will be displayed on the image. Note, this property is ignored unless the [enableVertSlice3Selection](#) property is true.

If true, the selected vertical slice 4 will be displayed on the image. Note, this property is ignored unless the [enableVertSlice4Selection](#) property is true.

If true, the selected vertical slice 5 will be displayed on the image. Note, this property is ignored unless the [enableVertSlice5Selection](#) property is true.

**9.64.6.37 QColor QEImage::ellipseColor [read, write]**

Used to select the color of the ellipse marker.

**9.64.6.38 QString QEImage::ellipseHVariable [read, write]**

EPICS variable name (CA PV). This variable is used to read an ellipse height

**9.64.6.39 QString QEImage::ellipseWVariable [read, write]**

EPICS variable name (CA PV). This variable is used to read an ellipse width.

**9.64.6.40 QString QEImage::ellipseXVariable [read, write]**

EPICS variable name (CA PV). This variable is used to read an ellipse X (center or top left corner of bounding rectangle depending on property [ellipseDefinition](#)).

**9.64.6.41 QString QEImage::ellipseYVariable [read, write]**

EPICS variable name (CA PV). This variable is used to read an ellipse Y (center or top left corner of bounding rectangle depending on property [ellipseDefinition](#)).

**9.64.6.42 bool QEImage::enableArea1Selection [read, write]**

If true, the user will be able to select area 1. These are used for selection of Region of Interests, and for zooming to area 1

**9.64.6.43 bool QEImage::enableArea2Selection [read, write]**

If true, the user will be able to select area 2. These are used for selection of Region of Interests, and for zooming to area 2

**9.64.6.44 bool QEImage::enableArea3Selection [read, write]**

If true, the user will be able to select area 3. These are used for selection of Region of Interests, and for zooming to area 3

**9.64.6.45 bool QEImage::enableArea4Selection [read, write]**

If true, the user will be able to select area 4. These are used for selection of Region of Interests, and for zooming to area 4

**9.64.6.46 bool QEImage::enableBeamSelection [read, write]**

If true, the user will be able to select points on the image to mark a beam position. This can be used for automatic beam positioning.

**9.64.6.47 bool QEImage::enableHozSlice1Selection [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, and write the position of the slice to the optional variable specified by the [profileHoz1Variable](#) property. The profile will only be presented to the user if [enableHozSlicePresentation](#) property is true.

**9.64.6.48 bool QEImage::enableHozSlice2Selection [read, write]**

If true, the option to select a second horizontal slice through the image will be available to the user. This will be used to write the position of the slice to the optional variable specified by the [profileHoz2Variable](#) property.

**9.64.6.49 bool QEImage::enableHozSlice3Selection [read, write]**

If true, the option to select a third horizontal slice through the image will be available to the user. This will be used to write the position of the slice to the optional variable specified by the [profileHoz3Variable](#) property.

**9.64.6.50 bool QEImage::enableHozSlice4Selection [read, write]**

If true, the option to select a fourth horizontal slice through the image will be available to the user. This will be used to write the position of the slice to the optional variable specified by the [profileHoz4Variable](#) property.

**9.64.6.51 bool QEImage::enableHozSlice5Selection [read, write]**

If true, the option to select a fifth horizontal slice through the image will be available to the user. This will be used to write the position of the slice to the optional variable specified by the [profileHoz5Variable](#) property.

**9.64.6.52 bool QEImage::enableProfileSelection [read, write]**

If true, the option to select an arbitrary line through any part of the image will be available to the user. This will be used to generate a pixel profile.

**9.64.6.53 bool QEImage::enableTargetSelection [read, write]**

If true, the user will be able to select points on the image to mark a target position. This can be used for automatic beam positioning.

**9.64.6.54 bool QEImage::enableVertSlice1Selection [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 horizontal pixel profile, and write the position of the slice to the optional variable specified by the [profileVert1Variable](#) property. The profile will only be presented to the user if #enableVertSlicePresentation property is true.

**9.64.6.55 bool QEImage::enableVertSlice2Selection [read, write]**

If true, the option to select a second vertical slice through the image will be available to the user. This will be used to write the position of the slice to the optional variable specified by the [profileVert2Variable](#) property.

**9.64.6.56 bool QEImage::enableVertSlice3Selection [read, write]**

If true, the option to select a third vertical slice through the image will be available to the user. This will be used to write the position of the slice to the optional variable specified by the [profileVert3Variable](#) property.

**9.64.6.57 bool QEImage::enableVertSlice4Selection [read, write]**

If true, the option to select a fourth vertical slice through the image will be available to the user. This will be used to write the position of the slice to the optional variable specified by the [profileVert4Variable](#) property.

**9.64.6.58 bool QEImage::enableVertSlice5Selection [read, write]**

If true, the option to select a fifth vertical slice through the image will be available to the user. This will be used to write the position of the slice to the optional variable specified by the [profileVert5Variable](#) property.

**9.64.6.59 bool QEImage::externalControls [read, write]**

If true, image controls and views such as brightness controls and profile plots are hosted by the application as dock windows, toolbars, etc. Refer to the [#ContainerProfile](#) class and the [#windowCustomisation](#) class to see how this class asks an application to act as a host.

**9.64.6.60 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.)

**9.64.6.61 QString QEImage::formatVariable [read, write]**

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

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

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

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

If true, flip image horizontally.

**9.64.6.64 QColor QEImage::hozSlice1Color [read, write]**

Used to select the color of the horizontal slice 1 markup.

**9.64.6.65 QColor QEImage::hozSlice2Color [read, write]**

Used to select the color of the horizontal slice 2 markup.

9.64.6.66 QColor QEImage::hozSlice3Color [read, write]

Used to select the color of the horizontal slice 3 markup.

9.64.6.67 QColor QEImage::hozSlice4Color [read, write]

Used to select the color of the horizontal slice 4 markup.

9.64.6.68 QColor QEImage::hozSlice5Color [read, write]

Used to select the color of the horizontal slice 5 markup.

9.64.6.69 QString QEImage::imageVariable [read, write]

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

9.64.6.70 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.

9.64.6.71 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.

Bit depth. Note, EPICS data type size will typically be adequate for the number of bits required (one byte for up to 8 bits, 2 bytes for up to 16 bits, etc), but can be larger (for example, 4 bytes for 24 bits) and may be larger than necessary (4 bytes for 8 bits).

9.64.6.72 QString QEImage::lineProfileArrayVariable [read, write]

EPICS variable name (CA PV). This variable is used to write the areadetector arbitrary line profile array.

9.64.6.73 QString QEImage::lineProfileThicknessVariable [read, write]

EPICS variable name (CA PV). This variable is used to write the areadetector arbitrary line profile end Y.

**9.64.6.74 QString QEImage::lineProfileX1Variable [read, write]**

EPICS variable name (CA PV). This variable is used to write the areadetector arbitrary line profile start X.

**9.64.6.75 QString QEImage::lineProfileX2Variable [read, write]**

EPICS variable name (CA PV). This variable is used to write the areadetector arbitrary line profile end X.

**9.64.6.76 QString QEImage::lineProfileY1Variable [read, write]**

EPICS variable name (CA PV). This variable is used to write the areadetector arbitrary line profile start Y.

**9.64.6.77 QString QEImage::lineProfileY2Variable [read, write]**

EPICS variable name (CA PV). This variable is used to write the areadetector arbitrary line profile end Y.

**9.64.6.78 bool QEImage::logBrightness [read, write]**

If true, the image will be displayed using a logarithmic brightness scale.

**9.64.6.79 QColor QEImage::profileColor [read, write]**

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

**9.64.6.80 QString QEImage::profileHoz1ThicknessVariable [read, write]**

EPICS variable name (CA PV). This variable is used to write the areadetector first horizontal profile thickness.

**9.64.6.81 QString QEImage::profileHoz1Variable [read, write]**

EPICS variable name (CA PV). This variable is used to write the areadetector first horizontal profile.

**9.64.6.82 QString QEImage::profileHoz2ThicknessVariable [read, write]**

EPICS variable name (CA PV). This variable is used to write the areadetector second horizontal profile thickness.

**9.64.6.83 QString QEImage::profileHoz2Variable [read, write]**

EPICS variable name (CA PV). This variable is used to write the areadetector second horizontal profile.

**9.64.6.84 QString QEImage::profileHoz3ThicknessVariable [read, write]**

EPICS variable name (CA PV). This variable is used to write the areadetector third horizontal profile thickness.

**9.64.6.85 QString QEImage::profileHoz3Variable [read, write]**

EPICS variable name (CA PV). This variable is used to write the areadetector third horizontal profile.

**9.64.6.86 QString QEImage::profileHoz4ThicknessVariable [read, write]**

EPICS variable name (CA PV). This variable is used to write the areadetector fourth horizontal profile thickness.

**9.64.6.87 QString QEImage::profileHoz4Variable [read, write]**

EPICS variable name (CA PV). This variable is used to write the areadetector fourth horizontal profile.

**9.64.6.88 QString QEImage::profileHoz5ThicknessVariable [read, write]**

EPICS variable name (CA PV). This variable is used to write the areadetector fifth horizontal profile thickness.

**9.64.6.89 QString QEImage::profileHoz5Variable [read, write]**

EPICS variable name (CA PV). This variable is used to write the areadetector fifth horizontal profile.

**9.64.6.90 QString QEImage::profileHozArrayVariable [read, write]**

EPICS variable name (CA PV). This variable is used to write the areadetector horizontal profile array.

**9.64.6.91 QString QEImage::profileVert1ThicknessVariable [read, write]**

EPICS variable name (CA PV). This variable is used to write the areadetector first vertical profile.

**9.64.6.92 QString QEImage::profileVert1Variable [read, write]**

EPICS variable name (CA PV). This variable is used to write the areadetector first vertical profile.

**9.64.6.93 QString QEImage::profileVert2ThicknessVariable [read, write]**

EPICS variable name (CA PV). This variable is used to write the areadetector second vertical profile.

**9.64.6.94 QString QEImage::profileVert2Variable [read, write]**

EPICS variable name (CA PV). This variable is used to write the areadetector second vertical profile.

**9.64.6.95 QString QEImage::profileVert3ThicknessVariable [read, write]**

EPICS variable name (CA PV). This variable is used to write the areadetector third vertical profile.

**9.64.6.96 QString QEImage::profileVert3Variable [read, write]**

EPICS variable name (CA PV). This variable is used to write the areadetector third vertical profile.

**9.64.6.97 QString QEImage::profileVert4ThicknessVariable [read, write]**

EPICS variable name (CA PV). This variable is used to write the areadetector fourth vertical profile.

**9.64.6.98 QString QEImage::profileVert4Variable [read, write]**

EPICS variable name (CA PV). This variable is used to write the areadetector fourth vertical profile.

**9.64.6.99 QString QEImage::profileVert5ThicknessVariable [read, write]**

EPICS variable name (CA PV). This variable is used to write the areadetector fifth vertical profile.

**9.64.6.100 QString QEImage::profileVert5Variable [read, write]**

EPICS variable name (CA PV). This variable is used to write the areadetector fifth vertical profile.

**9.64.6.101 QString QEImage::profileVertArrayVariable [read, write]**

EPICS variable name (CA PV). This variable is used to write the areadetector vertical profile array.

**9.64.6.102 QString QEImage::program1 [read, write]**

Program to run when a request is made to pass on the current image to the first external application. No attempt to run a program is made if this property is empty. Example: paint.exe

**9.64.6.103 QString QEImage::program2 [read, write]**

Program to run when a request is made to pass on the current image to the second external application. No attempt to run a program is made if this property is empty. Example: paint.exe

**9.64.6.104 ProgramStartupOptionNames QEImage::programStartupOption1 [read, write]**

Startup options for the program specified in the 'program1' property. Just run the command, run the command within a terminal, or display the output in QE message system.

**9.64.6.105 ProgramStartupOptionNames QEImage::programStartupOption2 [read, write]**

Startup options for the program specified in the 'program2' property. Just run the command, run the command within a terminal, or display the output in QE message system.

**9.64.6.106 QString QEImage::regionOfInterest1HVariable [read, write]**

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

**9.64.6.107 QString QEImage::regionOfInterest1WVariable [read, write]**

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

**9.64.6.108 QString QEImage::regionOfInterest1XVariable [read, write]**

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

**9.64.6.109 QString QEImage::regionOfInterest1YVariable [read, write]**

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

**9.64.6.119 QString QEImage::regionOfInterest4WVariable [read, write]**

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

**9.64.6.120 QString QEImage::regionOfInterest4XVariable [read, write]**

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

**9.64.6.121 QString QEImage::regionOfInterest4YVariable [read, write]**

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

**9.64.6.122 ResizeOptions QEImage::resizeOption [read, write]**

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

**9.64.6.123 RotationOptions QEImage::rotation [read, write]**

Image rotation option.

**9.64.6.124 bool QEImage::showTime [read, write]**

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

**9.64.6.125 QString QEImage::styleSheet [read, write]**

Hide style sheet from designer as style calculation by the styleManager and not directly setable per se. This also stops transient styles being saved to the ui file.

**9.64.6.126 QColor QEImage::targetColor [read, write]**

Used to select the color of the target marker.

**9.64.6.127 QString QEImage::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.

**9.64.6.128 QString QEImage::targetXVariable [read, write]**

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

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

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

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

Used to select the color of the timestamp.

**9.64.6.131 QString QEImage::URL [read, write]**

MPEG stream URL. If this is specified, this will be used as the source of the image in preference to variables (variables defining the image data, width, and height will be ignored)

**9.64.6.132 bool QEImage::useFalseColour [read, write]**

If true, the apply false colour to the image.

**9.64.6.133 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 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'.

**9.64.6.134 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.

**9.64.6.135 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.

**9.64.6.136 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.

**9.64.6.137 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 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'.

**9.64.6.138 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.

**9.64.6.139 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.

9.64.6.140 `bool QEImage::verticalFlip [read, write]`

If true, flip image vertically.

9.64.6.141 `QColor QEImage::vertSlice1Color [read, write]`

Used to select the color of the vertical slice 1 markup.

9.64.6.142 `QColor QEImage::vertSlice2Color [read, write]`

Used to select the color of the vertical slice 2 markup.

9.64.6.143 `QColor QEImage::vertSlice3Color [read, write]`

Used to select the color of the vertical slice 3 markup.

9.64.6.144 `QColor QEImage::vertSlice4Color [read, write]`

Used to select the color of the vertical slice 4 markup.

9.64.6.145 `QColor QEImage::vertSlice5Color [read, write]`

Used to select the color of the vertical slice 5 markup.

9.64.6.146 `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.

9.64.6.147 `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:

- /tmp/epicsqt/trunk/framework/widgets/QEImage/QEImage.h
- /tmp/epicsqt/trunk/framework/widgets/QEImage/QEImage.cpp

## 9.65 QEImageMarkupThickness Class Reference

### Public Member Functions

- **QEImageMarkupThickness** (QWidget \*parent=0)
- void **setThickness** (unsigned int thicknessIn)
- unsigned int **getThickness** ()

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

- /tmp/epicsqt/trunk/framework/widgets/QEImage/QEImageMarkupThickness.h
- /tmp/epicsqt/trunk/framework/widgets/QEImage/QEImageMarkupThickness.cpp

## 9.66 QEImageOptionsDialog Class Reference

### Signals

- void **optionChange** (imageContextMenu::imageContextMenuOptions option, bool checked)

### Public Member Functions

- **QEImageOptionsDialog** (QWidget \*parent=0)
- void **initialise** ()
- void **optionSet** (imageContextMenu::imageContextMenuOptions option, bool checked)
- bool **optionGet** (imageContextMenu::imageContextMenuOptions option)

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

- /tmp/epicsqt/trunk/framework/widgets/QEImage/QEImageOptionsDialog.h
- /tmp/epicsqt/trunk/framework/widgets/QEImage/QEImageOptionsDialog.cpp

## 9.67 QELabel Class Reference

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

### Public Types

- enum **updateOptions** { **UPDATE\_TEXT**, **UPDATE\_PIXMAP** }
- enum **UserLevels** { **User** = userLevelTypes::USERLEVEL\_USER, **Scientist** = userLevelTypes::USERLEVEL\_SCIENTIST, **Engineer** = userLevelTypes::USERLEVEL\_ENGINEER }
- enum **DisplayAlarmStateOptions** { **Never** = standardProperties::DISPLAY\_ALARM\_STATE\_NEVER, **Always** = standardProperties::DISPLAY\_ALARM\_STATE\_ALWAYS, **WhenInAlarm** = standardProperties::DISPLAY\_ALARM\_STATE\_WHEN\_IN\_ALARM }

- 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 **Separators** { **NoSeparator** = QQStringFormatting::SEPARATOR\_NONE, **Comma** = QQStringFormatting::SEPARATOR\_COMMA, **Underscore** = QQStringFormatting::SEPARATOR\_UNDERSCORE, **Space** = QQStringFormatting::SEPARATOR\_SPACE }
  - 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** = QELabel::UPDATE\_TEXT, **Picture** = QELabel::UPDATE\_PIXMAP }
- User friendly enumerations for updateOption property - refer to [QELabel::updateOptions](#) for details.*

## Public Slots

- void **setDefaultStyle** (const QString &style)
   
*Update the default style applied to this widget.*
- void **setManagedVisible** (bool v)

## Signals

- void **dbValueChanged** (const QString &out)
- void **dbValueChanged** (const int &out)
- void **dbValueChanged** (const long &out)
- void **dbValueChanged** (const qlonglong &out)
- void **dbValueChanged** (const double &out)
- void **dbValueChanged** (const bool &out)
- void **dbConnectionChanged** (const bool &isConnected)
   
*Sent when the widget state updated following a channel connection change.*
- 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)
- void **setVariableNameProperty** (QString variableName)

*Property access function for `variable` property. This has special behaviour to work well within designer.*

- `QString getVariableNameProperty ()`  
*Property access function for `variable` property. This has special behaviour to work well within designer.*
- `void setVariableNameSubstitutionsProperty (QString variableNameSubstitutions)`

*Property access function for `variableSubstitutions` property. This has special behaviour to work well within designer.*

- `QString getVariableNameSubstitutionsProperty ()`  
*Property access function for `variableSubstitutions` property. This has special behaviour to work well within designer.*
- `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.*
- `DisplayAlarmStateOptions getDisplayAlarmStateOptionProperty ()`  
*Access function for `displayAlarmStateOption` property - refer to `displayAlarmStateOption` property for details.*
- `void setDisplayAlarmStateOptionProperty (DisplayAlarmStateOptions option)`  
*Access function for `displayAlarmStateOption` property - refer to `displayAlarmStateOption` property for details.*
- `Formats getFormatProperty (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 setSeparatorProperty (const Separators notation)`  
*Access function for `separator` property - refer to `separator` property for details.*
- `Separators getSeparatorProperty () const`  
*Access function for `separator` property - refer to `separator` 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 allowDrop`
- `bool visible`
- `unsigned int`
- `QString styleSheet`
- `QString defaultStyle`
- `QString userLevelUserStyle`
- `QString userLevelScientistStyle`
- `QString userLevelEngineerStyle`
- `UserLevels userLevelVisibility`
- `UserLevels userLevelEnabled`
- `bool displayAlarmState`
- `DisplayAlarmStateOptions displayAlarmStateOption`
- `int precision`
- `bool useDbPrecision`
- `bool leadingZero`
- `bool trailingZeros`
- `bool addUnits`
- `QString localEnumeration`
- `Formats format`
- `int radix`
- `Separators separator`
- `Notations notation`
- `ArrayActions arrayAction`
- `QString text`
- `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 `QQStringFormatting::arrayActions` for details.

##### Enumerator:

**Append** Refer to `QQStringFormatting::APPEND` for details.

**Ascii** Refer to `QQStringFormatting::ASCII` for details.

**Index** Refer to `QQStringFormatting::INDEX` for details.

#### 9.67.2.2 enum QELabel::DisplayAlarmStateOptions

User friendly enumerations for `displayAlarmStateOption` property - refer to `displayAlarmStateOption` property and `displayAlarmStateOptions` enumeration for details.

##### Enumerator:

**Never** Refer to `DISPLAY_ALARM_STATE_NEVER` for details.

**Always** Refer to `DISPLAY_ALARM_STATE_ALWAYS` for details.

**WhenInAlarm** Refer to `DISPLAY_ALARM_STATE_WHEN_IN_ALARM` for details.

#### 9.67.2.3 enum QELabel::Formats

User friendly enumerations for format property - refer to `QQStringFormatting::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.

#### 9.67.2.4 enum QELabel::Notations

User friendly enumerations for notation property - refer to `QQStringFormatting::notations` for details.

##### Enumerator:

**Fixed** Refer to `QQStringFormatting::NOTATION_FIXED` for details.

**Scientific** Refer to `QQStringFormatting::NOTATION_SCIENTIFIC` for details.

**Automatic** Refer to `QQStringFormatting::NOTATION_AUTOMATIC` for details.

### 9.67.2.5 enum QELabel::Separators

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

#### Enumerator:

**NoSeparator** Use no separator.

**Comma** Use ',' as separator.

**Underscore** Use '\_' as separator.

**Space** Use ' ' as separator.

### 9.67.2.6 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.

### 9.67.2.7 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.

### 9.67.2.8 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::setManagedVisible ( bool v ) [inline, slot]`

Slot to set the visibility of a QE widget, taking into account the user level. Widget will be hidden if hidden by a call this slot, by will only be made visible by a call to this slot if the user level allows.

### 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.

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 **QString QELabel::defaultStyle** [read, write]

Style Sheet string to be applied before, i.e. lower priority than, any other style, e.g. alarm style and/or user level style. Default is an empty string.

#### 9.67.5.5 **bool QELabel::displayAlarmState** [read, write]

DEPRECATED. USE displayAlarmStateOption INSTEAD. If set (default) widget will indicate the alarm state of any variable data it is displaying. If clear widget will never indicate the alarm state of any variable data it is displaying. Typically the background colour is set to indicate the alarm state. Note, this property is included in the set of standard properties as it applies to most widgets. It will do nothing for widgets that don't display data.

#### 9.67.5.6 **DisplayAlarmStateOptions QELabel::displayAlarmStateOption** [read, write]

If 'Always' (default) widget will indicate the alarm state of any variable data it is displaying, including 'No Alarm'. If 'Never' widget will never indicate the alarm state of any variable data it is displaying. If 'WhenInAlarm' widget only indicate the alarm state of any variable data it is displaying if it is 'in alarm'. Typically the background colour is set to indicate the alarm state. Note, this property is included in the set of standard properties as it applies to most widgets. It will do nothing for widgets that don't display data.

#### 9.67.5.7 **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).

#### 9.67.5.8 **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.

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.

#### 9.67.5.9 bool QELabel::leadingZero [read, write]

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

#### 9.67.5.10 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)  
 >= Greater 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"

#### 9.67.5.11 Notations QELabel::notation [read, write]

Notation used for numerical formatting. Default is fixed.

#### 9.67.5.12 QPixmap QELabel::pixmap0 [read, write]

Pixmap displayed when updateOption property is 'Picture' and data is interpreted as 0.

**9.67.5.13 QPixmap QELabel:: pixmap1 [read, write]**

Pixmap displayed when updateOption property is 'Picture' and data is interpreted as 1.

**9.67.5.14 QPixmap QELabel:: pixmap2 [read, write]**

Pixmap displayed when updateOption property is 'Picture' and data is interpreted as 2.

**9.67.5.15 QPixmap QELabel:: pixmap3 [read, write]**

Pixmap displayed when updateOption property is 'Picture' and data is interpreted as 3.

**9.67.5.16 QPixmap QELabel:: pixmap4 [read, write]**

Pixmap displayed when updateOption property is 'Picture' and data is interpreted as 4.

**9.67.5.17 QPixmap QELabel:: pixmap5 [read, write]**

Pixmap displayed when updateOption property is 'Picture' and data is interpreted as 5.

**9.67.5.18 QPixmap QELabel:: pixmap6 [read, write]**

Pixmap displayed when updateOption property is 'Picture' and data is interpreted as 6.

**9.67.5.19 QPixmap QELabel:: pixmap7 [read, write]**

Pixmap displayed when updateOption property is 'Picture' and data is interpreted as 7.

**9.67.5.20 int QELabel:: precision [read, write]**

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

**9.67.5.21 int QELabel:: radix [read, write]**

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

**9.67.5.22 Separators QELabel::separator [read, write]**

Separators used for integer and fixed point formatting. Default is None.

**9.67.5.23 QString QELabel::styleSheet [read, write]**

Hide style sheet from designer as style calculation by the styleManager and not directly setable per se. This also stops transient styles being saved to the ui file.

**9.67.5.24 bool QELabel::trailingZeros [read, write]**

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

**9.67.5.25 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.

**9.67.5.26 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.

**9.67.5.27 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 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'.

**9.67.5.28 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.

**9.67.5.29 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.

**9.67.5.30 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.

**9.67.5.31 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 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'.

**9.67.5.32 QString QELabel::variable [read, write]**

EPICS variable name (CA PV)

**9.67.5.33 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.

**9.67.5.34 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.

### 9.67.5.35 bool QELabel::visible [read, write]

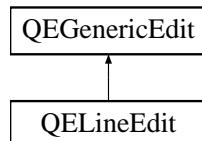
Display the widget. Default is true. Setting this property false is usefull if widget is only used to provide a signal - for example, when supplying data to a [QELink](#) widget. Note, when false the widget will still be visible in Qt Designer.

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

- /tmp/epicsqt/trunk/framework/widgets/QELabel/QELabel.h
- /tmp/epicsqt/trunk/framework/widgets/QELabel/QELabel.cpp

## 9.68 QELineEdit Class Reference

Inheritance diagram for QELineEdit:



### Public Types

- 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 [Separators](#) { [NoSeparator](#) = [QStringFormatting::SEPARATOR\\_NONE](#), [Comma](#) = [QStringFormatting::SEPARATOR\\_COMMA](#), [Underscore](#) = [QStringFormatting::SEPARATOR\\_UNDERSCORE](#), [Space](#) = [QStringFormatting::SEPARATOR\\_SPACE](#) }
- 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](#) }

### Signals

- void [dbValueChanged](#) (const QString &out)
- void [dbValueChanged](#) (const int &out)
- void [dbValueChanged](#) (const long &out)
- void [dbValueChanged](#) (const qlonglong &out)
- void [dbValueChanged](#) (const double &out)

- void **dbValueChanged** (const bool &out)
- void **dbConnectionChanged** (const bool &isConnected)
 

*Sent when the widget state updated following a channel connection change.*
- 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

- 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 **setSeparatorProperty** (const Separators notation)
 

*Access function for [separator](#) property - refer to [separator](#) property for details.*
- Separators **getSeparatorProperty** () const
 

*Access function for [separator](#) property - refer to [separator](#) 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)

## Properties

- int **precision**
- bool **useDbPrecision**
- bool **leadingZero**
- bool **trailingZeros**
- bool **addUnits**
- QString **localEnumeration**
- Formats **format**
- int **radix**
- Separators **separator**
- Notations **notation**
- ArrayActions **arrayAction**
- unsigned **int**

### 9.68.1 Member Enumeration Documentation

#### 9.68.1.1 enum QELineEdit::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.

#### 9.68.1.2 enum QELineEdit::Formats

User friendly enumerations for format property - refer to QEStringFormatting::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.

#### 9.68.1.3 enum QELineEdit::Notations

User friendly enumerations for notation property - refer to QEStringFormatting::notations for details.

##### Enumerator:

**Fixed** Refer to QEStringFormatting::NOTATION\_FIXED for details.

**Scientific** Refer to QEStringFormatting::NOTATION\_SCIENTIFIC for details.

**Automatic** Refer to QEStringFormatting::NOTATION\_AUTOMATIC for details.

#### 9.68.1.4 enum QELineEdit::Separators

User friendly enumerations for separator property - refer to QEStringFormatting::formats for details.

##### Enumerator:

**NoSeparator** Use no separator.

**Comma** Use ',' as separator.

**Underscore** Use '\_' as separator.

**Space** Use '' as separator.

## 9.68.2 Constructor & Destructor Documentation

### 9.68.2.1 QELlineEdit::QELlineEdit ( QWidget \* *parent* = 0 )

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

### 9.68.2.2 QELlineEdit::QELlineEdit ( 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 QELlineEdit::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.

## 9.68.4 Property Documentation

### 9.68.4.1 bool QELlineEdit::addUnits [read, write]

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

### 9.68.4.2 ArrayActions QELlineEdit::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'.

---

**9.68.4.3 Formats QELineEdit::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).

**9.68.4.4 unsigned QELineEdit::int [read, write]**

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.

Reimplemented from [QEGenericEdit](#).

**9.68.4.5 bool QELineEdit::leadingZero [read, write]**

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

**9.68.4.6 QString QELineEdit::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"

**9.68.4.7 Notations QELineEdit::notation [read, write]**

Notation used for numerical formatting. Default is fixed.

**9.68.4.8 int QELineEdit::precision [read, write]**

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

**9.68.4.9 int QELineEdit::radix [read, write]**

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

**9.68.4.10 Separators QELineEdit::separator [read, write]**

Separators used for integer and fixed point formatting. Default is None.

**9.68.4.11 bool QELineEdit::trailingZeros [read, write]**

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

**9.68.4.12 bool QELineEdit::useDbPrecision [read, write]**

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

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

- /tmp/epicsqt/trunk/framework/widgets/QELineEdit/QELineEdit.h
- /tmp/epicsqt/trunk/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:

- `/tmp/epicsqt/trunk/framework/widgets/QELLineEdit/QELLineEditManager.h`

## 9.70 QELink Class Reference

### 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 int &in)`
- `void in (const long &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 int &out)`
- `void out (const long &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 **setConditionProperty** (ConditionNames condition)
- ConditionNames **getConditionProperty** ()

## Protected Attributes

- conditions **condition**
- QVariant **comparisonValue**
- bool **signalTrue**
- bool **signalFalse**
- QVariant **outTrueValue**
- QVariant **outFalseValue**

## Properties

- ConditionNames **condition**
- QString **comparisonValue**
- QString **outTrueValue**
- QString **outFalseValue**
- bool **runVisible**

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

- /tmp/epicsqt/trunk/framework/widgets/QELink/QELink.h
- /tmp/epicsqt/trunk/framework/widgets/QELink/QELink.cpp

## 9.71 QELog Class Reference

### Public Types

- enum **optionsLayoutProperty** { **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 }
- enum **UserLevels** { **User** = userLevelTypes::USERLEVEL\_USER, **Scientist** = userLevelTypes::USERLEVEL\_SCIENTIST, **Engineer** = userLevelTypes::USERLEVEL\_ENGINEER }
- enum **DisplayAlarmStateOptions** { **Never** = standardProperties::DISPLAY\_ALARM\_STATE\_NEVER, **Always** = standardProperties::DISPLAY\_ALARM\_STATE\_ALWAYS, **WhenInAlarm** = standardProperties::DISPLAY\_ALARM\_STATE\_WHEN\_IN\_ALARM }

### Public Slots

- void **setManagedVisible** (bool v)

### Public Member Functions

- **QELog** (QWidget \*pParent=0)
- 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 **setOptionsLayout** (int pValue)
- int **getOptionsLayout** ()
- 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 **setOptionsLayoutProperty** (optionsLayoutProperty pOptionsLayout)
- optionsLayoutProperty **getOptionsLayoutProperty** ()
- MessageFilterOptions **getMessageFormFilter** ()
- void **setMessageFormFilter** (MessageFilterOptions messageFormFilter)
- MessageFilterOptions **getMessageSourceFilter** ()
- void **setMessageSourceFilter** (MessageFilterOptions messageSourceFilter)
- 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.*
- DisplayAlarmStateOptions **getDisplayAlarmStateOptionProperty** ()  
*Access function for displayAlarmStateOption property - refer to displayAlarmStateOption property for details.*
- void **setDisplayAlarmStateOptionProperty** (DisplayAlarmStateOptions option)  
*Access function for displayAlarmStateOption property - refer to displayAlarmStateOption property for details.*

## Protected Attributes

- **\_QTableWidgetLog** \* qTableWidgetLog
- QCheckBox \* **qCheckBoxInfoMessage**
- QCheckBox \* **qCheckBoxWarningMessage**
- QCheckBox \* **qCheckBoxErrorMessage**
- QPushButton \* **qPushButtonClear**
- QPushButton \* **qPushButtonSave**
- QColor **qColorInfo**
- QColor **qColorWarning**
- QColor **qColorError**
- bool **scrollToBottom**
- int **optionsLayout**

## Properties

- bool **showColumnTime**
- bool **showColumnType**
- bool **showColumnMessage**
- bool **showMessageFilter**
- bool **showClear**
- bool **showSave**
- optionsLayoutProperty **optionsLayout**
- QColor **infoColor**
- QColor **warningColor**
- QColor **errorColor**
- MessageFilterOptions **messageFormFilter**
- MessageFilterOptions **messageSourceFilter**
- unsigned **int**
- bool **variableAsToolTip**
- bool **allowDrop**
- bool **visible**
- QString **styleSheet**
- QString **defaultStyle**
- QString **userLevelUserStyle**
- QString **userLevelScientistStyle**
- QString **userLevelEngineerStyle**
- **UserLevels userLevelVisibility**
- **UserLevels userLevelEnabled**
- bool **displayAlarmState**
- **DisplayAlarmStateOptions displayAlarmStateOption**

### 9.71.1 Member Enumeration Documentation

#### 9.71.1.1 enum QELog::DisplayAlarmStateOptions

User friendly enumerations for **displayAlarmStateOption** property - refer to **displayAlarmStateOption** property and **displayAlarmStateOptions** enumeration for details.

##### Enumerator:

**Never** Refer to DISPLAY\_ALARM\_STATE\_NEVER for details.

**Always** Refer to DISPLAY\_ALARM\_STATE\_ALWAYS for details.

**WhenInAlarm** Refer to DISPLAY\_ALARM\_STATE\_WHEN\_IN\_ALARM for details.

### 9.71.1.2 enum QELog::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.

## 9.71.2 Member Function Documentation

### 9.71.2.1 void QELog::setManagedVisible ( bool v ) [inline, slot]

Slot to set the visibility of a QE widget, taking into account the user level. Widget will be hidden if hidden by a call this slot, by will only be made visible by a call to this slot if the user level allows.

## 9.71.3 Property Documentation

### 9.71.3.1 bool QELog::allowDrop [read, write]

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

### 9.71.3.2 QString QELog::defaultStyle [read, write]

Style Sheet string to be applied before, i.e. lower priority than, any other style, e.g. alarm style and/or user level style. Default is an empty string.

### 9.71.3.3 bool QELog::displayAlarmState [read, write]

DEPRECATED. USE `displayAlarmStateOption` INSTEAD. If set (default) widget will indicate the alarm state of any variable data it is displaying. If clear widget will never indicate the alarm state of any variable data it is displaying. Typically the background colour is set to indicate the alarm state. Note, this property is included in the set of standard properties as it applies to most widgets. It will do nothing for widgets that don't display data.

### 9.71.3.4 DisplayAlarmStateOptions QELog::displayAlarmStateOption [read, write]

If 'Always' (default) widget will indicate the alarm state of any variable data it is displaying, including 'No Alarm'. If 'Never' widget will never indicate the alarm state of any

variable data it is displaying. If 'WhenInAlarm' widget only indicate the alarm state of any variable data it is displaying if it is 'in alarm'. Typically the background colour is set to indicate the alarm state. Note, this property is included in the set of standard properties as it applies to most widgets. It will do nothing for widgets that don't display data.

#### 9.71.3.5 `unsigned QELog::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.

#### 9.71.3.6 `QString QELog::styleSheet` [read, write]

Hide style sheet from designer as style calculation by the styleManager and not directly setable per se. This also stops transient styles being saved to the ui file.

#### 9.71.3.7 `UserLevels QELog::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'.

#### 9.71.3.8 `QString QELog::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.

#### 9.71.3.9 `QString QELog::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.

**9.71.3.10 QString QELog::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.

**9.71.3.11 UserLevels QELog::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'.

**9.71.3.12 bool QELog::variableAsToolTip [read, write]**

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

**9.71.3.13 bool QELog::visible [read, write]**

Display the widget. Default is true. Setting this property false is useful 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:

- /tmp/epicsqt/trunk/framework/widgets/QELog/QELog.h
- /tmp/epicsqt/trunk/framework/widgets/QELog/QELog.cpp

## 9.72 QELogin Class Reference

### Signals

- void **login** ()

### Public Member Functions

- **QELogin** (QWidget \*pParent=0)
- bool **login** (userLevelTypes::userLevels level, QString password)
- QString **getPriorityUserPassword** ()

- `QString getPriorityScientistPassword ()`
- `QString getPriorityEngineerPassword ()`
- `void setUserPassword (QString pValue)`
- `QString getUserPassword ()`
- `void setScientistPassword (QString pValue)`
- `QString getScientistPassword ()`
- `void setEngineerPassword (QString pValue)`
- `QString getEngineerPassword ()`
- `void setCompactStyle (bool compactStyle)`
- `bool getCompactStyle ()`
- `void setStatusOnly (bool statusOnlyIn)`
- `bool getStatusOnly ()`
- `QString getUserTypeName (userLevelTypes::userLevels type)`

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

- `/tmp/epicsqt/trunk/framework/widgets/QELogin/QELogin.h`
- `/tmp/epicsqt/trunk/framework/widgets/QELogin/QELogin.cpp`

## 9.73 QELoginDialog Class Reference

### Public Member Functions

- `QELoginDialog (QELogin *ownerIn)`

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

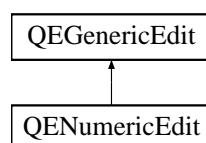
- `/tmp/epicsqt/trunk/framework/widgets/QELogin/QELogin.h`
- `/tmp/epicsqt/trunk/framework/widgets/QELogin/QELogin.cpp`

## 9.74 QENumericEdit Class Reference

The `QENumericEdit` class This class is similar to `QLineEdit` (both of which are derived from `QLineEdit`). However this class is tailored specifically for editing numerical values.

```
#include <QENumericEdit.h>
```

Inheritance diagram for QENumericEdit:



## Signals

- void **dbValueChanged** (const QString &out)
- void **dbValueChanged** (const int &out)
- void **dbValueChanged** (const long &out)
- void **dbValueChanged** (const qlonglong &out)
- void **dbValueChanged** (const double &out)
- void **dbValueChanged** (const bool &out)
- void **dbConnectionChanged** (const bool &isConnected)

## Public Member Functions

- **QENumericEdit** (QWidget \*parent=0)
- **QENumericEdit** (const QString &variableName, QWidget \*parent=0)
- virtual ~**QENumericEdit** ()  
*Destruktion.*
- double **getNumericValue** ()
- void **setNumericValue** (const double value, const bool isUserUpdate=false)
- void **setAutoScale** (const bool value)
- bool **getAutoScale** () const
- void **setPropertyPrecision** (const int value)
- int **getPropertyPrecision** () const
- void **setPropertyLeadingZeros** (const int value)
- int **getPropertyLeadingZeros** () const
- void **setPropertyMinimum** (const double value)
- double **getPropertyMinimum** () const
- void **setPropertyMaximum** (const double value)
- double **getPropertyMaximum** () const
- void **setAddUnits** (bool addUnits)
- bool **getAddUnits** () const
- void **setArrayIndex** (const int arrayIndexIn)
- int **getArrayIndex** () const
- void **setRadix** (const QEFixedPointRadix::Radicies value)
- QEFixedPointRadix::Radicies **getRadix** () const
- void **setSeparator** (const QEFixedPointRadix::Separators value)
- QEFixedPointRadix::Separators **getSeparator** () const

## Protected Member Functions

- void **keyPressEvent** (QKeyEvent \*event)
- void **focusInEvent** (QFocusEvent \*event)
- void **mouseReleaseEvent** (QMouseEvent \*event)
- void **establishConnection** (unsigned int variableIndex)
- qcaobject::QCaObject \* **createQcalItem** (unsigned int variableIndex)
- int **getPrecision** () const

- int **getLeadingZeros** () const
- double **getMinimum** () const
- double **getMaximum** () const
- int **maximumSignificance** () const
- int **getRadixValue** () const
- void **setValue** (const QVariant &value)
 

*Sets the underlying QLineEdit widget to the given value.*
- QVariant **getValue** ()
 

*Gets the underlying value.*
- bool **writeData** (const QVariant &value, QString &message)
 

*Write the data to the channel.*

## Protected Attributes

- QEFloatingFormatting **floatingFormatting**

## Properties

- bool **autoScale**
- QEFixedPointRadix::Radicies **radix**

*Specify radix, default is Decimal.*
- QEFixedPointRadix::Separators **separator**

*Specify digit 'thousands' separator character, default is none.*
- int **precision**
- int **leadingZeros**
- double **minimum**
- double **maximum**
- bool **addUnits**
- int **arrayIndex**

## Friends

- class **NumericValidator**

### 9.74.1 Detailed Description

The **QENumericUpDown** class This class is similar to **QELineEdit** (both of which are derived from **QLineEdit**). However this class is tailored specifically for editing numerical values.

Note: this class based on thumb\_wheel\_edits.pas by same author.

### 9.74.2 Constructor & Destructor Documentation

9.74.2.1 `QENumericEdit::QENumericEdit ( QWidget * parent = 0 )`

Create without a variable. Use `setVariableNameProperty()` and `setSubstitutionsProperty()` to define a variable and, optionally, macro substitutions later.

9.74.2.2 `QENumericEdit::QENumericEdit ( 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.

### 9.74.3 Member Function Documentation

9.74.3.1 `void QENumericEdit::dbConnectionChanged ( const bool & isConnected ) [signal]`

Sent when the widget state updated following a channel connection change Applied to provary varible.

9.74.3.2 `void QENumericEdit::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.

### 9.74.4 Property Documentation

9.74.4.1 `bool QENumericEdit::addUnits [read, write]`

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

9.74.4.2 `int QENumericEdit::arrayIndex [read, write]`

Array Index element to edit if variable is a waveform. Defaults to 0.

9.74.4.3 `bool QENumericEdit::autoScale [read, write]`

If true (default), display and editing of numbers using the precision, and control limits supplied with the data. If false, the precision, leadingZeros, minimum and maximum properties are used.

**9.74.4.4 int QENumericEdit::leadingZeros [read, write]**

Specifies the number of leading zeros. This is only used if autoScale is false. Strictly speaking, this should be an unsigned int, but designer properties editor much 'nicer' with integers.

**9.74.4.5 double QENumericEdit::maximum [read, write]**

Specifies the maximum allowed value. This is only used if autoScale is false.

**9.74.4.6 double QENumericEdit::minimum [read, write]**

Specifies the minimum allowed value. This is only used if autoScale is false.

**9.74.4.7 int QENumericEdit::precision [read, write]**

Precision used for the display and editing of numbers. The default is 4. This is only used if autoScale is false. Strictly speaking, this should be an unsigned int, but designer properties editor much 'nicer' with integers.

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

- /tmp/epicsqt/trunk/framework/widgets/QELineEdit/QENumericEdit.h
- /tmp/epicsqt/trunk/framework/widgets/QELineEdit/QENumericEdit.cpp

## 9.75 QENumericEditManager Class Reference

### Public Member Functions

- **QENumericEditManager** (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:

- /tmp/epicsqt/trunk/framework/widgets/QELineEdit/QENumericEditManager.h
- /tmp/epicsqt/trunk/framework/widgets/QELineEdit/QENumericEditManager.cpp

## 9.76 QEPeriodic Class Reference

### 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 = userLevelTypes::USERLEVEL\_USER, Scientist = userLevelTypes::USERLEVEL\_SCIENTIST, Engineer = userLevelTypes::USERLEVEL\_ENGINEER }
- enum **DisplayAlarmStateOptions** { Never = standardProperties::DISPLAY\_ALARM\_STATE\_NEVER, Always = standardProperties::DISPLAY\_ALARM\_STATE\_ALWAYS, WhenInAlarm = standardProperties::DISPLAY\_ALARM\_STATE\_WHEN\_IN\_ALARM }
- 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 **setElement** (const QString &symbol)

### Signals

- void **userElementChanged** (const QString &symbol)
   
*Sent when the element is changed by the user selecting an element.*
- 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 **getElementValues** (QString symbol, double \*value1, double \*value2)
- QString **getSelectedSymbol** ()
- void **setVariableNameSubstitutionsProperty** (QString variableNameSubstitutions)

*Property access function for **variableSubstitutions** property. This has special behaviour to work well within designer.*

- QString **getVariableNameSubstitutionsProperty** ()
- Property access function for **variableSubstitutions** property. This has special behaviour to work well within designer.*
- **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.*
- **DisplayAlarmStateOptions getDisplayAlarmStateOptionProperty** ()

Access function for `displayAlarmStateOption` property - refer to `displayAlarmStateOption` property for details.

- void `setDisplayAlarmStateOptionProperty` (`DisplayAlarmStateOptions` option)  
Access function for `displayAlarmStateOption` property - refer to `displayAlarmStateOption` 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 Types

- enum `variableIndexes` {  
`WRITE_VARIABLE_1, WRITE_VARIABLE_2, READ_VARIABLE_1, READ_VARIABLE_2,`  
`QEPPERIODIC_NUM_VARIABLES }`

### 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` ()
- void `paste` (QVariant s)

### Protected Attributes

- QEFormatting **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 **visible**
- unsigned **int**
- QString **userLevelUserStyle**
- QString **userLevelScientistStyle**
- QString **userLevelEngineerStyle**
- **UserLevels userLevelVisibility**
- **UserLevels userLevelEnabled**
- bool **displayAlarmState**
- **DisplayAlarmStateOptions displayAlarmStateOption**
- **PresentationOptions presentationOption**
- **VariableTypes variableType1**
- **VariableTypes variableType2**
- QString **userInfo**

### 9.76.1 Member Enumeration Documentation

#### 9.76.1.1 enum QEPeriodic::DisplayAlarmStateOptions

User friendly enumerations for **displayAlarmStateOption** property - refer to **displayAlarmStateOption** property and **displayAlarmStateOptions** enumeration for details.

##### Enumerator:

**Never** Refer to DISPLAY\_ALARM\_STATE\_NEVER for details.

**Always** Refer to DISPLAY\_ALARM\_STATE\_ALWAYS for details.

**WhenInAlarm** Refer to DISPLAY\_ALARM\_STATE\_WHEN\_IN\_ALARM for details.

### 9.76.1.2 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.

## 9.76.2 Member Function Documentation

### 9.76.2.1 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.

### 9.76.2.2 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.

## 9.76.3 Member Data Documentation

### 9.76.3.1 bool QEPeriodic::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.

## 9.76.4 Property Documentation

### 9.76.4.1 bool QEPeriodic::displayAlarmState [read, write]

DEPRECATED. USE `displayAlarmStateOption` INSTEAD. If set (default) widget will indicate the alarm state of any variable data it is displaying. If clear widget will never indicate the alarm state of any variable data it is displaying. Typically the background colour is set to indicate the alarm state. Note, this property is included in the set of standard properties as it applies to most widgets. It will do nothing for widgets that don't display data.

**9.76.4.2 DisplayAlarmStateOptions** `QEPeriodic::displayAlarmStateOption` [read, write]

If 'Always' (default) widget will indicate the alarm state of any variable data it is displaying, including 'No Alarm'. If 'Never' widget will never indicate the alarm state of any variable data it is displaying. If 'WhenInAlarm' widget only indicate the alarm state of any variable data it is displaying if it is 'in alarm'. Typically the background colour is set to indicate the alarm state. Note, this property is included in the set of standard properties as it applies to most widgets. It will do nothing for widgets that don't display data.

**9.76.4.3 unsigned** `QEPeriodic::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.

**9.76.4.4 QString** `QEPeriodic::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.

**9.76.4.5 QString** `QEPeriodic::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.

**9.76.4.6 bool** `QEPeriodic::subscribe` [read, write]

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

**9.76.4.7 UserLevels** `QEPeriodic::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'.

**9.76.4.8 QString QEPeriodic::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.

**9.76.4.9 QString QEPeriodic::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.

**9.76.4.10 QString QEPeriodic::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.

**9.76.4.11 UserLevels QEPeriodic::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'.

**9.76.4.12 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.

**9.76.4.13 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.

---

**9.76.4.14 bool QEPeriodic::visible [read, write]**

Display the widget. Default is true. Setting this property false is useful 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.

**9.76.4.15 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.

**9.76.4.16 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:

- /tmp/epicsqt/trunk/framework/widgets/QEPeriodic/QEPeriodic.h
- /tmp/epicsqt/trunk/framework/widgets/QEPeriodic/QEPeriodic.cpp

## 9.77 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:

- /tmp/epicsqt/trunk/framework/widgets/QEPeriodic/QEPeriodic.h

## 9.78 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:

- /tmp/epicsqt/trunk/framework/widgets/QEPeriodic/QEPeriodicTaskMenu.h
- /tmp/epicsqt/trunk/framework/widgets/QEPeriodic/QEPeriodicTaskMenuExtension.cpp

## 9.79 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:

- /tmp/epicsqt/trunk/framework/widgets/QEPeriodic/QEPeriodicTaskMenu.h
- /tmp/epicsqt/trunk/framework/widgets/QEPeriodic/QEPeriodicTaskMenuExtension.cpp

## 9.80 QEPlot Class Reference

### Public Types

- enum **UserLevels** { **User** = userLevelTypes::USERLEVEL\_USER, **Scientist** = userLevelTypes::USERLEVEL\_SCIENTIST, **Engineer** = userLevelTypes::USERLEVEL\_ENGINEER }
- enum **DisplayAlarmStateOptions** { **Never** = standardProperties::DISPLAY\_ALARM\_STATE\_NEVER, **Always** = standardProperties::DISPLAY\_ALARM\_STATE\_ALWAYS, **WhenInAlarm** = standardProperties::DISPLAY\_ALARM\_STATE\_WHEN\_IN\_ALARM }
- enum **TraceStyles** { **Lines** = QwtPlotCurve::Lines, **Sticks** = QwtPlotCurve::Sticks, **Steps** = QwtPlotCurve::Steps, **Dots** = QwtPlotCurve::Dots }

### Public Slots

- void **setManagedVisible** (bool v)

### 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)
- QSize **sizeHint** () const
- 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** ()
- void **setVariableNameSubstitutionsProperty** (QString variableNameSubstitutions)

*Property access function for `variableSubstitutions` property. This has special behaviour to work well within designer.*

- QString **getVariableNameSubstitutionsProperty** ()  
*Property access function for `variableSubstitutions` property. This has special behaviour to work well within designer.*
- 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.*
- DisplayAlarmStateOptions **getDisplayAlarmStateOptionProperty** ()  
*Access function for `displayAlarmStateOption` property - refer to `displayAlarmStateOption` property for details.*
- void **setDisplayAlarmStateOptionProperty** (DisplayAlarmStateOptions option)  
*Access function for `displayAlarmStateOption` property - refer to `displayAlarmStateOption` 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** ()
- QString **copyVariable** ()
- QVariant **copyData** ()
- void **paste** (QVariant s)

### Protected Attributes

- QEFloatingFormatting **floatingFormatting**
- bool **localEnabled**
- bool **allowDrop**

### Properties

- QString **variable1**
- QString **variable2**
- QString **variable3**
- QString **variable4**
- QString **variableSubstitutions**
- bool **variableAsToolTip**
- bool **visible**
- unsigned **int**
- QString **styleSheet**
- QString **defaultStyle**
- QString **userLevelUserStyle**
- QString **userLevelScientistStyle**
- QString **userLevelEngineerStyle**
- **UserLevels userLevelVisibility**
- **UserLevels userLevelEnabled**
- bool **displayAlarmState**
- **DisplayAlarmStateOptions displayAlarmStateOption**
- 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`

### 9.80.1 Member Enumeration Documentation

#### 9.80.1.1 enum QEPlot::DisplayAlarmStateOptions

User friendly enumerations for `displayAlarmStateOption` property - refer to `displayAlarmStateOption` property and `displayAlarmStateOptions` enumeration for details.

##### Enumerator:

- Never** Refer to `DISPLAY_ALARM_STATE_NEVER` for details.  
**Always** Refer to `DISPLAY_ALARM_STATE_ALWAYS` for details.  
**WhenInAlarm** Refer to `DISPLAY_ALARM_STATE_WHEN_IN_ALARM` for details.

#### 9.80.1.2 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.

### 9.80.2 Member Function Documentation

#### 9.80.2.1 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.

#### 9.80.2.2 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.

**9.80.2.3 void QEPlot::setManagedVisible( bool v ) [inline, slot]**

Slot to set the visibility of a QE widget, taking into account the user level. Widget will be hidden if hidden by a call this slot, by will only be made visible by a call to this slot if the user level allows.

**9.80.3 Member Data Documentation****9.80.3.1 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.

**9.80.4 Property Documentation****9.80.4.1 QString QEPlot::defaultStyle [read, write]**

Style Sheet string to be applied before, i.e. lower priority than, any other style, e.g. alarm style and/or user level style. Default is an empty string.

**9.80.4.2 bool QEPlot::displayAlarmState [read, write]**

DEPRECATED. USE displayAlarmStateOption INSTEAD. If set (default) widget will indicate the alarm state of any variable data it is displaying. If clear widget will never indicate the alarm state of any variable data it is displaying. Typically the background colour is set to indicate the alarm state. Note, this property is included in the set of standard properties as it applies to most widgets. It will do nothing for widgets that don't display data.

**9.80.4.3 DisplayAlarmStateOptions QEPlot::displayAlarmStateOption [read, write]**

If 'Always' (default) widget will indicate the alarm state of any variable data it is displaying, including 'No Alarm'. If 'Never' widget will never indicate the alarm state of any variable data it is displaying. If 'WhenInAlarm' widget only indicate the alarm state of any variable data it is displaying if it is 'in alarm'. Typically the background colour is set to indicate the alarm state. Note, this property is included in the set of standard properties as it applies to most widgets. It will do nothing for widgets that don't display data.

**9.80.4.4 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.

#### 9.80.4.5 `QString QEPlot::styleSheet [read, write]`

Hide style sheet from designer as style calculation by the styleManager and not directly setable per se. This also stops transient styles being saved to the ui file.

#### 9.80.4.6 `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 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'.

#### 9.80.4.7 `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.

#### 9.80.4.8 `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.

#### 9.80.4.9 `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.

**9.80.4.10 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 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'.

**9.80.4.11 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.

**9.80.4.12 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.

**9.80.4.13 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.

**9.80.4.14 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.

**9.80.4.15 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.

**9.80.4.16 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.

**9.80.4.17 bool QEPlot::visible [read, write]**

Display the widget. Default is true. Setting this property false is useful if widget is only used to provide a signal - for example, when supplying data to a [QELink](#) widget. Note,

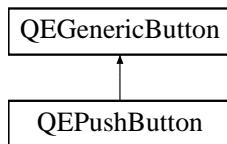
when false the widget will still be visible in Qt Designer.

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

- /tmp/epicsqt/trunk/framework/widgets/QEPlot/QEPlot.h
- /tmp/epicsqt/trunk/framework/widgets/QEPlot/QEPlot.cpp

## 9.81 QEPushButton Class Reference

Inheritance diagram for QEPushButton:



### Public Types

- enum `UserLevels` { `User` = userLevelTypes::USERLEVEL\_USER, `Scientist` = userLevelTypes::USERLEVEL\_SCIENTIST, `Engineer` = userLevelTypes::USERLEVEL\_ENGINEER }
  - enum `DisplayAlarmStateOptions` { `Never` = standardProperties::DISPLAY\_ALARM\_STATE\_NEVER, `Always` = standardProperties::DISPLAY\_ALARM\_STATE\_ALWAYS, `WhenInAlarm` = standardProperties::DISPLAY\_ALARM\_STATE\_WHEN\_IN\_ALARM }
  - 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 `ProgramStartupOptionNames` { `None` = applicationLauncher::PSO\_NONE, `Terminal` = applicationLauncher::PSO\_TERMINAL, `LogOutput` = applicationLauncher::PSO\_LOGOUTPUT, `StdOutput` = applicationLauncher::PSO\_STDOUTPUT }

- enum `CreationOptionNames` {
   
`Open` = QEActionRequests::OptionOpen, `NewTab` = QEActionRequests::OptionNewTab,
   
`NewWindow` = QEActionRequests::OptionNewWindow, `DockTop` = QEActionRequests::OptionTopDockWindow,
   
`DockBottom` = QEActionRequests::OptionBottomDockWindow, `DockLeft` = QEActionRequests::OptionLeftDockWindow, `DockRight` = QEActionRequests::OptionRightDockWindow,
   
`DockTopTabbed` = QEActionRequests::OptionTopDockWindowTabbed,
   
`DockBottomTabbed` = QEActionRequests::OptionBottomDockWindowTabbed, `DockLeftTabbed` = QEActionRequests::OptionLeftDockWindowTabbed, `DockRightTabbed` = QEActionRequests::OptionRightDockWindowTabbed, `DockFloating` = QEActionRequests::OptionFloatingDockWindow }

*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 `requestAction` (const QEActionRequests &request)
- void `setDefaultStyle` (const QString &style)
   
*Update the default style applied to this widget.*
- void `setManagedVisible` (bool v)

## 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` (const QEActionRequests &request)
   
*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)
- void `programCompleted` ()
   
*Program started by button has completed.*

## Public Member Functions

- `QEPushButton` (QWidget \*parent=0)
- `QEPushButton` (const QString &variableName, QWidget \*parent=0)
- void `writeNow` ()
- void `setVariableNameSubstitutionsProperty` (QString variableNameSubstitutions)

*Property access function for `variableSubstitutions` property. This has special behaviour to work well within designer.*

- **QString getVariableNameSubstitutionsProperty ()**  
*Property access function for `variableSubstitutions` property. This has special behaviour to work well within designer.*
- **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.*
- **DisplayAlarmStateOptions getDisplayAlarmStateOptionProperty ()**  
*Access function for `displayAlarmStateOption` property - refer to `displayAlarmStateOption` property for details.*
- **void setDisplayAlarmStateOptionProperty (DisplayAlarmStateOptions option)**  
*Access function for `displayAlarmStateOption` property - refer to `displayAlarmStateOption` 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 allowDrop**
- **bool visible**
- **unsigned int**
- **QString styleSheet**

- `QString defaultStyle`
- `QString userLevelUserStyle`
- `QString userLevelScientistStyle`
- `QString userLevelEngineerStyle`
- `UserLevels userLevelVisibility`
- `UserLevels userLevelEnabled`
- `bool displayAlarmState`
- `DisplayAlarmStateOptions displayAlarmStateOption`
- `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`
- `QString confirmText`
- `bool writeOnPress`
- `bool writeOnRelease`
- `bool writeOnClick`
- `QString pressText`
- `QString releaseText`
- `QString clickText`
- `QString clickCheckedText`
- `QString labelText`
- `QString program`
- `QStringList arguments`
- `ProgramStartupOptionNames programStartupOption`
- `QString guiFile`
- `CreationOptionNames creationOption`
- `QString prioritySubstitutions`
- `QString customisationName`

### 9.81.1 Member Enumeration Documentation

#### 9.81.1.1 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.

#### 9.81.1.2 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.

**DockTop** Open new GUI in a top dock window.

**DockBottom** Open new GUI in a bottom dock window.

**DockLeft** Open new GUI in a left dock window.

**DockRight** Open new GUI in a right dock window.

**DockTopTabbed** Open new GUI in a top dock window (tabbed with any existing dock in that area)

**DockBottomTabbed** Open new GUI in a bottom dock window (tabbed with any existing dock in that area)

**DockLeftTabbed** Open new GUI in a left dock window (tabbed with any existing dock in that area)

**DockRightTabbed** Open new GUI in a right dock window (tabbed with any existing dock in that area)

**DockFloating** Open new GUI in a floating dock window.

#### 9.81.1.3 enum QEPushButton::DisplayAlarmStateOptions

User friendly enumerations for `displayAlarmStateOption` property - refer to `displayAlarmStateOption` property and `displayAlarmStateOptions` enumeration for details.

##### Enumerator:

**Never** Refer to `DISPLAY_ALARM_STATE_NEVER` for details.

**Always** Refer to `DISPLAY_ALARM_STATE_ALWAYS` for details.

**WhenInAlarm** Refer to `DISPLAY_ALARM_STATE_WHEN_IN_ALARM` for details.

#### 9.81.1.4 enum QEPushButton::Formats

User friendly enumerations for format property - refer to QEStringFormatting::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.

#### 9.81.1.5 enum QEPushButton::Notations

User friendly enumerations for notation property - refer to QEStringFormatting::notations for details.

##### Enumerator:

**Fixed** Refer to QEStringFormatting::NOTATION\_FIXED for details.

**Scientific** Refer to QEStringFormatting::NOTATION\_SCIENTIFIC for details.

**Automatic** Refer to QEStringFormatting::NOTATION\_AUTOMATIC for details.

#### 9.81.1.6 enum QEPushButton::ProgramStartupOptionNames

Startup options. Just run the command, run the command within a terminal, or display the output in QE message system.

##### Enumerator:

**None** Just run the program.

**Terminal** Run the program in a terminal (in Windows a command interpreter will also be started, so the program may be a built-in command like 'dir')

**LogOutput** Run the program, and log the output in the QE message system.

**StdOutput** Run the program, and send output to standard output and standard error.

#### 9.81.1.7 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)

**9.81.1.8 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.

**9.81.2 Constructor & Destructor Documentation****9.81.2.1 QEPushButton::QEPushButton ( QWidget \* *parent* = 0 )**

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

**9.81.2.2 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.

**9.81.3 Member Function Documentation****9.81.3.1 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)

**9.81.3.2 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.

**9.81.3.3 void QEPushButton::pressed ( int value ) [signal]**

Button has been Pressed. The value emitted is the integer interpretation of the press-Text property

**9.81.3.4 void QEPushButton::released ( int value ) [signal]**

Button has been Released The value emitted is the integer interpretation of the release-Text property

**9.81.3.5 void QEPushButton::requestAction ( const QEActionRequests & request )  
[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.

**9.81.3.6 void QEPushButton::setManagedVisible ( bool v ) [inline, slot]**

Slot to set the visibility of a QE widget, taking into account the user level. Widget will be hidden if hidden by a call this slot, by will only be made visible by a call to this slot if the user level allows.

#### 9.81.4 Property Documentation

**9.81.4.1 bool QEPushButton::addUnits [read, write]**

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

**9.81.4.2 Qt::Alignment QEPushButton::alignment [read, write]**

Set the buttons text alignment. Left justification is particularly useful when displaying quickly changing numeric data updates.

**9.81.4.3 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.

**9.81.4.4 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.

**9.81.4.5 QStringList QEPushButton::arguments [read, write]**

Arguments for program specified in the 'program' property.

**9.81.4.6 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'.

**9.81.4.7 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](#).

**9.81.4.8 QString QEPushButton::clickText [read, write]**

Value written when user clicks button if 'writeOnClick' property is true

Reimplemented from [QEGenericButton](#).

**9.81.4.9 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

**9.81.4.10 QString QEPushButton::confirmText [read, write]**

Text used to confirm action if confirmation dialog is presented

Reimplemented from [QEGenericButton](#).

**9.81.4.11 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](#).

**9.81.4.12 QString QEPushButton::customisationName [read, write]**

Window customisation name. This name will be used to select a set of window customisations including menu items and tool bar buttons. Applications such as QEGui can load .xml files containing named sets of window customisations. This property is used to select a set loaded from these files. The selected set of customisations will be applied to the main window containing the new GUI. Customisations are not applied if the GUI is opened as a dock.

Reimplemented from [QEGenericButton](#).

**9.81.4.13 QString QEPushButton::defaultStyle [read, write]**

Style Sheet string to be applied before, i.e. lower priority than, any other style, e.g. alarm style and/or user level style. Default is an empty string.

**9.81.4.14 bool QEPushButton::displayAlarmState [read, write]**

DEPRECATED. USE displayAlarmStateOption INSTEAD. If set (default) widget will indicate the alarm state of any variable data it is displaying. If clear widget will never indicate the alarm state of any variable data it is displaying. Typically the background colour is set to indicate the alarm state. Note, this property is included in the set of standard properties as it applies to most widgets. It will do nothing for widgets that don't display data.

**9.81.4.15 DisplayAlarmStateOptions QEPushButton::displayAlarmStateOption [read, write]**

If 'Always' (default) widget will indicate the alarm state of any variable data it is displaying, including 'No Alarm'. If 'Never' widget will never indicate the alarm state of any variable data it is displaying. If 'WhenInAlarm' widget only indicate the alarm state of any variable data it is displaying if it is 'in alarm'. Typically the background colour is set to indicate the alarm state. Note, this property is included in the set of standard properties as it applies to most widgets. It will do nothing for widgets that don't display data.

**9.81.4.16 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).

**9.81.4.17 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 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::openQEFfile\(\)](#) in [QEWidget.cpp](#) for details.

**9.81.4.18 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.

**9.81.4.19 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. IF NOT LEFT EMPTY, THIS TEXT WILL TAKE PRIORITY OVER THE PUSH BUTTON 'text' PROPERTY! 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](#).

#### 9.81.4.20 bool QEPushButton::leadingZero [read, write]

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

#### 9.81.4.21 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)  
 >= Greater 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"

#### 9.81.4.22 Notations QEPushButton::notation [read, write]

Notation used for numerical formatting. Default is fixed.

#### 9.81.4.23 QString QEPushButton::password [read, write]

Password user will need to enter before any action is taken

Reimplemented from [QEGenericButton](#).

**9.81.4.24 QPixmap QEPushButton:: pixmap0 [read, write]**

Pixmap to display if updateOption is Icon or TextAndIcon and data value translates to an index of 0

**9.81.4.25 QPixmap QEPushButton:: pixmap1 [read, write]**

Pixmap to display if updateOption is Icon or TextAndIcon and data value translates to an index of 1

**9.81.4.26 QPixmap QEPushButton:: pixmap2 [read, write]**

Pixmap to display if updateOption is Icon or TextAndIcon and data value translates to an index of 2

**9.81.4.27 QPixmap QEPushButton:: pixmap3 [read, write]**

Pixmap to display if updateOption is Icon or TextAndIcon and data value translates to an index of 3

**9.81.4.28 QPixmap QEPushButton:: pixmap4 [read, write]**

Pixmap to display if updateOption is Icon or TextAndIcon and data value translates to an index of 4

**9.81.4.29 QPixmap QEPushButton:: pixmap5 [read, write]**

Pixmap to display if updateOption is Icon or TextAndIcon and data value translates to an index of 5

**9.81.4.30 QPixmap QEPushButton:: pixmap6 [read, write]**

Pixmap to display if updateOption is Icon or TextAndIcon and data value translates to an index of 6

**9.81.4.31 QPixmap QEPushButton:: pixmap7 [read, write]**

Pixmap to display if updateOption is Icon or TextAndIcon and data value translates to an index of 7

**9.81.4.32 int QEPushButton:: precision [read, write]**

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

**9.81.4.33 QString QEPushButton::pressText [read, write]**

Value written when user presses button if 'writeOnPress' property is true

Reimplemented from [QEGenericButton](#).

**9.81.4.34 QString QEPushButton::prioritySubstitutions [read, write]**

Overriding macro substitutions. These macro substitions take precedence over any existing macro substitutions defined by the variableSubstitutions property, any parent forms, or the application containing the button. These macro substitutions are particularly usefull when the button's function is to reload the same form but with different macro substitutions. The variableSubstitutions property cannot be used for this since, although they are added to the list of macro substitutions applied to the new form, they are appended to the list and the existing macro substitutions take precedence.

Reimplemented from [QEGenericButton](#).

**9.81.4.35 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

**9.81.4.36 ProgramStartupOptionNames QEPushButton::programStartupOption [read, write]**

Startup options. Just run the command, run the command within a terminal, or display the output in QE message system.

**9.81.4.37 QString QEPushButton::releaseText [read, write]**

Value written when user releases button if 'writeOnRelease' property is true

Reimplemented from [QEGenericButton](#).

**9.81.4.38 QString QEPushButton::styleSheet [read, write]**

Hide style sheet from designer as style calculation by the styleManager and not directly setable per se. This also stops transient styles being saved to the ui file.

**9.81.4.39 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)

**9.81.4.40 bool QEPushButton::trailingZeros [read, write]**

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

**9.81.4.41 UpdateOptions QEPushButton::updateOption [read, write]**

Update options (text, pixmap, both, or state (checked or unchecked)

Reimplemented from [QEGenericButton](#).

**9.81.4.42 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.

**9.81.4.43 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 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'.

**9.81.4.44 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.

**9.81.4.45 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.

**9.81.4.46 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.

**9.81.4.47 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 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'.

**9.81.4.48 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.

**9.81.4.49 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.

**9.81.4.50 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.

**9.81.4.51 bool QEPushButton::visible [read, write]**

Display the widget. Default is true. Setting this property false is useful 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.

**9.81.4.52 bool QEPushButton::writeOnClick [read, write]**

If true, the 'clickText' property is written when the button is clicked. Default is true

Reimplemented from [QEGenericButton](#).

**9.81.4.53 bool QEPushButton::writeOnPress [read, write]**

If true, the 'pressText' property is written when the button is pressed. Default is false

Reimplemented from [QEGenericButton](#).

**9.81.4.54 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:

- /tmp/epicsqt/trunk/framework/widgets/QEButton/QEPushButton.h
- /tmp/epicsqt/trunk/framework/widgets/QEButton/QEPushButton.cpp

## 9.82 QEPVNameLists Class Reference

### Public Member Functions

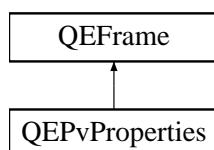
- void **prependOrMoveToFirst** (const QString &item)
- void **saveConfiguration** (PMElement &parentElement)
- void **restoreConfiguration** (PMElement &parentElement)

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

- /tmp/epicsqt/trunk/framework/widgets/QEStripChart/QEStripChart.cpp

## 9.83 QEPvProperties Class Reference

Inheritance diagram for QEPvProperties:



### Signals

- void **setCurrentBoxIndex** (int index)

## Public Member Functions

- void [setVariableNameProperty](#) (QString variableName)
 

*Property access function for `variable` property. This has special behaviour to work well within designer.*
- QString [getVariableNameProperty](#) ()
 

*Property access function for `variable` property. This has special behaviour to work well within designer.*
- void [setVariableNameSubstitutionsProperty](#) (QString variableNameSubstitutions)
 

*Property access function for `variableSubstitutions` property. This has special behaviour to work well within designer.*
- QString [getVariableNameSubstitutionsProperty](#) ()
 

*Property access function for `variableSubstitutions` property. This has special behaviour to work well within designer.*
- **QEPvProperties** (QWidget \*parent=0)
- **QEPvProperties** (const QString &variableName, QWidget \*parent=0)
- QSize [sizeHint](#) () const

## Protected Member Functions

- void [resizeEvent](#) (QResizeEvent \*event)
- void [establishConnection](#) (unsigned int variableIndex)
- qcaobject::QCaObject \* [createQcalItem](#) (unsigned int variableIndex)
- void [mousePressEvent](#) (QMouseEvent \*event)
- void [dragEnterEvent](#) (QDragEnterEvent \*event)
- void [dropEvent](#) (QDropEvent \*event)
- void [saveConfiguration](#) (PersistanceManager \*pm)
- void [restoreConfiguration](#) (PersistanceManager \*pm, restorePhases restorePhase)
- QString [copyVariable](#) ()
- QVariant [copyData](#) ()
- void [paste](#) (QVariant s)

## Properties

- QString [variable](#)
- QString [variableSubstitutions](#)

### 9.83.1 Property Documentation

#### 9.83.1.1 QString QEPvProperties::variable [read, write]

EPICS variable name (CA PV)

### 9.83.1.2 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.

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

- /tmp/epicsqt/trunk/framework/widgets/QEPvProperties/QEPvProperties.h
- /tmp/epicsqt/trunk/framework/widgets/QEPvProperties/QEPvProperties.cpp

## 9.84 QEPvPropertiesManager Class Reference

### Public Member Functions

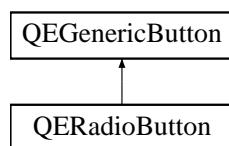
- **QEPvPropertiesManager** (QObject \*parent=0)
- bool **isContainer** () const
- bool **isInitialized** () const
- QIcon **icon** () const
- QString **group** () const
- QString **includeFile** () const
- QString **name** () const
- QString **toolTip** () const
- QString **whatsThis** () const
- QWidget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

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

- /tmp/epicsqt/trunk/framework/widgets/QEPvProperties/QEPvPropertiesManager.h
- /tmp/epicsqt/trunk/framework/widgets/QEPvProperties/QEPvPropertiesManager.cpp

## 9.85 QERadioButton Class Reference

Inheritance diagram for QERadioButton:



## Public Types

- enum `UserLevels` { `User` = userLevelTypes::USERLEVEL\_USER, `Scientist` = userLevelTypes::USERLEVEL\_SCIENTIST, `Engineer` = userLevelTypes::USERLEVEL\_ENGINEER }
- enum `DisplayAlarmStateOptions` { `Never` = standardProperties::DISPLAY\_ALARM\_STATE\_NEVER, `Always` = standardProperties::DISPLAY\_ALARM\_STATE\_ALWAYS, `WhenInAlarm` = standardProperties::DISPLAY\_ALARM\_STATE\_WHEN\_IN\_ALARM }
- 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 `Separators` { `NoSeparator` = QEStringFormatting::SEPARATOR\_NONE, `Comma` = QEStringFormatting::SEPARATOR\_COMMA, `Underscore` = QEStringFormatting::SEPARATOR\_UNDERSCORE, `Space` = QEStringFormatting::SEPARATOR\_SPACE }
- 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` = 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 `ProgramStartupOptionNames` { `None` = applicationLauncher::PSO\_NONE, `Terminal` = applicationLauncher::PSO\_TERMINAL, `LogOutput` = applicationLauncher::PSO\_LOGOUTPUT, `StdOutput` = applicationLauncher::PSO\_STDOUTPUT }
- enum `CreationOptionNames` {
   
`Open` = QEActionRequests::OptionOpen, `NewTab` = QEActionRequests::OptionNewTab, `NewWindow` = QEActionRequests::OptionNewWindow, `DockTop` = QEActionRequests::OptionTopDockWindow,
   
`DockBottom` = QEActionRequests::OptionBottomDockWindow, `DockLeft` = QEActionRequests::OptionLeftDockWindow, `DockRight` = QEActionRequests::OptionRightDockWindow, `DockTopTabbed` = QEActionRequests::OptionTopDockWindowTabbed,
   
`DockBottomTabbed` = QEActionRequests::OptionBottomDockWindowTabbed, `DockLeftTabbed` = QEActionRequests::OptionLeftDockWindowTabbed, `DockRightTabbed` = QEActionRequests::OptionRightDockWindowTabbed, `DockFloating` = QEActionRequests::OptionFloatingDockWindow }

*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 `requestAction` (const QEActionRequests &request)

- void `setDefaultStyle` (const QString &style)  
*Update the default style applied to this widget.*
- void `setManagedVisible` (bool v)

## 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` (const QEActionRequests &request)  
*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)
- void `programCompleted` ()  
*Program started by button has completed.*

## Public Member Functions

- `QERadioButton` (QWidget \*parent=0)
- `QERadioButton` (const QString &variableName, QWidget \*parent=0)
- void `writeNow` ()
- void `setVariableNameProperty` (QString variableName)  
*Property access function for `variable` property. This has special behaviour to work well within designer.*
- QString `getVariableNameProperty` ()  
*Property access function for `variable` property. This has special behaviour to work well within designer.*
- void `setVariableNameSubstitutionsProperty` (QString variableNameSubstitutions)  
*Property access function for `variableSubstitutions` property. This has special behaviour to work well within designer.*
- QString `getVariableNameSubstitutionsProperty` ()  
*Property access function for `variableSubstitutions` property. This has special behaviour to work well within designer.*
- `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.
- `DisplayAlarmStateOptions getDisplayAlarmStateOptionProperty ()`  
Access function for `displayAlarmStateOption` property - refer to `displayAlarmStateOption` property for details.
- void `setDisplayAlarmStateOptionProperty (DisplayAlarmStateOptions option)`  
Access function for `displayAlarmStateOption` property - refer to `displayAlarmStateOption` 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 `setSeparatorProperty (const Separators notation)`  
Access function for `separator` property - refer to `separator` property for details.
- `Separators getSeparatorProperty () const`  
Access function for `separator` property - refer to `separator` 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 allowDrop`
- `bool visible`
- `unsigned int`
- `QString styleSheet`
- `QString defaultStyle`
- `QString userLevelUserStyle`
- `QString userLevelScientistStyle`
- `QString userLevelEngineerStyle`
- `UserLevels userLevelVisibility`
- `UserLevels userLevelEnabled`
- `bool displayAlarmState`

- `DisplayAlarmStateOptions displayAlarmStateOption`
- `int precision`
- `bool useDbPrecision`
- `bool leadingZero`
- `bool trailingZeros`
- `bool addUnits`
- `QString localEnumeration`
- `Formats format`
- `int radix`
- `Separators separator`
- `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`
- `QString confirmText`
- `bool writeOnPress`
- `bool writeOnRelease`
- `bool writeOnClick`
- `QString pressText`
- `QString releaseText`
- `QString clickText`
- `QString clickCheckedText`
- `QString labelText`
- `QString program`
- `QStringList arguments`
- `ProgramStartupOptionNames programStartupOption`
- `QString guiFile`
- `CreationOptionNames creationOption`
- `QString prioritySubstitutions`
- `QString customisationName`

### 9.85.1 Member Enumeration Documentation

#### 9.85.1.1 enum QERadioButton::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.

#### 9.85.1.2 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.

**DockTop** Open new GUI in a top dock window.

**DockBottom** Open new GUI in a bottom dock window.

**DockLeft** Open new GUI in a left dock window.

**DockRight** Open new GUI in a right dock window.

**DockTopTabbed** Open new GUI in a top dock window (tabbed with any existing dock in that area)

**DockBottomTabbed** Open new GUI in a bottom dock window (tabbed with any existing dock in that area)

**DockLeftTabbed** Open new GUI in a left dock window (tabbed with any existing dock in that area)

**DockRightTabbed** Open new GUI in a right dock window (tabbed with any existing dock in that area)

**DockFloating** Open new GUI in a floating dock window.

#### 9.85.1.3 enum QERadioButton::DisplayAlarmStateOptions

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

##### Enumerator:

**Never** Refer to DISPLAY\_ALARM\_STATE\_NEVER for details.

**Always** Refer to DISPLAY\_ALARM\_STATE\_ALWAYS for details.

**WhenInAlarm** Refer to DISPLAY\_ALARM\_STATE\_WHEN\_IN\_ALARM for details.

#### 9.85.1.4 enum QERadioButton::Formats

User friendly enumerations for format property - refer to QEStringFormatting::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.

#### 9.85.1.5 enum QERadioButton::Notations

User friendly enumerations for notation property - refer to QEStringFormatting::notations for details.

##### Enumerator:

**Fixed** Refer to QEStringFormatting::NOTATION\_FIXED for details.

**Scientific** Refer to QEStringFormatting::NOTATION\_SCIENTIFIC for details.

**Automatic** Refer to QEStringFormatting::NOTATION\_AUTOMATIC for details.

#### 9.85.1.6 enum QERadioButton::ProgramStartupOptionNames

Startup options. Just run the command, run the command within a terminal, or display the output in QE message system.

##### Enumerator:

**None** Just run the program.

**Terminal** Run the program in a terminal (in Windows a command interpreter will also be started, so the program may be a built-in command like 'dir')

**LogOutput** Run the program, and log the output in the QE message system.

**StdOutput** Run the program, and send output to standard output and standard error.

#### 9.85.1.7 enum QERadioButton::Separators

User friendly enumerations for separator property - refer to QEStringFormatting::formats for details.

**Enumerator:**

- NoSeparator** Use no separator.
- Comma** Use ',' as separator.
- Underscore** Use '\_' as separator.
- Space** Use ' ' as separator.

**9.85.1.8 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)

**9.85.1.9 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.

**9.85.2 Constructor & Destructor Documentation****9.85.2.1 QERadioButton::QERadioButton ( QWidget \* *parent* = 0 )**

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

**9.85.2.2 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.

### 9.85.3 Member Function Documentation

9.85.3.1 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)

9.85.3.2 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.

9.85.3.3 void QERadioButton::pressed ( int *value* ) [signal]

Button has been Pressed. The value emitted is the integer interpretation of the pressText property

9.85.3.4 void QERadioButton::released ( int *value* ) [signal]

Button has been Released The value emitted is the integer interpretation of the releaseText property

9.85.3.5 void QERadioButton::requestAction ( const QEActionRequests & *request* )  
[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.

9.85.3.6 void QERadioButton::setManagedVisible ( bool *v* ) [inline, slot]

Slot to set the visibility of a QE widget, taking into account the user level. Widget will be hidden if hidden by a call this slot, by will only be made visible by a call to this slot if the user level allows.

### 9.85.4 Property Documentation

**9.85.4.1 bool QERadioButton::addUnits [read, write]**

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

**9.85.4.2 Qt::Alignment QERadioButton::alignment [read, write]**

Set the buttons text alignment. Left justification is particularly useful when displaying quickly changing numeric data updates.

**9.85.4.3 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.

**9.85.4.4 QStringList QERadioButton::arguments [read, write]**

Arguments for program specified in the 'program' property.

**9.85.4.5 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'.

**9.85.4.6 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](#).

#### 9.85.4.7 `QString QERadioButton::clickText [read, write]`

Value written when user clicks button if 'writeOnClick' property is true

Reimplemented from [QEGenericButton](#).

#### 9.85.4.8 `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

#### 9.85.4.9 `QString QERadioButton::confirmText [read, write]`

Text used to confirm action if confirmation dialog is presented

Reimplemented from [QEGenericButton](#).

#### 9.85.4.10 `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](#).

#### 9.85.4.11 `QString QERadioButton::customisationName [read, write]`

Window customisation name. This name will be used to select a set of window customisations including menu items and tool bar buttons. Applications such as QEgui can load .xml files containing named sets of window customisations. This property is used to select a set loaded from these files. The selected set of customisations will be applied to the main window containing the new GUI.

Reimplemented from [QEGenericButton](#).

#### 9.85.4.12 `QString QERadioButton::defaultStyle [read, write]`

Style Sheet string to be applied before, i.e. lower priority than, any other style, e.g. alarm style and/or user level style. Default is an empty string.

**9.85.4.13 bool QERadioButton::displayAlarmState [read, write]**

DEPRECATED. USE `displayAlarmStateOption` INSTEAD. If set (default) widget will indicate the alarm state of any variable data it is displaying. If clear widget will never indicate the alarm state of any variable data it is displaying. Typically the background colour is set to indicate the alarm state. Note, this property is included in the set of standard properties as it applies to most widgets. It will do nothing for widgets that don't display data.

**9.85.4.14 DisplayAlarmStateOptions QERadioButton::displayAlarmStateOption  
[read, write]**

If 'Always' (default) widget will indicate the alarm state of any variable data it is displaying, including 'No Alarm'. If 'Never' widget will never indicate the alarm state of any variable data it is displaying. If 'WhenInAlarm' widget only indicate the alarm state of any variable data it is displaying if it is 'in alarm'. Typically the background colour is set to indicate the alarm state. Note, this property is included in the set of standard properties as it applies to most widgets. It will do nothing for widgets that don't display data.

**9.85.4.15 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).

**9.85.4.16 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::openQEFfile()` in `QEWidget.cpp` for details.

**9.85.4.17 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.

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.

**9.85.4.18 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. IF NOT LEFT EMPTY, THIS TEXT WILL TAKE PRIORITY OVER THE PUSH BUTTON 'text' PROPERTY! 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](#).

**9.85.4.19 bool QERadioButton::leadingZero [read, write]**

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

**9.85.4.20 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)  
>= Greater 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"

**9.85.4.21 Notations** `QERadioButton::notation` [read, write]

Notation used for numerical formatting. Default is fixed.

**9.85.4.22 QString** `QERadioButton::password` [read, write]

Password user will need to enter before any action is taken

Reimplemented from [QEGenericButton](#).

**9.85.4.23 QPixmap** `QERadioButton:: pixmap0` [read, write]

Pixmap to display if updateOption is Icon or TextAndIcon and data value translates to an index of 0

**9.85.4.24 QPixmap** `QERadioButton:: pixmap1` [read, write]

Pixmap to display if updateOption is Icon or TextAndIcon and data value translates to an index of 1

**9.85.4.25 QPixmap** `QERadioButton:: pixmap2` [read, write]

Pixmap to display if updateOption is Icon or TextAndIcon and data value translates to an index of 2

**9.85.4.26 QPixmap** `QERadioButton:: pixmap3` [read, write]

Pixmap to display if updateOption is Icon or TextAndIcon and data value translates to an index of 3

**9.85.4.27 QPixmap** `QERadioButton:: pixmap4` [read, write]

Pixmap to display if updateOption is Icon or TextAndIcon and data value translates to an index of 4

**9.85.4.28 QPixmap** `QERadioButton:: pixmap5` [read, write]

Pixmap to display if updateOption is Icon or TextAndIcon and data value translates to an index of 5

**9.85.4.29 QPixmap** `QERadioButton:: pixmap6` [read, write]

Pixmap to display if updateOption is Icon or TextAndIcon and data value translates to an index of 6

**9.85.4.30 QPixmap QERadioButton:: pixmap7 [read, write]**

Pixmap to display if updateOption is Icon or TextAndIcon and data value translates to an index of 7

**9.85.4.31 int QERadioButton:: precision [read, write]**

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

**9.85.4.32 QString QERadioButton:: pressText [read, write]**

Value written when user presses button if 'writeOnPress' property is true

Reimplemented from [QEGenericButton](#).

**9.85.4.33 QString QERadioButton:: prioritySubstitutions [read, write]**

Overriding macro substitutions. These macro substitutions take precedence over any existing macro substitutions defined by the variableSubstitutions property, any parent forms, or the application containing the button. These macro substitutions are particularly useful when the button's function is to reload the same form but with different macro substitutions. The variableSubstitutions property cannot be used for this since, although they are added to the list of macro substitutions applied to the new form, they are appended to the list and the existing macro substitutions take precedence.

Reimplemented from [QEGenericButton](#).

**9.85.4.34 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

**9.85.4.35 ProgramStartupOptionNames QERadioButton:: programStartupOption [read, write]**

Startup options. Just run the command, run the command within a terminal, or display the output in QE message system.

**9.85.4.36 int QERadioButton:: radix [read, write]**

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

**9.85.4.37 QString QERadioButton::releaseText [read, write]**

Value written when user releases button if 'writeOnRelease' property is true

Reimplemented from [QEGenericButton](#).

**9.85.4.38 Separators QERadioButton::separator [read, write]**

Separators used for integer and fixed point formatting. Default is None.

**9.85.4.39 QString QERadioButton::styleSheet [read, write]**

Hide style sheet from designer as style calculation by the styleManager and not directly setable per se. This also stops transient styles being saved to the ui file.

**9.85.4.40 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)

**9.85.4.41 bool QERadioButton::trailingZeros [read, write]**

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

**9.85.4.42 UpdateOptions QERadioButton::updateOption [read, write]**

Update options (text, pixmap, both, or state (checked or unchecked)

Reimplemented from [QEGenericButton](#).

**9.85.4.43 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.

**9.85.4.44 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 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'.

**9.85.4.45 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.

**9.85.4.46 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.

**9.85.4.47 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.

**9.85.4.48 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 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'.

**9.85.4.49 QString QERadioButton::variable [read, write]**

EPICS variable name (CA PV)

**9.85.4.50 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.

**9.85.4.51 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.

**9.85.4.52 bool QERadioButton::visible [read, write]**

Display the widget. Default is true. Setting this property false is useful 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.

**9.85.4.53 bool QERadioButton::writeOnClick [read, write]**

If true, the 'clickText' property is written when the button is clicked. Default is true  
Reimplemented from [QEGenericButton](#).

**9.85.4.54 bool QERadioButton::writeOnPress [read, write]**

If true, the 'pressText' property is written when the button is pressed. Default is false  
Reimplemented from [QEGenericButton](#).

**9.85.4.55 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:

- /tmp/epicsqt/trunk/framework/widgets/QEButton/QERadioButton.h
- /tmp/epicsqt/trunk/framework/widgets/QEButton/QERadioButton.cpp

## 9.86 QERecipe Class Reference

### Public Types

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

## Public Member Functions

- **QEReipe** (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 **setOptionsLayout** (int pValue)
- int **getOptionsLayout** ()
- void **setCurrentUserType** (int pValue)
- int **getCurrentUserType** ()
- bool **saveRecipeList** ()
- void **refreshRecipeList** ()
- void **refreshButton** ()
- void **userLevelChanged** (userLevelTypes::userLevels pValue)
- void **setConfigurationTypeProperty** (configurationTypesProperty pConfigurationType)
  - configurationTypesProperty **getConfigurationTypeProperty** ()
  - void **setOptionsLayoutProperty** (optionsLayoutProperty pOptionsLayout)
  - optionsLayoutProperty **getOptionsLayoutProperty** ()
  - 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 **optionsLayout**
- 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**
- optionsLayoutProperty **optionsLayout**
- userTypesProperty **currentUserType**

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

- /tmp/epicsqt/trunk/framework/widgets/QEReipe/QEReipe.h
- /tmp/epicsqt/trunk/framework/widgets/QEReipe/QEReipe.cpp

## 9.87 QERecordFieldName Class Reference

### Static Public Member Functions

- static QString **recordName** (const QString &pvName)
- static QString **fieldName** (const QString &pvName)
- static QString **fieldPvName** (const QString &pvName, const QString &field)
- static QString **rtypePvName** (const QString &pvName)

- static bool **pvNameIsValid** (const QString &pvName)
- static bool **extractPvName** (const QString &item, QString &pvName)

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

- /tmp/epicsqt/trunk/framework/widgets/QEPvProperties/QEPvPropertiesUtilities.h
- /tmp/epicsqt/trunk/framework/widgets/QEPvProperties/QEPvPropertiesUtilities.cpp

## 9.88 QERecordSpec Class Reference

### Public Member Functions

- **QERecordSpec** (const QString recordType)
- QString **getRecordType** ()
- QString **getFieldName** (const int index)

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

- /tmp/epicsqt/trunk/framework/widgets/QEPvProperties/QEPvPropertiesUtilities.h
- /tmp/epicsqt/trunk/framework/widgets/QEPvProperties/QEPvPropertiesUtilities.cpp

## 9.89 QERecordSpecList Class Reference

### Public Member Functions

- **QERecordSpec** \* **find** (const QString recordType)
- void **appendOrReplace** (**QERecordSpec** \*recordSpec)
- bool **processRecordSpecFile** (const QString &filename)

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

- /tmp/epicsqt/trunk/framework/widgets/QEPvProperties/QEPvPropertiesUtilities.h
- /tmp/epicsqt/trunk/framework/widgets/QEPvProperties/QEPvPropertiesUtilities.cpp

## 9.90 QEScript Class Reference

```
#include <QEScript.h>
```

## Public Types

- enum **scriptTypesProperty** { **File** = FROM\_FILE, **Text** = FROM\_TEXT }
- enum **optionsLayoutProperty** { **Top** = TOP, **Bottom** = BOTTOM, **Left** = LEFT, **Right** = RIGHT }
- enum **UserLevels** { **User** = userLevelTypes::USERLEVEL\_USER, **Scientist** = userLevelTypes::USERLEVEL\_SCIENTIST, **Engineer** = userLevelTypes::USERLEVEL\_ENGINEER }
- enum **DisplayAlarmStateOptions** { **Never** = standardProperties::DISPLAY\_ALARM\_STATE\_NEVER, **Always** = standardProperties::DISPLAY\_ALARM\_STATE\_ALWAYS, **WhenInAlarm** = standardProperties::DISPLAY\_ALARM\_STATE\_WHEN\_IN\_ALARM }

## Public Slots

- void **setManagedVisible** (bool v)

## Signals

- void **selected** (QString pFilename)

## Public Member Functions

- **QEScript** (QWidget \*pParent=0)
- void **setShowScriptList** (bool pValue)
- bool **getShowScriptList** ()
- void **setShowNew** (bool pValue)
- bool **getShowNew** ()
- void **setShowSave** (bool pValue)
- bool **getShowSave** ()
- void **setShowDelete** (bool pValue)
- bool **getShowDelete** ()
- void **setShowExecute** (bool pValue)
- bool **getShowExecute** ()
- void **setShowAbort** (bool pValue)
- bool **getShowAbort** ()
- void **setEditableTable** (bool pValue)
- bool **getEditableTable** ()
- void **setShowTable** (bool pValue)
- bool **getShowTable** ()
- void **setShowTableControl** (bool pValue)
- bool **getShowTableControl** ()
- void **setShowColumnNumber** (bool pValue)
- bool **getShowColumnNumber** ()
- void **setShowColumnEnable** (bool pValue)
- bool **getShowColumnEnable** ()

- void **setShowColumnProgram** (bool pValue)
- bool **getShowColumnProgram** ()
- void **setShowColumnParameters** (bool pValue)
- bool **getShowColumnParameters** ()
- void **setShowColumnWorkingDirectory** (bool pValue)
- bool **getShowColumnWorkingDirectory** ()
- void **setShowColumnTimeout** (bool pValue)
- bool **getShowColumnTimeout** ()
- void **setShowColumnStop** (bool pValue)
- bool **getShowColumnStop** ()
- void **setShowColumnLog** (bool pValue)
- bool **getShowColumnLog** ()
- void **setScriptType** (int pValue)
- int **getScriptType** ()
- void **setScriptFile** (QString pValue)
- QString **getScriptFile** ()
- void **setScriptText** (QString pValue)
- QString **getScriptText** ()
- void **setScriptDefault** (QString pValue)
- QString **getScriptDefault** ()
- void **setExecuteText** (QString pValue)
- QString **getExecuteText** ()
- void **setOptionsLayout** (int pValue)
- int **getOptionsLayout** ()
- void **insertRow** (bool pEnable, QString pProgram, QString pParameter, QString pWorkingDirectory, int pTimeOut, bool pStop, bool pLog)
- bool **saveScriptList** ()
- void **refreshScriptList** ()
- void **refreshWidgets** ()
- void **setScriptTypeProperty** (scriptTypesProperty pScriptType)
- scriptTypesProperty **getScriptTypeProperty** ()
- void **setOptionsLayoutProperty** (optionsLayoutProperty pOptionsLayout)
- optionsLayoutProperty **getOptionsLayoutProperty** ()
- 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.*
- DisplayAlarmStateOptions **getDisplayAlarmStateOptionProperty** ()

- Access function for `displayAlarmStateOption` property - refer to `displayAlarmStateOption` property for details.
- void `setDisplayAlarmStateOptionProperty (DisplayAlarmStateOptions option)`  
Access function for `displayAlarmStateOption` property - refer to `displayAlarmStateOption` property for details.

## Protected Attributes

- QComboBox \* `qComboBoxScriptList`
- QPushButton \* `qPushButtonNew`
- QPushButton \* `qPushButtonSave`
- QPushButton \* `qPushButtonDelete`
- QPushButton \* `qPushButtonExecute`
- QPushButton \* `qPushButtonAbort`
- QPushButton \* `qPushButtonAdd`
- QPushButton \* `qPushButtonRemove`
- QPushButton \* `qPushButtonUp`
- QPushButton \* `qPushButtonDown`
- QPushButton \* `qPushButtonCopy`
- QPushButton \* `qPushButtonPaste`
- `_QTableWidgetScript` \* `qTableWidgetScript`
- QString `scriptFile`  
*Define the file where to save the scripts (if not defined then the scripts will be saved in a file named "QEScript.xml")*
- QString `scriptText`  
*Define the XML text that contains the scripts.*
- QString `scriptDefault`  
*Define the script (previously saved by the user) that will be load as the default script when the widget starts.*
- int `scriptType`
- int `optionsLayout`
- QDomDocument `document`
- QString `filename`
- QList< `_CopyPaste` \* > `copyPasteList`
- bool `editableTable`  
*Enable/disable table edition.*
- bool `isExecuting`

## Properties

- bool `showScriptList`  
*Show/hide combobox that contains the list of existing scripts created by the user.*
- bool `showNew`  
*Show/hide button to reset (initialize) the table that contains the sequence of programs to be executed.*

- bool **showSave**  
*Show/hide button to save/overwrite a new/existing script.*
- bool **showDelete**  
*Show/hide button to delete an existing script.*
- bool **showExecute**  
*Show/hide button to execute a sequence of programs.*
- bool **showAbort**  
*Show/hide button to abort the execution of a sequence of programs.*
- bool **showTable**  
*Show/hide table that contains a sequence of programs to be executed.*
- bool **showTableControl**  
*Show/hide the controls of the table that contains a sequence of programs to be executed.*
- bool **showColumnName**  
*Show/hide the column '#' that displays the sequential number of programs.*
- bool **showColumnEnable**  
*Show/hide the column 'Enable' that enables the execution of programs.*
- bool **showColumnProgram**  
*Show/hide the column 'Program' that contains the external programs to be executed.*
- bool **showColumnParameters**  
*Show/hide the column 'Parameters' that contains the parameters that are passed to external programs to be executed.*
- bool **showColumnWorkingDirectory**  
*Show/hide the column 'Directory' that defines the working directory to be used when external programs are executed.*
- bool **showColumnTimeout**  
*Show/hide the column 'Timeout' that defines a time out period in seconds (if equal to 0 then the program runs until it finishes; otherwise if greater than 0 then the program will only run during this amount of seconds and will be aborted beyond this time)*
- bool **showColumnStop**  
*Show/hide the column 'Stop' that enables stopping the execution of subsequent programs when the current one exited with an error code different from 0.*
- bool **showColumnLog**  
*Show/hide the column 'Log' that enables the generation of log messages (these messages may be displayed using the [QELog](#) widget)*
- scriptTypesProperty **scriptType**  
*Select if the scripts are to be loaded/saved from an XML file or from an XML text.*
- QString **executeText**  
*Define the caption of the button responsible for starting the execution of external programs (if not defined then the caption will be "Execute")*
- optionsLayoutProperty **optionsLayout**  
*Change the order of the widgets. Valid orders are: TOP, BOTTOM, LEFT and RIGHT.*
- bool **variableAsToolTip**
- bool **allowDrop**
- bool **visible**

- unsigned int
- QString styleSheet
- QString defaultStyle
- QString userLevelUserStyle
- QString userLevelScientistStyle
- QString userLevelEngineerStyle
- UserLevels userLevelVisibility
- UserLevels userLevelEnabled
- bool displayAlarmState
- DisplayAlarmStateOptions displayAlarmStateOption

### 9.90.1 Detailed Description

This class is a EPICS aware widget. The [QEScript](#) widget allows the user to define a certain sequence of external programs to be executed. This sequence may be saved, modified or loaded for future usage.

### 9.90.2 Member Enumeration Documentation

#### 9.90.2.1 enum QEScript::DisplayAlarmStateOptions

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

##### Enumerator:

**Never** Refer to DISPLAY\_ALARM\_STATE\_NEVER for details.

**Always** Refer to DISPLAY\_ALARM\_STATE\_ALWAYS for details.

**WhenInAlarm** Refer to DISPLAY\_ALARM\_STATE\_WHEN\_IN\_ALARM for details.

#### 9.90.2.2 enum QEScript::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.

### 9.90.3 Member Function Documentation

9.90.3.1 **void QEScript::setManagedVisible ( bool v ) [inline, slot]**

Slot to set the visibility of a QE widget, taking into account the user level. Widget will be hidden if hidden by a call this slot, by will only be made visible by a call to this slot if the user level allows.

### 9.90.4 Property Documentation

9.90.4.1 **bool QEScript::allowDrop [read, write]**

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

9.90.4.2 **QString QEScript::defaultStyle [read, write]**

Style Sheet string to be applied before, i.e. lower priority than, any other style, e.g. alarm style and/or user level style. Default is an empty string.

9.90.4.3 **bool QEScript::displayAlarmState [read, write]**

DEPRECATED. USE displayAlarmStateOption INSTEAD. If set (default) widget will indicate the alarm state of any variable data it is displaying. If clear widget will never indicate the alarm state of any variable data it is displaying. Typically the background colour is set to indicate the alarm state. Note, this property is included in the set of standard properties as it applies to most widgets. It will do nothing for widgets that don't display data.

9.90.4.4 **DisplayAlarmStateOptions QEScript::displayAlarmStateOption [read, write]**

If 'Always' (default) widget will indicate the alarm state of any variable data it is displaying, including 'No Alarm'. If 'Never' widget will never indicate the alarm state of any variable data it is displaying. If 'WhenInAlarm' widget only indicate the alarm state of any variable data it is displaying if it is 'in alarm'. Typically the background colour is set to indicate the alarm state. Note, this property is included in the set of standard properties as it applies to most widgets. It will do nothing for widgets that don't display data.

9.90.4.5 **unsigned QEScript::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.

#### 9.90.4.6 `QString QEScript::styleSheet [read, write]`

Hide style sheet from designer as style calculation by the styleManager and not directly setable per se. This also stops transient styles being saved to the ui file.

#### 9.90.4.7 `UserLevels QEScript::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'.

#### 9.90.4.8 `QString QEScript::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.

#### 9.90.4.9 `QString QEScript::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.

#### 9.90.4.10 `QString QEScript::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.

9.90.4.11 **UserLevels** `QEScript::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'.

9.90.4.12 `bool QEScript::variableAsToolTip` [read, write]

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

9.90.4.13 `bool QEScript::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:

- /tmp/epicsqt/trunk/framework/widgets/QEScript/QEScript.h
- /tmp/epicsqt/trunk/framework/widgets/QEScript/QEScript.cpp

## 9.91 QEShape Class Reference

```
#include <QEShape.h>
```

### Public Types

- enum `shapeOptions` {
 **Line, Points, Polyline, Polygon,**  
**Rect, RoundedRect, Ellipse, Arc,**  
**Chord, Pie, Path** }
- enum `animationOptions` {
 **Width, Height, X, Y,**  
**Transparency, Rotation, ColourHue, ColourSaturation,**  
**ColourValue, ColourIndex, Penwidth** }
- enum `UserLevels` { **User** = userLevelTypes::USERLEVEL\_USER, **Scientist** = userLevelTypes::USERLEVEL\_SCIENTIST, **Engineer** = userLevelTypes::USERLEVEL\_ENGINEER }
- enum `DisplayAlarmStateOptions` { **Never** = standardProperties::DISPLAY\_ALARM\_STATE\_NEVER, **Always** = standardProperties::DISPLAY\_ALARM\_STATE\_ALWAYS, **WhenInAlarm** = standardProperties::DISPLAY\_ALARM\_STATE\_WHEN\_IN\_ALARM }

## Public Slots

- void `setManagedVisible` (bool v)

## Signals

- 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 `scaleBy` (const int m, const int d)
 

*Scale the widgets my m/d.*
- 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 setVariableNameSubstitutionsProperty (QString variableNameSubstitutions)`

*Property access function for `variableSubstitutions` property. This has special behaviour to work well within designer.*

---

- `QString getVariableNameSubstitutionsProperty ()`  
*Property access function for `variableSubstitutions` property. This has special behaviour to work well within designer.*
- `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.*
- `DisplayAlarmStateOptions getDisplayAlarmStateOptionProperty ()`  
*Access function for `displayAlarmStateOption` property - refer to `displayAlarmStateOption` property for details.*
- `void setDisplayAlarmStateOptionProperty (DisplayAlarmStateOptions option)`  
*Access function for `displayAlarmStateOption` property - refer to `displayAlarmStateOption` property for details.*

## Properties

- `QString variable1`
- `QString variable2`
- `QString variable3`
- `QString variable4`
- `QString variable5`
- `QString variable6`
- `QString variableSubstitutions`
- `bool variableAsToolTip`
- `bool allowDrop`
- `bool visible`
- `unsigned int`
- `QString styleSheet`
- `QString defaultStyle`
- `QString userLevelUserStyle`
- `QString userLevelScientistStyle`
- `QString userLevelEngineerStyle`
- `UserLevels userLevelVisibility`
- `UserLevels userLevelEnabled`
- `bool displayAlarmState`
- `DisplayAlarmStateOptions displayAlarmStateOption`
- `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`

### 9.91.1 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 vertical and horizontal size and position. It is tighly integrated with the base class QEWidget which provides generic support such as macro substitutions, drag/drop, and standard properties.

### 9.91.2 Member Enumeration Documentation

#### 9.91.2.1 enum QEShape::animationOptions

Options for how a variable will animate the shape.

#### 9.91.2.2 enum QEShape::DisplayAlarmStateOptions

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

**Enumerator:**

**Never** Refer to DISPLAY\_ALARM\_STATE\_NEVER for details.

**Always** Refer to DISPLAY\_ALARM\_STATE\_ALWAYS for details.

**WhenInAlarm** Refer to DISPLAY\_ALARM\_STATE\_WHEN\_IN\_ALARM for details.

#### 9.91.2.3 enum QEShape::shapeOptions

Options for the type of shape.

#### 9.91.2.4 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.

### 9.91.3 Constructor & Destructor Documentation

#### 9.91.3.1 QEShape::QEShape ( QWidget \* *parent* = 0 )

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

#### 9.91.3.2 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.

### 9.91.4 Member Function Documentation

#### 9.91.4.1 void QEShape::dbValueChanged1 ( const qulonglong & 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.

#### 9.91.4.2 void QEShape::dbValueChanged2 ( const qulonglong & 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.

#### 9.91.4.3 void QEShape::dbValueChanged3 ( const qulonglong & 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.

#### 9.91.4.4 void QEShape::dbValueChanged4 ( const qulonglong & 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.

#### 9.91.4.5 void QEShape::dbValueChanged5 ( const qulonglong & 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.

#### 9.91.4.6 void QEShape::dbValueChanged6 ( const qulonglong & 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.

#### 9.91.4.7 void QEShape::setManagedVisible ( bool v ) [inline, slot]

Slot to set the visibility of a QE widget, taking into account the user level. Widget will be hidden if hidden by a call this slot, by will only be made visible by a call to this slot if the user level allows.

### 9.91.5 Property Documentation

#### 9.91.5.1 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.

#### 9.91.5.2 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.

#### 9.91.5.3 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.

#### 9.91.5.4 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.

#### 9.91.5.5 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.

#### 9.91.5.6 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.

#### 9.91.5.7 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.

#### 9.91.5.8 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'.

**9.91.5.9 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'.

**9.91.5.10 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'.

**9.91.5.11 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'.

**9.91.5.12 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'.

**9.91.5.13 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'.

**9.91.5.14 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'.

**9.91.5.15 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'.

**9.91.5.16 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'.

**9.91.5.17 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'.

**9.91.5.18 QString QEShape::defaultStyle [read, write]**

Style Sheet string to be applied before, i.e. lower priority than, any other style, e.g. alarm style and/or user level style. Default is an empty string.

**9.91.5.19 bool QEShape::displayAlarmState [read, write]**

DEPRECATED. USE `displayAlarmStateOption` INSTEAD. If set (default) widget will indicate the alarm state of any variable data it is displaying. If clear widget will never indicate the alarm state of any variable data it is displaying. Typically the background colour is set to indicate the alarm state. Note, this property is included in the set of standard properties as it applies to most widgets. It will do nothing for widgets that don't display data.

**9.91.5.20 DisplayAlarmStateOptions QEShape::displayAlarmStateOption [read, write]**

If 'Always' (default) widget will indicate the alarm state of any variable data it is displaying, including 'No Alarm'. If 'Never' widget will never indicate the alarm state of any variable data it is displaying. If 'WhenInAlarm' widget only indicate the alarm state of any variable data it is displaying if it is 'in alarm'. Typically the background colour is set to indicate the alarm state. Note, this property is included in the set of standard properties as it applies to most widgets. It will do nothing for widgets that don't display data.

**9.91.5.21 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, RoundedRectangle, Ellipse, Arc, Chord, Pie, Path

**9.91.5.22 double QEShape::offset1 [read, write]**

Offset applied to data from the 1st variable before it is used to animate the shape

**9.91.5.23 double QEShape::offset2 [read, write]**

Offset applied to data from the 2nd variable before it is used to animate the shape

**9.91.5.24 double QEShape::offset3 [read, write]**

Offset applied to data from the 3rd variable before it is used to animate the shape

**9.91.5.25 double QEShape::offset4 [read, write]**

Offset applied to data from the 4th variable before it is used to animate the shape

**9.91.5.26 double QEShape::offset5 [read, write]**

Offset applied to data from the 5th variable before it is used to animate the shape

**9.91.5.27 double QEShape::offset6 [read, write]**

Offset applied to data from the 6th variable before it is used to animate the shape

**9.91.5.28 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, Pixmap

**9.91.5.29 QPoint QEShape::point10 [read, write]**

10th coordinate used when drawing the shape. Used for the following shapes: Points, Polyline, Polygon, Path

**9.91.5.30 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, Pixmap

**9.91.5.31 QPoint QEShape::point3 [read, write]**

3rd coordinate used when drawing the shape. Used for the following shapes: Points, Polyline, Polygon, Path

**9.91.5.32 QPoint QEShape::point4 [read, write]**

4th coordinate used when drawing the shape. Used for the following shapes: Points, Polyline, Polygon, Path

**9.91.5.33 QPoint QEShape::point5 [read, write]**

5th coordinate used when drawing the shape. Used for the following shapes: Points, Polyline, Polygon, Path

**9.91.5.34 QPoint QEShape::point6 [read, write]**

6th coordinate used when drawing the shape. Used for the following shapes: Points, Polyline, Polygon, Path

**9.91.5.35 QPoint QEShape::point7 [read, write]**

7th coordinate used when drawing the shape. Used for the following shapes: Points, Polyline, Polygon, Path

**9.91.5.36 QPoint QEShape::point8 [read, write]**

8th coordinate used when drawing the shape. Used for the following shapes: Points, Polyline, Polygon, Path

**9.91.5.37 QPoint QEShape::point9 [read, write]**

9th coordinate used when drawing the shape. Used for the following shapes: Points, Polyline, Polygon, Path

**9.91.5.38 double QEShape::scale2 [read, write]**

Scale factor applied to data from the 2nd variable before it is used to animate the shape

**9.91.5.39 double QEShape::scale3 [read, write]**

Scale factor applied to data from the 3rd variable before it is used to animate the shape

**9.91.5.40 double QEShape::scale4 [read, write]**

Scale factor applied to data from the 4th variable before it is used to animate the shape

**9.91.5.41 double QEShape::scale5 [read, write]**

Scale factor applied to data from the 5th variable before it is used to animate the shape

**9.91.5.42 double QEShape::scale6 [read, write]**

Scale factor applied to data from the 6th variable before it is used to animate the shape

**9.91.5.43 QString QEShape::styleSheet [read, write]**

Hide style sheet from designer as style calculation by the styleManager and not directly setable per se. This also stops transient styles being saved to the ui file.

**9.91.5.44 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 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'.

**9.91.5.45 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.

**9.91.5.46 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.

**9.91.5.47 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.

**9.91.5.48 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 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'.

**9.91.5.49 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.

**9.91.5.50 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.

**9.91.5.51 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.

**9.91.5.52 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.

**9.91.5.53 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.

**9.91.5.54 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.

**9.91.5.55 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.

**9.91.5.56 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.

**9.91.5.57 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:

- /tmp/epicsqt/trunk/framework/widgets/QEShape/QEShape.h
- /tmp/epicsqt/trunk/framework/widgets/QEShape/QEShape.cpp

## 9.92 QESlider Class Reference

### Public Types

- enum [UserLevels](#) { [User](#) = userLevelTypes::USERLEVEL\_USER, [Scientist](#) = userLevelTypes::USERLEVEL\_SCIENTIST, [Engineer](#) = userLevelTypes::USERLEVEL\_ENGINEER }
- enum [DisplayAlarmStateOptions](#) { [Never](#) = standardProperties::DISPLAY\_ALARM\_STATE\_NEVER, [Always](#) = standardProperties::DISPLAY\_ALARM\_STATE\_ALWAYS, [WhenInAlarm](#) = standardProperties::DISPLAY\_ALARM\_STATE\_WHEN\_IN\_ALARM }

### Public Slots

- void [setDefaultStyle](#) (const QString &style)  
*Update the default style applied to this widget.*
- void [setManagedVisible](#) (bool v)

### Signals

- void [dbValueChanged](#) (const qulonglong &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** ()
- void **setVariableNameProperty** (QString variableName)  
*Property access function for `variable` property. This has special behaviour to work well within designer.*
- QString **getVariableNameProperty** ()  
*Property access function for `variable` property. This has special behaviour to work well within designer.*
- void **setVariableNameSubstitutionsProperty** (QString variableNameSubstitutions)  
*Property access function for `variableSubstitutions` property. This has special behaviour to work well within designer.*
- QString **getVariableNameSubstitutionsProperty** ()  
*Property access function for `variableSubstitutions` property. This has special behaviour to work well within designer.*
- 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.*
- DisplayAlarmStateOptions **getDisplayAlarmStateOptionProperty** ()  
*Access function for `displayAlarmStateOption` property - refer to `displayAlarmStateOption` property for details.*
- void **setDisplayAlarmStateOptionProperty** (DisplayAlarmStateOptions option)  
*Access function for `displayAlarmStateOption` property - refer to `displayAlarmStateOption` 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** ()
- QString **copyVariable** ()
- QVariant **copyData** ()
- void **paste** (QVariant s)

### Protected Attributes

- QEFloatingFormatting **floatingFormatting**
- bool **writeOnChange**

### Properties

- QString **variable**
- QString **variableSubstitutions**
- bool **subscribe**
- bool **variableAsToolTip**
- bool **allowDrop**
- bool **visible**
- unsigned **int**
- QString **styleSheet**
- QString **defaultStyle**
- QString **userLevelUserStyle**
- QString **userLevelScientistStyle**
- QString **userLevelEngineerStyle**
- UserLevels **userLevelVisibility**
- UserLevels **userLevelEnabled**
- bool **displayAlarmState**
- DisplayAlarmStateOptions **displayAlarmStateOption**
- double **value**
- int **sliderPosition**

#### 9.92.1 Member Enumeration Documentation

##### 9.92.1.1 enum QESlider::DisplayAlarmStateOptions

User friendly enumerations for `displayAlarmStateOption` property - refer to `displayAlarmStateOption` property and `displayAlarmStateOptions` enumeration for details.

**Enumerator:**

- Never** Refer to DISPLAY\_ALARM\_STATE\_NEVER for details.  
**Always** Refer to DISPLAY\_ALARM\_STATE\_ALWAYS for details.  
**WhenInAlarm** Refer to DISPLAY\_ALARM\_STATE\_WHEN\_IN\_ALARM for details.

**9.92.1.2 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.

**9.92.2 Member Function Documentation****9.92.2.1 void QESlider::dbValueChanged ( const qulonglong & 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.

**9.92.2.2 void QESlider::setManagedVisible ( bool v ) [inline, slot]**

Slot to set the visibility of a QE widget, taking into account the user level. Widget will be hidden if hidden by a call this slot, by will only be made visible by a call to this slot if the user level allows.

**9.92.3 Member Data Documentation****9.92.3.1 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).

**9.92.4 Property Documentation****9.92.4.1 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.

**9.92.4.2 QString QESlider::defaultStyle [read, write]**

Style Sheet string to be applied before, i.e. lower priority than, any other style, e.g. alarm style and/or user level style. Default is an empty string.

**9.92.4.3 bool QESlider::displayAlarmState [read, write]**

DEPRECATED. USE `displayAlarmStateOption` INSTEAD. If set (default) widget will indicate the alarm state of any variable data it is displaying. If clear widget will never indicate the alarm state of any variable data it is displaying. Typically the background colour is set to indicate the alarm state. Note, this property is included in the set of standard properties as it applies to most widgets. It will do nothing for widgets that don't display data.

**9.92.4.4 DisplayAlarmStateOptions QESlider::displayAlarmStateOption [read, write]**

If 'Always' (default) widget will indicate the alarm state of any variable data it is displaying, including 'No Alarm'. If 'Never' widget will never indicate the alarm state of any variable data it is displaying. If 'WhenInAlarm' widget only indicate the alarm state of any variable data it is displaying if it is 'in alarm'. Typically the background colour is set to indicate the alarm state. Note, this property is included in the set of standard properties as it applies to most widgets. It will do nothing for widgets that don't display data.

**9.92.4.5 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.

**9.92.4.6 QString QESlider::styleSheet [read, write]**

Hide style sheet from designer as style calculation by the `styleManager` and not directly setable per se. This also stops transient styles being saved to the ui file.

**9.92.4.7 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)

**9.92.4.8 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 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'.

**9.92.4.9 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.

**9.92.4.10 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.

**9.92.4.11 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.

**9.92.4.12 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 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'.

9.92.4.13 **QString QESlider::variable** [read, write]

EPICS variable name (CA PV)

9.92.4.14 **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.

9.92.4.15 **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.

9.92.4.16 **bool QESlider::visible** [read, write]

Display the widget. Default is true. Setting this property false is useful 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:

- /tmp/epicsqt/trunk/framework/widgets/QESlider/QESlider.h
- /tmp/epicsqt/trunk/framework/widgets/QESlider/QESlider.cpp

## 9.93 QESpinBox Class Reference

### Public Types

- enum [UserLevels](#) { [User](#) = userLevelTypes::USERLEVEL\_USER, [Scientist](#) = userLevelTypes::USERLEVEL\_SCIENTIST, [Engineer](#) = userLevelTypes::USERLEVEL\_ENGINEER }
- enum [DisplayAlarmStateOptions](#) { [Never](#) = standardProperties::DISPLAY\_ALARM\_STATE\_NEVER, [Always](#) = standardProperties::DISPLAY\_ALARM\_STATE\_ALWAYS, [WhenInAlarm](#) = standardProperties::DISPLAY\_ALARM\_STATE\_WHEN\_IN\_ALARM }

### Public Slots

- void [setDefaultStyle](#) (const QString &style)  
*Update the default style applied to this widget.*
- void [setManagedVisible](#) (bool v)

## 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 writeOnChangeln)
- bool `getWriteOnChange` ()
- void `setSubscribe` (bool subscribe)
- bool `getSubscribe` ()
- void `setAddUnitsAsSuffix` (bool addUnitsAsSuffixIn)
- bool `getAddUnitsAsSuffix` ()
- void `setUseDbPrecisionForDecimals` (bool useDbPrecisionForDecimalln)
- bool `getUseDbPrecisionForDecimals` ()
- void `setVariableNameProperty` (QString variableName)
 

*Property access function for `variable` property. This has special behaviour to work well within designer.*
- QString `getVariableNameProperty` ()
 

*Property access function for `variable` property. This has special behaviour to work well within designer.*
- void `setVariableNameSubstitutionsProperty` (QString variableNameSubstitutions)
 

*Property access function for `variableSubstitutions` property. This has special behaviour to work well within designer.*
- QString `getVariableNameSubstitutionsProperty` ()
 

*Property access function for `variableSubstitutions` property. This has special behaviour to work well within designer.*
- `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.*
- `DisplayAlarmStateOptions getDisplayAlarmStateOptionProperty` ()

- Access function for `displayAlarmStateOption` property - refer to `displayAlarmStateOption` property for details.
- void **setDisplayAlarmStateOptionProperty** (`DisplayAlarmStateOptions` option)  
Access function for `displayAlarmStateOption` property - refer to `displayAlarmStateOption` 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** ()
- QString **copyVariable** ()
- QVariant **copyData** ()
- void **paste** (QVariant s)
- QMenu \* **getDefaultContextMenu** ()

### Protected Attributes

- QEFloatingFormatting **floatingFormatting**
- bool **writeOnChange**
- bool **addUnitsAsSuffix**
- bool **useDbPrecisionForDecimal**

### Properties

- QString **variable**
- QString **variableSubstitutions**
- bool **variableAsToolTip**
- bool **allowDrop**
- bool **visible**
- unsigned **int**
- QString **styleSheet**
- QString **defaultStyle**
- QString **userLevelUserStyle**
- QString **userLevelScientistStyle**
- QString **userLevelEngineerStyle**
- **UserLevels** **userLevelVisibility**
- **UserLevels** **userLevelEnabled**
- bool **displayAlarmState**
- `DisplayAlarmStateOptions` **displayAlarmStateOption**
- bool **subscribe**
- bool **useDbPrecision**
- bool **addUnits**
- double **value**

### 9.93.1 Member Enumeration Documentation

#### 9.93.1.1 enum QESpinBox::DisplayAlarmStateOptions

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

##### Enumerator:

**Never** Refer to DISPLAY\_ALARM\_STATE\_NEVER for details.

**Always** Refer to DISPLAY\_ALARM\_STATE\_ALWAYS for details.

**WhenInAlarm** Refer to DISPLAY\_ALARM\_STATE\_WHEN\_IN\_ALARM for details.

#### 9.93.1.2 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.

### 9.93.2 Member Function Documentation

#### 9.93.2.1 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.

#### 9.93.2.2 void QESpinBox::setManagedVisible ( bool v ) [inline, slot]

Slot to set the visibility of a QE widget, taking into account the user level. Widget will be hidden if hidden by a call this slot, by will only be made visible by a call to this slot if the user level allows.

### 9.93.3 Property Documentation

#### 9.93.3.1 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.

**9.93.3.2 QString QESpinBox::defaultStyle [read, write]**

Style Sheet string to be applied before, i.e. lower priority than, any other style, e.g. alarm style and/or user level style. Default is an empty string.

**9.93.3.3 bool QESpinBox::displayAlarmState [read, write]**

DEPRECATED. USE `displayAlarmStateOption` INSTEAD. If set (default) widget will indicate the alarm state of any variable data it is displaying. If clear widget will never indicate the alarm state of any variable data it is displaying. Typically the background colour is set to indicate the alarm state. Note, this property is included in the set of standard properties as it applies to most widgets. It will do nothing for widgets that don't display data.

**9.93.3.4 DisplayAlarmStateOptions QESpinBox::displayAlarmStateOption [read, write]**

If 'Always' (default) widget will indicate the alarm state of any variable data it is displaying, including 'No Alarm'. If 'Never' widget will never indicate the alarm state of any variable data it is displaying. If 'WhenInAlarm' widget only indicate the alarm state of any variable data it is displaying if it is 'in alarm'. Typically the background colour is set to indicate the alarm state. Note, this property is included in the set of standard properties as it applies to most widgets. It will do nothing for widgets that don't display data.

**9.93.3.5 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.

**9.93.3.6 QString QESpinBox::styleSheet [read, write]**

Hide style sheet from designer as style calculation by the `styleManager` and not directly setable per se. This also stops transient styles being saved to the ui file.

**9.93.3.7 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)

**9.93.3.8 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 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'.

**9.93.3.9 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.

**9.93.3.10 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.

**9.93.3.11 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.

**9.93.3.12 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 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'.

**9.93.3.13 QString QESpinBox::variable [read, write]**

EPICS variable name (CA PV)

**9.93.3.14 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.

**9.93.3.15 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 also used for other purposes.

**9.93.3.16 bool QESpinBox::visible [read, write]**

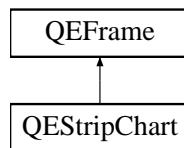
Display the widget. Default is true. Setting this property false is useful if widget is only used to provide a signal - for example, when supplying data to a [QELink](#) widget. Note, when false the widget will still be visible in Qt Designer.

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

- /tmp/epicsqt/trunk/framework/widgets/QESpinBox/QESpinBox.h
- /tmp/epicsqt/trunk/framework/widgets/QESpinBox/QESpinBox.cpp

## 9.94 QEStripChart Class Reference

Inheritance diagram for QEStripChart:



### Public Types

- enum **PropertyChartYRanges** { **manual** = QEStripChartNames::manual, **dynamic** = QEStripChartNames::dynamic }
- enum **Constants** { **NUMBER\_OF\_PVS** = 12 }

## Public Slots

- void **videoModeSelected** (const QEStripChartNames::VideoModes mode)
- void **yRangeSelected** (const QEStripChartNames::ChartYRanges scale)
- void **yScaleModeSelected** (const QEStripChartNames::YScaleModes mode)

## Public Member Functions

- **QEStripChart** (QWidget \*parent=0)
- QSize **sizeHint** () const
- QDateTime **getStartTime** () const
- QDateTime **getEndTime** () const
- void **setEndTime** (QDateTime endDateIn)
- int **getDuration** () const
- void **setDuration** (int durationIn)
- double **getYMinimum** () const
- void **setYMinimum** (const double yMinimumIn)
- double **getYMaximum** () const
- void **setYMaximum** (const double yMaximumIn)
- void **setYRange** (const double yMinimumIn, const double yMaximumIn)
- void **setPvName** (unsigned int slot, const QString &pvName)
- QString **getPvName** (unsigned int slot) const
- int **addPvName** (const QString &pvName)
- PropertyChartYRanges **getYRangeMode** () const
- void **setYRangeMode** (const PropertyChartYRanges scale)
- QEStripChartNames::VideoModes **getVideoMode** () const
- QEStripChartNames::YScaleModes **getYScaleMode** () const

## Protected Member Functions

- void **mousePressEvent** (QMouseEvent \*event)
- void **dragEnterEvent** (QDragEnterEvent \*event)
- void **dropEvent** (QDropEvent \*event)
- QString **copyVariable** ()
- QVariant **copyData** ()
- void **paste** (QVariant s)
- qcaobject::QCaObject \* **createQcalItem** (unsigned int variableIndex)
- void **establishConnection** (unsigned int variableIndex)
- void **saveConfiguration** (PersistanceManager \*pm)
- void **restoreConfiguration** (PersistanceManager \*pm, restorePhases restorePhase)
  
- void **addToPredefinedList** (const QString &pvName)
- QStringList **getPredefinedPVNameList** () const
- QString **getPredefinedItem** (int i) const
- void **setRecalcIsRequired** ()
- void **setReplotIsRequired** ()
- void **evaluateAllowDrop** ()

## Properties

- int **duration**
- double **yMinimum**
- double **yMaximum**
- QEStripChartNames::VideoModes **videoMode**
- PropertyChartYRanges **chartRange**
- QEStripChartNames::YScaleModes **scaleMode**
- QString **variable1**
- QString **variable2**
- QString **variable3**
- QString **variable4**
- QString **variable5**
- QString **variable6**
- QString **variable7**
- QString **variable8**
- QString **variable9**
- QString **variable10**
- QString **variable11**
- QString **variable12**
- QString **variableSubstitutions**
- 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 [QEStripChartItem](#)

### 9.94.1 Property Documentation

#### 9.94.1.1 QString QEStripChart::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.

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

- /tmp/epicsqt/trunk/framework/widgets/QEStripChart/QEStripChart.h
- /tmp/epicsqt/trunk/framework/widgets/QEStripChart/QEStripChart.cpp

## 9.95 QEStripChartAdjustPVDialog Class Reference

### Public Member Functions

- **QEStripChartAdjustPVDialog** (QWidget \*parent=0)
- void **setValueScaling** (const ValueScaling &valueScale)
- ValueScaling **getValueScaling** () const
- void **setSupport** (const double min, const double max, const QEDisplayRanges &loprHopr, const QEDisplayRanges &plotted, const QEDisplayRanges &buffered)

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

- /tmp/epicsqt/trunk/framework/widgets/QEStripChart/QEStripChartAdjustPVDialog.h
- /tmp/epicsqt/trunk/framework/widgets/QEStripChart/QEStripChartAdjustPVDialog.cpp

## 9.96 QEStripChartContextMenu Class Reference

### Signals

- void **contextMenuSelected** (const QEStripChartNames::ContextMenuOptions)

### Public Member Functions

- **QEStripChartContextMenu** (bool inUse, QWidget \*parent=0)
- void **setPredefinedNames** (const QStringList &pvlList)
- void **setUseReceiveTime** (const bool useReceiveTime)
- void **setArchiveReadHow** (const QEArchiveInterface::How how)
- void **setLineDrawMode** (const QEStripChartNames::LineDrawModes mode)

### 9.96.1 Constructor & Destructor Documentation

#### 9.96.1.1 QEStripChartContextMenu::QEStripChartContextMenu ( bool *inUse*, QWidget \* *parent* = 0 ) [explicit]

Construct strip chart item context menu. This menu item creates all required sub menu items. inUse set true for an inuse slot, i.e. already has a PV allocated. inUse set false for an empty slot.

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

- /tmp/epicsqt/trunk/framework/widgets/QEStripChart/QEStripChartContextMenu.h
- /tmp/epicsqt/trunk/framework/widgets/QEStripChart/QEStripChartContextMenu.cpp

## 9.97 QEStripChartDurationDialog Class Reference

### Public Member Functions

- **QEStripChartDurationDialog** (QWidget \*parent=0)
- void **setDuration** (int secs)
- int **getDuration** () const

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

- /tmp/epicsqt/trunk/framework/widgets/QEStripChart/QEStripChartDurationDialog.h
- /tmp/epicsqt/trunk/framework/widgets/QEStripChart/QEStripChartDurationDialog.cpp

## 9.98 QEStripChartItem Class Reference

### Public Slots

- void **setColour** (const QColor &colour)

### Signals

- void **itemContextMenuRequested** (const unsigned int, const QPoint &)
- void **requestAction** (const QEActionRequests &)

### Public Member Functions

- **QEStripChartItem** (QEStripChart \*chart, unsigned int slot, QWidget \*parent)
- bool **isInUse** ()
- bool **isCalculation** ()
- void **setPvName** (QString pvName, QString substitutions)
- QString **getPvName** ()
- bool **isScaled** ()
- bool **getUseReceiveTime** ()
- QEArchiveInterface::How **getArchiveReadHow** ()
- QEStripChartNames::LineDrawModes **getLineDrawMode** ()
- QColor **getColour** ()
- QEDisplayRanges **getLoprHopr** (bool doScale)
- QEDisplayRanges **getDisplayedMinMax** (bool doScale)
- QEDisplayRanges **getBufferedMinMax** (bool doScale)
- QCaDataPointList **determinePlotPoints** ()

- void **readArchive** ()
- void **normalise** ()
- void **plotData** ()
- void **saveConfiguration** (PMElement &parentElement)
- void **restoreConfiguration** (PMElement &parentElement)

### Public Attributes

- QCaVariableNamePropertyManager **pvNamePropertyManager**

### Protected Member Functions

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

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

- /tmp/epicsqt/trunk/framework/widgets/QEStripChart/QEStripChartItem.h
- /tmp/epicsqt/trunk/framework/widgets/QEStripChart/QEStripChartItem.cpp

## 9.99 QEStripChartNames Class Reference

### Public Types

- enum **ChartTimeModes** { **tmRealTime**, **tmPaused**, **tmHistorical** }
- enum **ChartYRanges** {
 **manual**, **operatingRange**, **plotted**, **buffered**,
 **dynamic**, **normalised** }
- enum **PlayModes** {
 **play**, **pause**, **forward**, **backward**,
 **selectTimes** }
- enum **StateModes** { **previous**, **next** }
- enum **VideoModes** { **normal**, **reverse** }
- enum **YScaleModes** { **linear**, **log** }
- enum **LineDrawModes** { **ldmHide**, **ldmRegular**, **ldmBold** }
- enum **ContextMenuOptions** {
 **SCCM\_NONE** = contextMenu::CM\_SPECIFIC\_WIDGETS\_START\_HERE, **SCCM\_READ\_ARCHIVE**, **SCCM\_SCALE\_CHART\_AUTO**, **SCCM\_SCALE\_CHART\_PLOTTED**, **SCCM\_SCALE\_CHART\_BUFFERED**, **SCCM\_SCALE\_PV\_RESET**, **SCCM\_SCALE\_PV\_GENERAL**, **SCCM\_SCALE\_PV\_AUTO**,
 **SCCM\_SCALE\_PV\_PLOTTED**, **SCCM\_SCALE\_PV\_BUFFERED**, **SCCM\_SCALE\_PV\_CENTRE**, **SCCM\_PLOT\_RECTANGULAR**,
 **SCCM\_PLOT\_SMOOTH**, **SCCM\_PLOT\_SERVER\_TIME**, **SCCM\_PLOT\_CLIENT\_TIME**, **SCCM\_ARCH\_LINEAR**,

```

SCCM_ARCH_PLOTBIN, SCCM_ARCH_RAW, SCCM_ARCH_SHEET, SCCM_-
ARCH_AVERAGED,
SCCM_LINE_HIDE, SCCM_LINE_REGULAR, SCCM_LINE_BOLD, SCCM_LINE_-
COLOUR,
SCCM_PV_EDIT_NAME, SCCM_ADD_TO_PREDEFINED, SCCM_PV_WRITE_-
TRACE, SCCM_PV_STATS,
SCCM_PV_CLEAR, SCCM_PV_ADD_NAME, SCCM_PV_PASTE_NAME, SCCM_-
PREDEFINED_01,
SCCM_PREDEFINED_02, SCCM_PREDEFINED_03, SCCM_PREDEFINED_-
04, SCCM_PREDEFINED_05,
SCCM_PREDEFINED_06, SCCM_PREDEFINED_07, SCCM_PREDEFINED_-
08, SCCM_PREDEFINED_09,
SCCM_PREDEFINED_10 }

```

### Static Public Attributes

- static const ContextMenuOptions **ContextMenuItemFirst** = SCCM\_READ\_ARCHIVE
- static const ContextMenuOptions **ContextMenuItemLast** = SCCM\_PREDEFINED\_-
10
- static const int **NumberPrefefinedItems** = (SCCM\_PREDEFINED\_10 - SCCM\_-
PREDEFINED\_01 + 1)

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

- /tmp/epicsqt/trunk/framework/widgets/QEStripChart/QEStripChartNames.h

## 9.100 QEStripChartPushButtonSpecifications Struct Reference

### Public Attributes

- int **gap**
- int **width**
- int **value**
- bool **isIcon**
- const QString **captionOrIcon**
- const QString **toolTip**
- const char \* **member**

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

- /tmp/epicsqt/trunk/framework/widgets/QEStripChart/QEStripChartToolBar.cpp

## 9.101 QEStripChartRangeDialog Class Reference

### Public Member Functions

- **QEStripChartRangeDialog** (QWidget \*parent=0)
- void **setRange** (const double min, const double max)
- double **getMinimum** ()
- double **getMaximum** ()

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

- /tmp/epicsqt/trunk/framework/widgets/QEStripChart/QEStripChartRangeDialog.h
- /tmp/epicsqt/trunk/framework/widgets/QEStripChart/QEStripChartRangeDialog.cpp

## 9.102 QEStripChartState Class Reference

### Public Member Functions

- void **saveConfiguration** (PMElement &parentElement)
- void **restoreConfiguration** (PMElement &parentElement)

### Public Attributes

- bool **isNormalVideo**
- QEStripChartNames::ChartTimeModes **chartTimeMode**
- QEStripChartNames::YScaleModes **yScaleMode**
- QEStripChartNames::ChartYRanges **chartYScale**
- double **yMinimum**
- double **yMaximum**
- int **duration**
- Qt::TimeSpec **timeZoneSpec**
- QDateTime **endDateTime**

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

- /tmp/epicsqt/trunk/framework/widgets/QEStripChart/QEStripChartState.h
- /tmp/epicsqt/trunk/framework/widgets/QEStripChart/QEStripChartState.cpp

## 9.103 QEStripChartStateList Class Reference

### Public Member Functions

- void **clear** ()

- void **push** (const QEStripChartState &state)
- bool **prev** (QEStripChartState &state)
- bool **next** (QEStripChartState &state)
- bool **prevAvailable** ()
- bool **nextAvailable** ()

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

- /tmp/epicsqt/trunk/framework/widgets/QEStripChart/QEStripChartState.h
- /tmp/epicsqt/trunk/framework/widgets/QEStripChart/QEStripChartState.cpp

## 9.104 QEStripChartStatistics Class Reference

### Public Member Functions

- **QEStripChartStatistics** (const QString &pvName, const QString &egu, const QCaDataPointList &dataList, QEStripChartItem \*owner, QWidget \*parent=0)

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

- /tmp/epicsqt/trunk/framework/widgets/QEStripChart/QEStripChartStatistics.h
- /tmp/epicsqt/trunk/framework/widgets/QEStripChart/QEStripChartStatistics.cpp

## 9.105 QEStripChartTimeDialog Class Reference

### Public Member Functions

- **QEStripChartTimeDialog** (QWidget \*parent=0)
- void **setMaximumDateTime** (QDateTime datetime)
- void **setStartTime** (QDateTime datetime)
- QDateTime **getStartTime** ()
- void **setEndTime** (QDateTime datetime)
- QDateTime **getEndTime** ()

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

- /tmp/epicsqt/trunk/framework/widgets/QEStripChart/QEStripChartTimeDialog.h
- /tmp/epicsqt/trunk/framework/widgets/QEStripChart/QEStripChartTimeDialog.cpp

## 9.106 QEStripChartToolBar Class Reference

This class holds all the StripChart tool bar widgets.

```
#include <QEStripChartToolBar.h>
```

## Classes

- class [OwnWidgets](#)

## Signals

- void **stateSelected** (const QEStripChartNames::StateModes mode)
- void **videoModeSelected** (const QEStripChartNames::VideoModes mode)
- void **yScaleModeSelected** (const QEStripChartNames::YScaleModes mode)
- void **yRangeSelected** (const QEStripChartNames::ChartYRanges scale)
- void **durationSelected** (const int seconds)
- void **selectDuration** ()
- void **timeZoneSelected** (const Qt::TimeSpec timeSpec)
- void **playModeSelected** (const QEStripChartNames::PlayModes mode)
- void **readArchiveSelected** ()

## Public Member Functions

- **QEStripChartToolBar** (QWidget \*parent=0)
- void **setYRangeStatus** (const QString &status)
- void **setTimeStatus** (const QString &timeStatus)
- void **setDurationStatus** (const QString &durationStatus)
- void **setStateSelectionEnabled** (const QEStripChartNames::StateModes mode, const bool enabled)

## Static Public Attributes

- static const int **designHeight** = 44

## Protected Member Functions

- void **resizeEvent** (QResizeEvent \*event)

### 9.106.1 Detailed Description

This class holds all the StripChart tool bar widgets.

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

- /tmp/epicsqt/trunk/framework/widgets/QEStripChart/QEStripChartToolBar.h
- /tmp/epicsqt/trunk/framework/widgets/QEStripChart/QEStripChartToolBar.cpp

## 9.107 QESubstitutedLabel Class Reference

### Public Member Functions

- **QESubstitutedLabel** (QWidget \*parent=0)
- void **setLabelTextProperty** (QString labelTextIn)
- QString **getLabelTextProperty** ()
- void **setSubstitutionsProperty** (QString macroSubstitutionsIn)
- QString **getSubstitutionsProperty** ()
- QString **getLabelTextPropertyFormat** ()
- void **setLabelTextPropertyFormat** (QString labelTextIn)

### Protected Attributes

- QString **labelText**

### Properties

- QString **textSubstitutions**

#### 9.107.1 Member Data Documentation

##### 9.107.1.1 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

#### 9.107.2 Property Documentation

##### 9.107.2.1 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:

- /tmp/epicsqt/trunk/framework/widgets/QESubstitutedLabel/QESubstitutedLabel.h
- /tmp/epicsqt/trunk/framework/widgets/QESubstitutedLabel/QESubstitutedLabel.cpp

## 9.108 recording Class Reference

---

## Signals

- void **byteArrayChanged** (const QByteArray &value, unsigned long dataSize, QCaAlarmInfo &alarmInfo, QCaDateTime &timeStamp, const unsigned int &variableIndex)
- void **playingBack** (bool playing)

## Public Member Functions

- **recording** (QWidget \*parent=0)
- bool **isRecording** ()
- void **recordImage** (QByteArray image, unsigned long dataSize, QCaAlarmInfo &alarmInfo, QCaDateTime &time)
- void **nextFrameDue** ()

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

- /tmp/epicsqt/trunk/framework/widgets/QElImage/recording.h
- /tmp/epicsqt/trunk/framework/widgets/QElImage/recording.cpp

## 9.109 imageDisplayProperties::rgbPixel Struct Reference

### Public Attributes

- unsigned char **p** [4]

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

- /tmp/epicsqt/trunk/framework/widgets/QElImage/brightnessContrast.h

## 9.110 screenSelectDialog Class Reference

### Public Types

- enum **screens** { **PRIMARY\_SCREEN** = -3, **THIS\_SCREEN** = -2, **ALL\_SCREENS** = -1 }

## Public Member Functions

- **screenSelectDialog** (int numScreens, QWidget \*parent=0)
- int **getScreenNum** ()

### Static Public Member Functions

- static bool **getFullscreenGeometry** (QWidget \*target, QRect &geom)

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

- /tmp/epicsqt/trunk/framework/widgets/QElImage/screenSelectDialog.h
- /tmp/epicsqt/trunk/framework/widgets/QElImage/screenSelectDialog.cpp

## 9.111 selectMenu Class Reference

### Public Member Functions

- **selectMenu** (QWidget \*parent=0)
- imageContextMenu::imageContextMenuOptions **getSelectOption** (const QPoint &pos)
- void **enable** (imageContextMenu::imageContextMenuOptions option, bool state)
- bool **isEnabled** (imageContextMenu::imageContextMenuOptions option)
- void **setChecked** (const int mode)
- void **setItemText** (imageContextMenu::imageContextMenuOptions option, QString title)

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

- /tmp/epicsqt/trunk/framework/widgets/QElImage/selectMenu.h
- /tmp/epicsqt/trunk/framework/widgets/QElImage/selectMenu.cpp

## 9.112 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:

- /tmp/epicsqt/trunk/framework/widgets/QEPlot/QEPlot.h

## 9.113 userInfoStruct Class Reference

### Public Attributes

- bool **enable**
- double **value1**
- double **value2**
- QString **elementText**

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

- /tmp/epicsqt/trunk/framework/widgets/QEPeriodic/QEPeriodic.h

## 9.114 QEPeriodic::userInfoStructArray Struct Reference

### Public Attributes

- **userInfoStruct array [NUM\_ELEMENTS]**

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

- /tmp/epicsqt/trunk/framework/widgets/QEPeriodic/QEPeriodic.h

## 9.115 ValueScaling Class Reference

### Public Member Functions

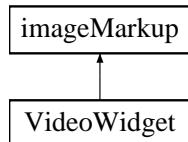
- void **reset ()**
- void **assign** (const **ValueScaling** &s)
- void **set** (const double dIn, const double mIn, const double cIn)
- void **get** (double &dOut, double &mOut, double &cOut) const
- void **map** (const double fromLower, const double fromUpper, const double toLower, const double toUpper)
- bool **isScaled ()** const
- double **value** (const double x) const
- QEDisplayRanges **value** (const QEDisplayRanges &x) const
- void **saveConfiguration** (PMElement &parentElement) const
- void **restoreConfiguration** (PMElement &parentElement)

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

- /tmp/epicsqt/trunk/framework/widgets/QEStripChart/QEStripChartUtilities.h
- /tmp/epicsqt/trunk/framework/widgets/QEStripChart/QEStripChartUtilities.cpp

## 9.116 VideoWidget Class Reference

Inheritance diagram for VideoWidget:



### Signals

- void **userSelection** (imageMarkup::markupIds mode, bool complete, bool clearing, QPoint point1, QPoint point2, unsigned int thickness)
- void **zoomInOut** (int zoomAmount)
- void **currentPixelInfo** (QPoint pos)
- void **pan** (QPoint pos)
- void **redraw** ()

### Public Member Functions

- **VideoWidget** (QWidget \*parent=0)
- void **setNewImage** (QImage image, QCaDateTime &time)
- void **setPanning** (bool panningIn)
- bool **getPanning** ()
- QPoint **scalePoint** (QPoint pnt)
- int **scaleOrdinate** (int ord)
- QPoint **scaleImagePoint** (QPoint pnt)
- QRect **scaleImageRectangle** (QRect r)
- int **scaleImageOrdinate** (int ord)
- QImage **getImage** ()
- QSize **getImageSize** ()
- bool **hasCurrentImage** ()
- void **markupChange** ()

### Protected Member Functions

- void **paintEvent** (QPaintEvent \*)
- void **mousePressEvent** (QMouseEvent \*event)
- void **mouseReleaseEvent** (QMouseEvent \*event)
- void **mouseMoveEvent** (QMouseEvent \*event)
- void **wheelEvent** (QWheelEvent \*event)
- void **keyPressEvent** (QKeyEvent \*event)
- void **markupChange** (QVector<QRect> &changedAreas)

- void **resizeEvent** (QResizeEvent \*event)
- void **markupSetCursor** (QCursor cursor)
- void **markupAction** (markuplds mode, bool complete, bool clearing, QPoint point1, QPoint point2, unsigned int thickness)

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

- /tmp/epicsqt/trunk/framework/widgets/QElImage/videowidget.h
- /tmp/epicsqt/trunk/framework/widgets/QElImage/videowidget.cpp

## 9.117 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:

- /tmp/epicsqt/trunk/framework/widgets/QElImage/zoomMenu.h
- /tmp/epicsqt/trunk/framework/widgets/QElImage/zoomMenu.cpp

# Index

\_CopyPaste, 27  
\_Field, 27  
\_Item, 28  
\_QDialogItem, 29  
\_QPushButtonGroup, 29  
\_QTableWidgetFileBrowser, 29  
\_QTableWidgetLog, 30  
\_QTableWidgetScript, 30

addUnits  
    QEAnalogProgressBar, 74  
    QECheckBox, 93  
    QELabel, 198  
    QELineEdit, 207  
    QENumericEdit, 221  
    QEPushButton, 246  
    QERadioButton, 265

alarmSeverityDisplayMode  
    QEAnalogProgressBar, 74

alignment  
    QECheckBox, 93  
    QEPushButton, 246  
    QERadioButton, 266

allowDrop  
    QEAnalogProgressBar, 74  
    QEBitStatus, 82  
    QECheckBox, 93  
    QEComboBox, 106  
    QEConfiguredLayout, 112  
    QEFileBrowser, 119  
    QEForm, 124  
    QEFrame, 128  
    QEGenericEdit, 137  
    QEGroupBox, 142  
    QEImage, 172  
    QELabel, 198  
    QELog, 215  
    QEPeriodic, 227  
    QEPlot, 236  
    QEPushButton, 246  
    QERadioButton, 266

QEScript, 283  
QEShape, 292  
QESlider, 302  
QESpinBox, 308

altReadbackVariable  
    QEPushButton, 246

Always  
    QEAnalogProgressBar, 72  
    QEBitStatus, 81  
    QECheckBox, 90  
    QEComboBox, 105  
    QEConfiguredLayout, 111  
    QEFileBrowser, 118  
    QEFrame, 127  
    QEGenericEdit, 135  
    QEGroupBox, 141  
    QEImage, 168  
    QELabel, 196  
    QELog, 214  
    QEPeriodic, 226  
    QEPlot, 235  
    QEPushButton, 243  
    QERadioButton, 262

QEScript, 282  
QEShape, 290  
QESlider, 302  
QESpinBox, 308

animation1  
    QEShape, 292

animation2  
    QEShape, 292

animation3  
    QEShape, 292

animation4  
    QEShape, 292

animation5  
    QEShape, 292

animation6  
    QEShape, 292

animationOptions  
    QEShape, 290

Append  
 QEAnalogProgressBar, 72  
 QECheckBox, 89  
 QELabel, 196  
 QELineEdit, 206  
 QEPushButton, 243  
 QERadioButton, 262  
 areaColor  
 QEImage, 172  
 arealInfo, 30  
 arguments  
 QECheckBox, 93  
 QEPushButton, 247  
 QERadioButton, 266  
 arguments1  
 QEImage, 173  
 arguments2  
 QEImage, 173  
 arrayAction  
 QEAnalogProgressBar, 74  
 QECheckBox, 93  
 QELabel, 198  
 QELineEdit, 207  
 QEPushButton, 247  
 QERadioButton, 266  
 ArrayActions  
 QEAnalogProgressBar, 72  
 QECheckBox, 89  
 QELabel, 196  
 QELineEdit, 206  
 QEPushButton, 243  
 QERadioButton, 262  
 arrayIndex  
 QEBitStatus, 82  
 QENumericEdit, 221  
 Ascii  
 QEAnalogProgressBar, 72  
 QECheckBox, 89  
 QELabel, 196  
 QELineEdit, 206  
 QEPushButton, 243  
 QERadioButton, 262  
 autoBrightnessContrast  
 QEImage, 173  
 Automatic  
 QEAnalogProgressBar, 73  
 QECheckBox, 91  
 QELabel, 196  
 QELineEdit, 206  
 QEPushButton, 244  
 QERadioButton, 263  
 autoScale  
 QENumericEdit, 221  
 backgroundColour  
 QEAnalogIndicator, 66  
 Bar  
 QEAnalogIndicator, 66  
 Bayer  
 QEImage, 169  
 BayerBG  
 QEImage, 169  
 BayerGR  
 QEImage, 169  
 BayerRG  
 QEImage, 169  
 beamColor  
 QEImage, 173  
 beamXVariable  
 QEImage, 173  
 beamYVariable  
 QEImage, 173  
 bitDepthVariable  
 QEImage, 173  
 borderColour  
 QEAnalogIndicator, 66  
 Bottom\_To\_Top  
 QEAnalogIndicator, 66  
 BOUNDING\_RECTANGLE  
 QEImage, 169  
 BoundingRectangle  
 QEImage, 168  
 briefInfoArea  
 QEImage, 173  
 buildImageCore  
 imagePropertiesCore, 47  
 CenterAndSize  
 QEImage, 168  
 centreAngle  
 QEAnalogIndicator, 66  
 clickCheckedText  
 QECheckBox, 94  
 QEPushButton, 247  
 QERadioButton, 266  
 clicked  
 QECheckBox, 92  
 QEPushButton, 245

QERadioButton, 265  
clickText  
    QECheckBox, 94  
    QEPushButton, 247  
    QERadioButton, 267  
clippingHighVariable  
    QEImage, 173  
clippingLowVariable  
    QEImage, 174  
clippingOnOffVariable  
    QEImage, 174  
color1  
    QEShape, 292  
color10  
    QEShape, 292  
color2  
    QEShape, 293  
color3  
    QEShape, 293  
color4  
    QEShape, 293  
color5  
    QEShape, 293  
color6  
    QEShape, 293  
color7  
    QEShape, 293  
color8  
    QEShape, 293  
color9  
    QEShape, 293  
Comma  
    QEAnalogProgressBar, 73  
    QECheckBox, 91  
    QELabel, 197  
    QELineEdit, 207  
    QERadioButton, 264  
confirmAction  
    QECheckBox, 94  
    QEPushButton, 247  
    QERadioButton, 267  
confirmText  
    QECheckBox, 94  
    QEPushButton, 248  
    QERadioButton, 267  
confirmWrite  
    QEGenericEdit, 137  
contrastReversal  
    QEImage, 174  
creationOption  
    QECheckBox, 94  
    QEPushButton, 248  
    QERadioButton, 267  
CreationOptionNames  
    QECheckBox, 89  
    QEPushButton, 243  
    QERadioButton, 262  
customisationName  
    QECheckBox, 95  
    QEPushButton, 248  
    QERadioButton, 267  
dbConnectionChanged  
    QEAnalogProgressBar, 74  
    QEBitStatus, 82  
    QENumericEdit, 221  
dbElementChanged  
    QEPeriodic, 227  
dbValueChanged  
    QEAnalogProgressBar, 74  
    QEBitStatus, 82  
    QECheckBox, 92  
    QEComboBox, 106  
    QEImage, 172  
    QELabel, 198  
    QELineEdit, 207  
    QENumericEdit, 221  
    QEPeriodic, 227  
    QEPlot, 235  
    QEPushButton, 245  
    QERadioButton, 265  
    QESlider, 302  
    QESpinBox, 308  
dbValueChanged1  
    QEShape, 291  
dbValueChanged2  
    QEShape, 291  
dbValueChanged3  
    QEShape, 291  
dbValueChanged4  
    QEShape, 291  
dbValueChanged5  
    QEShape, 291  
dbValueChanged6  
    QEShape, 291  
Default  
    QEAnalogProgressBar, 72  
    QECheckBox, 90  
    QELabel, 196  
    QELineEdit, 206

QEPushButton, 244  
 QERadioButton, 263  
 defaultStyle  
   QEAnalogProgressBar, 75  
   QEBitStatus, 82  
   QECheckBox, 95  
   QEComboBox, 106  
   QEConfiguredLayout, 112  
   QEFileBrowser, 119  
   QEFrame, 128  
   QEGenericEdit, 137  
   QEGroupBox, 142  
   QEImage, 174  
   QELabel, 199  
   QELog, 215  
   QEPlot, 236  
   QEPushButton, 248  
   QERadioButton, 267  
   QEScript, 283  
   QEShape, 293  
   QESlider, 302  
   QESpinBox, 308  
 dimension1Variable  
   QEImage, 174  
 dimension2Variable  
   QEImage, 174  
 dimension3Variable  
   QEImage, 174  
 dimensionsVariable  
   QEImage, 174  
 displayAlarmState  
   QEAnalogProgressBar, 75  
   QEBitStatus, 83  
   QECheckBox, 95  
   QEComboBox, 107  
   QEConfiguredLayout, 112  
   QEFileBrowser, 119  
   QEFrame, 128  
   QEGenericEdit, 137  
   QEGroupBox, 142  
   QEImage, 175  
   QELabel, 199  
   QELog, 215  
   QEPeriodic, 227  
   QEPlot, 236  
   QEPushButton, 248  
   QERadioButton, 268  
   QEScript, 283  
   QEShape, 294  
   QESlider, 303  
   QESpinBox, 309  
 DisplayAlarmStateOptions  
   QEAnalogProgressBar, 72  
   QEBitStatus, 81  
   QECheckBox, 90  
   QEComboBox, 105  
   QEConfiguredLayout, 111  
   QEFileBrowser, 118  
   QEFrame, 127  
   QEGenericEdit, 135  
   QEGroupBox, 141  
   QEImage, 168  
   QELabel, 196  
   QELog, 214  
   QEPeriodic, 226  
   QEPlot, 235  
   QEPushButton, 243  
   QERadioButton, 262  
   QEScript, 282  
   QEShape, 290  
   QESlider, 301  
   QESpinBox, 308  
 displayArea1Selection  
   QEImage, 175  
 displayArea2Selection  
   QEImage, 175  
 displayArea3Selection  
   QEImage, 175

displayArea4Selection  
QEImage, 175  
displayBeamSelection  
QEImage, 175  
displayButtonBar  
QEImage, 172  
displayCursorPixelInfo  
QEImage, 176  
displayEllipse  
QEImage, 176  
displayHozSlice1Selection  
QEImage, 176  
displayHozSlice2Selection  
QEImage, 176  
displayHozSlice3Selection  
QEImage, 176  
displayHozSlice4Selection  
QEImage, 176  
displayHozSlice5Selection  
QEImage, 176  
displayProfileSelection  
QEImage, 176  
displayTargetSelection  
QEImage, 176  
displayVertSliceSelection  
QEImage, 177  
DockBottom  
QECheckBox, 90  
QEPushButton, 243  
QERadioButton, 262  
DockBottomTabbed  
QECheckBox, 90  
QEPushButton, 243  
QERadioButton, 262  
DockFloating  
QECheckBox, 90  
QEPushButton, 243  
QERadioButton, 262  
DockLeft  
QECheckBox, 90  
QEPushButton, 243  
QERadioButton, 262  
DockLeftTabbed  
QECheckBox, 90  
QEPushButton, 243  
QERadioButton, 262  
DockRight  
QECheckBox, 90  
QEPushButton, 243  
QERadioButton, 262

DockRightTabbed  
QECheckBox, 90  
QEPushButton, 243  
QERadioButton, 262  
DockTop  
QECheckBox, 90  
QEPushButton, 243  
QERadioButton, 262

DockTopTabbed  
QECheckBox, 90  
QEPushButton, 243  
QERadioButton, 262

DottedFullCrosshair  
QEImage, 171  
drawMarkup  
markupHLine, 51  
markupVLine, 56

ellipseColor  
QEImage, 177  
ellipseHVariable  
QEImage, 177  
EllipseVariableDefinitions  
QEImage, 168  
ellipseVariableDefinitions  
QEImage, 168  
ellipseWVariable  
QEImage, 177  
ellipseXVariable  
QEImage, 177  
ellipseYVariable  
QEImage, 177  
enableArea1Selection  
QEImage, 177  
enableArea2Selection  
QEImage, 178  
enableArea3Selection  
QEImage, 178  
enableArea4Selection  
QEImage, 178  
enableBeamSelection  
QEImage, 178  
enableHozSlice1Selection  
QEImage, 178  
enableHozSlice2Selection  
QEImage, 178  
enableHozSlice3Selection  
QEImage, 178  
enableHozSlice4Selection  
QEImage, 178

enableHozSlice5Selection  
     QEImage, 179  
 enableProfileSelection  
     QEImage, 179  
 enableTargetSelection  
     QEImage, 179  
 enableVertSlice1Selection  
     QEImage, 179  
 enableVertSlice2Selection  
     QEImage, 179  
 enableVertSlice3Selection  
     QEImage, 179  
 enableVertSlice4Selection  
     QEImage, 179  
 enableVertSlice5Selection  
     QEImage, 180  
 Engineer  
     QEAnalogProgressBar, 73  
     QEBitStatus, 82  
     QECheckBox, 92  
     QEComboBox, 106  
     QEConfiguredLayout, 112  
     QEFileBrowser, 118  
     QEFrame, 127  
     QEGenericEdit, 135  
     QEGroupBox, 142  
     QEImage, 171  
     QELabel, 197  
     QELog, 215  
     QEPeriodic, 227  
     QEPlot, 235  
     QEPushButton, 245  
     QERadioButton, 264  
     QEScript, 282  
     QEShape, 290  
     QESlider, 302  
     QESpinBox, 308  
 externalControls  
     QEImage, 180  
  
 FFBuffer, 32  
 FFThread, 32  
 Fit  
     QEImage, 170  
 Fixed  
     QEAnalogProgressBar, 73  
     QECheckBox, 91  
     QELabel, 196  
     QELineEdit, 206  
     QEPushButton, 244  
  
 QERadioButton, 263  
 flipRotateMenu, 33  
 Floating  
     QEAnalogProgressBar, 72  
     QECheckBox, 90  
     QELabel, 196  
     QELineEdit, 206  
     QEPushButton, 244  
     QERadioButton, 263  
 fontColour  
     QEAnalogIndicator, 66  
 foregroundColour  
     QEAnalogIndicator, 67  
 format  
     QEAnalogProgressBar, 75  
     QECheckBox, 95  
     QELabel, 199  
     QELineEdit, 207  
     QEPushButton, 249  
     QERadioButton, 268  
 formatOption  
     QEImage, 180  
 FormatOptions  
     QEImage, 169  
 Formats  
     QEAnalogProgressBar, 72  
     QECheckBox, 90  
     QELabel, 196  
     QELineEdit, 206  
     QEPushButton, 243  
     QERadioButton, 262  
 formatVariable  
     QEImage, 180  
 fullScreenWindow, 33  
  
 getConfirmWrite  
     QEGenericEdit, 135  
 getPixelValueFromData  
     imageProcessor, 44  
 getSubscribe  
     QEGenericEdit, 135  
 getWriteOnEnter  
     QEGenericEdit, 136  
 getWriteOnFinish  
     QEGenericEdit, 136  
 getWriteOnLoseFocus  
     QEGenericEdit, 136  
 guiFile  
     QECheckBox, 96  
     QEPushButton, 249

QERadioButton, 268  
handleGuiLaunchRequests  
    QEForm, 123  
heightVariable  
    QEImage, 180  
histogram, 34  
histogramScroll, 34  
historicImage, 34  
horizontalFlip  
    QEImage, 180  
hozSlice1Color  
    QEImage, 180  
hozSlice2Color  
    QEImage, 180  
hozSlice3Color  
    QEImage, 180  
hozSlice4Color  
    QEImage, 181  
hozSlice5Color  
    QEImage, 181

Icon  
    QECheckBox, 91  
    QEPushButton, 245  
    QERadioButton, 264  
imageContextMenu, 35  
imageDisplayProperties, 36  
imageDisplayProperties::rgbPixel, 322  
imageInfo, 37  
imageMarkup, 38  
imageMarkupLegendSetText, 40  
imageProcessor, 41  
    getPixelValueFromData, 44  
imageProperties, 44  
    imageProperties, 46  
    ROTATION\_0, 46  
    ROTATION\_180, 46  
    ROTATION\_90\_LEFT, 46  
    ROTATION\_90\_RIGHT, 46  
    rotationOptions, 46  
imagePropertiesCore, 46  
    buildImageCore, 47  
imageUpdateIndicator, 47  
imageVariable  
    QEImage, 181

Index  
    QEAnalogProgressBar, 72  
    QECheckBox, 89  
    QELabel, 196

    QELineEdit, 206  
    QEPushButton, 243  
    QERadioButton, 262  
initialHosScrollPos  
    QEImage, 181  
initialVertScrollPos  
    QEImage, 172  
int  
    QEAnalogProgressBar, 75  
    QEBitStatus, 83  
    QECheckBox, 96  
    QEComboBox, 107  
    QEConfiguredLayout, 113  
    QEFileBrowser, 119  
    QEForm, 124  
    QEFrame, 128  
    QEGenericEdit, 137  
    QEGroupBox, 143  
    QEImage, 181  
    QELabel, 199  
    QELineEdit, 208  
    QELog, 216  
    QEPeriodic, 228  
    QEPlot, 236  
    QEPushButton, 249  
    QERadioButton, 268  
    QEScript, 283  
    QEShape, 294  
    QESlider, 303  
    QESpinBox, 309

Integer  
    QEAnalogProgressBar, 72  
    QECheckBox, 90  
    QELabel, 196  
    QELineEdit, 206  
    QEPushButton, 244  
    QERadioButton, 263

labelText  
    QECheckBox, 96  
    QEPushButton, 249  
    QERadioButton, 268  
    QESubstitutedLabel, 321

leadingZero  
    QEAnalogProgressBar, 76  
    QECheckBox, 96  
    QELabel, 200  
    QELineEdit, 208  
    QEPushButton, 250  
    QERadioButton, 269

leadingZeros  
     QENumericEdit, 221

Left\_To\_Right  
     QEAnalogIndicator, 66

lineProfileArrayVariable  
     QEImage, 181

lineProfileThicknessVariable  
     QEImage, 181

lineProfileX1Variable  
     QEImage, 181

lineProfileX2Variable  
     QEImage, 182

lineProfileY1Variable  
     QEImage, 182

lineProfileY2Variable  
     QEImage, 182

LocalEnumeration  
     QEAnalogProgressBar, 73

QECheckBox, 90

QELabel, 196

QELineEdit, 206

QEPushButton, 244

QERadioButton, 263

localEnumeration  
     QEAnalogProgressBar, 76

QECheckBox, 96

QEComboBox, 107

QELabel, 200

QELineEdit, 208

QEPushButton, 250

QERadioButton, 269

logBrightness  
     QEImage, 182

loginWidget, 47

LogOutput  
     QECheckBox, 91

QEImage, 169

QEPushButton, 244

QERadioButton, 263

logScale  
     QEAnalogIndicator, 67

logScaleInterval  
     QEAnalogIndicator, 67

majorInterval  
     QEAnalogIndicator, 67

markupCrosshair1, 48

markupCrosshair2, 48

markupDisplayMenu, 49

markupEllipse, 49

markupHLine, 50  
     drawMarkup, 51

markupItem, 51

markupLine, 54

markupRegion, 54

markupText, 55

markupVLine, 56  
     drawMarkup, 56

maximum  
     QEAnalogIndicator, 67

QENumericEdit, 222

messageFormFilter  
     QEForm, 124

messageSourceFilter  
     QEForm, 124

Meter  
     QEAnalogIndicator, 66

minimum  
     QEAnalogIndicator, 67

QENumericEdit, 222

minorInterval  
     QEAnalogIndicator, 67

mode  
     QEAnalogIndicator, 67

Modes  
     QEAnalogIndicator, 66

Mono  
     QEImage, 169

mpegSource, 57  
     updateImage, 57

mpegSourceObject, 57

Never  
     QEAnalogProgressBar, 72

QEBitStatus, 81

QECheckBox, 90

QEComboBox, 105

QEConfiguredLayout, 111

QEFileBrowser, 118

QEFrame, 127

QEGenericEdit, 135

QEGroupBox, 141

QEImage, 168

QELabel, 196

QELog, 214

QEPeriodic, 226

QEPlot, 235

QEPushButton, 243

QERadioButton, 262

QEScript, 282

QEShape, 290  
QESlider, 302  
QESpinBox, 308  
NewTab  
    QECheckBox, 89  
    QEPushButton, 243  
    QERadioButton, 262  
NewWindow  
    QECheckBox, 90  
    QEPushButton, 243  
    QERadioButton, 262  
None  
    QECheckBox, 91  
    QEImage, 169  
    QEPushButton, 244  
    QERadioButton, 263  
NoRotation  
    QEImage, 170  
NoSeparator  
    QEAnalogProgressBar, 73  
    QECheckBox, 91  
    QELabel, 197  
    QELineEdit, 206  
    QERadioButton, 264  
notation  
    QEAnalogProgressBar, 76  
    QECheckBox, 97  
    QELabel, 200  
    QELineEdit, 208  
    QEPushButton, 250  
    QERadioButton, 269  
Notations  
    QEAnalogProgressBar, 73  
    QECheckBox, 90  
    QELabel, 196  
    QELineEdit, 206  
    QEPushButton, 244  
    QERadioButton, 263  
offset1  
    QEShape, 294  
offset2  
    QEShape, 294  
offset3  
    QEShape, 294  
offset4  
    QEShape, 295  
offset5  
    QEShape, 295  
offset6  
    QEShape, 295  
Open  
    QECheckBox, 89  
    QEPushButton, 243  
    QERadioButton, 262  
orientation  
    QEAnalogIndicator, 67  
Orientations  
    QEAnalogIndicator, 66  
password  
    QECheckBox, 97  
    QEPushButton, 250  
    QERadioButton, 270  
PeriodicDialog, 58  
PeriodicElementSetupForm, 59  
PeriodicSetupDialog, 59  
Picture  
    QELabel, 197  
 pixmap0  
    QECheckBox, 97  
    QELabel, 200  
    QEPushButton, 250  
    QERadioButton, 270  
 pixmap1  
    QECheckBox, 97  
    QELabel, 200  
    QEPushButton, 251  
    QERadioButton, 270  
 pixmap2  
    QECheckBox, 97  
    QELabel, 201  
    QEPushButton, 251  
    QERadioButton, 270  
 pixmap3  
    QECheckBox, 98  
    QELabel, 201  
    QEPushButton, 251  
    QERadioButton, 270  
 pixmap4  
    QECheckBox, 98  
    QELabel, 201  
    QEPushButton, 251  
    QERadioButton, 270  
 pixmap5  
    QECheckBox, 98  
    QELabel, 201  
    QEPushButton, 251  
    QERadioButton, 270  
 pixmap6

QECheckBox, 98  
QELabel, 201  
QEPushButton, 251  
QERadioButton, 270  
pixmap7  
    QECheckBox, 98  
    QELabel, 201  
    QEPushButton, 251  
    QERadioButton, 270  
playbackTimer, 59  
point1  
    QEShape, 295  
point10  
    QEShape, 295  
point2  
    QEShape, 295  
point3  
    QEShape, 295  
point4  
    QEShape, 295  
point5  
    QEShape, 295  
point6  
    QEShape, 296  
point7  
    QEShape, 296  
point8  
    QEShape, 296  
point9  
    QEShape, 296  
pointInfo, 60  
precision  
    QEAnalogProgressBar, 77  
    QECheckBox, 98  
    QELabel, 201  
    QELineEdit, 209  
    QENumericEdit, 222  
    QEPushButton, 251  
    QERadioButton, 271  
pressed  
    QECheckBox, 92  
    QEPushButton, 245  
    QERadioButton, 265  
pressText  
    QECheckBox, 98  
    QEPushButton, 251  
    QERadioButton, 271  
prioritySubstitutions  
    QECheckBox, 98  
    QEPushButton, 252  
    QERadioButton, 271  
QERadioButton, 271  
profileColor  
    QEImage, 182  
profileHoz1ThicknessVariable  
    QEImage, 182  
profileHoz1Variable  
    QEImage, 182  
profileHoz2ThicknessVariable  
    QEImage, 182  
profileHoz2Variable  
    QEImage, 182  
profileHoz3ThicknessVariable  
    QEImage, 183  
profileHoz3Variable  
    QEImage, 183  
profileHoz4ThicknessVariable  
    QEImage, 183  
profileHoz4Variable  
    QEImage, 183  
profileHoz5ThicknessVariable  
    QEImage, 183  
profileHoz5Variable  
    QEImage, 183  
profileHozArrayVariable  
    QEImage, 183  
profilePlot, 60  
profileVert1ThicknessVariable  
    QEImage, 183  
profileVert1Variable  
    QEImage, 183  
profileVert2ThicknessVariable  
    QEImage, 184  
profileVert2Variable  
    QEImage, 184  
profileVert3ThicknessVariable  
    QEImage, 184  
profileVert3Variable  
    QEImage, 184  
profileVert4ThicknessVariable  
    QEImage, 184  
profileVert4Variable  
    QEImage, 184  
profileVert5ThicknessVariable  
    QEImage, 184  
profileVert5Variable  
    QEImage, 184  
profileVertArrayListVariable  
    QEImage, 184  
program  
    QECheckBox, 99

QEPushButton, 252  
QERadioButton, 271  
program1  
    QEImage, 185  
program2  
    QEImage, 185  
programStartupOption  
    QECheckBox, 99  
    QEPushButton, 252  
    QERadioButton, 271  
programStartupOption1  
    QEImage, 185  
programStartupOption2  
    QEImage, 185  
ProgramStartupOptionNames  
    QECheckBox, 91  
    QEImage, 169  
    QEPushButton, 244  
    QERadioButton, 263  
  
QBitStatus, 61  
QEAnalogIndicator, 63  
    backgroundColour, 66  
    Bar, 66  
    borderColour, 66  
    Bottom\_To\_Top, 66  
    centreAngle, 66  
    fontColour, 66  
    foregroundColour, 67  
    Left\_To\_Right, 66  
    logScale, 67  
    logScaleInterval, 67  
    majorInterval, 67  
    maximum, 67  
    Meter, 66  
    minimum, 67  
    minorInterval, 67  
    mode, 67  
    Modes, 66  
    orientation, 67  
    Orientations, 66  
    Right\_To\_Left, 66  
    Scale, 66  
    showScale, 67  
    showText, 68  
    spanAngle, 68  
    Top\_To\_Bottom, 66  
    value, 68  
QEAnalogIndicator::Band, 31  
QEAnalogIndicator::BandList, 31  
  
QEAnalogProgressBar, 68  
    addUnits, 74  
    alarmSeverityDisplayMode, 74  
    allowDrop, 74  
    Always, 72  
    Append, 72  
    arrayAction, 74  
    ArrayActions, 72  
    Ascii, 72  
    Automatic, 73  
    Comma, 73  
    dbConnectionChanged, 74  
    dbValueChanged, 74  
    Default, 72  
    defaultStyle, 75  
    displayAlarmState, 75  
    displayAlarmStateOption, 75  
    DisplayAlarmStateOptions, 72  
    Engineer, 73  
    Fixed, 73  
    Floating, 72  
    format, 75  
    Formats, 72  
    Index, 72  
    int, 75  
    Integer, 72  
    leadingZero, 76  
    LocalEnumeration, 73  
    localEnumeration, 76  
    Never, 72  
    NoSeparator, 73  
    notation, 76  
    Notations, 73  
    precision, 77  
    QEAnalogProgressBar, 73  
    radix, 77  
    Scientific, 73  
    Scientist, 73  
    separator, 77  
    Separators, 73  
    setManagedVisible, 74  
    Space, 73  
    styleSheet, 77  
    Time, 73  
    trailingZeros, 77  
    Underscore, 73  
    UnsignedInteger, 73  
    useDbDisplayLimits, 77  
    useDbPrecision, 77  
    User, 73

userLevelEnabled, 77  
userLevelEngineerStyle, 77  
UserLevels, 73  
userLevelScientistStyle, 78  
userLevelUserStyle, 78  
userLevelVisibility, 78  
value, 78  
variable, 78  
variableAsToolTip, 78  
variableSubstitutions, 79  
visible, 79  
WhenInAlarm, 72  
QEBitStatus, 79  
    allowDrop, 82  
    Always, 81  
    arrayIndex, 82  
    dbConnectionChanged, 82  
    dbValueChanged, 82  
    defaultStyle, 82  
    displayAlarmState, 83  
    displayAlarmStateOption, 83  
    DisplayAlarmStateOptions, 81  
    Engineer, 82  
    int, 83  
    Never, 81  
    Scientist, 82  
    setManagedVisible, 82  
    styleSheet, 83  
    User, 82  
    userLevelEnabled, 83  
    userLevelEngineerStyle, 83  
    UserLevels, 81  
    userLevelScientistStyle, 84  
    userLevelUserStyle, 84  
    userLevelVisibility, 84  
    variable, 84  
    variableAsToolTip, 84  
    variableSubstitutions, 84  
    visible, 85  
    WhenInAlarm, 81  
QECheckBox, 85  
    addUnits, 93  
    alignment, 93  
    allowDrop, 93  
    Always, 90  
    Append, 89  
    arguments, 93  
    arrayAction, 93  
    ArrayActions, 89  
    Ascii, 89  
Automatic, 91  
clickCheckedText, 94  
clicked, 92  
clickText, 94  
Comma, 91  
confirmAction, 94  
confirmText, 94  
creationOption, 94  
CreationOptionNames, 89  
customisationName, 95  
dbValueChanged, 92  
Default, 90  
defaultStyle, 95  
displayAlarmState, 95  
displayAlarmStateOption, 95  
DisplayAlarmStateOptions, 90  
DockBottom, 90  
DockBottomTabbed, 90  
DockFloating, 90  
DockLeft, 90  
DockLeftTabbed, 90  
DockRight, 90  
DockRightTabbed, 90  
DockTop, 90  
DockTopTabbed, 90  
Engineer, 92  
Fixed, 91  
Floating, 90  
format, 95  
Formats, 90  
guiFile, 96  
Icon, 91  
Index, 89  
int, 96  
Integer, 90  
labelText, 96  
leadingZero, 96  
LocalEnumeration, 90  
localEnumeration, 96  
LogOutput, 91  
Never, 90  
NewTab, 89  
NewWindow, 90  
None, 91  
NoSeparator, 91  
notation, 97  
Notations, 90  
Open, 89  
password, 97  
pixmap0, 97

pixmap1, 97  
 pixmap2, 97  
 pixmap3, 98  
 pixmap4, 98  
 pixmap5, 98  
 pixmap6, 98  
 pixmap7, 98  
 precision, 98  
 pressed, 92  
 pressText, 98  
 prioritySubstitutions, 98  
 program, 99  
 programStartupOption, 99  
 ProgramStartupOptionNames, 91  
 QECheckBox, 92  
 radix, 99  
 released, 92  
 releaseText, 99  
 requestAction, 93  
 Scientific, 91  
 Scientist, 92  
 separator, 99  
 Separators, 91  
 setManagedVisible, 93  
 Space, 91  
 State, 91  
 StdOutput, 91  
 styleSheet, 99  
 subscribe, 99  
 Terminal, 91  
 Text, 91  
 TextAndIcon, 91  
 Time, 90  
 trailingZeros, 100  
 Underscore, 91  
 UnsignedInteger, 90  
 updateOption, 100  
 UpdateOptions, 91  
 useDbPrecision, 100  
 User, 92  
 userLevelEnabled, 100  
 userLevelEngineerStyle, 100  
 UserLevels, 91  
 userLevelScientistStyle, 100  
 userLevelUserStyle, 101  
 userLevelVisibility, 101  
 variable, 101  
 variableAsToolTip, 101  
 variableSubstitutions, 101  
 visible, 101  
 WhenInAlarm, 90  
 writeOnClick, 101  
 writeOnPress, 102  
 writeOnRelease, 102  
 QECheckBoxManager, 102  
 QEComboBox, 103  
 allowDrop, 106  
 Always, 105  
 dbValueChanged, 106  
 defaultStyle, 106  
 displayAlarmState, 107  
 displayAlarmStateOption, 107  
 DisplayAlarmStateOptions, 105  
 Engineer, 106  
 int, 107  
 localEnumeration, 107  
 Never, 105  
 Scientist, 106  
 setManagedVisible, 106  
 styleSheet, 107  
 subscribe, 107  
 useDbEnumerations, 106  
 User, 106  
 userLevelEnabled, 107  
 userLevelEngineerStyle, 108  
 UserLevels, 105  
 userLevelScientistStyle, 108  
 userLevelUserStyle, 108  
 userLevelVisibility, 108  
 variable, 108  
 variableAsToolTip, 109  
 variableSubstitutions, 109  
 visible, 109  
 WhenInAlarm, 105  
 writeOnChange, 106  
 QEConfiguredLayout, 109  
 allowDrop, 112  
 Always, 111  
 defaultStyle, 112  
 displayAlarmState, 112  
 displayAlarmStateOption, 112  
 DisplayAlarmStateOptions, 111  
 Engineer, 112  
 int, 113  
 Never, 111  
 Scientist, 112  
 setManagedVisible, 112  
 styleSheet, 113  
 User, 112  
 userLevelEnabled, 113

userLevelEngineerStyle, 113  
UserLevels, 111  
userLevelScientistStyle, 113  
userLevelUserStyle, 113  
userLevelVisibility, 114  
variableAsToolTip, 114  
visible, 114  
WhenInAlarm, 111  
QEConfiguredLayoutManager, 114  
QEFileBrowser, 115  
allowDrop, 119  
Always, 118  
defaultStyle, 119  
displayAlarmState, 119  
displayAlarmStateOption, 119  
DisplayAlarmStateOptions, 118  
Engineer, 118  
int, 119  
Never, 118  
Scientist, 118  
selected, 118  
setManagedVisible, 118  
styleSheet, 119  
User, 118  
userLevelEnabled, 120  
userLevelEngineerStyle, 120  
UserLevels, 118  
userLevelScientistStyle, 120  
userLevelUserStyle, 120  
userLevelVisibility, 120  
variable, 121  
variableAsToolTip, 121  
variableSubstitutions, 121  
visible, 121  
WhenInAlarm, 118  
QEForm, 121  
allowDrop, 124  
displayAlarmStateOption, 124  
handleGuiLaunchRequests, 123  
int, 124  
messageFormFilter, 124  
messageSourceFilter, 124  
resizeContents, 123  
uiFile, 125  
variableAsToolTip, 125  
variableSubstitutions, 125  
QEFrame, 125  
allowDrop, 128  
Always, 127  
defaultStyle, 128  
displayAlarmState, 128  
displayAlarmStateOption, 128  
DisplayAlarmStateOptions, 127  
Engineer, 127  
int, 128  
Never, 127  
Scientist, 127  
setManagedVisible, 127  
styleSheet, 128  
User, 127  
userLevelEnabled, 128  
userLevelEngineerStyle, 129  
UserLevels, 127  
userLevelScientistStyle, 129  
userLevelUserStyle, 129  
userLevelVisibility, 129  
variableAsToolTip, 129  
visible, 130  
WhenInAlarm, 127  
QEGenericButton, 130  
QEGenericEdit, 132  
allowDrop, 137  
Always, 135  
confirmWrite, 137  
defaultStyle, 137  
displayAlarmState, 137  
displayAlarmStateOption, 137  
DisplayAlarmStateOptions, 135  
Engineer, 135  
getConfirmWrite, 135  
getSubscribe, 135  
getWriteOnEnter, 136  
getWriteOnFinish, 136  
getWriteOnLoseFocus, 136  
int, 137  
Never, 135  
QEGenericEdit, 135  
Scientist, 135  
setConfirmWrite, 136  
setManagedVisible, 136  
setSubscribe, 136  
setWriteOnEnter, 136  
setWriteOnFinish, 136  
setWriteOnLoseFocus, 136  
styleSheet, 138  
subscribe, 138  
User, 135  
userLevelEnabled, 138  
userLevelEngineerStyle, 138  
UserLevels, 135

userLevelScientistStyle, 138  
userLevelUserStyle, 138  
userLevelVisibility, 139  
variable, 139  
variableAsToolTip, 139  
variableSubstitutions, 139  
visible, 139  
WhenInAlarm, 135  
writeOnEnter, 139  
writeOnFinish, 139  
writeOnLoseFocus, 140

QEGroupBox, 140  
allowDrop, 142  
Always, 141  
defaultStyle, 142  
displayAlarmState, 142  
displayAlarmStateOption, 142  
DisplayAlarmStateOptions, 141  
Engineer, 142  
int, 143  
Never, 141  
Scientist, 142  
setManagedVisible, 142  
styleSheet, 143  
substitutedTitle, 143  
textSubstitutions, 143  
User, 142  
userLevelEnabled, 143  
userLevelEngineerStyle, 143  
UserLevels, 141  
userLevelScientistStyle, 144  
userLevelUserStyle, 144  
userLevelVisibility, 144  
variableAsToolTip, 144  
visible, 144  
WhenInAlarm, 141

QEImage, 145  
allowDrop, 172  
Always, 168  
areaColor, 172  
arguments1, 173  
arguments2, 173  
autoBrightnessContrast, 173  
Bayer, 169  
BayerBG, 169  
BayerGB, 169  
BayerGR, 169  
BayerRG, 169  
beamColor, 173  
beamXVariable, 173

beamYVariable, 173  
bitDepthVariable, 173  
BOUNDING\_RECTANGLE, 169  
BoundingRectangle, 168  
briefInfoArea, 173  
CenterAndSize, 168  
clippingHighVariable, 173  
clippingLowVariable, 174  
clippingOnOffVariable, 174  
contrastReversal, 174  
dbValueChanged, 172  
defaultStyle, 174  
dimension1Variable, 174  
dimension2Variable, 174  
dimension3Variable, 174  
dimensionsVariable, 174  
displayAlarmState, 175  
displayAlarmStateOption, 175  
DisplayAlarmStateOptions, 168  
displayArea1Selection, 175  
displayArea2Selection, 175  
displayArea3Selection, 175  
displayArea4Selection, 175  
displayBeamSelection, 175  
displayButtonBar, 172  
displayCursorPixelInfo, 176  
displayEllipse, 176  
displayHozSlice1Selection, 176  
displayHozSlice2Selection, 176  
displayHozSlice3Selection, 176  
displayHozSlice4Selection, 176  
displayHozSlice5Selection, 176  
displayProfileSelection, 176  
displayTargetSelection, 176  
displayVertSliceSelection, 177  
DottedFullCrosshair, 171  
ellipseColor, 177  
ellipseHVariable, 177  
EllipseVariableDefinitions, 168  
ellipseVariableDefinitions, 168  
ellipseWVariable, 177  
ellipseXVariable, 177  
ellipseYVariable, 177  
enableArea1Selection, 177  
enableArea2Selection, 178  
enableArea3Selection, 178  
enableArea4Selection, 178  
enableBeamSelection, 178  
enableHozSlice1Selection, 178  
enableHozSlice2Selection, 178

enableHozSlice3Selection, 178  
enableHozSlice4Selection, 178  
enableHozSlice5Selection, 179  
enableProfileSelection, 179  
enableTargetSelection, 179  
enableVertSlice1Selection, 179  
enableVertSlice2Selection, 179  
enableVertSlice3Selection, 179  
enableVertSlice4Selection, 179  
enableVertSlice5Selection, 180  
Engineer, 171  
externalControls, 180  
Fit, 170  
formatOption, 180  
FormatOptions, 169  
formatVariable, 180  
heightVariable, 180  
horizontalFlip, 180  
hozSlice1Color, 180  
hozSlice2Color, 180  
hozSlice3Color, 180  
hozSlice4Color, 181  
hozSlice5Color, 181  
imageVariable, 181  
initialHosScrollPos, 181  
initialVertScrollPos, 172  
int, 181  
lineProfileArrayVariable, 181  
lineProfileThicknessVariable, 181  
lineProfileX1Variable, 181  
lineProfileX2Variable, 182  
lineProfileY1Variable, 182  
lineProfileY2Variable, 182  
logBrightness, 182  
LogOutput, 169  
Mono, 169  
Never, 168  
None, 169  
NoRotation, 170  
profileColor, 182  
profileHoz1ThicknessVariable, 182  
profileHoz1Variable, 182  
profileHoz2ThicknessVariable, 182  
profileHoz2Variable, 182  
profileHoz3ThicknessVariable, 183  
profileHoz3Variable, 183  
profileHoz4ThicknessVariable, 183  
profileHoz4Variable, 183  
profileHoz5ThicknessVariable, 183  
profileHoz5Variable, 183  
profileHozArrayVariable, 183  
profileVert1ThicknessVariable, 183  
profileVert1Variable, 183  
profileVert2ThicknessVariable, 184  
profileVert2Variable, 184  
profileVert3ThicknessVariable, 184  
profileVert3Variable, 184  
profileVert4ThicknessVariable, 184  
profileVert4Variable, 184  
profileVert5ThicknessVariable, 184  
profileVert5Variable, 184  
profileVertArrayVariable, 184  
program1, 185  
program2, 185  
programStartupOption1, 185  
programStartupOption2, 185  
ProgramStartupOptionNames, 169  
QEImage, 171  
regionOfInterest1HVariable, 185  
regionOfInterest1WVariable, 185  
regionOfInterest1XVariable, 185  
regionOfInterest1YVariable, 186  
regionOfInterest2HVariable, 186  
regionOfInterest2WVariable, 186  
regionOfInterest2XVariable, 186  
regionOfInterest2YVariable, 186  
regionOfInterest3HVariable, 186  
regionOfInterest3WVariable, 186  
regionOfInterest3XVariable, 186  
regionOfInterest3YVariable, 186  
regionOfInterest4HVariable, 187  
regionOfInterest4WVariable, 187  
regionOfInterest4XVariable, 187  
regionOfInterest4YVariable, 187  
RESIZE\_OPTION\_FIT, 170  
RESIZE\_OPTION\_ZOOM, 170  
resizeOption, 187  
ResizeOptions, 169  
resizeOptions, 170  
rgb1, 169  
rgb2, 169  
rgb3, 169  
Rotate180, 170  
Rotate90Left, 170  
Rotate90Right, 170  
rotation, 187  
RotationOptions, 170  
Scientist, 171  
selectOptions, 170  
setImageFile, 172

setManagedVisible, 172  
showTime, 187  
SO\_AREA4, 171  
SO\_BEAM, 171  
SO\_HSLICE1, 170  
SO\_HSLICE2, 170  
SO\_HSLICE3, 171  
SO\_HSLICE4, 171  
SO\_HSLICE5, 171  
SO\_NONE, 170  
SO\_PANNING, 170  
SO\_PROFILE, 171  
SO\_TARGET, 171  
SO\_VSLICE1, 170  
SO\_VSLICE2, 170  
SO\_VSLICE3, 170  
SO\_VSLICE4, 170  
SO\_VSLICE5, 170  
SolidSmallCrosshair, 171  
StdOutput, 169  
styleSheet, 187  
targetColor, 187  
TargetOptions, 171  
targetTriggerVariable, 188  
targetXVariable, 188  
targetYVariable, 188  
Terminal, 169  
timeColor, 188  
URL, 188  
useFalseColour, 188  
User, 171  
userLevelEnabled, 188  
userLevelEngineerStyle, 188  
UserLevels, 171  
userLevelScientistStyle, 189  
userLevelUserStyle, 189  
userLevelVisibility, 189  
variableAsToolTip, 189  
variableSubstitutions, 189  
verticalFlip, 189  
vertSlice1Color, 190  
vertSlice2Color, 190  
vertSlice3Color, 190  
vertSlice4Color, 190  
vertSlice5Color, 190  
visible, 190  
WhenInAlarm, 168  
widthVariable, 190  
yuv422, 169  
yuv444, 169  
Zoom, 170  
QEImageMarkupThickness, 190  
QEImageOptionsDialog, 191  
QELabel, 191  
    addUnits, 198  
    allowDrop, 198  
    Always, 196  
    Append, 196  
    arrayAction, 198  
    ArrayActions, 196  
    Ascii, 196  
    Automatic, 196  
    Comma, 197  
    dbValueChanged, 198  
    Default, 196  
    defaultStyle, 199  
    displayAlarmState, 199  
    displayAlarmStateOption, 199  
    DisplayAlarmStateOptions, 196  
    Engineer, 197  
    Fixed, 196  
    Floating, 196  
    format, 199  
    Formats, 196  
    Index, 196  
    int, 199  
    Integer, 196  
    leadingZero, 200  
    LocalEnumeration, 196  
    localEnumeration, 200  
    Never, 196  
    NoSeparator, 197  
    notation, 200  
    Notations, 196  
    Picture, 197  
     pixmap0, 200  
     pixmap1, 200  
     pixmap2, 201  
     pixmap3, 201  
     pixmap4, 201  
     pixmap5, 201  
     pixmap6, 201  
     pixmap7, 201  
    precision, 201  
    QELabel, 198  
    radix, 201  
    Scientific, 196  
    Scientist, 197  
    separator, 201  
    Separators, 196

setManagedVisible, 198  
Space, 197  
styleSheet, 202  
Text, 197  
Time, 196  
trailingZeros, 202  
Underscore, 197  
UnsignedInteger, 196  
UPDATE\_PIXMAP, 197  
UPDATE\_TEXT, 197  
updateOption, 202  
UpdateOptions, 197  
updateOptions, 197  
useDbPrecision, 202  
User, 197  
userLevelEnabled, 202  
userLevelEngineerStyle, 202  
UserLevels, 197  
userLevelScientistStyle, 202  
userLevelUserStyle, 203  
userLevelVisibility, 203  
variable, 203  
variableAsToolTip, 203  
variableSubstitutions, 203  
visible, 203  
WhenInAlarm, 196  
QELineEdit, 204  
addUnits, 207  
Append, 206  
arrayAction, 207  
ArrayActions, 206  
Ascii, 206  
Automatic, 206  
Comma, 206  
dbValueChanged, 207  
Default, 206  
Fixed, 206  
Floating, 206  
format, 207  
Formats, 206  
Index, 206  
int, 208  
Integer, 206  
leadingZero, 208  
LocalEnumeration, 206  
localEnumeration, 208  
NoSeparator, 206  
notation, 208  
Notations, 206  
precision, 209  
QELineEdit, 207  
radix, 209  
Scientific, 206  
separator, 209  
Separators, 206  
Space, 207  
Time, 206  
trailingZeros, 209  
Underscore, 207  
UnsignedInteger, 206  
useDbPrecision, 209  
QELineEditManager, 209  
QELink, 210  
QELog, 212  
allowDrop, 215  
Always, 214  
defaultStyle, 215  
displayAlarmState, 215  
displayAlarmStateOption, 215  
DisplayAlarmStateOptions, 214  
Engineer, 215  
int, 216  
Never, 214  
Scientist, 215  
setManagedVisible, 215  
styleSheet, 216  
User, 215  
userLevelEnabled, 216  
userLevelEngineerStyle, 216  
UserLevels, 214  
userLevelScientistStyle, 216  
userLevelUserStyle, 216  
userLevelVisibility, 217  
variableAsToolTip, 217  
visible, 217  
WhenInAlarm, 214  
QELogin, 217  
QELoginDialog, 218  
QENumericEdit, 218  
addUnits, 221  
arrayIndex, 221  
autoScale, 221  
dbConnectionChanged, 221  
dbValueChanged, 221  
leadingZeros, 221  
maximum, 222  
minimum, 222  
precision, 222  
QENumericEdit, 221  
QENumericEditManager, 222

QEPeriodic, 223  
allowDrop, 227  
Always, 226  
dbElementChanged, 227  
dbValueChanged, 227  
displayAlarmState, 227  
displayAlarmStateOption, 227  
DisplayAlarmStateOptions, 226  
Engineer, 227  
int, 228  
Never, 226  
readbackLabelVariable1, 228  
readbackLabelVariable2, 228  
Scientist, 227  
subscribe, 228  
User, 227  
userLevelEnabled, 228  
userLevelEngineerStyle, 228  
UserLevels, 226  
userLevelScientistStyle, 229  
userLevelUserStyle, 229  
userLevelVisibility, 229  
variableAsToolTip, 229  
variableSubstitutions, 229  
visible, 229  
WhenInAlarm, 226  
writeButtonVariable1, 230  
writeButtonVariable2, 230  
QEPeriodic::elementInfoStruct, 31  
QEPeriodic::userInfoStructArray, 324  
QEPeriodicComponentData, 230  
QEPeriodicTaskMenu, 230  
QEPeriodicTaskMenuFactory, 231  
QEPlot, 231  
    allowDrop, 236  
    Always, 235  
    dbValueChanged, 235  
    defaultStyle, 236  
    displayAlarmState, 236  
    displayAlarmStateOption, 236  
    DisplayAlarmStateOptions, 235  
    Engineer, 235  
    int, 236  
    Never, 235  
    Scientist, 235  
    setManagedVisible, 235  
    styleSheet, 237  
    User, 235  
    userLevelEnabled, 237  
    userLevelEngineerStyle, 237  
UserLevels, 235  
userLevelScientistStyle, 237  
userLevelUserStyle, 237  
userLevelVisibility, 237  
variable1, 238  
variable2, 238  
variable3, 238  
variable4, 238  
variableAsToolTip, 238  
variableSubstitutions, 238  
visible, 238  
WhenInAlarm, 235  
QEPushButton, 239  
    addUnits, 246  
    alignment, 246  
    allowDrop, 246  
    altReadbackVariable, 246  
    Always, 243  
    Append, 243  
    arguments, 247  
    arrayAction, 247  
    ArrayActions, 243  
    Ascii, 243  
    Automatic, 244  
    clickCheckedText, 247  
    clicked, 245  
    clickText, 247  
    confirmAction, 247  
    confirmText, 248  
    creationOption, 248  
    CreationOptionNames, 243  
    customisationName, 248  
    dbValueChanged, 245  
    Default, 244  
    defaultStyle, 248  
    displayAlarmState, 248  
    displayAlarmStateOption, 248  
    DisplayAlarmStateOptions, 243  
    DockBottom, 243  
    DockBottomTabbed, 243  
    DockFloating, 243  
    DockLeft, 243  
    DockLeftTabbed, 243  
    DockRight, 243  
    DockRightTabbed, 243  
    DockTop, 243  
    DockTopTabbed, 243  
    Engineer, 245  
    Fixed, 244  
    Floating, 244

format, 249  
Formats, 243  
guiFile, 249  
Icon, 245  
Index, 243  
int, 249  
Integer, 244  
labelText, 249  
leadingZero, 250  
LocalEnumeration, 244  
localEnumeration, 250  
LogOutput, 244  
Never, 243  
NewTab, 243  
NewWindow, 243  
None, 244  
notation, 250  
Notations, 244  
Open, 243  
password, 250  
pixmap0, 250  
pixmap1, 251  
pixmap2, 251  
pixmap3, 251  
pixmap4, 251  
pixmap5, 251  
pixmap6, 251  
pixmap7, 251  
precision, 251  
pressed, 245  
pressText, 251  
prioritySubstitutions, 252  
program, 252  
programStartupOption, 252  
ProgramStartupOptionNames, 244  
QEPushButton, 245  
released, 246  
releaseText, 252  
requestAction, 246  
Scientific, 244  
Scientist, 245  
setManagedVisible, 246  
State, 245  
StdOutput, 244  
styleSheet, 252  
subscribe, 252  
Terminal, 244  
Text, 245  
TextAndIcon, 245  
Time, 244  
trailingZeros, 252  
UnsignedInteger, 244  
updateOption, 253  
UpdateOptions, 244  
useDbPrecision, 253  
User, 245  
userLevelEnabled, 253  
userLevelEngineerStyle, 253  
UserLevels, 245  
userLevelScientistStyle, 253  
userLevelUserStyle, 253  
userLevelVisibility, 254  
variable, 254  
variableAsToolTip, 254  
variableSubstitutions, 254  
visible, 254  
WhenInAlarm, 243  
writeOnClick, 254  
writeOnPress, 254  
writeOnRelease, 255  
QEPVNameLists, 255  
QEPvProperties, 255  
variable, 256  
variableSubstitutions, 256  
QEPvPropertiesManager, 257  
QERadioButton, 257  
    addUnits, 265  
    alignment, 266  
    allowDrop, 266  
    Always, 262  
    Append, 262  
    arguments, 266  
    arrayAction, 266  
    ArrayActions, 262  
    Ascii, 262  
    Automatic, 263  
    clickCheckedText, 266  
    clicked, 265  
    clickText, 267  
    Comma, 264  
    confirmAction, 267  
    confirmText, 267  
    creationOption, 267  
    CreationOptionNames, 262  
    customisationName, 267  
    dbValueChanged, 265  
    Default, 263  
    defaultStyle, 267  
    displayAlarmState, 267  
    displayAlarmStateOption, 268

DisplayAlarmStateOptions, 262  
DockBottom, 262  
DockBottomTabbed, 262  
DockFloating, 262  
DockLeft, 262  
DockLeftTabbed, 262  
DockRight, 262  
DockRightTabbed, 262  
DockTop, 262  
DockTopTabbed, 262  
Engineer, 264  
Fixed, 263  
Floating, 263  
format, 268  
Formats, 262  
guiFile, 268  
Icon, 264  
Index, 262  
int, 268  
Integer, 263  
labelText, 268  
leadingZero, 269  
LocalEnumeration, 263  
localEnumeration, 269  
LogOutput, 263  
Never, 262  
NewTab, 262  
NewWindow, 262  
None, 263  
NoSeparator, 264  
notation, 269  
Notations, 263  
Open, 262  
password, 270  
pixmap0, 270  
pixmap1, 270  
pixmap2, 270  
pixmap3, 270  
pixmap4, 270  
pixmap5, 270  
pixmap6, 270  
pixmap7, 270  
precision, 271  
pressed, 265  
pressText, 271  
prioritySubstitutions, 271  
program, 271  
programStartupOption, 271  
ProgramStartupOptionNames, 263  
QERadioButton, 264  
radix, 271  
released, 265  
releaseText, 271  
requestAction, 265  
Scientific, 263  
Scientist, 264  
separator, 272  
Separators, 263  
setManagedVisible, 265  
Space, 264  
State, 264  
StdOutput, 263  
styleSheet, 272  
subscribe, 272  
Terminal, 263  
Text, 264  
TextAndIcon, 264  
Time, 263  
trailingZeros, 272  
Underscore, 264  
UnsignedInteger, 263  
updateOption, 272  
UpdateOptions, 264  
useDbPrecision, 272  
User, 264  
userLevelEnabled, 272  
userLevelEngineerStyle, 272  
UserLevels, 264  
userLevelScientistStyle, 273  
userLevelUserStyle, 273  
userLevelVisibility, 273  
variable, 273  
variableAsToolTip, 273  
variableSubstitutions, 273  
visible, 274  
WhenInAlarm, 262  
writeOnClick, 274  
writeOnPress, 274  
writeOnRelease, 274  
QERecipe, 274  
QERecordFieldName, 276  
QERecordSpec, 277  
QERecordSpecList, 277  
QEScript, 277  
    allowDrop, 283  
    Always, 282  
    defaultStyle, 283  
    displayAlarmState, 283  
    displayAlarmStateOption, 283  
    DisplayAlarmStateOptions, 282

Engineer, 282  
int, 283  
Never, 282  
Scientist, 282  
setManagedVisible, 283  
styleSheet, 284  
User, 282  
userLevelEnabled, 284  
userLevelEngineerStyle, 284  
UserLevels, 282  
userLevelScientistStyle, 284  
userLevelUserStyle, 284  
userLevelVisibility, 284  
variableAsToolTip, 285  
visible, 285  
WhenInAlarm, 282  
QEShape, 285  
allowDrop, 292  
Always, 290  
animation1, 292  
animation2, 292  
animation3, 292  
animation4, 292  
animation5, 292  
animation6, 292  
animationOptions, 290  
color1, 292  
color10, 292  
color2, 293  
color3, 293  
color4, 293  
color5, 293  
color6, 293  
color7, 293  
color8, 293  
color9, 293  
dbValueChanged1, 291  
dbValueChanged2, 291  
dbValueChanged3, 291  
dbValueChanged4, 291  
dbValueChanged5, 291  
dbValueChanged6, 291  
defaultStyle, 293  
displayAlarmState, 294  
displayAlarmStateOption, 294  
DisplayAlarmStateOptions, 290  
Engineer, 290  
int, 294  
Never, 290  
offset1, 294  
offset2, 294  
offset3, 294  
offset4, 295  
offset5, 295  
offset6, 295  
point1, 295  
point10, 295  
point2, 295  
point3, 295  
point4, 295  
point5, 295  
point6, 296  
point7, 296  
point8, 296  
point9, 296  
QEShape, 290  
scale2, 296  
scale3, 296  
scale4, 296  
scale5, 296  
scale6, 296  
Scientist, 290  
setManagedVisible, 291  
shapeOptions, 290  
styleSheet, 297  
User, 290  
userLevelEnabled, 297  
userLevelEngineerStyle, 297  
UserLevels, 290  
userLevelScientistStyle, 297  
userLevelUserStyle, 297  
userLevelVisibility, 297  
variable1, 298  
variable2, 298  
variable3, 298  
variable4, 298  
variable5, 298  
variable6, 298  
variableAsToolTip, 298  
variableSubstitutions, 299  
visible, 299  
WhenInAlarm, 290  
QESlider, 299  
allowDrop, 302  
Always, 302  
dbValueChanged, 302  
defaultStyle, 302  
displayAlarmState, 303  
displayAlarmStateOption, 303  
DisplayAlarmStateOptions, 301

Engineer, 302  
int, 303  
Never, 302  
Scientist, 302  
setManagedVisible, 302  
styleSheet, 303  
subscribe, 303  
User, 302  
userLevelEnabled, 303  
userLevelEngineerStyle, 304  
UserLevels, 302  
userLevelScientistStyle, 304  
userLevelUserStyle, 304  
userLevelVisibility, 304  
variable, 304  
variableAsToolTip, 305  
variableSubstitutions, 305  
visible, 305  
WhenInAlarm, 302  
writeOnChange, 302

QESpinBox, 305  
allowDrop, 308  
Always, 308  
dbValueChanged, 308  
defaultStyle, 308  
displayAlarmState, 309  
displayAlarmStateOption, 309  
DisplayAlarmStateOptions, 308  
Engineer, 308  
int, 309  
Never, 308  
Scientist, 308  
setManagedVisible, 308  
styleSheet, 309  
subscribe, 309  
User, 308  
userLevelEnabled, 309  
userLevelEngineerStyle, 310  
UserLevels, 308  
userLevelScientistStyle, 310  
userLevelUserStyle, 310  
userLevelVisibility, 310  
variable, 310  
variableAsToolTip, 311  
variableSubstitutions, 311  
visible, 311  
WhenInAlarm, 308

QEStripChart, 311  
variableSubstitutions, 313

QEStripChartAdjustPVDialog, 314

QEStripChartContextMenu, 314  
QEStripChartContextMenu, 314  
QEStripChartDurationDialog, 315  
QEStripChartItem, 315  
QEStripChartNames, 316  
QEStripChartPushButtonSpecifications, 317  
QEStripChartRangeDialog, 318  
QEStripChartState, 318  
QEStripChartStateList, 318  
QEStripChartStatistics, 319  
QEStripChartTimeDialog, 319  
QEStripChartToolBar, 319  
QEStripChartToolBar::OwnWidgets, 58  
QESubstitutedLabel, 321  
    labelText, 321  
    textSubstitutions, 321

radix  
QEAnalogProgressBar, 77  
QECheckBox, 99  
QELabel, 201  
QELineEdit, 209  
QERadioButton, 271

readbackLabelVariable1  
    QEPeriodic, 228

readbackLabelVariable2  
    QEPeriodic, 228

recording, 321

regionOfInterest1HVariable  
    QEImage, 185

regionOfInterest1WVariable  
    QEImage, 185

regionOfInterest1XVariable  
    QEImage, 185

regionOfInterest1YVariable  
    QEImage, 186

regionOfInterest2HVariable  
    QEImage, 186

regionOfInterest2WVariable  
    QEImage, 186

regionOfInterest2XVariable  
    QEImage, 186

regionOfInterest2YVariable  
    QEImage, 186

regionOfInterest3HVariable  
    QEImage, 186

regionOfInterest3WVariable  
    QEImage, 186

regionOfInterest3XVariable  
    QEImage, 186

regionOfInterest3YVariable  
     QEImage, 186  
 regionOfInterest4HVariable  
     QEImage, 187  
 regionOfInterest4WVariable  
     QEImage, 187  
 regionOfInterest4XVariable  
     QEImage, 187  
 regionOfInterest4YVariable  
     QEImage, 187  
 released  
     QECheckBox, 92  
     QEPushButton, 246  
     QERadioButton, 265  
 releaseText  
     QECheckBox, 99  
     QEPushButton, 252  
     QERadioButton, 271  
 requestAction  
     QECheckBox, 93  
     QEPushButton, 246  
     QERadioButton, 265  
 RESIZE\_OPTION\_FIT  
     QEImage, 170  
 RESIZE\_OPTION\_ZOOM  
     QEImage, 170  
 resizeContents  
     QEForm, 123  
 resizeOption  
     QEImage, 187  
 ResizeOptions  
     QEImage, 169  
 resizeOptions  
     QEImage, 170  
 rgb1  
     QEImage, 169  
 rgb2  
     QEImage, 169  
 rgb3  
     QEImage, 169  
 Right\_To\_Left  
     QEAnalogIndicator, 66  
 Rotate180  
     QEImage, 170  
 Rotate90Left  
     QEImage, 170  
 Rotate90Right  
     QEImage, 170  
 rotation  
     QEImage, 187  
 ROTATION\_0  
     imageProperties, 46  
 ROTATION\_180  
     imageProperties, 46  
 ROTATION\_90\_LEFT  
     imageProperties, 46  
 ROTATION\_90\_RIGHT  
     imageProperties, 46  
 RotationOptions  
     QEImage, 170  
 rotationOptions  
     imageProperties, 46  
 Scale  
     QEAnalogIndicator, 66  
 scale2  
     QEShape, 296  
 scale3  
     QEShape, 296  
 scale4  
     QEShape, 296  
 scale5  
     QEShape, 296  
 scale6  
     QEShape, 296  
 Scientific  
     QEAnalogProgressBar, 73  
     QECheckBox, 91  
     QELabel, 196  
     QELineEdit, 206  
     QEPushButton, 244  
     QERadioButton, 263  
 Scientist  
     QEAnalogProgressBar, 73  
     QEBitStatus, 82  
     QECheckBox, 92  
     QEComboBox, 106  
     QEConfiguredLayout, 112  
     QEFileBrowser, 118  
     QEFrame, 127  
     QEGenericEdit, 135  
     QEGroupBox, 142  
     QEImage, 171  
     QELabel, 197  
     QELog, 215  
     QEPeriodic, 227  
     QEPlot, 235  
     QEPushButton, 245  
     QERadioButton, 264  
     QEScript, 282

QEShape, 290  
QESlider, 302  
QESpinBox, 308  
screenSelectDialog, 322  
selected  
    QEFileBrowser, 118  
selectMenu, 323  
selectOptions  
    QEImage, 170  
separator  
    QEAnalogProgressBar, 77  
    QECheckBox, 99  
    QELabel, 201  
    QELineEdit, 209  
    QERadioButton, 272  
Separators  
    QEAnalogProgressBar, 73  
    QECheckBox, 91  
    QELabel, 196  
    QELineEdit, 206  
    QERadioButton, 263  
setConfirmWrite  
    QEGenericEdit, 136  
setImageFile  
    QEImage, 172  
setManagedVisible  
    QEAnalogProgressBar, 74  
    QEBitStatus, 82  
    QECheckBox, 93  
    QEComboBox, 106  
    QEConfiguredLayout, 112  
    QEFileBrowser, 118  
    QEFrame, 127  
    QEGenericEdit, 136  
    QEGroupBox, 142  
    QEImage, 172  
    QELabel, 198  
    QELog, 215  
    QEPlot, 235  
    QEPushButton, 246  
    QERadioButton, 265  
    QEScript, 283  
    QEShape, 291  
    QESlider, 302  
    QESpinBox, 308  
setSubscribe  
    QEGenericEdit, 136  
setWriteOnEnter  
    QEGenericEdit, 136  
setWriteOnFinish

QEGenericEdit, 136  
setWriteOnLoseFocus  
    QEGenericEdit, 136  
shapeOptions  
    QEShape, 290  
showScale  
    QEAnalogIndicator, 67  
showText  
    QEAnalogIndicator, 68  
showTime  
    QEImage, 187  
SO\_AREA4  
    QEImage, 171  
SO\_BEAM  
    QEImage, 171  
SO\_HSLICE1  
    QEImage, 170  
SO\_HSLICE2  
    QEImage, 170  
SO\_HSLICE3  
    QEImage, 171  
SO\_HSLICE4  
    QEImage, 171  
SO\_HSLICE5  
    QEImage, 171  
SO\_NONE  
    QEImage, 170  
SO\_PANNING  
    QEImage, 170  
SO\_PROFILE  
    QEImage, 171  
SO\_TARGET  
    QEImage, 171  
SO\_VSLICE1  
    QEImage, 170  
SO\_VSLICE2  
    QEImage, 170  
SO\_VSLICE3  
    QEImage, 170  
SO\_VSLICE4  
    QEImage, 170  
SO\_VSLICE5  
    QEImage, 170  
SolidSmallCrosshair  
    QEImage, 171  
Space  
    QEAnalogProgressBar, 73  
    QECheckBox, 91  
    QELabel, 197  
    QELineEdit, 207

QERadioButton, 264  
 spanAngle  
   QEAnalogIndicator, 68  
 State  
   QECheckBox, 91  
   QEPushButton, 245  
   QERadioButton, 264  
 StdOutput  
   QECheckBox, 91  
   QEImage, 169  
   QEPushButton, 244  
   QERadioButton, 263  
 styleSheet  
   QEAnalogProgressBar, 77  
   QEBitStatus, 83  
   QECheckBox, 99  
   QEComboBox, 107  
   QEConfiguredLayout, 113  
   QEFileBrowser, 119  
   QEFrame, 128  
   QEGenericEdit, 138  
   QEGroupBox, 143  
   QEImage, 187  
   QELabel, 202  
   QELog, 216  
   QEPlot, 237  
   QEPushButton, 252  
   QERadioButton, 272  
   QEScript, 284  
   QEShape, 297  
   QESlider, 303  
   QESpinBox, 309  
 subscribe  
   QECheckBox, 99  
   QEComboBox, 107  
   QEGenericEdit, 138  
   QEPeriodic, 228  
   QEPushButton, 252  
   QERadioButton, 272  
   QESlider, 303  
   QESpinBox, 309  
 substitutedTitle  
   QEGroupBox, 143  
 targetColor  
   QEImage, 187  
 TargetOptions  
   QEImage, 171  
 targetTriggerVariable  
   QEImage, 188  
 targetXVariable  
   QEImage, 188  
 targetYVariable  
   QEImage, 188  
 Terminal  
   QECheckBox, 91  
   QEImage, 169  
   QEPushButton, 244  
   QERadioButton, 263  
 Text  
   QECheckBox, 91  
   QELabel, 197  
   QEPushButton, 245  
   QERadioButton, 264  
 TextAndIcon  
   QECheckBox, 91  
   QEPushButton, 245  
   QERadioButton, 264  
 textSubstitutions  
   QEGroupBox, 143  
   QESubstitutedLabel, 321  
 Time  
   QEAnalogProgressBar, 73  
   QECheckBox, 90  
   QELabel, 196  
   QELineEdit, 206  
   QEPushButton, 244  
   QERadioButton, 263  
 timeColor  
   QEImage, 188  
 Top\_To\_Bottom  
   QEAnalogIndicator, 66  
 trace, 323  
 trailingZeros  
   QEAnalogProgressBar, 77  
   QECheckBox, 100  
   QELabel, 202  
   QELineEdit, 209  
   QEPushButton, 252  
   QERadioButton, 272  
 uiFile  
   QEForm, 125  
 Underscore  
   QEAnalogProgressBar, 73  
   QECheckBox, 91  
   QELabel, 197  
   QELineEdit, 207  
   QERadioButton, 264  
 UnsignedInteger

QEAnalogProgressBar, 73  
QECheckBox, 90  
QELabel, 196  
QELineEdit, 206  
QEPushButton, 244  
QERadioButton, 263  
UPDATE\_PIXMAP  
QELabel, 197  
UPDATE\_TEXT  
QELabel, 197  
updateImage  
mpegSource, 57  
updateOption  
QECheckBox, 100  
QELabel, 202  
QEPushButton, 253  
QERadioButton, 272  
UpdateOptions  
QECheckBox, 91  
QELabel, 197  
QEPushButton, 244  
QERadioButton, 264  
updateOptions  
QELabel, 197  
URL  
QEImage, 188  
useDbDisplayLimits  
QEAnalogProgressBar, 77  
useDbEnumerations  
QEComboBox, 106  
useDbPrecision  
QEAnalogProgressBar, 77  
QECheckBox, 100  
QELabel, 202  
QELineEdit, 209  
QEPushButton, 253  
QERadioButton, 272  
useFalseColour  
QEImage, 188  
User  
QEAnalogProgressBar, 73  
QEBitStatus, 82  
QECheckBox, 92  
QEComboBox, 106  
QEConfiguredLayout, 112  
QEFileBrowser, 118  
QEFrame, 127  
QEGenericEdit, 135  
QEGroupBox, 142  
QEImage, 171  
QELabel, 197  
QELog, 215  
QEPeriodic, 227  
QEPlot, 235  
QEPushButton, 245  
QERadioButton, 264  
QEScript, 282  
QEShape, 290  
QESlider, 302  
QESpinBox, 308  
userInfoStruct, 324  
userLevelEnabled  
QEAnalogProgressBar, 77  
QEBitStatus, 83  
QECheckBox, 100  
QEComboBox, 107  
QEConfiguredLayout, 113  
QEFileBrowser, 120  
QEFrame, 128  
QEGenericEdit, 138  
QEGroupBox, 143  
QEImage, 188  
QELabel, 202  
QELog, 216  
QEPeriodic, 228  
QEPlot, 237  
QEPushButton, 253  
QERadioButton, 272  
QEScript, 284  
QEShape, 297  
QESlider, 303  
QESpinBox, 309  
userLevelEngineerStyle  
QEAnalogProgressBar, 77  
QEBitStatus, 83  
QECheckBox, 100  
QEComboBox, 108  
QEConfiguredLayout, 113  
QEFileBrowser, 120  
QEFrame, 129  
QEGenericEdit, 138  
QEGroupBox, 143  
QEImage, 188  
QELabel, 202  
QELog, 216  
QEPeriodic, 228  
QEPlot, 237  
QEPushButton, 253  
QERadioButton, 272  
QEScript, 284

QEShape, 297  
QESlider, 304  
QESpinBox, 310  
**UserLevels**  
QEAnalogProgressBar, 73  
QEBitStatus, 81  
QECheckBox, 91  
QEComboBox, 105  
QEConfiguredLayout, 111  
QEFileBrowser, 118  
QEFrame, 127  
QEGenericEdit, 135  
QEGroupBox, 141  
QEImage, 171  
QELabel, 197  
QELog, 214  
QEPeriodic, 226  
QEPlot, 235  
QEPushButton, 245  
QERadioButton, 264  
QEScript, 282  
QEShape, 290  
QESlider, 302  
QESpinBox, 308  
**userLevelScientistStyle**  
QEAnalogProgressBar, 78  
QEBitStatus, 84  
QECheckBox, 100  
QEComboBox, 108  
QEConfiguredLayout, 113  
QEFileBrowser, 120  
QEFrame, 129  
QEGenericEdit, 138  
QEGroupBox, 144  
QEImage, 189  
QELabel, 202  
QELog, 216  
QEPeriodic, 229  
QEPlot, 237  
QEPushButton, 253  
QERadioButton, 273  
QEScript, 284  
QEShape, 297  
QESlider, 304  
QESpinBox, 310  
**userLevelUserStyle**  
QEAnalogProgressBar, 78  
QEBitStatus, 84  
QECheckBox, 101  
QEComboBox, 108  
QEConfiguredLayout, 113  
QEFileBrowser, 120  
QEFrame, 129  
QEGenericEdit, 135  
QEGroupBox, 141  
QEImage, 189  
QELabel, 202  
QELog, 216  
QEPeriodic, 229  
QEPlot, 237  
QEPushButton, 253  
QERadioButton, 273  
QEScript, 284  
QEShape, 297  
QESlider, 304  
QESpinBox, 310  
**value**  
QEAnalogIndicator, 68  
QEAnalogProgressBar, 78  
**ValueScaling**, 324  
**variable**  
QEAnalogProgressBar, 78  
QEBitStatus, 84  
QECheckBox, 101  
QEComboBox, 108  
QEFileBrowser, 121  
QEGenericEdit, 139  
QELabel, 203

QEPushButton, 254  
QE PvProperties, 256  
QERadioButton, 273  
QESlider, 304  
QESpinBox, 310  
variable1  
QEPlot, 238  
QEShape, 298  
variable2  
QEPlot, 238  
QEShape, 298  
variable3  
QEPlot, 238  
QEShape, 298  
variable4  
QEPlot, 238  
QEShape, 298  
variable5  
QEShape, 298  
variable6  
QEShape, 298  
variableAsToolTip  
QEAnalogProgressBar, 78  
QEBitStatus, 84  
QECheckBox, 101  
QEComboBox, 109  
QEConfiguredLayout, 114  
QEFileBrowser, 121  
QEForm, 125  
QEFrame, 129  
QEGenericEdit, 139  
QEGroupBox, 144  
QEImage, 189  
QELabel, 203  
QELog, 217  
QEPeriodic, 229  
QEPlot, 238  
QEPushButton, 254  
QERadioButton, 273  
QEScript, 285  
QEShape, 298  
QESlider, 305  
QESpinBox, 311  
variableSubstitutions  
QEAnalogProgressBar, 79  
QEBitStatus, 84  
QECheckBox, 101  
QEComboBox, 109  
QEFileBrowser, 121  
QEForm, 125  
QEGenericEdit, 139  
QEGroupBox, 144  
QEImage, 189  
QELabel, 203  
QELog, 217  
QEPeriodic, 229  
QEPlot, 238  
QEPushButton, 254  
QERadioButton, 273  
QEScript, 285  
QEShape, 298  
QESlider, 305  
QESpinBox, 311  
WhenInAlarm  
QEAnalogProgressBar, 72  
QEBitStatus, 81

QECheckBox, 90  
QEComboBox, 105  
QEConfiguredLayout, 111  
QEFileDialog, 118  
QEFrame, 127  
QEGenericEdit, 135  
QEGroupBox, 141  
QEImage, 168  
QELabel, 196  
QELog, 214  
QEPeriodic, 226  
QEPlot, 235  
QEPushButton, 243  
QERadioButton, 262  
QEScript, 282  
QEShape, 290  
QESlider, 302  
QESpinBox, 308  
widthVariable  
    QEImage, 190  
writeButtonVariable1  
    QEPeriodic, 230  
writeButtonVariable2  
    QEPeriodic, 230  
writeOnChange  
    QEComboBox, 106  
    QESlider, 302  
writeOnClick  
    QECheckBox, 101  
    QEPushButton, 254  
    QERadioButton, 274  
writeOnEnter  
    QEGenericEdit, 139  
writeOnFinish  
    QEGenericEdit, 139  
writeOnLoseFocus  
    QEGenericEdit, 140  
writeOnPress  
    QECheckBox, 102  
    QEPushButton, 254  
    QERadioButton, 274  
writeOnRelease  
    QECheckBox, 102  
    QEPushButton, 255  
    QERadioButton, 274  
  
yuv422  
    QEImage, 169  
yuv444  
    QEImage, 169