

# EPICS QT Framework

## 3.3.1

Generated by Doxygen 1.6.1

Wed Jun 29 17:29:04 2016



# 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>17</b>
<b>6 Qt Creator</b>	<b>19</b>
<b>7 Class Index</b>	<b>21</b>
7.1 Class Hierarchy . . . . .	21
<b>8 Class Index</b>	<b>25</b>
8.1 Class List . . . . .	25
<b>9 Class Documentation</b>	<b>29</b>
9.1 _CopyPaste Class Reference . . . . .	29

9.2	_Field Class Reference . . . . .	30
9.3	_Item Class Reference . . . . .	31
9.4	_QDialogItem Class Reference . . . . .	32
9.5	_QPushButtonGroup Class Reference . . . . .	33
9.6	_QTableWidgetFileBrowser Class Reference . . . . .	34
9.7	_QTableWidgetLog Class Reference . . . . .	35
9.8	_QTableWidgetScript Class Reference . . . . .	36
9.9	areaInfo Class Reference . . . . .	37
9.10	QEAnalogIndicator::Band Struct Reference . . . . .	38
9.11	QEAnalogIndicator::BandList Class Reference . . . . .	39
9.12	QEPeriodic::elementInfoStruct Struct Reference . . . . .	40
9.13	FFBuffer Class Reference . . . . .	41
9.14	FFThread Class Reference . . . . .	42
9.15	flipRotateMenu Class Reference . . . . .	43
9.16	fullScreenWindow Class Reference . . . . .	44
9.17	histogram Class Reference . . . . .	45
9.18	histogramScroll Class Reference . . . . .	46
9.19	historicImage Class Reference . . . . .	47
9.20	imageContextMenu Class Reference . . . . .	48
9.21	imageDisplayProperties Class Reference . . . . .	50
9.22	imageInfo Class Reference . . . . .	52
9.23	imageMarkup Class Reference . . . . .	53
9.24	imageMarkupLegendSetText Class Reference . . . . .	56
9.25	imageProcessor Class Reference . . . . .	57
9.25.1	Detailed Description . . . . .	60
9.25.2	Member Function Documentation . . . . .	60
9.25.2.1	getPixelValueFromData . . . . .	60
9.26	imageProperties Class Reference . . . . .	61
9.26.1	Detailed Description . . . . .	62
9.26.2	Member Enumeration Documentation . . . . .	63
9.26.2.1	rotationOptions . . . . .	63
9.26.3	Constructor & Destructor Documentation . . . . .	63
9.26.3.1	imageProperties . . . . .	63
9.27	imagePropertiesCore Class Reference . . . . .	64

9.27.1 Member Function Documentation . . . . .	64
9.27.1.1 buildImageCore . . . . .	64
9.28 imageUpdateIndicator Class Reference . . . . .	65
9.29 loginWidget Class Reference . . . . .	66
9.30 markupCrosshair1 Class Reference . . . . .	67
9.31 markupCrosshair2 Class Reference . . . . .	68
9.32 markupDisplayMenu Class Reference . . . . .	69
9.33 markupEllipse Class Reference . . . . .	70
9.34 markupHLine Class Reference . . . . .	71
9.34.1 Member Function Documentation . . . . .	71
9.34.1.1 drawMarkup . . . . .	71
9.35 markupItem Class Reference . . . . .	72
9.36 markupLine Class Reference . . . . .	75
9.37 markupRegion Class Reference . . . . .	76
9.38 markupText Class Reference . . . . .	77
9.39 markupVLine Class Reference . . . . .	78
9.39.1 Member Function Documentation . . . . .	78
9.39.1.1 drawMarkup . . . . .	78
9.40 mpegSource Class Reference . . . . .	79
9.40.1 Member Function Documentation . . . . .	79
9.40.1.1 updateImage . . . . .	79
9.41 mpegSourceObject Class Reference . . . . .	80
9.42 QEStripChartToolBar::OwnTabWidget Class Reference . . . . .	81
9.43 PeriodicDialog Class Reference . . . . .	82
9.44 PeriodicElementSetupForm Class Reference . . . . .	83
9.45 PeriodicSetupDialog Class Reference . . . . .	84
9.46 playbackTimer Class Reference . . . . .	85
9.47 pointInfo Class Reference . . . . .	86
9.48 profilePlot Class Reference . . . . .	87
9.49 QBitStatus Class Reference . . . . .	88
9.50 QEAnalogIndicator Class Reference . . . . .	90
9.50.1 Detailed Description . . . . .	93
9.50.2 Member Enumeration Documentation . . . . .	94
9.50.2.1 Modes . . . . .	94

9.50.2.2	Orientations . . . . .	94
9.50.3	Property Documentation . . . . .	94
9.50.3.1	backgroundColour . . . . .	94
9.50.3.2	borderColour . . . . .	94
9.50.3.3	centreAngle . . . . .	94
9.50.3.4	fontColour . . . . .	94
9.50.3.5	foregroundColour . . . . .	94
9.50.3.6	logScale . . . . .	95
9.50.3.7	logScaleInterval . . . . .	95
9.50.3.8	majorInterval . . . . .	95
9.50.3.9	maximum . . . . .	95
9.50.3.10	minimum . . . . .	95
9.50.3.11	minorInterval . . . . .	95
9.50.3.12	mode . . . . .	95
9.50.3.13	orientation . . . . .	95
9.50.3.14	showScale . . . . .	95
9.50.3.15	showText . . . . .	95
9.50.3.16	spanAngle . . . . .	95
9.50.3.17	value . . . . .	96
9.51	QEAnalogProgressBar Class Reference . . . . .	97
9.51.1	Member Enumeration Documentation . . . . .	100
9.51.1.1	ArrayActions . . . . .	100
9.51.1.2	DisplayAlarmStateOptions . . . . .	101
9.51.1.3	Formats . . . . .	101
9.51.1.4	Notations . . . . .	101
9.51.1.5	Separators . . . . .	101
9.51.1.6	UserLevels . . . . .	102
9.51.2	Constructor & Destructor Documentation . . . . .	102
9.51.2.1	QEAnalogProgressBar . . . . .	102
9.51.2.2	QEAnalogProgressBar . . . . .	102
9.51.3	Member Function Documentation . . . . .	102
9.51.3.1	dbConnectionChanged . . . . .	102
9.51.3.2	dbValueChanged . . . . .	102
9.51.3.3	setManagedVisible . . . . .	103

9.51.4 Property Documentation . . . . .	103
9.51.4.1 addUnits . . . . .	103
9.51.4.2 alarmSeverityDisplayMode . . . . .	103
9.51.4.3 allowDrop . . . . .	103
9.51.4.4 arrayAction . . . . .	103
9.51.4.5 arrayIndex . . . . .	103
9.51.4.6 defaultStyle . . . . .	104
9.51.4.7 displayAlarmState . . . . .	104
9.51.4.8 displayAlarmStateOption . . . . .	104
9.51.4.9 format . . . . .	104
9.51.4.10 int . . . . .	104
9.51.4.11 leadingZero . . . . .	104
9.51.4.12 localEnumeration . . . . .	105
9.51.4.13 notation . . . . .	105
9.51.4.14 precision . . . . .	105
9.51.4.15 radix . . . . .	105
9.51.4.16 separator . . . . .	105
9.51.4.17 styleSheet . . . . .	106
9.51.4.18 trailingZeros . . . . .	106
9.51.4.19 useDbDisplayLimits . . . . .	106
9.51.4.20 useDbPrecision . . . . .	106
9.51.4.21 userLevelEnabled . . . . .	106
9.51.4.22 userLevelEngineerStyle . . . . .	106
9.51.4.23 userLevelScientistStyle . . . . .	106
9.51.4.24 userLevelUserStyle . . . . .	107
9.51.4.25 userLevelVisibility . . . . .	107
9.51.4.26 value . . . . .	107
9.51.4.27 variable . . . . .	107
9.51.4.28 variableAsToolTip . . . . .	107
9.51.4.29 variableSubstitutions . . . . .	107
9.51.4.30 visible . . . . .	108
9.52 QEBitStatus Class Reference . . . . .	109
9.52.1 Member Enumeration Documentation . . . . .	111
9.52.1.1 DisplayAlarmStateOptions . . . . .	111

9.52.1.2	UserLevels . . . . .	111
9.52.2	Member Function Documentation . . . . .	111
9.52.2.1	dbConnectionChanged . . . . .	111
9.52.2.2	dbValueChanged . . . . .	111
9.52.2.3	setManagedVisible . . . . .	112
9.52.3	Property Documentation . . . . .	112
9.52.3.1	allowDrop . . . . .	112
9.52.3.2	arrayIndex . . . . .	112
9.52.3.3	defaultStyle . . . . .	112
9.52.3.4	displayAlarmState . . . . .	112
9.52.3.5	displayAlarmStateOption . . . . .	112
9.52.3.6	int . . . . .	113
9.52.3.7	styleSheet . . . . .	113
9.52.3.8	userLevelEnabled . . . . .	113
9.52.3.9	userLevelEngineerStyle . . . . .	113
9.52.3.10	userLevelScientistStyle . . . . .	113
9.52.3.11	userLevelUserStyle . . . . .	113
9.52.3.12	userLevelVisibility . . . . .	114
9.52.3.13	variable . . . . .	114
9.52.3.14	variableAsToolTip . . . . .	114
9.52.3.15	variableSubstitutions . . . . .	114
9.52.3.16	visible . . . . .	114
9.53	QECheckBox Class Reference . . . . .	115
9.53.1	Member Enumeration Documentation . . . . .	119
9.53.1.1	ArrayActions . . . . .	119
9.53.1.2	CreationOptionNames . . . . .	119
9.53.1.3	DisplayAlarmStateOptions . . . . .	120
9.53.1.4	Formats . . . . .	120
9.53.1.5	Notations . . . . .	121
9.53.1.6	ProgramStartupOptionNames . . . . .	121
9.53.1.7	Separators . . . . .	121
9.53.1.8	UpdateOptions . . . . .	121
9.53.1.9	UserLevels . . . . .	122
9.53.2	Constructor & Destructor Documentation . . . . .	122

9.53.2.1	QECheckBox . . . . .	122
9.53.2.2	QECheckBox . . . . .	122
9.53.3	Member Function Documentation . . . . .	122
9.53.3.1	clicked . . . . .	122
9.53.3.2	dbValueChanged . . . . .	122
9.53.3.3	pressed . . . . .	123
9.53.3.4	released . . . . .	123
9.53.3.5	requestAction . . . . .	123
9.53.3.6	setManagedVisible . . . . .	123
9.53.4	Property Documentation . . . . .	123
9.53.4.1	addUnits . . . . .	123
9.53.4.2	alignment . . . . .	123
9.53.4.3	allowDrop . . . . .	123
9.53.4.4	arguments . . . . .	124
9.53.4.5	arrayAction . . . . .	124
9.53.4.6	arrayIndex . . . . .	124
9.53.4.7	clickCheckedText . . . . .	124
9.53.4.8	clickText . . . . .	124
9.53.4.9	confirmAction . . . . .	125
9.53.4.10	confirmText . . . . .	125
9.53.4.11	creationOption . . . . .	125
9.53.4.12	customisationName . . . . .	125
9.53.4.13	defaultStyle . . . . .	125
9.53.4.14	disabledRecordPolicy . . . . .	125
9.53.4.15	displayAlarmState . . . . .	126
9.53.4.16	displayAlarmStateOption . . . . .	126
9.53.4.17	format . . . . .	126
9.53.4.18	guiFile . . . . .	126
9.53.4.19	int . . . . .	126
9.53.4.20	labelText . . . . .	126
9.53.4.21	leadingZero . . . . .	127
9.53.4.22	localEnumeration . . . . .	127
9.53.4.23	notation . . . . .	127
9.53.4.24	password . . . . .	128

9.53.4.25 pixmap0 . . . . .	128
9.53.4.26 pixmap1 . . . . .	128
9.53.4.27 pixmap2 . . . . .	128
9.53.4.28 pixmap3 . . . . .	128
9.53.4.29 pixmap4 . . . . .	128
9.53.4.30 pixmap5 . . . . .	128
9.53.4.31 pixmap6 . . . . .	128
9.53.4.32 pixmap7 . . . . .	128
9.53.4.33 precision . . . . .	129
9.53.4.34 pressText . . . . .	129
9.53.4.35 prioritySubstitutions . . . . .	129
9.53.4.36 program . . . . .	129
9.53.4.37 programStartupOption . . . . .	129
9.53.4.38 radix . . . . .	129
9.53.4.39 releaseText . . . . .	129
9.53.4.40 separator . . . . .	130
9.53.4.41 styleSheet . . . . .	130
9.53.4.42 subscribe . . . . .	130
9.53.4.43 trailingZeros . . . . .	130
9.53.4.44 updateOption . . . . .	130
9.53.4.45 useDbPrecision . . . . .	130
9.53.4.46 userLevelEnabled . . . . .	130
9.53.4.47 userLevelEngineerStyle . . . . .	130
9.53.4.48 userLevelScientistStyle . . . . .	131
9.53.4.49 userLevelUserStyle . . . . .	131
9.53.4.50 userLevelVisibility . . . . .	131
9.53.4.51 variable . . . . .	131
9.53.4.52 variableAsToolTip . . . . .	131
9.53.4.53 variableSubstitutions . . . . .	131
9.53.4.54 visible . . . . .	132
9.53.4.55 writeOnClick . . . . .	132
9.53.4.56 writeOnPress . . . . .	132
9.53.4.57 writeOnRelease . . . . .	132
9.54 QECheckBoxManager Class Reference . . . . .	133

9.55 QEComboBox Class Reference . . . . .	134
9.55.1 Member Enumeration Documentation . . . . .	136
9.55.1.1 DisplayAlarmStateOptions . . . . .	136
9.55.1.2 UserLevels . . . . .	136
9.55.2 Member Function Documentation . . . . .	137
9.55.2.1 dbValueChanged . . . . .	137
9.55.2.2 setManagedVisible . . . . .	137
9.55.3 Member Data Documentation . . . . .	137
9.55.3.1 useDbEnumerations . . . . .	137
9.55.3.2 writeOnChange . . . . .	137
9.55.4 Property Documentation . . . . .	137
9.55.4.1 allowDrop . . . . .	137
9.55.4.2 allowFocusUpdate . . . . .	137
9.55.4.3 arrayIndex . . . . .	137
9.55.4.4 defaultStyle . . . . .	138
9.55.4.5 displayAlarmState . . . . .	138
9.55.4.6 displayAlarmStateOption . . . . .	138
9.55.4.7 int . . . . .	138
9.55.4.8 localEnumeration . . . . .	138
9.55.4.9 styleSheet . . . . .	138
9.55.4.10 subscribe . . . . .	138
9.55.4.11 userLevelEnabled . . . . .	139
9.55.4.12 userLevelEngineerStyle . . . . .	139
9.55.4.13 userLevelScientistStyle . . . . .	139
9.55.4.14 userLevelUserStyle . . . . .	139
9.55.4.15 userLevelVisibility . . . . .	139
9.55.4.16 variable . . . . .	140
9.55.4.17 variableAsToolTip . . . . .	140
9.55.4.18 variableSubstitutions . . . . .	140
9.55.4.19 visible . . . . .	140
9.56 QEConfiguredLayout Class Reference . . . . .	141
9.56.1 Member Enumeration Documentation . . . . .	143
9.56.1.1 DisplayAlarmStateOptions . . . . .	143
9.56.1.2 UserLevels . . . . .	143

9.56.2 Member Function Documentation . . . . .	144
9.56.2.1 setManagedVisible . . . . .	144
9.56.3 Property Documentation . . . . .	144
9.56.3.1 allowDrop . . . . .	144
9.56.3.2 defaultStyle . . . . .	144
9.56.3.3 displayAlarmState . . . . .	144
9.56.3.4 displayAlarmStateOption . . . . .	144
9.56.3.5 int . . . . .	144
9.56.3.6 styleSheet . . . . .	145
9.56.3.7 userLevelEnabled . . . . .	145
9.56.3.8 userLevelEngineerStyle . . . . .	145
9.56.3.9 userLevelScientistStyle . . . . .	145
9.56.3.10 userLevelUserStyle . . . . .	145
9.56.3.11 userLevelVisibility . . . . .	146
9.56.3.12 variableAsToolTip . . . . .	146
9.56.3.13 visible . . . . .	146
9.57 QEConfiguredLayoutManager Class Reference . . . . .	147
9.58 QEFileBrowser Class Reference . . . . .	148
9.58.1 Detailed Description . . . . .	151
9.58.2 Member Enumeration Documentation . . . . .	151
9.58.2.1 DisplayAlarmStateOptions . . . . .	151
9.58.2.2 UserLevels . . . . .	151
9.58.3 Member Function Documentation . . . . .	152
9.58.3.1 selected . . . . .	152
9.58.3.2 setManagedVisible . . . . .	152
9.58.4 Property Documentation . . . . .	152
9.58.4.1 allowDrop . . . . .	152
9.58.4.2 defaultStyle . . . . .	152
9.58.4.3 displayAlarmState . . . . .	152
9.58.4.4 displayAlarmStateOption . . . . .	152
9.58.4.5 int . . . . .	153
9.58.4.6 styleSheet . . . . .	153
9.58.4.7 userLevelEnabled . . . . .	153
9.58.4.8 userLevelEngineerStyle . . . . .	153

9.58.4.9	userLevelScientistStyle . . . . .	153
9.58.4.10	userLevelUserStyle . . . . .	153
9.58.4.11	userLevelVisibility . . . . .	154
9.58.4.12	variable . . . . .	154
9.58.4.13	variableAsToolTip . . . . .	154
9.58.4.14	variableSubstitutions . . . . .	154
9.58.4.15	visible . . . . .	154
9.59	QEForm Class Reference . . . . .	155
9.59.1	Member Data Documentation . . . . .	157
9.59.1.1	handleGuiLaunchRequests . . . . .	157
9.59.1.2	resizeContents . . . . .	157
9.59.2	Property Documentation . . . . .	157
9.59.2.1	allowDrop . . . . .	157
9.59.2.2	displayAlarmStateOption . . . . .	158
9.59.2.3	int . . . . .	158
9.59.2.4	messageFormFilter . . . . .	158
9.59.2.5	messageSourceFilter . . . . .	158
9.59.2.6	uiFile . . . . .	158
9.59.2.7	variableAsToolTip . . . . .	158
9.59.2.8	variableSubstitutions . . . . .	159
9.60	QEFrame Class Reference . . . . .	160
9.60.1	Member Enumeration Documentation . . . . .	162
9.60.1.1	DisplayAlarmStateOptions . . . . .	162
9.60.1.2	UserLevels . . . . .	162
9.60.2	Member Function Documentation . . . . .	162
9.60.2.1	setManagedVisible . . . . .	162
9.60.3	Property Documentation . . . . .	162
9.60.3.1	allowDrop . . . . .	162
9.60.3.2	defaultStyle . . . . .	162
9.60.3.3	displayAlarmState . . . . .	163
9.60.3.4	displayAlarmStateOption . . . . .	163
9.60.3.5	int . . . . .	163
9.60.3.6	pixmap . . . . .	163
9.60.3.7	pixmap0 . . . . .	163

9.60.3.8 pixmap1 . . . . .	163
9.60.3.9 pixmap2 . . . . .	163
9.60.3.10 pixmap3 . . . . .	164
9.60.3.11 pixmap4 . . . . .	164
9.60.3.12 pixmap5 . . . . .	164
9.60.3.13 pixmap6 . . . . .	164
9.60.3.14 pixmap7 . . . . .	164
9.60.3.15 scaledContents . . . . .	164
9.60.3.16 styleSheet . . . . .	164
9.60.3.17 userLevelEnabled . . . . .	164
9.60.3.18 userLevelEngineerStyle . . . . .	165
9.60.3.19 userLevelScientistStyle . . . . .	165
9.60.3.20 userLevelUserStyle . . . . .	165
9.60.3.21 userLevelVisibility . . . . .	165
9.60.3.22 variableAsToolTip . . . . .	165
9.60.3.23 visible . . . . .	165
9.61 QEGenericButton Class Reference . . . . .	167
9.62 QEGenericEdit Class Reference . . . . .	170
9.62.1 Member Enumeration Documentation . . . . .	172
9.62.1.1 DisplayAlarmStateOptions . . . . .	172
9.62.1.2 UserLevels . . . . .	173
9.62.2 Constructor & Destructor Documentation . . . . .	173
9.62.2.1 QEGenericEdit . . . . .	173
9.62.2.2 QEGenericEdit . . . . .	173
9.62.3 Member Function Documentation . . . . .	173
9.62.3.1 getConfirmWrite . . . . .	173
9.62.3.2 getSubscribe . . . . .	173
9.62.3.3 getWriteOnEnter . . . . .	173
9.62.3.4 getWriteOnFinish . . . . .	174
9.62.3.5 getWriteOnLoseFocus . . . . .	174
9.62.3.6 setAllowFocusUpdate . . . . .	174
9.62.3.7 setConfirmWrite . . . . .	174
9.62.3.8 setManagedVisible . . . . .	174
9.62.3.9 setSubscribe . . . . .	174

9.62.3.10	setWriteOnEnter . . . . .	174
9.62.3.11	setWriteOnFinish . . . . .	174
9.62.3.12	setWriteOnLoseFocus . . . . .	175
9.62.4	Property Documentation . . . . .	175
9.62.4.1	allowDrop . . . . .	175
9.62.4.2	arrayIndex . . . . .	175
9.62.4.3	confirmWrite . . . . .	175
9.62.4.4	defaultStyle . . . . .	175
9.62.4.5	displayAlarmState . . . . .	175
9.62.4.6	displayAlarmStateOption . . . . .	175
9.62.4.7	int . . . . .	176
9.62.4.8	styleSheet . . . . .	176
9.62.4.9	subscribe . . . . .	176
9.62.4.10	userLevelEnabled . . . . .	176
9.62.4.11	userLevelEngineerStyle . . . . .	176
9.62.4.12	userLevelScientistStyle . . . . .	176
9.62.4.13	userLevelUserStyle . . . . .	177
9.62.4.14	userLevelVisibility . . . . .	177
9.62.4.15	variable . . . . .	177
9.62.4.16	variableAsToolTip . . . . .	177
9.62.4.17	variableSubstitutions . . . . .	177
9.62.4.18	visible . . . . .	177
9.62.4.19	writeOnEnter . . . . .	178
9.62.4.20	writeOnFinish . . . . .	178
9.62.4.21	writeOnLoseFocus . . . . .	178
9.63	QEGroupBox Class Reference . . . . .	179
9.63.1	Member Enumeration Documentation . . . . .	180
9.63.1.1	DisplayAlarmStateOptions . . . . .	180
9.63.1.2	UserLevels . . . . .	180
9.63.2	Member Function Documentation . . . . .	181
9.63.2.1	setManagedVisible . . . . .	181
9.63.3	Property Documentation . . . . .	181
9.63.3.1	allowDrop . . . . .	181
9.63.3.2	defaultStyle . . . . .	181

9.63.3.3	displayAlarmState . . . . .	181
9.63.3.4	displayAlarmStateOption . . . . .	181
9.63.3.5	int . . . . .	181
9.63.3.6	styleSheet . . . . .	182
9.63.3.7	substitutedTitle . . . . .	182
9.63.3.8	textSubstitutions . . . . .	182
9.63.3.9	userLevelEnabled . . . . .	182
9.63.3.10	userLevelEngineerStyle . . . . .	182
9.63.3.11	userLevelScientistStyle . . . . .	182
9.63.3.12	userLevelUserStyle . . . . .	183
9.63.3.13	userLevelVisibility . . . . .	183
9.63.3.14	variableAsToolTip . . . . .	183
9.63.3.15	visible . . . . .	183
9.64	QEImage Class Reference . . . . .	184
9.64.1	Detailed Description . . . . .	214
9.64.2	Member Enumeration Documentation . . . . .	214
9.64.2.1	DisplayAlarmStateOptions . . . . .	214
9.64.2.2	EllipseVariableDefinitions . . . . .	214
9.64.2.3	ellipseVariableDefinitions . . . . .	214
9.64.2.4	FormatOptions . . . . .	215
9.64.2.5	ProgramStartupOptionNames . . . . .	215
9.64.2.6	ResizeOptions . . . . .	215
9.64.2.7	resizeOptions . . . . .	216
9.64.2.8	RotationOptions . . . . .	216
9.64.2.9	selectOptions . . . . .	216
9.64.2.10	TargetOptions . . . . .	217
9.64.2.11	UserLevels . . . . .	217
9.64.3	Constructor & Destructor Documentation . . . . .	217
9.64.3.1	QEImage . . . . .	217
9.64.3.2	QEImage . . . . .	217
9.64.4	Member Function Documentation . . . . .	217
9.64.4.1	dbValueChanged . . . . .	217
9.64.4.2	setImageFile . . . . .	218
9.64.4.3	setManagedVisible . . . . .	218

9.64.5 Member Data Documentation . . . . .	218
9.64.5.1 displayButtonBar . . . . .	218
9.64.5.2 initialVertScrollPos . . . . .	218
9.64.6 Property Documentation . . . . .	218
9.64.6.1 allowDrop . . . . .	218
9.64.6.2 areaColor . . . . .	218
9.64.6.3 arguments1 . . . . .	218
9.64.6.4 arguments2 . . . . .	218
9.64.6.5 autoBrightnessContrast . . . . .	219
9.64.6.6 beamColor . . . . .	219
9.64.6.7 beamXVariable . . . . .	219
9.64.6.8 beamYVariable . . . . .	219
9.64.6.9 bitDepthVariable . . . . .	219
9.64.6.10 briefInfoArea . . . . .	219
9.64.6.11 clippingHighVariable . . . . .	219
9.64.6.12 clippingLowVariable . . . . .	219
9.64.6.13 clippingOnOffVariable . . . . .	219
9.64.6.14 contrastReversal . . . . .	220
9.64.6.15 dataTypeVariable . . . . .	220
9.64.6.16 defaultStyle . . . . .	220
9.64.6.17 dimension1Variable . . . . .	220
9.64.6.18 dimension2Variable . . . . .	220
9.64.6.19 dimension3Variable . . . . .	220
9.64.6.20 dimensionsVariable . . . . .	220
9.64.6.21 displayAlarmState . . . . .	220
9.64.6.22 displayAlarmStateOption . . . . .	221
9.64.6.23 displayArea1Selection . . . . .	221
9.64.6.24 displayArea2Selection . . . . .	221
9.64.6.25 displayArea3Selection . . . . .	221
9.64.6.26 displayArea4Selection . . . . .	221
9.64.6.27 displayBeamSelection . . . . .	221
9.64.6.28 displayCursorPixelInfo . . . . .	221
9.64.6.29 displayEllipse . . . . .	222
9.64.6.30 displayHozSlice1Selection . . . . .	222

9.64.6.31 displayHozSlice2Selection . . . . .	222
9.64.6.32 displayHozSlice3Selection . . . . .	222
9.64.6.33 displayHozSlice4Selection . . . . .	222
9.64.6.34 displayHozSlice5Selection . . . . .	222
9.64.6.35 displayProfileSelection . . . . .	222
9.64.6.36 displayTargetSelection . . . . .	222
9.64.6.37 displayVertSlice1Selection . . . . .	222
9.64.6.38 displayVertSlice2Selection . . . . .	223
9.64.6.39 displayVertSlice3Selection . . . . .	223
9.64.6.40 displayVertSlice4Selection . . . . .	223
9.64.6.41 displayVertSlice5Selection . . . . .	223
9.64.6.42 ellipseColor . . . . .	223
9.64.6.43 ellipseHVariable . . . . .	223
9.64.6.44 ellipseWVariable . . . . .	223
9.64.6.45 ellipseXVariable . . . . .	223
9.64.6.46 ellipseYVariable . . . . .	223
9.64.6.47 enableArea1Selection . . . . .	224
9.64.6.48 enableArea2Selection . . . . .	224
9.64.6.49 enableArea3Selection . . . . .	224
9.64.6.50 enableArea4Selection . . . . .	224
9.64.6.51 enableBeamSelection . . . . .	224
9.64.6.52 enableHozSlice1Selection . . . . .	224
9.64.6.53 enableHozSlice2Selection . . . . .	224
9.64.6.54 enableHozSlice3Selection . . . . .	224
9.64.6.55 enableHozSlice4Selection . . . . .	225
9.64.6.56 enableHozSlice5Selection . . . . .	225
9.64.6.57 enableProfileSelection . . . . .	225
9.64.6.58 enableTargetSelection . . . . .	225
9.64.6.59 enableVertSlice1Selection . . . . .	225
9.64.6.60 enableVertSlice2Selection . . . . .	225
9.64.6.61 enableVertSlice3Selection . . . . .	225
9.64.6.62 enableVertSlice4Selection . . . . .	226
9.64.6.63 enableVertSlice5Selection . . . . .	226
9.64.6.64 externalControls . . . . .	226

9.64.6.65 formatOption . . . . .	226
9.64.6.66 formatVariable . . . . .	226
9.64.6.67 heightVariable . . . . .	226
9.64.6.68 horizontalFlip . . . . .	226
9.64.6.69 hozSlice1Color . . . . .	226
9.64.6.70 hozSlice2Color . . . . .	226
9.64.6.71 hozSlice3Color . . . . .	227
9.64.6.72 hozSlice4Color . . . . .	227
9.64.6.73 hozSlice5Color . . . . .	227
9.64.6.74 imageVariable . . . . .	227
9.64.6.75 initialHosScrollPos . . . . .	227
9.64.6.76 int . . . . .	227
9.64.6.77 lineProfileArrayVariable . . . . .	227
9.64.6.78 lineProfileThicknessVariable . . . . .	227
9.64.6.79 lineProfileX1Variable . . . . .	228
9.64.6.80 lineProfileX2Variable . . . . .	228
9.64.6.81 lineProfileY1Variable . . . . .	228
9.64.6.82 lineProfileY2Variable . . . . .	228
9.64.6.83 logBrightness . . . . .	228
9.64.6.84 profileColor . . . . .	228
9.64.6.85 profileHoz1ThicknessVariable . . . . .	228
9.64.6.86 profileHoz1Variable . . . . .	228
9.64.6.87 profileHoz2ThicknessVariable . . . . .	228
9.64.6.88 profileHoz2Variable . . . . .	229
9.64.6.89 profileHoz3ThicknessVariable . . . . .	229
9.64.6.90 profileHoz3Variable . . . . .	229
9.64.6.91 profileHoz4ThicknessVariable . . . . .	229
9.64.6.92 profileHoz4Variable . . . . .	229
9.64.6.93 profileHoz5ThicknessVariable . . . . .	229
9.64.6.94 profileHoz5Variable . . . . .	229
9.64.6.95 profileHozArrayVariable . . . . .	229
9.64.6.96 profileVert1ThicknessVariable . . . . .	230
9.64.6.97 profileVert1Variable . . . . .	230
9.64.6.98 profileVert2ThicknessVariable . . . . .	230

9.64.6.99 profileVert2Variable . . . . .	230
9.64.6.100profileVert3ThicknessVariable . . . . .	230
9.64.6.101profileVert3Variable . . . . .	230
9.64.6.102profileVert4ThicknessVariable . . . . .	230
9.64.6.103profileVert4Variable . . . . .	230
9.64.6.104profileVert5ThicknessVariable . . . . .	231
9.64.6.105profileVert5Variable . . . . .	231
9.64.6.106profileVertArrayVariable . . . . .	231
9.64.6.107program1 . . . . .	231
9.64.6.108program2 . . . . .	231
9.64.6.109programStartupOption1 . . . . .	231
9.64.6.110programStartupOption2 . . . . .	231
9.64.6.111regionOfInterest1HVariable . . . . .	232
9.64.6.112regionOfInterest1WVariable . . . . .	232
9.64.6.113regionOfInterest1XVariable . . . . .	232
9.64.6.114regionOfInterest1YVariable . . . . .	232
9.64.6.115regionOfInterest2HVariable . . . . .	232
9.64.6.116regionOfInterest2WVariable . . . . .	232
9.64.6.117regionOfInterest2XVariable . . . . .	232
9.64.6.118regionOfInterest2YVariable . . . . .	232
9.64.6.119regionOfInterest3HVariable . . . . .	233
9.64.6.120regionOfInterest3WVariable . . . . .	233
9.64.6.121regionOfInterest3XVariable . . . . .	233
9.64.6.122regionOfInterest3YVariable . . . . .	233
9.64.6.123regionOfInterest4HVariable . . . . .	233
9.64.6.124regionOfInterest4WVariable . . . . .	233
9.64.6.125regionOfInterest4XVariable . . . . .	233
9.64.6.126regionOfInterest4YVariable . . . . .	233
9.64.6.127resizeOption . . . . .	234
9.64.6.128rotation . . . . .	234
9.64.6.129showTime . . . . .	234
9.64.6.130styleSheet . . . . .	234
9.64.6.131targetColor . . . . .	234
9.64.6.132targetTriggerVariable . . . . .	234

9.64.6.133targetXVariable . . . . .	234
9.64.6.134targetYVariable . . . . .	234
9.64.6.135timeColor . . . . .	234
9.64.6.136URL . . . . .	235
9.64.6.137useFalseColour . . . . .	235
9.64.6.138userLevelEnabled . . . . .	235
9.64.6.139userLevelEngineerStyle . . . . .	235
9.64.6.140userLevelScientistStyle . . . . .	235
9.64.6.141userLevelUserStyle . . . . .	235
9.64.6.142userLevelVisibility . . . . .	236
9.64.6.143variableAsToolTip . . . . .	236
9.64.6.144variableSubstitutions . . . . .	236
9.64.6.145verticalFlip . . . . .	236
9.64.6.146vertSlice1Color . . . . .	236
9.64.6.147vertSlice2Color . . . . .	236
9.64.6.148vertSlice3Color . . . . .	236
9.64.6.149vertSlice4Color . . . . .	236
9.64.6.150vertSlice5Color . . . . .	237
9.64.6.151visible . . . . .	237
9.64.6.152widthVariable . . . . .	237
9.65 QEImageMarkupThickness Class Reference . . . . .	238
9.66 QEImageOptionsDialog Class Reference . . . . .	239
9.67 QELabel Class Reference . . . . .	240
9.67.1 Detailed Description . . . . .	243
9.67.2 Member Enumeration Documentation . . . . .	243
9.67.2.1 ArrayActions . . . . .	243
9.67.2.2 DisplayAlarmStateOptions . . . . .	244
9.67.2.3 Formats . . . . .	244
9.67.2.4 Notations . . . . .	244
9.67.2.5 Separators . . . . .	244
9.67.2.6 UpdateOptions . . . . .	245
9.67.2.7 updateOptions . . . . .	245
9.67.2.8 UserLevels . . . . .	245
9.67.3 Constructor & Destructor Documentation . . . . .	245

9.67.3.1	QELabel	245
9.67.3.2	QELabel	246
9.67.4	Member Function Documentation	246
9.67.4.1	dbValueChanged	246
9.67.4.2	setManagedVisible	246
9.67.5	Property Documentation	246
9.67.5.1	addUnits	246
9.67.5.2	allowDrop	246
9.67.5.3	arrayAction	246
9.67.5.4	arrayIndex	247
9.67.5.5	defaultStyle	247
9.67.5.6	displayAlarmState	247
9.67.5.7	displayAlarmStateOption	247
9.67.5.8	format	247
9.67.5.9	int	247
9.67.5.10	leadingZero	247
9.67.5.11	localEnumeration	248
9.67.5.12	notation	248
9.67.5.13	pixmap0	248
9.67.5.14	pixmap1	248
9.67.5.15	pixmap2	249
9.67.5.16	pixmap3	249
9.67.5.17	pixmap4	249
9.67.5.18	pixmap5	249
9.67.5.19	pixmap6	249
9.67.5.20	pixmap7	249
9.67.5.21	precision	249
9.67.5.22	radix	249
9.67.5.23	separator	249
9.67.5.24	styleSheet	250
9.67.5.25	trailingZeros	250
9.67.5.26	updateOption	250
9.67.5.27	useDbPrecision	250
9.67.5.28	userLevelEnabled	250

9.67.5.29 userLevelEngineerStyle . . . . .	250
9.67.5.30 userLevelScientistStyle . . . . .	250
9.67.5.31 userLevelUserStyle . . . . .	251
9.67.5.32 userLevelVisibility . . . . .	251
9.67.5.33 variable . . . . .	251
9.67.5.34 variableAsToolTip . . . . .	251
9.67.5.35 variableSubstitutions . . . . .	251
9.67.5.36 visible . . . . .	251
9.68 QLELineEdit Class Reference . . . . .	253
9.68.1 Member Enumeration Documentation . . . . .	255
9.68.1.1 ArrayActions . . . . .	255
9.68.1.2 Formats . . . . .	255
9.68.1.3 Notations . . . . .	255
9.68.1.4 Separators . . . . .	255
9.68.2 Constructor & Destructor Documentation . . . . .	256
9.68.2.1 QLELineEdit . . . . .	256
9.68.2.2 QLELineEdit . . . . .	256
9.68.3 Member Function Documentation . . . . .	256
9.68.3.1 dbValueChanged . . . . .	256
9.68.4 Property Documentation . . . . .	256
9.68.4.1 addUnits . . . . .	256
9.68.4.2 arrayAction . . . . .	256
9.68.4.3 format . . . . .	257
9.68.4.4 leadingZero . . . . .	257
9.68.4.5 localEnumeration . . . . .	257
9.68.4.6 notation . . . . .	258
9.68.4.7 precision . . . . .	258
9.68.4.8 radix . . . . .	258
9.68.4.9 separator . . . . .	258
9.68.4.10 trailingZeros . . . . .	258
9.68.4.11 useDbPrecision . . . . .	258
9.69 QLELineEditManager Class Reference . . . . .	259
9.70 QELink Class Reference . . . . .	260
9.71 QELog Class Reference . . . . .	262

9.71.1 Member Enumeration Documentation . . . . .	264
9.71.1.1 DisplayAlarmStateOptions . . . . .	264
9.71.1.2 UserLevels . . . . .	265
9.71.2 Member Function Documentation . . . . .	265
9.71.2.1 setManagedVisible . . . . .	265
9.71.3 Property Documentation . . . . .	265
9.71.3.1 allowDrop . . . . .	265
9.71.3.2 defaultStyle . . . . .	265
9.71.3.3 displayAlarmState . . . . .	265
9.71.3.4 displayAlarmStateOption . . . . .	265
9.71.3.5 int . . . . .	266
9.71.3.6 styleSheet . . . . .	266
9.71.3.7 userLevelEnabled . . . . .	266
9.71.3.8 userLevelEngineerStyle . . . . .	266
9.71.3.9 userLevelScientistStyle . . . . .	266
9.71.3.10 userLevelUserStyle . . . . .	267
9.71.3.11 userLevelVisibility . . . . .	267
9.71.3.12 variableAsToolTip . . . . .	267
9.71.3.13 visible . . . . .	267
9.72 QELogin Class Reference . . . . .	268
9.73 QELoginDialog Class Reference . . . . .	269
9.74 QEPersistent Class Reference . . . . .	270
9.74.1 Member Enumeration Documentation . . . . .	274
9.74.1.1 DisplayAlarmStateOptions . . . . .	274
9.74.1.2 UserLevels . . . . .	274
9.74.2 Member Function Documentation . . . . .	274
9.74.2.1 dbElementChanged . . . . .	274
9.74.2.2 dbValueChanged . . . . .	275
9.74.3 Member Data Documentation . . . . .	275
9.74.3.1 allowDrop . . . . .	275
9.74.4 Property Documentation . . . . .	275
9.74.4.1 displayAlarmState . . . . .	275
9.74.4.2 displayAlarmStateOption . . . . .	275
9.74.4.3 int . . . . .	275

9.74.4.4	readbackLabelVariable1 . . . . .	275
9.74.4.5	readbackLabelVariable2 . . . . .	276
9.74.4.6	subscribe . . . . .	276
9.74.4.7	userLevelEnabled . . . . .	276
9.74.4.8	userLevelEngineerStyle . . . . .	276
9.74.4.9	userLevelScientistStyle . . . . .	276
9.74.4.10	userLevelUserStyle . . . . .	276
9.74.4.11	userLevelVisibility . . . . .	277
9.74.4.12	variableAsToolTip . . . . .	277
9.74.4.13	variableSubstitutions . . . . .	277
9.74.4.14	visible . . . . .	277
9.74.4.15	writeButtonVariable1 . . . . .	277
9.74.4.16	writeButtonVariable2 . . . . .	277
9.75	QEPeriodicComponentData Class Reference . . . . .	278
9.76	QEPeriodicTaskMenu Class Reference . . . . .	279
9.77	QEPeriodicTaskMenuFactory Class Reference . . . . .	280
9.78	QEPlot Class Reference . . . . .	281
9.78.1	Member Enumeration Documentation . . . . .	285
9.78.1.1	DisplayAlarmStateOptions . . . . .	285
9.78.1.2	UserLevels . . . . .	285
9.78.2	Member Function Documentation . . . . .	285
9.78.2.1	dbValueChanged . . . . .	285
9.78.2.2	dbValueChanged . . . . .	285
9.78.2.3	setManagedVisible . . . . .	285
9.78.3	Member Data Documentation . . . . .	286
9.78.3.1	allowDrop . . . . .	286
9.78.4	Property Documentation . . . . .	286
9.78.4.1	defaultStyle . . . . .	286
9.78.4.2	displayAlarmState . . . . .	286
9.78.4.3	displayAlarmStateOption . . . . .	286
9.78.4.4	int . . . . .	286
9.78.4.5	styleSheet . . . . .	286
9.78.4.6	userLevelEnabled . . . . .	287
9.78.4.7	userLevelEngineerStyle . . . . .	287

9.78.4.8 userLevelScientistStyle . . . . .	287
9.78.4.9 userLevelUserStyle . . . . .	287
9.78.4.10 userLevelVisibility . . . . .	287
9.78.4.11 variable1 . . . . .	288
9.78.4.12 variable2 . . . . .	288
9.78.4.13 variable3 . . . . .	288
9.78.4.14 variable4 . . . . .	288
9.78.4.15 variableAsToolTip . . . . .	288
9.78.4.16 variableSubstitutions . . . . .	288
9.78.4.17 visible . . . . .	288
9.79 QEPushButton Class Reference . . . . .	289
9.79.1 Member Enumeration Documentation . . . . .	293
9.79.1.1 ArrayActions . . . . .	293
9.79.1.2 CreationOptionNames . . . . .	293
9.79.1.3 DisplayAlarmStateOptions . . . . .	294
9.79.1.4 Formats . . . . .	294
9.79.1.5 Notations . . . . .	294
9.79.1.6 ProgramStartupOptionNames . . . . .	295
9.79.1.7 UpdateOptions . . . . .	295
9.79.1.8 UserLevels . . . . .	295
9.79.2 Constructor & Destructor Documentation . . . . .	295
9.79.2.1 QEPushButton . . . . .	295
9.79.2.2 QEPushButton . . . . .	296
9.79.3 Member Function Documentation . . . . .	296
9.79.3.1 clicked . . . . .	296
9.79.3.2 dbValueChanged . . . . .	296
9.79.3.3 pressed . . . . .	296
9.79.3.4 released . . . . .	296
9.79.3.5 requestAction . . . . .	296
9.79.3.6 setManagedVisible . . . . .	297
9.79.4 Property Documentation . . . . .	297
9.79.4.1 addUnits . . . . .	297
9.79.4.2 alignment . . . . .	297
9.79.4.3 allowDrop . . . . .	297

9.79.4.4 altReadbackVariable . . . . .	297
9.79.4.5 arguments . . . . .	297
9.79.4.6 arrayAction . . . . .	297
9.79.4.7 arrayIndex . . . . .	298
9.79.4.8 clickCheckedText . . . . .	298
9.79.4.9 clickText . . . . .	298
9.79.4.10 confirmAction . . . . .	298
9.79.4.11 confirmText . . . . .	298
9.79.4.12 creationOption . . . . .	298
9.79.4.13 customisationName . . . . .	299
9.79.4.14 defaultStyle . . . . .	299
9.79.4.15 disabledRecordPolicy . . . . .	299
9.79.4.16 displayAlarmState . . . . .	299
9.79.4.17 displayAlarmStateOption . . . . .	299
9.79.4.18 format . . . . .	300
9.79.4.19 guiFile . . . . .	300
9.79.4.20 int . . . . .	300
9.79.4.21 labelText . . . . .	300
9.79.4.22 leadingZero . . . . .	300
9.79.4.23 localEnumeration . . . . .	300
9.79.4.24 notation . . . . .	301
9.79.4.25 password . . . . .	301
9.79.4.26 pixmap0 . . . . .	301
9.79.4.27 pixmap1 . . . . .	301
9.79.4.28 pixmap2 . . . . .	301
9.79.4.29 pixmap3 . . . . .	302
9.79.4.30 pixmap4 . . . . .	302
9.79.4.31 pixmap5 . . . . .	302
9.79.4.32 pixmap6 . . . . .	302
9.79.4.33 pixmap7 . . . . .	302
9.79.4.34 precision . . . . .	302
9.79.4.35 pressText . . . . .	302
9.79.4.36 prioritySubstitutions . . . . .	302
9.79.4.37 program . . . . .	303

9.79.4.38	programStartupOption . . . . .	303
9.79.4.39	releaseText . . . . .	303
9.79.4.40	styleSheet . . . . .	303
9.79.4.41	subscribe . . . . .	303
9.79.4.42	trailingZeros . . . . .	303
9.79.4.43	updateOption . . . . .	303
9.79.4.44	useDbPrecision . . . . .	303
9.79.4.45	userLevelEnabled . . . . .	304
9.79.4.46	userLevelEngineerStyle . . . . .	304
9.79.4.47	userLevelScientistStyle . . . . .	304
9.79.4.48	userLevelUserStyle . . . . .	304
9.79.4.49	userLevelVisibility . . . . .	304
9.79.4.50	variable . . . . .	305
9.79.4.51	variableAsToolTip . . . . .	305
9.79.4.52	variableSubstitutions . . . . .	305
9.79.4.53	visible . . . . .	305
9.79.4.54	writeOnClick . . . . .	305
9.79.4.55	writeOnPress . . . . .	305
9.79.4.56	writeOnRelease . . . . .	305
9.80	QEPvFrame Class Reference . . . . .	306
9.80.1	Member Function Documentation . . . . .	306
9.80.1.1	dbConnectionChanged . . . . .	306
9.80.1.2	dbValueChanged . . . . .	307
9.80.2	Property Documentation . . . . .	307
9.80.2.1	arrayIndex . . . . .	307
9.80.2.2	variable . . . . .	307
9.80.2.3	variableSubstitutions . . . . .	307
9.81	QEPvFrameManager Class Reference . . . . .	308
9.82	QEPVNameLists Class Reference . . . . .	309
9.83	QEPvProperties Class Reference . . . . .	310
9.83.1	Property Documentation . . . . .	311
9.83.1.1	variable . . . . .	311
9.83.1.2	variableSubstitutions . . . . .	311
9.84	QEPvPropertiesManager Class Reference . . . . .	312

9.85 QERadioButton Class Reference . . . . .	313
9.85.1 Member Enumeration Documentation . . . . .	317
9.85.1.1 ArrayActions . . . . .	317
9.85.1.2 CreationOptionNames . . . . .	317
9.85.1.3 DisplayAlarmStateOptions . . . . .	318
9.85.1.4 Formats . . . . .	318
9.85.1.5 Notations . . . . .	319
9.85.1.6 ProgramStartupOptionNames . . . . .	319
9.85.1.7 Separators . . . . .	319
9.85.1.8 UpdateOptions . . . . .	319
9.85.1.9 UserLevels . . . . .	320
9.85.2 Constructor & Destructor Documentation . . . . .	320
9.85.2.1 QERadioButton . . . . .	320
9.85.2.2 QERadioButton . . . . .	320
9.85.3 Member Function Documentation . . . . .	320
9.85.3.1 clicked . . . . .	320
9.85.3.2 dbValueChanged . . . . .	320
9.85.3.3 pressed . . . . .	321
9.85.3.4 released . . . . .	321
9.85.3.5 requestAction . . . . .	321
9.85.3.6 setManagedVisible . . . . .	321
9.85.4 Property Documentation . . . . .	321
9.85.4.1 addUnits . . . . .	321
9.85.4.2 alignment . . . . .	321
9.85.4.3 allowDrop . . . . .	321
9.85.4.4 arguments . . . . .	322
9.85.4.5 arrayAction . . . . .	322
9.85.4.6 arrayIndex . . . . .	322
9.85.4.7 clickCheckedText . . . . .	322
9.85.4.8 clickText . . . . .	322
9.85.4.9 confirmAction . . . . .	323
9.85.4.10 confirmText . . . . .	323
9.85.4.11 creationOption . . . . .	323
9.85.4.12 customisationName . . . . .	323

9.85.4.13 defaultStyle . . . . .	323
9.85.4.14 disabledRecordPolicy . . . . .	323
9.85.4.15 displayAlarmState . . . . .	324
9.85.4.16 displayAlarmStateOption . . . . .	324
9.85.4.17 format . . . . .	324
9.85.4.18 guiFile . . . . .	324
9.85.4.19 int . . . . .	324
9.85.4.20 labelText . . . . .	324
9.85.4.21 leadingZero . . . . .	325
9.85.4.22 localEnumeration . . . . .	325
9.85.4.23 notation . . . . .	325
9.85.4.24 password . . . . .	326
9.85.4.25 pixmap0 . . . . .	326
9.85.4.26 pixmap1 . . . . .	326
9.85.4.27 pixmap2 . . . . .	326
9.85.4.28 pixmap3 . . . . .	326
9.85.4.29 pixmap4 . . . . .	326
9.85.4.30 pixmap5 . . . . .	326
9.85.4.31 pixmap6 . . . . .	326
9.85.4.32 pixmap7 . . . . .	326
9.85.4.33 precision . . . . .	327
9.85.4.34 pressText . . . . .	327
9.85.4.35 prioritySubstitutions . . . . .	327
9.85.4.36 program . . . . .	327
9.85.4.37 programStartupOption . . . . .	327
9.85.4.38 radix . . . . .	327
9.85.4.39 releaseText . . . . .	327
9.85.4.40 separator . . . . .	328
9.85.4.41 styleSheet . . . . .	328
9.85.4.42 subscribe . . . . .	328
9.85.4.43 trailingZeros . . . . .	328
9.85.4.44 updateOption . . . . .	328
9.85.4.45 useDbPrecision . . . . .	328
9.85.4.46 userLevelEnabled . . . . .	328

9.85.4.47 userLevelEngineerStyle . . . . .	328
9.85.4.48 userLevelScientistStyle . . . . .	329
9.85.4.49 userLevelUserStyle . . . . .	329
9.85.4.50 userLevelVisibility . . . . .	329
9.85.4.51 variable . . . . .	329
9.85.4.52 variableAsToolTip . . . . .	329
9.85.4.53 variableSubstitutions . . . . .	329
9.85.4.54 visible . . . . .	330
9.85.4.55 writeOnClick . . . . .	330
9.85.4.56 writeOnPress . . . . .	330
9.85.4.57 writeOnRelease . . . . .	330
9.86 QERecipe Class Reference . . . . .	331
9.87 QERecordSpec Class Reference . . . . .	333
9.88 QERecordSpecList Class Reference . . . . .	334
9.89 QEScript Class Reference . . . . .	335
9.89.1 Detailed Description . . . . .	339
9.89.2 Member Enumeration Documentation . . . . .	340
9.89.2.1 DisplayAlarmStateOptions . . . . .	340
9.89.2.2 UserLevels . . . . .	340
9.89.3 Member Function Documentation . . . . .	340
9.89.3.1 setManagedVisible . . . . .	340
9.89.4 Property Documentation . . . . .	340
9.89.4.1 allowDrop . . . . .	340
9.89.4.2 defaultStyle . . . . .	340
9.89.4.3 displayAlarmState . . . . .	341
9.89.4.4 displayAlarmStateOption . . . . .	341
9.89.4.5 int . . . . .	341
9.89.4.6 styleSheet . . . . .	341
9.89.4.7 userLevelEnabled . . . . .	341
9.89.4.8 userLevelEngineerStyle . . . . .	341
9.89.4.9 userLevelScientistStyle . . . . .	342
9.89.4.10 userLevelUserStyle . . . . .	342
9.89.4.11 userLevelVisibility . . . . .	342
9.89.4.12 variableAsToolTip . . . . .	342

9.89.4.13	visible	342
9.90	QEShape Class Reference	344
9.90.1	Detailed Description	349
9.90.2	Member Enumeration Documentation	349
9.90.2.1	animationOptions	349
9.90.2.2	DisplayAlarmStateOptions	349
9.90.2.3	shapeOptions	349
9.90.2.4	UserLevels	349
9.90.3	Constructor & Destructor Documentation	350
9.90.3.1	QEShape	350
9.90.3.2	QEShape	350
9.90.4	Member Function Documentation	350
9.90.4.1	dbValueChanged1	350
9.90.4.2	dbValueChanged2	350
9.90.4.3	dbValueChanged3	350
9.90.4.4	dbValueChanged4	351
9.90.4.5	dbValueChanged5	351
9.90.4.6	dbValueChanged6	351
9.90.4.7	setManagedVisible	351
9.90.5	Property Documentation	351
9.90.5.1	allowDrop	351
9.90.5.2	animation1	351
9.90.5.3	animation2	351
9.90.5.4	animation3	352
9.90.5.5	animation4	352
9.90.5.6	animation5	352
9.90.5.7	animation6	352
9.90.5.8	color1	352
9.90.5.9	color10	352
9.90.5.10	color2	352
9.90.5.11	color3	352
9.90.5.12	color4	352
9.90.5.13	color5	353
9.90.5.14	color6	353

9.90.5.15 color7 . . . . .	353
9.90.5.16 color8 . . . . .	353
9.90.5.17 color9 . . . . .	353
9.90.5.18 defaultStyle . . . . .	353
9.90.5.19 displayAlarmState . . . . .	353
9.90.5.20 displayAlarmStateOption . . . . .	353
9.90.5.21 int . . . . .	354
9.90.5.22 offset1 . . . . .	354
9.90.5.23 offset2 . . . . .	354
9.90.5.24 offset3 . . . . .	354
9.90.5.25 offset4 . . . . .	354
9.90.5.26 offset5 . . . . .	354
9.90.5.27 offset6 . . . . .	354
9.90.5.28 point1 . . . . .	355
9.90.5.29 point10 . . . . .	355
9.90.5.30 point2 . . . . .	355
9.90.5.31 point3 . . . . .	355
9.90.5.32 point4 . . . . .	355
9.90.5.33 point5 . . . . .	355
9.90.5.34 point6 . . . . .	355
9.90.5.35 point7 . . . . .	355
9.90.5.36 point8 . . . . .	356
9.90.5.37 point9 . . . . .	356
9.90.5.38 scale2 . . . . .	356
9.90.5.39 scale3 . . . . .	356
9.90.5.40 scale4 . . . . .	356
9.90.5.41 scale5 . . . . .	356
9.90.5.42 scale6 . . . . .	356
9.90.5.43 styleSheet . . . . .	356
9.90.5.44 userLevelEnabled . . . . .	356
9.90.5.45 userLevelEngineerStyle . . . . .	357
9.90.5.46 userLevelScientistStyle . . . . .	357
9.90.5.47 userLevelUserStyle . . . . .	357
9.90.5.48 userLevelVisibility . . . . .	357

9.90.5.49 variable1 . . . . .	358
9.90.5.50 variable2 . . . . .	358
9.90.5.51 variable3 . . . . .	358
9.90.5.52 variable4 . . . . .	358
9.90.5.53 variable5 . . . . .	358
9.90.5.54 variable6 . . . . .	358
9.90.5.55 variableAsToolTip . . . . .	358
9.90.5.56 variableSubstitutions . . . . .	358
9.90.5.57 visible . . . . .	359
9.91 QESlider Class Reference . . . . .	360
9.91.1 Member Enumeration Documentation . . . . .	362
9.91.1.1 DisplayAlarmStateOptions . . . . .	362
9.91.1.2 UserLevels . . . . .	362
9.91.2 Member Function Documentation . . . . .	362
9.91.2.1 dbValueChanged . . . . .	362
9.91.2.2 setManagedVisible . . . . .	363
9.91.3 Member Data Documentation . . . . .	363
9.91.3.1 writeOnChange . . . . .	363
9.91.4 Property Documentation . . . . .	363
9.91.4.1 allowDrop . . . . .	363
9.91.4.2 allowFocusUpdate . . . . .	363
9.91.4.3 arrayIndex . . . . .	363
9.91.4.4 defaultStyle . . . . .	363
9.91.4.5 displayAlarmState . . . . .	363
9.91.4.6 displayAlarmStateOption . . . . .	364
9.91.4.7 int . . . . .	364
9.91.4.8 styleSheet . . . . .	364
9.91.4.9 subscribe . . . . .	364
9.91.4.10 userLevelEnabled . . . . .	364
9.91.4.11 userLevelEngineerStyle . . . . .	364
9.91.4.12 userLevelScientistStyle . . . . .	365
9.91.4.13 userLevelUserStyle . . . . .	365
9.91.4.14 userLevelVisibility . . . . .	365
9.91.4.15 variable . . . . .	365

9.91.4.16 variableAsToolTip . . . . .	365
9.91.4.17 variableSubstitutions . . . . .	365
9.91.4.18 visible . . . . .	366
9.92 QESpinBox Class Reference . . . . .	367
9.92.1 Member Enumeration Documentation . . . . .	369
9.92.1.1 DisplayAlarmStateOptions . . . . .	369
9.92.1.2 UserLevels . . . . .	369
9.92.2 Member Function Documentation . . . . .	370
9.92.2.1 dbValueChanged . . . . .	370
9.92.2.2 setManagedVisible . . . . .	370
9.92.3 Property Documentation . . . . .	370
9.92.3.1 allowDrop . . . . .	370
9.92.3.2 allowFocusUpdate . . . . .	370
9.92.3.3 arrayIndex . . . . .	370
9.92.3.4 defaultStyle . . . . .	370
9.92.3.5 displayAlarmState . . . . .	370
9.92.3.6 displayAlarmStateOption . . . . .	371
9.92.3.7 int . . . . .	371
9.92.3.8 styleSheet . . . . .	371
9.92.3.9 subscribe . . . . .	371
9.92.3.10 userLevelEnabled . . . . .	371
9.92.3.11 userLevelEngineerStyle . . . . .	371
9.92.3.12 userLevelScientistStyle . . . . .	372
9.92.3.13 userLevelUserStyle . . . . .	372
9.92.3.14 userLevelVisibility . . . . .	372
9.92.3.15 variable . . . . .	372
9.92.3.16 variableAsToolTip . . . . .	372
9.92.3.17 variableSubstitutions . . . . .	372
9.92.3.18 visible . . . . .	373
9.93 QEStripChart Class Reference . . . . .	374
9.93.1 Property Documentation . . . . .	376
9.93.1.1 variableSubstitutions . . . . .	376
9.94 QEStripChartAdjustPVDialo g Class Reference . . . . .	377
9.95 QEStripChartContextMenu Class Reference . . . . .	378

9.95.1 Constructor & Destructor Documentation . . . . .	378
9.95.1.1 QEStripContextMenu . . . . .	378
9.96 QEStripChartDurationDialog Class Reference . . . . .	379
9.97 QEStripChartItem Class Reference . . . . .	380
9.98 QEStripChartNames Class Reference . . . . .	381
9.99 QEStripChartPushButtonSpecifications Struct Reference . . . . .	383
9.100QEStripChartRangeDialog Class Reference . . . . .	384
9.101QEStripChartState Class Reference . . . . .	385
9.102QEStripChartStateList Class Reference . . . . .	386
9.103QEStripChartStatistics Class Reference . . . . .	387
9.104QEStripChartTimeDialog Class Reference . . . . .	388
9.105QEStripChartToolBar Class Reference . . . . .	389
9.105.1 Detailed Description . . . . .	390
9.106QESubstitutedLabel Class Reference . . . . .	391
9.106.1 Member Data Documentation . . . . .	391
9.106.1.1 labelText . . . . .	391
9.106.2 Property Documentation . . . . .	391
9.106.2.1 textSubstitutions . . . . .	391
9.107recording Class Reference . . . . .	392
9.108imageDisplayProperties::rgbPixel Struct Reference . . . . .	393
9.109screenSelectDialog Class Reference . . . . .	394
9.110selectMenu Class Reference . . . . .	395
9.111trace Class Reference . . . . .	396
9.112userInfoStruct Class Reference . . . . .	397
9.113QEPeriodic::userInfoStructArray Struct Reference . . . . .	398
9.114ValueScaling Class Reference . . . . .	399
9.115VideoWidget Class Reference . . . . .	400
9.116zoomMenu Class Reference . . . . .	402

# 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 >= 4.4.

## 1.4 Screenshots

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

Screenshots are only available in the HTML docs.

## 1.5 Downloads

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

For getting a development snapshot from the SVN repository:

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

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

## 1.6 Installation

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

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

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

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

## 1.7 Support

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

## 1.8 Related Projects

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

## 1.9 Credits:

### Authors:

Andrew Rhyder, Anthony Owen, Glenn Jackson

### Project admin:

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



## **Chapter 2**

# **GNU General Public License**

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

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

See the GNU General Public License for more details.

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

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

## **Chapter 3**

### **ASgui screen shots**

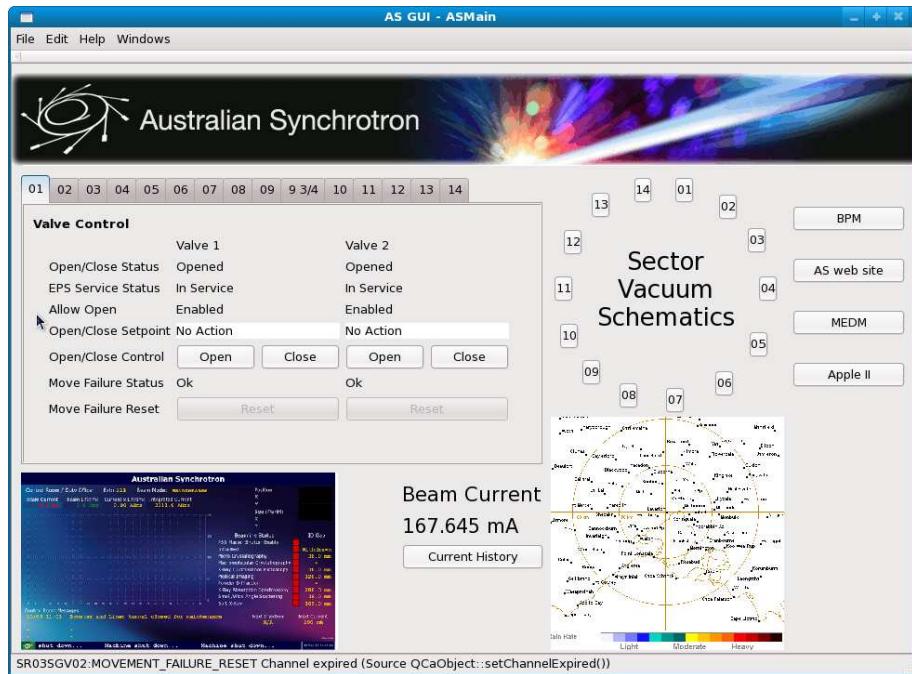


Figure 3.1: Australian Synchrotron mock up

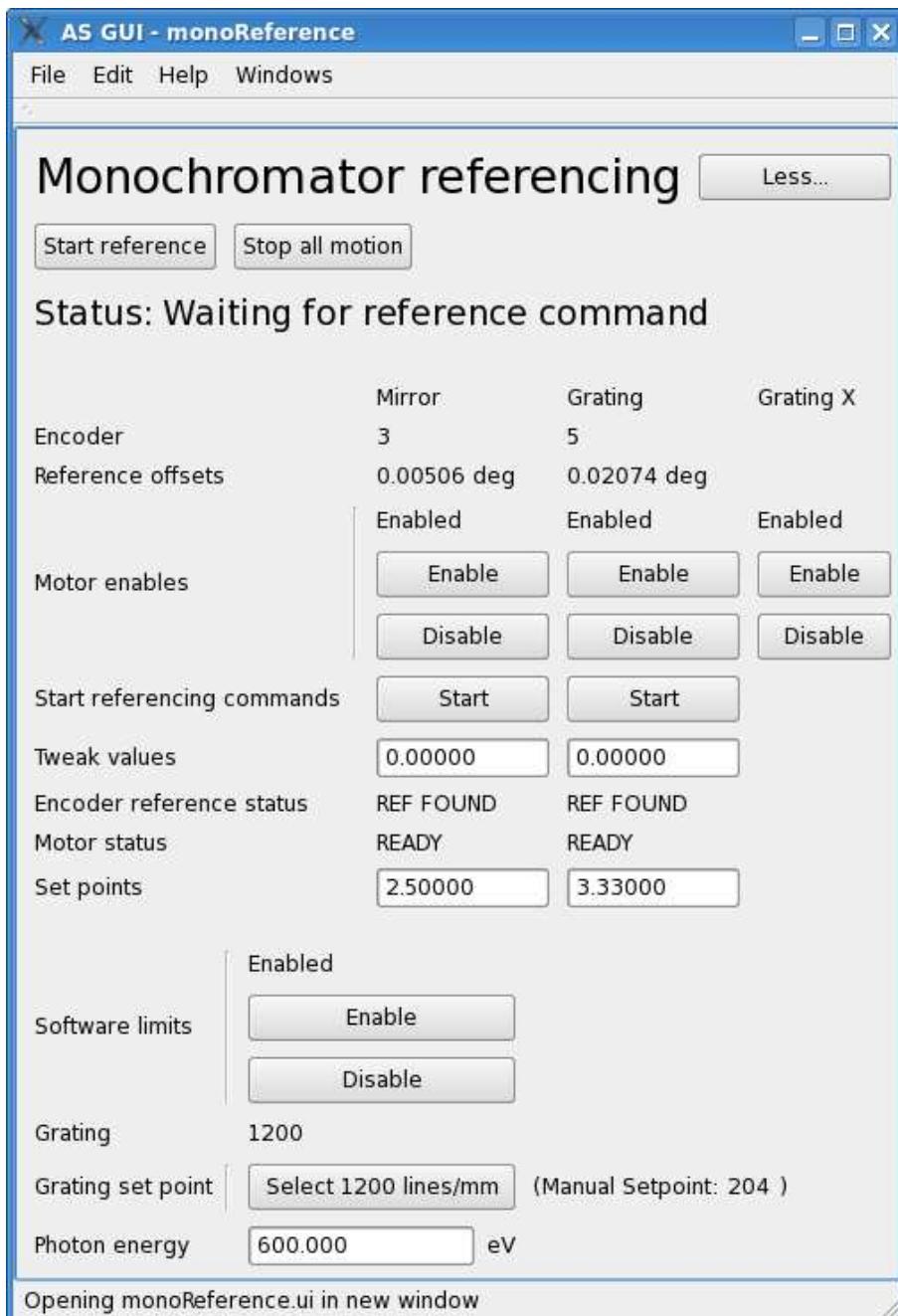


Figure 3.2: Monochromator referencing

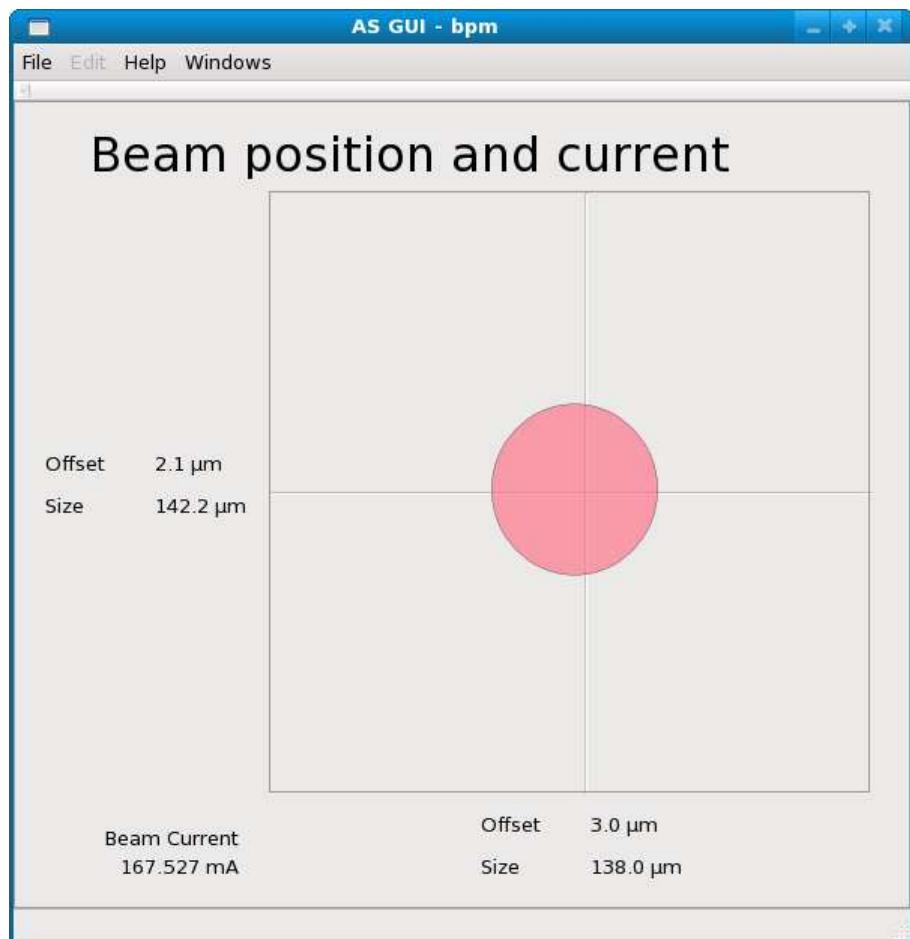


Figure 3.3: Beam position monitor

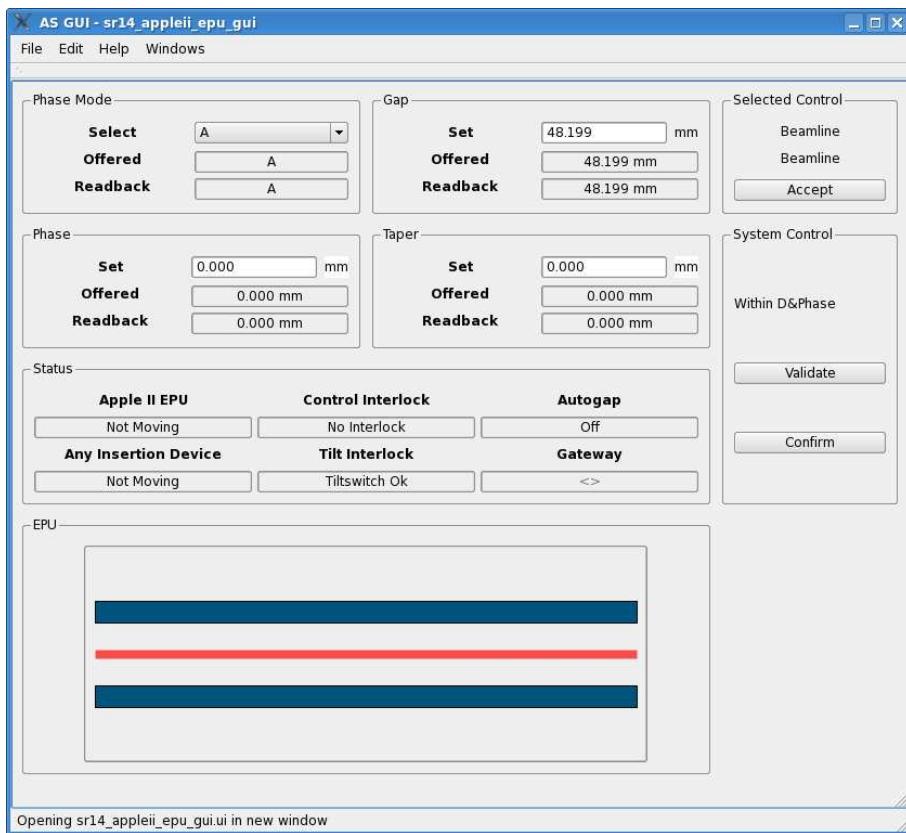


Figure 3.4: Insertion device



Figure 3.5: Injection efficiency monitor

## **Chapter 4**

# **other applications using epicsqt widgets**

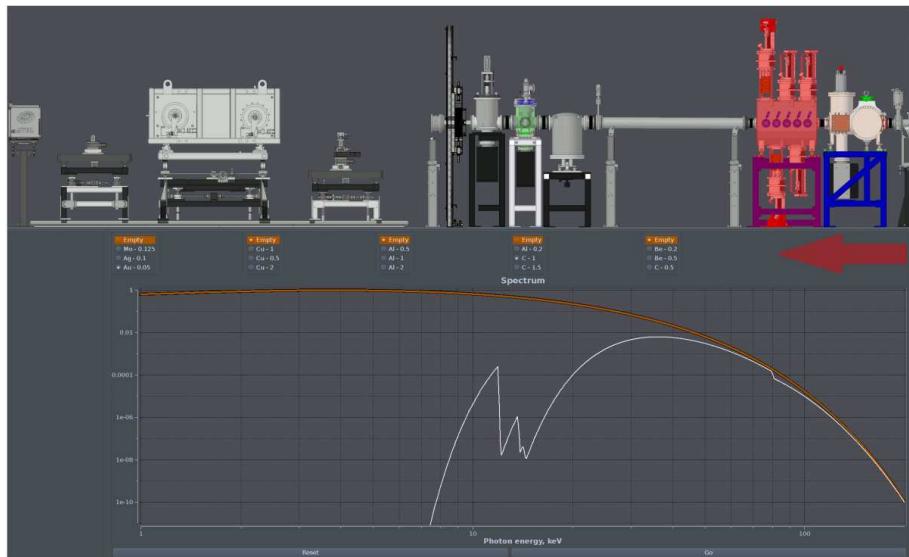


Figure 4.1: Medical Imaging beamline

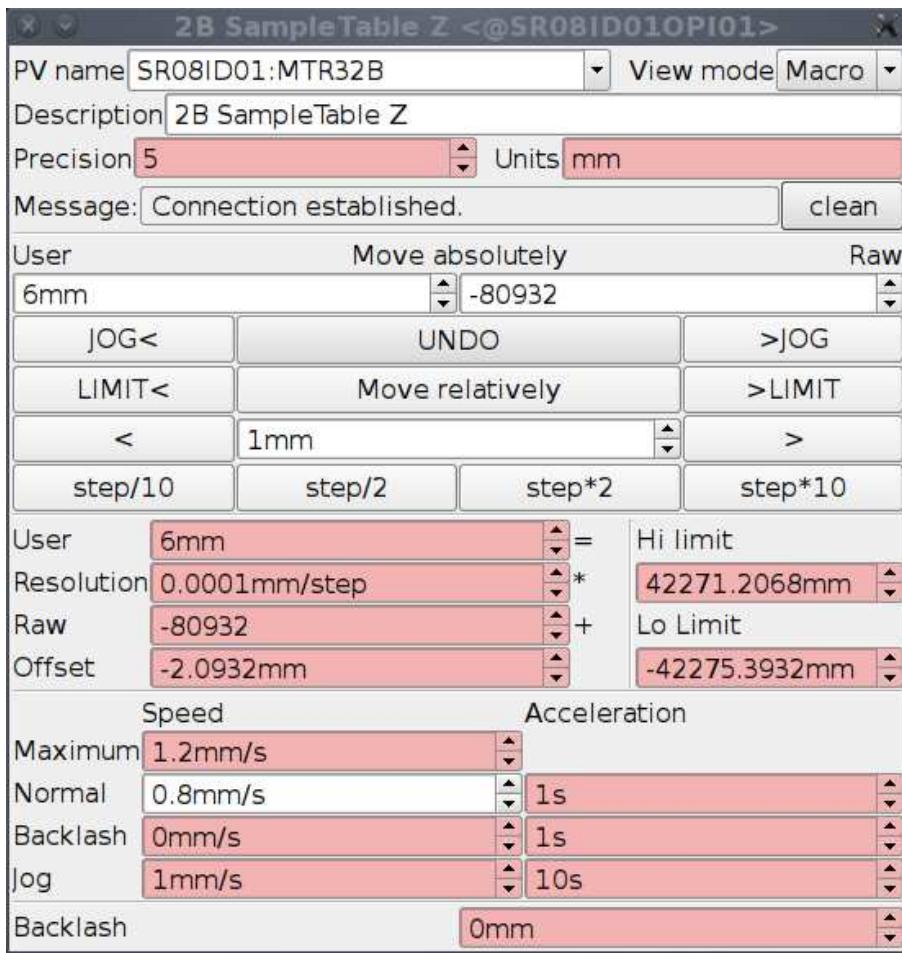


Figure 4.2: Motor controller

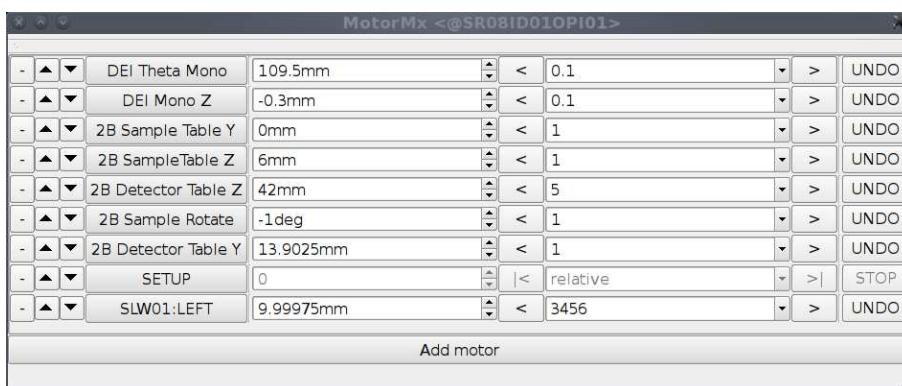


Figure 4.3: Motor controller



# **Chapter 5**

## **Qt Designer**

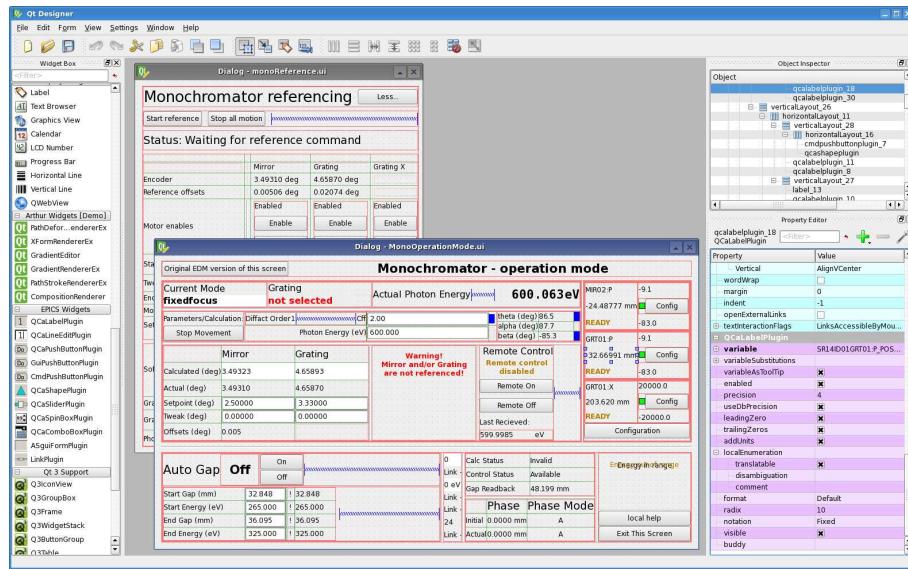


Figure 5.1: Editing multiple GUIs

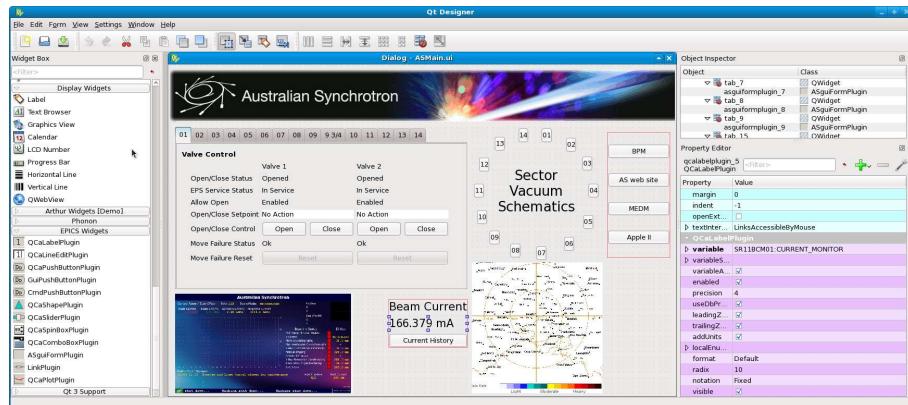
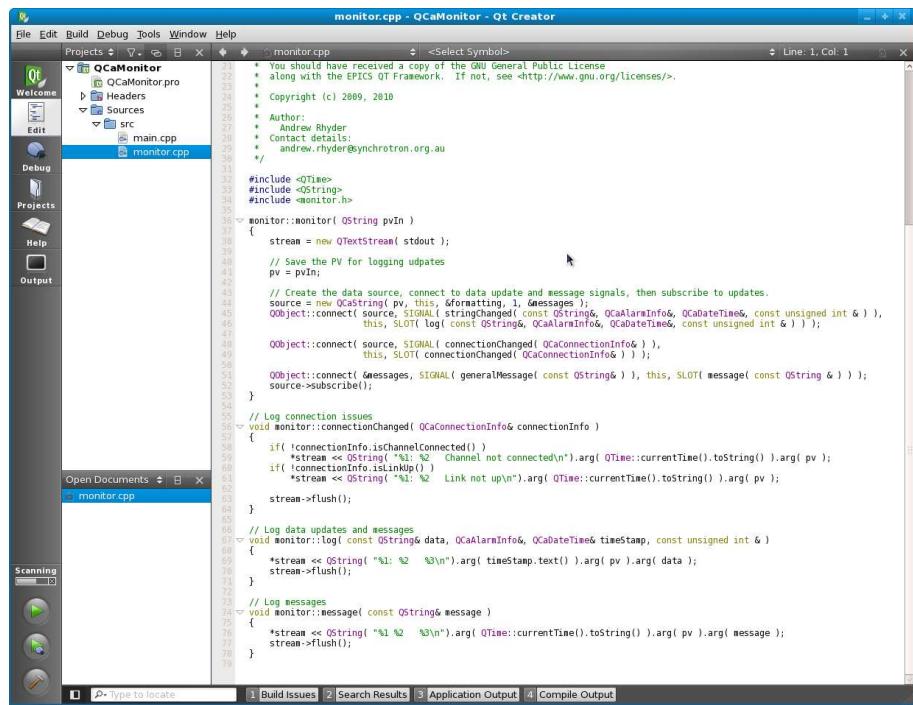


Figure 5.2: Editing a GUI

## **Chapter 6**

### **Qt Creator**



```

monitor.cpp - QCaMonitor - Qt Creator

File Edit Build Debug Tools Window Help
Projects < monitor.cpp <Select Symbols> Line: 1, Col: 1
File Edit Build Debug Tools Window Help
Welcome
QT Projects Sources src monitor.cpp
Edit
Debug
Help
Output
Open Documents monitor.cpp
Scanning
Type to locate 1 Build Issues 2 Search Results 3 Application Output 4 Compile Output

/*
 * 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/>.
 *
 * Copyright (c) 2009, 2010
 *
 * Author:
 * Andrew Ryder
 * Contact details:
 * andrew.ryder@synchrotron.org.au
 */
#ifndef MONITOR_H
#define MONITOR_H

#include <QTime>
#include <QString>
#include <qca.h>

class monitor : public monitorI {
public:
    monitor(QString pvIn)
    {
        stream = new QTextStream(stdout);
        // Save the PV for logging updates
        pv = pvIn;
        // Create the data source, connect to data update and message signals, then subscribe to updates.
        source = new QCString(pv, this, &formatting, 1, &messages);
        QObject::connect(source, SIGNAL(stringChanged(const QString&, QCaAlarmInfo&, QCaDateTime&, const unsigned int &)), this, SLOT(log(const QString&, QCaAlarmInfo&, QCaDateTime&, const unsigned int &)));
        QObject::connect(source, SIGNAL(connectionChanged(QCaConnectionInfo&)), this, SLOT(connectionChanged(QCaConnectionInfo&)));
        QObject::connect(&messages, SIGNAL(generalMessage(const QString&)), this, SLOT(message(const QString&)));
        source->subscribe();
    }
    // Log connection issues
    void monitor::connectionChanged(QCaConnectionInfo& connectionInfo)
    {
        if (!connectionInfo.isChannelConnected())
            *stream << QString("%%: %% Channel not connected\n").arg(QTime::currentTime().toString()).arg(pv);
        if (!connectionInfo.isLinkUp())
            *stream << QString("%%: %% Link not up\n").arg(QTime::currentTime().toString()).arg(pv);
        stream->flush();
    }
    // Log data updates and messages
    void monitor::log(const QString& data, QCaAlarmInfo&, QCaDateTime& timeStamp, const unsigned int & )
    {
        *stream << QString("%%: %% %3\n").arg(timeStamp.text()).arg(pv).arg(data);
        stream->flush();
    }
    // Log messages
    void monitor::message(const QString& message)
    {
        *stream << QString("%%: %% %3\n").arg(QTime::currentTime().toString()).arg(pv).arg(message);
        stream->flush();
    }
};

#endif

```

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 . . . . .	29
_Field . . . . .	30
_Item . . . . .	31
_QDialogItem . . . . .	32
_QPushButtonGroup . . . . .	33
_QTableWidgetFileBrowser . . . . .	34
_QTableWidgetLog . . . . .	35
_QTableWidgetScript . . . . .	36
areaInfo . . . . .	37
QEAnalogIndicator::Band . . . . .	38
QEAnalogIndicator::BandList . . . . .	39
QEPeriodic::elementInfoStruct . . . . .	40
FFBuffer . . . . .	41
FFThread . . . . .	42
flipRotateMenu . . . . .	43
fullScreenWindow . . . . .	44
histogram . . . . .	45
histogramScroll . . . . .	46
historicImage . . . . .	47
imageContextMenu . . . . .	48
imageDisplayProperties . . . . .	50
imageInfo . . . . .	52
QEImage . . . . .	184
imageMarkup . . . . .	53
VideoWidget . . . . .	400
imageMarkupLegendSetText . . . . .	56
imageProperties . . . . .	61
imageProcessor . . . . .	57

imagePropertiesCore . . . . .	64
imageUpdateIndicator . . . . .	65
loginWidget . . . . .	66
markupDisplayMenu . . . . .	69
markupItem . . . . .	72
markupCrosshair1 . . . . .	67
markupCrosshair2 . . . . .	68
markupEllipse . . . . .	70
markupHLine . . . . .	71
markupLine . . . . .	75
markupRegion . . . . .	76
markupText . . . . .	77
markupVLine . . . . .	78
mpegSource . . . . .	79
QEImage . . . . .	184
mpegSourceObject . . . . .	80
QEStripChartToolBar::OwnTabWidget . . . . .	81
PeriodicDialog . . . . .	82
PeriodicElementSetupForm . . . . .	83
PeriodicSetupDialog . . . . .	84
playbackTimer . . . . .	85
pointInfo . . . . .	86
profilePlot . . . . .	87
QBitStatus . . . . .	88
QEBitStatus . . . . .	109
QEAnalogIndicator . . . . .	90
QEAnalogProgressBar . . . . .	97
QECheckBoxManager . . . . .	133
QEComboBox . . . . .	134
QEConfiguredLayout . . . . .	141
QEConfiguredLayoutManager . . . . .	147
QEFileBrowser . . . . .	148
QEForm . . . . .	155
QEFrame . . . . .	160
QEPvFrame . . . . .	306
QEPvProperties . . . . .	310
QEStripChart . . . . .	374
QEGenericButton . . . . .	167
QECheckBox . . . . .	115
QEPushButton . . . . .	289
QERadioButton . . . . .	313
QEGenericEdit . . . . .	170
QELineEdit . . . . .	253
QEGroupBox . . . . .	179
QEImageMarkupThickness . . . . .	238
QEImageOptionsDialog . . . . .	239
QELabel . . . . .	240
QLineEditManager . . . . .	259

QELink . . . . .	260
QELog . . . . .	262
QELogin . . . . .	268
QELoginDialog . . . . .	269
QEPeriodic . . . . .	270
QEPeriodicComponentData . . . . .	278
QEPeriodicTaskMenu . . . . .	279
QEPeriodicTaskMenuFactory . . . . .	280
QEPlot . . . . .	281
QEPvFrameManager . . . . .	308
QEPVNameLists . . . . .	309
QEPvPropertiesManager . . . . .	312
QERecipe . . . . .	331
QERecordSpec . . . . .	333
QERecordSpecList . . . . .	334
QEScript . . . . .	335
QEShape . . . . .	344
QESlider . . . . .	360
QESpinBox . . . . .	367
QEStripChartAdjustPVDialog . . . . .	377
QEStripChartContextMenu . . . . .	378
QEStripChartDurationDialog . . . . .	379
QEStripChartItem . . . . .	380
QEStripChartNames . . . . .	381
QEStripChartPushButtonSpecifications . . . . .	383
QEStripChartRangeDialog . . . . .	384
QEStripChartState . . . . .	385
QEStripChartStateList . . . . .	386
QEStripChartStatistics . . . . .	387
QEStripChartTimeDialog . . . . .	388
QEStripChartToolBar . . . . .	389
QESubstitutedLabel . . . . .	391
recording . . . . .	392
imageDisplayProperties::rgbPixel . . . . .	393
screenSelectDialog . . . . .	394
selectMenu . . . . .	395
trace . . . . .	396
userInfoStruct . . . . .	397
QEPeriodic::userInfoStructArray . . . . .	398
ValueScaling . . . . .	399
zoomMenu . . . . .	402



# Chapter 8

## Class Index

### 8.1 Class List

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

_CopyPaste . . . . .	29
_Field . . . . .	30
_Item . . . . .	31
_QDialogItem . . . . .	32
_QPushButtonGroup . . . . .	33
_QTableWidgetFileBrowser . . . . .	34
_QTableWidgetLog . . . . .	35
_QTableWidgetScript . . . . .	36
areaInfo . . . . .	37
QEAnalogIndicator::Band . . . . .	38
QEAnalogIndicator::BandList . . . . .	39
QEPeriodic::elementInfoStruct . . . . .	40
FFBuffer . . . . .	41
FFThread . . . . .	42
flipRotateMenu . . . . .	43
fullScreenWindow . . . . .	44
histogram . . . . .	45
histogramScroll . . . . .	46
historicImage . . . . .	47
imageContextMenu . . . . .	48
imageDisplayProperties . . . . .	50
imageInfo . . . . .	52
imageMarkup . . . . .	53
imageMarkupLegendSetText . . . . .	56
imageProcessor . . . . .	57
imageProperties . . . . .	61
imagePropertiesCore . . . . .	64
imageUpdateIndicator . . . . .	65
loginWidget . . . . .	66

markupCrosshair1	67
markupCrosshair2	68
markupDisplayMenu	69
markupEllipse	70
markupHLine	71
markupItem	72
markupLine	75
markupRegion	76
markupText	77
markupVLine	78
mpegSource	79
mpegSourceObject	80
QEStripChartToolBar::OwnTabWidget	81
PeriodicDialog	82
PeriodicElementSetupForm	83
PeriodicSetupDialog	84
playbackTimer	85
pointInfo	86
profilePlot	87
QBitStatus	88
QEAnalogIndicator	90
QEAnalogProgressBar	97
QEBitStatus	109
QECheckBox	115
QECheckBoxManager	133
QEComboBox	134
QEConfiguredLayout	141
QEConfiguredLayoutManager	147
QEFileBrowser	148
QEForm	155
QEFrame	160
QEGenericButton	167
QEGenericEdit	170
QEGroupBox	179
QEImage	184
QEImageMarkupThickness	238
QEImageOptionsDialog	239
QELabel	240
QELineEdit	253
QELineEditManager	259
QELink	260
QELog	262
QELogin	268
QELoginDialog	269
QEPeriodic	270
QEPeriodicComponentData	278
QEPeriodicTaskMenu	279
QEPeriodicTaskMenuFactory	280
QEPlot	281
QEPushButton	289

QEPvFrame . . . . .	306
QEPvFrameManager . . . . .	308
QEPVNameLists . . . . .	309
QEPvProperties . . . . .	310
QEPvPropertiesManager . . . . .	312
QERadioButton . . . . .	313
QERecipe . . . . .	331
QERecordSpec . . . . .	333
QERecordSpecList . . . . .	334
QEScript . . . . .	335
QEShape . . . . .	344
QESlider . . . . .	360
QESpinBox . . . . .	367
QEStripChart . . . . .	374
QEStripChartAdjustPVDialo	377
QEStripChartContextMenu . . . . .	378
QEStripChartDurationDialog . . . . .	379
QEStripChartItem . . . . .	380
QEStripChartNames . . . . .	381
QEStripChartPushButtonSpecifications . . . . .	383
QEStripChartRangeDialog . . . . .	384
QEStripChartState . . . . .	385
QEStripChartStateList . . . . .	386
QEStripChartStatistics . . . . .	387
QEStripChartTimeDialog . . . . .	388
QEStripChartToolBar (This class holds all the StripChart tool bar widgets ) . . . . .	389
QESubstitutedLabel . . . . .	391
recording . . . . .	392
imageDisplayProperties::rgbPixel . . . . .	393
screenSelectDialog . . . . .	394
selectMenu . . . . .	395
trace . . . . .	396
userInfoStruct . . . . .	397
QEPeriodic::userInfoStructArray . . . . .	398
ValueScaling . . . . .	399
VideoWidget . . . . .	400
zoomMenu . . . . .	402



# 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 areaInfo 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/QEImage/mpeg.h
- /tmp/epicsqt/trunk/framework/widgets/QEImage/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/QEImage/flipRotateMenu.h
- /tmp/epicsqt/trunk/framework/widgets/QEImage/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/QEImage/fullScreenWindow.h
- /tmp/epicsqt/trunk/framework/widgets/QEImage/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/QEImage/brightnessContrast.h
- /tmp/epicsqt/trunk/framework/widgets/QEImage/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/QEImage/brightnessContrast.h](#)
- [/tmp/epicsqt/trunk/framework/widgets/QEImage/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/QEImage/recording.h
- /tmp/epicsqt/trunk/framework/widgets/QEImage/QEImage.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 **getContextMenuItemOption** (const QPoint &, imageContextMenuOptions \*option, bool \*checked)
- void **addMenuItem** (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/QEImage/imageContextMenu.h
- /tmp/epicsqt/trunk/framework/widgets/QEImage/imageContextMenu.cpp

## 9.21 imageDisplayProperties Class Reference

### Classes

- struct [rgbPixel](#)

### Signals

- void **brightnessContrastAutoImage** ()
- 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 minPIn, unsigned int maxPIn, unsigned int bitDepth, unsigned int binsIn[HISTOGRAM\_BINS], [rgbPixel](#) pixelLookup[256])
- void **showStatistics** ()
- void **setHistZoom** (int value)
- int **getHistZoom** ()
- bool **statisticsValid** ()

### Public Attributes

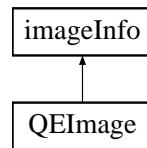
- int **zeroValue**
- int **fullValue**
- bool **defaultFullValue**
- unsigned int **range**
- unsigned int **maxP**
- unsigned int **minP**
- unsigned int **depth**
- unsigned int **bins** [HISTOGRAM\_BINS]
- bool **statisticsSet**
- [rgbPixel](#) \* **pixelLookup**
- QLabel \* **histXLabel**

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

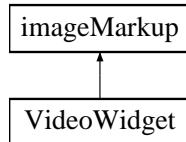
- void **showInfo** (bool show)
- QLayout \* **getInfoWidget** ()
- void **infoShow** (const bool show)
- void **infoUpdateTarget** ()
- void **infoUpdateTarget** (const int x, const int y)
- void **infoUpdateBeam** ()
- void **infoUpdateBeam** (const int x, const int y)
- void **infoUpdateVertProfile** ()
- 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, const double XStretch, const double YStretch)
  
- 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/QEImage/imageInfo.h
- /tmp/epicsqt/trunk/framework/widgets/QEImage/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 areaIndex, QRect area, bool display-  
Markups)
- 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 **markupMouseEvent** (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/QEImage/imageMarkup.h
- /tmp/epicsqt/trunk/framework/widgets/QEImage/imageMarkup.cpp

## 9.24 imageMarkupLegendSetText Class Reference

### Public Member Functions

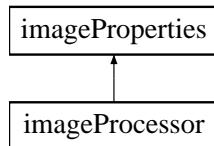
- **imageMarkupLegendSetText** (QString existingLegend, QWidget \*parent=0)
- **QString getLegend ()**

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

- /tmp/epicsqt/trunk/framework/widgets/QEImage/imageMarkupLegendSetText.h
- /tmp/epicsqt/trunk/framework/widgets/QEImage/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 in 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 rotateFlipToPoint (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** ()

## 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.

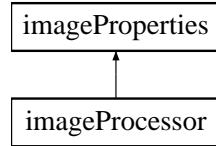
!! not done - copy of RGB1

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

- /tmp/epicsqt/trunk/framework/widgets/QEImage/imageProcessor.h
- /tmp/epicsqt/trunk/framework/widgets/QEImage/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 imageSize`
- `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/QEImage/imageProperties.h
- /tmp/epicsqt/trunk/framework/widgets/QEImage/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 imageSizeIn, [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

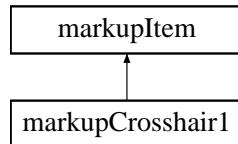
- **loginWidget** ([QELogin](#) \*ownerIn)
- userLevelTypes::userLevels **getUserType** ()
- **QString getPassword** ()
- **void clearPassword** ()

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

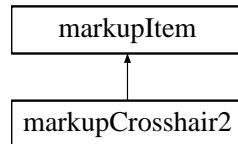
- **markupCrosshair1** (`imageMarkup *ownerIn`, `const bool interactiveIn`, `const bool reportOnMoveIn`, `const QString legendIn`)
- **void startDrawing** (`const QPoint pos`)
- **void setArea** ()
- **void drawMarkup** (`QPainter &p`)
- **void moveTo** (`const QPoint pos`)
- **bool isOver** (`const 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/QEImage/markupTarget.h`
- `/tmp/epicsqt/trunk/framework/widgets/QEImage/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** ()
- 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/QEImage/markupBeam.h
- /tmp/epicsqt/trunk/framework/widgets/QEImage/markupBeam.cpp

## 9.32 markupDisplayMenu Class Reference

### Public Member Functions

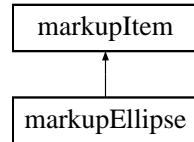
- **markupDisplayMenu** (QWidget \*parent=0)
- void **setDisplayed** (imageContextMenu::imageContextMenuOptions option, bool state)
- void **setItemText** (imageContextMenu::imageContextMenuOptions option, QString title)
- bool **isDisplayed** (imageContextMenu::imageContextMenuOptions option)
- void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

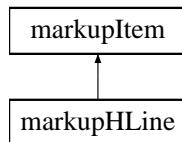
- **markupEllipse** ([imageMarkup](#) \*ownerIn, const bool interactiveIn, const bool reportOnMoveIn, const QString legendIn)
- void **startDrawing** (const QPoint pos)
- void **setArea** ()
- void **drawMarkup** ( QPainter &p )
- void **moveTo** (const QPoint pos)
- bool **isOver** (const QPoint point, 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/QEImage/markupEllipse.h
- /tmp/epicsqt/trunk/framework/widgets/QEImage/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, QCursors *cursor)`
- **QPoint origin** ()
- **QCursors cursorForHandle** (`const markupItem::markupHandles handle)`
- **QPoint getPoint1** ()
- **QPoint getPoint2** ()
- **QCursors 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

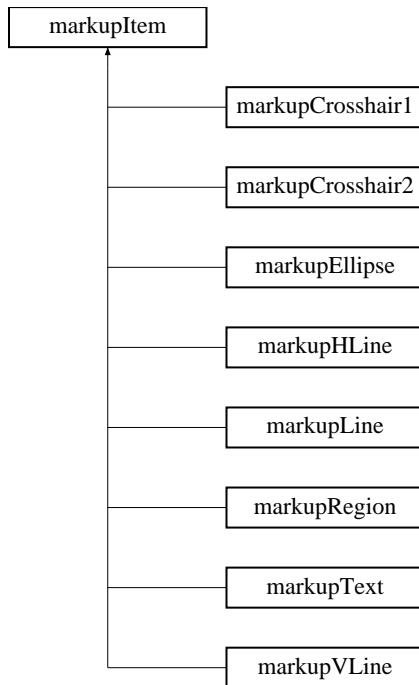
Implements [markupItem](#).

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 QCursor **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

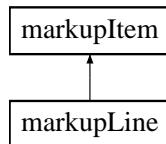
- markupHandles **activeHandle**
- **imageMarkup** \* **owner**
- unsigned int **thickness**
- unsigned int **maxThickness**

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

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

## 9.36 markupLine Class Reference

Inheritance diagram for markupLine::



### Public Member Functions

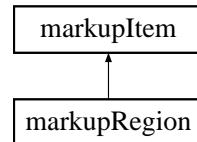
- **markupLine** ([imageMarkup](#) \*ownerIn, const bool interactiveIn, const bool reportOnMoveIn, const QString legendIn)
- void **startDrawing** (const QPoint pos)
- void **setArea** ()
- void **drawMarkup** ( QPainter &p )
- void **moveTo** (const QPoint pos)
- bool **isOver** (const QPoint point, 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/QEImage/markupLine.h
- /tmp/epicsqt/trunk/framework/widgets/QEImage/markupLine.cpp

## 9.37 markupRegion Class Reference

Inheritance diagram for markupRegion::



### Public Member Functions

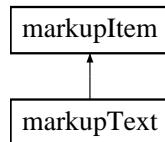
- **markupRegion** ([imageMarkup](#) \*ownerIn, const bool interactiveIn, const bool reportOnMoveIn, const QString legendIn)
- void **startDrawing** (const QPoint pos)
- void **setArea** ()
- void **drawMarkup** ( QPainter &p )
- void **moveTo** (const QPoint pos)
- bool **isOver** (const QPoint point, QCursor \*cursor)
- QPoint **origin** ()
- 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/QEImage/markupRegion.h
- /tmp/epicsqt/trunk/framework/widgets/QEImage/markupRegion.cpp

## 9.38 markupText Class Reference

Inheritance diagram for markupText::



### Public Member Functions

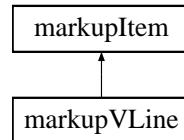
- **markupText** ([imageMarkup](#) \*ownerIn, const bool interactiveIn, const bool reportOnMoveIn, const QString legendIn)
- void **setText** (QString textIn)
- void **startDrawing** (const QPoint pos)
- void **setArea** ()
- void **drawMarkup** ( QPainter &p )
- void **moveTo** (const QPoint pos)
- bool **isOver** (const QPoint point, QCursor \*cursor)
- QPoint **origin** ()
- QCursor **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/QEImage/markupText.h
- /tmp/epicsqt/trunk/framework/widgets/QEImage/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, QCursor \*cursor)
- QPoint **origin** ()
- QCursors **cursorForHandle** (const markupItem::markupHandles handle)
- QPoint **getPoint1** ()
- QPoint **getPoint2** ()
- QCursors **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

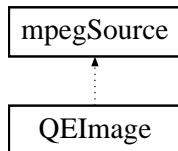
Implements [markupItem](#).

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

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

## 9.42 QEStripChartToolBar::OwnTabWidget Class Reference

### Public Member Functions

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

### Public Attributes

- QPushButton \* **pushButtons** [NUMBER\_OF\_BUTTONS]
- QLabel \* **yScaleStatus**
- QLabel \* **timeStatus**
- QLabel \* **durationStatus**
- QLabel \* **numberOfOutstandingRequests**
- QLabel \* **timeModeStatus**
- QComboBox \* **predefinedComboBox**
- QPushButton \* **loadButton**
- QPushButton \* **saveAsButton**
- QLabel \* **timeRefLabel**
- QLabel \* **time1**
- QLabel \* **time2**
- QLabel \* **timeDeltaLabel**
- QLabel \* **timeDelta**
- QLabel \* **valueRefLabel**
- QLabel \* **value1**
- QLabel \* **value2**
- QLabel \* **valueDelta1**
- QLabel \* **value3**
- QLabel \* **value4**
- QLabel \* **valueDelta2**
- QLabel \* **placeHolder2**
- QLabel \* **placeHolder3**

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

## 9.47 pointInfo Class Reference

### Public Member Functions

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

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

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

## 9.48 profilePlot Class Reference

### Public Types

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

### Public Member Functions

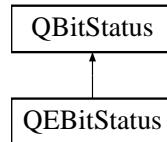
- **profilePlot** (`plotDirections plotDirectionIn`)
- **void setProfile** (`QVector< QPointF > *profile, double minX, double maxX, double minY, double maxY, QString title, QPoint start, QPoint end, unsigned int thicknessIn`)
- **void clearProfile ()**

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

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

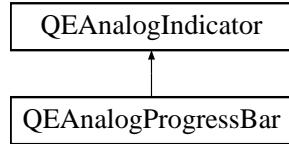
- `int value`
- `int numberOfBits`
- `int shift`
- `Orientations Orientation`
- `Shapes shape`
- `int gap`
- `QString reversePolarityMask`
- `QString onClearMask`
- `QString offClearMask`
- `QColor boarderColour`
- `QColor invalidColour`
- `QColor onColour`
- `QColor offColour`
- `QColor clearColour`
- `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 degreed 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.

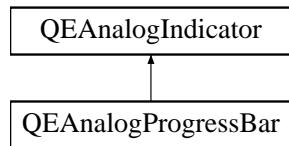
Reimplemented in [QEAnalogProgressBar](#).

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, `Under_score` = 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

- 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 [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 [AlarmSeverityDisplayModes](#) property - refer to [AlarmSeverityDisplayModes](#) property for details.*
- AlarmSeverityDisplayModes [getAlarmSeverityDisplayStyle \(\)](#)  
*Access function for [AlarmSeverityDisplayModes](#) property - refer to [AlarmSeverityDisplayModes](#) 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`
- `int arrayIndex`
- `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 `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.

### 9.51.1.2 enum QEAnalogProgressBar::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.

### 9.51.1.3 enum QEAnalogProgressBar::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.

### 9.51.1.4 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 call 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::alarmSeverityDisplayMode [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 int QEAnalogProgressBar::arrayIndex [read, write]**

Index used to select a single item of data for processing. The default is 0.

**9.51.4.6 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.

**9.51.4.7 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.

**9.51.4.8 DisplayAlarmStateOptions QEAnalogProgress-  
Bar::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.

**9.51.4.9 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).

**9.51.4.10 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.

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

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

**9.51.4.12 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"

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

Notation used for numerical formatting. Default is fixed.

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

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

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

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

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

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

**9.51.4.17 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.

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

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

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

Use the EPICS database display limits

**9.51.4.20 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.

**9.51.4.21 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' .

**9.51.4.22 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.

**9.51.4.23 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.

#### 9.51.4.24 **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.

#### 9.51.4.25 **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 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'.

#### 9.51.4.26 **int QEAnalogProgressBar::value [read, write]**

Current indicated value.

Reimplemented from [QEAnalogIndicator](#).

#### 9.51.4.27 **QString QEAnalogProgressBar::variable [read, write]**

EPICS variable name (CA PV)

#### 9.51.4.28 **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.

#### 9.51.4.29 **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 exam-

ple, '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.

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

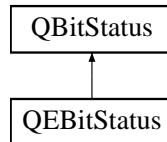
Display the widget. Default is true. Setting this property false is useful if widget is only used to provide a signal - for example, when supplying data to a [QELink](#) widget. Note, when false the widget will still be visible in Qt Designer.

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

- `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)`

## Protected Member Functions

- `qcaobject::QCaObject * createQcaItem (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`
- `int arrayIndex`
- `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**
- 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]**

Index used to select a single item of data for processing. The default is 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]**

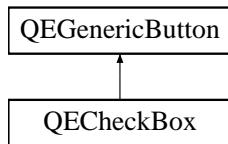
Display the widget. Default is true. Setting this property false is useful if widget is only used to provide a signal - for example, when supplying data to a [QELink](#) widget. Note, when false the widget will still be visible in Qt Designer.

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, `Under_score` = 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\_STDOUPUT }

- 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

- `QECheckBox (QWidget *parent=0)`
- `QECheckBox (const QString &variableName, QWidget *parent=0)`
- `void writeNow ()`
- `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
- int `arrayIndex`
- 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](#)
- `QWidgetProperties::DisabledRecordPolicy disabledRecordPolicy`
- `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 releaseText 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 int QECheckBox::arrayIndex [read, write]**

Index used to select a single item of data for processing. The default is 0.

**9.53.4.7 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](#).

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

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

Reimplemented from [QEGenericButton](#).

**9.53.4.9 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

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

Text used to confirm action if confirmation dialog is presented

Reimplemented from [QEGenericButton](#).

**9.53.4.11 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](#).

**9.53.4.12 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. Customisations are not applied if the GUI is opened as a dock.

Reimplemented from [QEGenericButton](#).

**9.53.4.13 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.

**9.53.4.14 QEWidgetProperties::DisabledRecordPolicy  
QECheckBox::disabledRecordPolicy [read, write]**

Set the widget's disabled record policy, i.e. the action to be taken when the underlying record is disabled, i.e. when the associated record's DISA and DISV field values are equal. Note: this is only applicable IOC process variables. When the policy is ignore, then no special action is taken. This is the default policy. When the policy is grayout, the widget is styled as if disconnected when the record is disabled.

Reimplemented from [QEGenericButton](#).

**9.53.4.15 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.

**9.53.4.16 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.

**9.53.4.17 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).

**9.53.4.18 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 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.

**9.53.4.19 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.

**9.53.4.20 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](#).

#### **9.53.4.21 bool QECheckBox::leadingZero [read, write]**

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

#### **9.53.4.22 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!, 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"

#### **9.53.4.23 Notations QECheckBox::notation [read, write]**

Notation used for numerical formatting. Default is fixed.

**9.53.4.24 QString QECheckBox::password [read, write]**

Password user will need to enter before any action is taken

Reimplemented from [QEGenericButton](#).

**9.53.4.25 QPixmap QECheckBox:: pixmap0 [read, write]**

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

**9.53.4.26 QPixmap QECheckBox:: pixmap1 [read, write]**

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

Reimplemented from [QEGenericButton](#).

**9.53.4.35 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](#).

**9.53.4.36 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

**9.53.4.37 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.

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

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

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

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

Reimplemented from [QEGenericButton](#).

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

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

**9.53.4.41 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.

**9.53.4.42 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)

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

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

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

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

Reimplemented from [QEGenericButton](#).

**9.53.4.45 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.

**9.53.4.46 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'.

**9.53.4.47 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.

#### 9.53.4.48 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.

#### 9.53.4.49 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.

#### 9.53.4.50 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'.

#### 9.53.4.51 QString QECheckBox::variable [read, write]

EPICS variable name (CA PV)

#### 9.53.4.52 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.

#### 9.53.4.53 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.

**9.53.4.54 bool QECheckBox::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.

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

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

Reimplemented from [QEGenericButton](#).

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

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

Reimplemented from [QEGenericButton](#).

**9.53.4.57 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` () const
- void `setSubscribe` (bool subscribe)
- bool `getSubscribe` () const
- void `setUseDbEnumerations` (bool `useDbEnumerations`)

- bool **getUseDbEnumerations () const**
- void **setLocalEnumerations (const QString &localEnumerations)**
- QString **getLocalEnumerations () const**
- void **setAllowFocusUpdate (bool allowFocusUpdate)**
- bool **getAllowFocusUpdate () const**
- void **writeNow ()**
- 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
- int arrayIndex
- bool subscribe
- bool allowFocusUpdate
- 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 bool QEComboBox::allowFocusUpdate [read, write]**

Allow updated while widget has focus - defaults to false

**9.55.4.3 int QEComboBox::arrayIndex [read, write]**

Index used to select a single item of data for processing. The default is 0.

**9.55.4.4 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.

**9.55.4.5 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.

**9.55.4.6 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.

**9.55.4.7 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.

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

Enumrations values used when useDbEnumerations is false.

**9.55.4.9 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.

**9.55.4.10 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)

**9.55.4.11 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'.

**9.55.4.12 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.

**9.55.4.13 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.

**9.55.4.14 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.

**9.55.4.15 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'.

**9.55.4.16 QString QEComboBox::variable [read, write]**

EPICS variable name (CA PV)

**9.55.4.17 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.

**9.55.4.18 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.

**9.55.4.19 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 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`

### 9.56.1 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 programmaticaly through setUserLevel(). Widgets that are always accessible should be visible at 'User'. Widgets that are only accessable to scientists managing the facility should be visible at 'Scientist'. Widgets that are only accessable 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

- QELLineEdit \* **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 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`

### 9.58.1 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 to this slot, but 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 \(const bool alertIfUINoFoundIn, const bool loadManuallyIn\)](#)
- 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 **getUniqueIdentifier** ()
 

*Get the version of the first QE widget (if any) of QE widgets by QUILoader.*
- void **setUniqueIdentifier** (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 Member Functions

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

## 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 resend with the messageSourceId 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]**

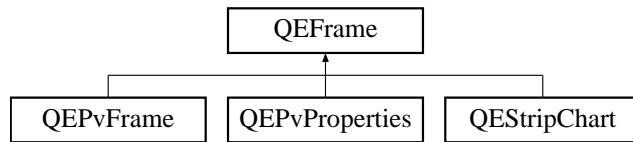
Macro substitutions to be applied to this widget, and all QE widgets that are opened when the .ui file is presented. Note, despite the name, the macro substitutions are general macro substitutions, and do not just apply to a variable name (in fact a [QEForm](#) widget does not even have a variable name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

### 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.*
- **QEFrame** (QWidget \*parent=0)
- QSize **sizeHint** () const
- void **setScaledContents** (bool scaledContentsIn)
- bool **getScaledContents** () const
- int **getSelectedPixmap** () const

## Protected Member Functions

- void **paintEvent** (QPaintEvent \*event)
- void  **pixmapUpdated** (const int index)

## 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**
- QPixmap  **pixmap**
- bool  **scaledContents**
- QPixmap  **pixmap0**
- QPixmap  **pixmap1**
- QPixmap  **pixmap2**
- QPixmap  **pixmap3**
- QPixmap  **pixmap4**
- QPixmap  **pixmap5**
- QPixmap  **pixmap6**
- QPixmap  **pixmap7**

### 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 QPixmap QEFrame:: pixmap [read, write]**

Pixmap for frame background. Similar operation to pixmap property for a QLabel. Deprecated, and is an alias for pixmap0

**9.60.3.7 QPixmap QEFrame:: pixmap0 [read, write]**

Pixmap displayed when updateOption property is 'Picture' and data is interpreted as 0.

**9.60.3.8 QPixmap QEFrame:: pixmap1 [read, write]**

Pixmap displayed when updateOption property is 'Picture' and data is interpreted as 1.

**9.60.3.9 QPixmap QEFrame:: pixmap2 [read, write]**

Pixmap displayed when updateOption property is 'Picture' and data is interpreted as 2.

**9.60.3.10 QPixmap QEFrame:: pixmap3 [read, write]**

Pixmap displayed when updateOption property is 'Picture' and data is interpreted as 3.

**9.60.3.11 QPixmap QEFrame:: pixmap4 [read, write]**

Pixmap displayed when updateOption property is 'Picture' and data is interpreted as 4.

**9.60.3.12 QPixmap QEFrame:: pixmap5 [read, write]**

Pixmap displayed when updateOption property is 'Picture' and data is interpreted as 5.

**9.60.3.13 QPixmap QEFrame:: pixmap6 [read, write]**

Pixmap displayed when updateOption property is 'Picture' and data is interpreted as 6.

**9.60.3.14 QPixmap QEFrame:: pixmap7 [read, write]**

Pixmap displayed when updateOption property is 'Picture' and data is interpreted as 7.

**9.60.3.15 bool QEFrame:: scaledContents [read, write]**

Flag the pixmap for the background is to be scaled to fit the frame. Similar operation to scaledContents property for a QLabel.

**9.60.3.16 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.

**9.60.3.17 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'.

**9.60.3.18 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.

**9.60.3.19 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.

**9.60.3.20 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.

**9.60.3.21 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'.

**9.60.3.22 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.

**9.60.3.23 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.

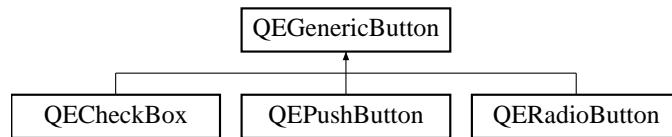
Note, when false the widget will still be visible in Qt Designer.

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 **VariableAllocation** {
   
    **VAR\_PRIMARY** = 0, **VAR\_READBACK**, **VAR\_DISA**, **VAR\_DISV**,
   
    **NUMBER\_OF\_VARIABLES** }
- enum **updateOptions** { **UPDATE\_TEXT**, **UPDATE\_ICON**, **UPDATE\_TEXT\_AND\_ICON**, **UPDATE\_STATE** }

### Public Member Functions

- **QEGenericButton** (QAbstractButton \*owner)
- void **setSubscribe** (bool subscribe)
- bool **getSubscribe** ()
- void **setUpdateOption** (updateOptions updateOptionIn)
- updateOptions **getUpdateOption** ()
- void **setTextAlignment** (Qt::Alignment alignment)
- Qt::Alignment **getTextAlignment** ()
- void **setPassword** (QString password)
- QString **getPassword** ()
- void **setConfirmAction** (bool confirmRequiredIn)
- bool **getConfirmAction** ()
- void **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** ()
- void **setDisabledRecordPolicy** (const QEWidgetProperties::DisabledRecordPolicy disabledRecordPolicy)
- QEWidgetProperties::DisabledRecordPolicy **getDisabledRecordPolicy** () const
- void **writeClickedNow** (const bool checked=false)

## Protected Member Functions

- void **useGenericNewVariableName** (const QString &variableName, const QString &variableNameSubstitutions, const unsigned int variableIndex)
- void **connectionChanged** (QCaConnectionInfo &connectionInfo, const unsigned int &variableIndex)
- void **setGenericButtonText** (const QString &text, QCaAlarmInfo &alarmInfo, QCaDateTime &, const unsigned int &variableIndex)
- void **setGenericDISAvalue** (const long &value, QCaAlarmInfo &alarmInfo, QCaDateTime &, const unsigned int &variableIndex)
- void **setGenericDISVvalue** (const long &value, 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

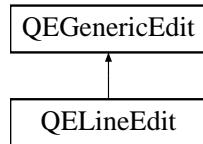
- applicationLauncher **programLauncher**
- QESingleVariableMethods \* **altReadback**

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

- `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 `setAllowFocusUpdate (bool allowFocusUpdate)`
- bool `getAllowFocusUpdate () const`  
*Returns 'true' if this widget configured to allow updates while it has focus.*
- 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**
- int **arrayIndex**
- bool **subscribe**
- bool **writeOnLoseFocus**
- bool **writeOnEnter**
- bool **writeOnFinish**
- bool **confirmWrite**
- bool **allowFocusUpdate**

*Allow updated while widget has focus - defaults to false.*

- 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::setAllowFocusUpdate (bool *allowFocusUpdate*)**

Sets if this widget configured to allow updates while it has focus. Default is 'false'.

**9.62.3.7 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).

**9.62.3.8 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 calll to this slot if the user level allows.

**9.62.3.9 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)

**9.62.3.10 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').

**9.62.3.11 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).

**9.62.3.12 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 int QEGenericEdit::arrayIndex [read, write]**

Index used to select a single item of data for processing. The default is 0.

**9.62.4.3 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).

**9.62.4.4 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.

**9.62.4.5 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.

**9.62.4.6 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.

#### **9.62.4.7 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.

#### **9.62.4.8 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.

#### **9.62.4.9 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)

#### **9.62.4.10 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'.

#### **9.62.4.11 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.

#### **9.62.4.12 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.

#### **9.62.4.13 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.

#### **9.62.4.14 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'.

#### **9.62.4.15 QString QEGenericEdit::variable [read, write]**

EPICS variable name (CA PV)

#### **9.62.4.16 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.

#### **9.62.4.17 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.

#### **9.62.4.18 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.

**9.62.4.19 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').

**9.62.4.20 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).

**9.62.4.21 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]**

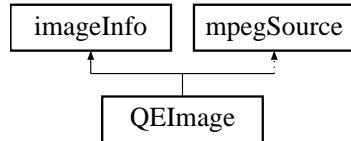
Display the widget. Default is true. Setting this property false is usefull if widget is only used to provide a signal - for example, when supplying data to a [QELink](#) widget. Note, when false the widget will still be visible in Qt Designer.

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

- /tmp/epicsqt/trunk/framework/widgets/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` { `BOUNding_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,`
  
    `BayerGB = imageDataFormats::BAYERGB, BayerBG = imageDataFor-`
  
    `mats::BAYERBG,`
  
    `BayerGR = imageDataFormats::BAYERGR, BayerRG = imageDataFor-`
  
    `mats::BAYERRG, rgb1 = imageDataFormats::RGB1, rgb2 = imageDataFor-`
  
    `mats::RGB2,`
  
    `rgb3 = imageDataFormats::RGB3, yuv444 = imageDataFormats::YUV444,`
  
    `yuv422 = imageDataFormats::YUV422, yuv421 = imageDataFor-`
  
    `mats::YUV421 }`
- enum `EllipseVariableDefinitions` { `BoundingRectangle = BOUNDING_-`
`RECTANGLE, CenterAndSize = CENTRE_AND_SIZE` }
- enum `TargetOptions` { `DottedFullCrosshair = VideoWidget::CROSSHAIR1,`
`SolidSmallCrosshair = 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)
- 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 [enableProfileSelection](#) property and [displayMarkups](#) property for details.*

- void **showProfile** (bool show)

*Show or hide the arbitrary line (profile) markup. Note that when hiding if its PV changes it will reshown unless [DisplayMarkups](#) has been set to off - 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 **showArea1** (bool show)

*Show or hide the area1 markup. Note that when hiding if its PV changes it will reshown unless [DisplayMarkups](#) has been set to off - 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 **setDisplayMarkupsOn** (bool on)

*Set markup display to on or off 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 `setXStretch` (double XStretchIn)  
*Access function for `XStretch` property - refer to `XStretch` property for details.*
- double `getXStretch ()`  
*Access function for `XStretch` property - refer to `XStretch` property for details.*
- void `setYStretch` (double YStretchIn)  
*Access function for `YStretch` property - refer to `YStretch` property for details.*
- double `getYStretch ()`  
*Access function for `YStretch` property - refer to `YStretch` 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.*
- 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.*
- 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 ()**  
*Access function for `hozSlice5Legend` property - refer to `hozSlice5Legend` property for details.*
- **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 details.*
- QString [getSubstitutedUrl](#) ()  
*Access function for URL property - refer to URL property 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.*
- 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,`  
`DATA_TYPE_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](#).*
- double **XStretch**  
*Stretch X factor. Used when generating canvas in which fully processed image is presented.*
- double **YStretch**  
*Stretch Y factor. Used when generating canvas in which fully processed image is presented.*
- int **initialHozScrollPos**
- int **initialVertScrollPos**
- bool **displayButtonBar**

## Properties

- QString `imageVariable`
- QString `formatVariable`
- QString `bitDepthVariable`
- QString `dataTypeVariable`
- 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**

*Name of arbitrary profile markup.*

- **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**  
*Name of target markup.*
- **QString beamLegend**  
*Name of beam markup.*
- **QString ellipseLegend**  
*Name of ellipse markup.*
- **bool displayVertSliceSelection**
- **bool displayVertSlice1Selection**
- **bool displayVertSlice2Selection**
- **bool displayVertSlice3Selection**
- **bool displayVertSlice4Selection**
- **bool displayVertSlice5Selection**
- **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 consistent with other widgets, but will not result in an updating image as the width and height variables are required as a minimum.

## 9.64.4 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::dataTypeVariable [read, write]**

EPICS variable name (CA PV). This variable is used to infer the bit depth of the image.

**9.64.6.16 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.

**9.64.6.17 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.

**9.64.6.18 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.

**9.64.6.19 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.

**9.64.6.20 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)

**9.64.6.21 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.

#### **9.64.6.22 bool QEImage::displayAlarmStateOptions [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.

#### **9.64.6.23 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.

#### **9.64.6.24 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.

#### **9.64.6.25 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.

#### **9.64.6.26 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.

#### **9.64.6.27 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.

#### **9.64.6.28 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.

**9.64.6.29 bool QEImage::displayEllipse [read, write]**

If true, the ellipse markup will be displayed on the image.

**9.64.6.30 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.

**9.64.6.31 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.

**9.64.6.32 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.

**9.64.6.33 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.

**9.64.6.34 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.

**9.64.6.35 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.

**9.64.6.36 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.

**9.64.6.37 bool QEImage::displayVertSlice1Selection [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.

**9.64.6.38 bool QEImage::displayVertSlice2Selection [read, write]**

If true, the selected vertical slice 2 will be displayed on the image. Note, this property is ignored unless the [enableVertSlice2Selection](#) property is true.

**9.64.6.39 bool QEImage::displayVertSlice3Selection [read, write]**

If true, the selected vertical slice 3 will be displayed on the image. Note, this property is ignored unless the [enableVertSlice3Selection](#) property is true.

**9.64.6.40 bool QEImage::displayVertSlice4Selection [read, write]**

If true, the selected vertical slice 4 will be displayed on the image. Note, this property is ignored unless the [enableVertSlice4Selection](#) property is true.

**9.64.6.41 bool QEImage::displayVertSlice5Selection [read, write]**

If true, the selected vertical slice 5 will be displayed on the image. Note, this property is ignored unless the [enableVertSlice5Selection](#) property is true.

**9.64.6.42 QColor QEImage::ellipseColor [read, write]**

Used to select the color of the ellipse marker.

**9.64.6.43 QString QEImage::ellipseHVariable [read, write]**

EPICS variable name (CA PV). This variable is used to read an ellipse height

**9.64.6.44 QString QEImage::ellipseWVariable [read, write]**

EPICS variable name (CA PV). This variable is used to read an ellipse width.

**9.64.6.45 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](#)).

**9.64.6.46 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](#)).

**9.64.6.47 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

**9.64.6.48 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

**9.64.6.49 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

**9.64.6.50 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

**9.64.6.51 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.

**9.64.6.52 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.

**9.64.6.53 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.

**9.64.6.54 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.

**9.64.6.55 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.

**9.64.6.56 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.

**9.64.6.57 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.

**9.64.6.58 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.

**9.64.6.59 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.

**9.64.6.60 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.

**9.64.6.61 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.

**9.64.6.62 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.

**9.64.6.63 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.

**9.64.6.64 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.

**9.64.6.65 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.)

**9.64.6.66 QString QEImage::formatVariable [read, write]**

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

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

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

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

If true, flip image horizontally.

**9.64.6.69 QColor QEImage::hozSlice1Color [read, write]**

Used to select the color of the horizontal slice 1 markup.

**9.64.6.70 QColor QEImage::hozSlice2Color [read, write]**

Used to select the color of the horizontal slice 2 markup.

**9.64.6.71 QColor QEImage::hozSlice3Color [read, write]**

Used to select the color of the horizontal slice 3 markup.

**9.64.6.72 QColor QEImage::hozSlice4Color [read, write]**

Used to select the color of the horizontal slice 4 markup.

**9.64.6.73 QColor QEImage::hozSlice5Color [read, write]**

Used to select the color of the horizontal slice 5 markup.

**9.64.6.74 QString QEImage::imageVariable [read, write]**

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

**9.64.6.75 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.

**9.64.6.76 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).

**9.64.6.77 QString QEImage::lineProfileArrayVariable [read, write]**

EPICS variable name (CA PV). This variable is used to write the areadetector arbitrary line profile array.

**9.64.6.78 QString QEImage::lineProfileThicknessVariable [read, write]**

EPICS variable name (CA PV). This variable is used to write the areadetector arbitrary line profile end Y.

**9.64.6.79 QString QEImage::lineProfileX1Variable [read, write]**

EPICS variable name (CA PV). This variable is used to write the areadetector arbitrary line profile start X.

**9.64.6.80 QString QEImage::lineProfileX2Variable [read, write]**

EPICS variable name (CA PV). This variable is used to write the areadetector arbitrary line profile end X.

**9.64.6.81 QString QEImage::lineProfileY1Variable [read, write]**

EPICS variable name (CA PV). This variable is used to write the areadetector arbitrary line profile start Y.

**9.64.6.82 QString QEImage::lineProfileY2Variable [read, write]**

EPICS variable name (CA PV). This variable is used to write the areadetector arbitrary line profile end Y.

**9.64.6.83 bool QEImage::logBrightness [read, write]**

If true, the image will be displayed using a logarithmic brightness scale.

**9.64.6.84 QColor QEImage::profileColor [read, write]**

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

**9.64.6.85 QString QEImage::profileHoz1ThicknessVariable [read, write]**

EPICS variable name (CA PV). This variable is used to write the areadetector first horizontal profile thickness.

**9.64.6.86 QString QEImage::profileHoz1Variable [read, write]**

EPICS variable name (CA PV). This variable is used to write the areadetector first horizontal profile.

**9.64.6.87 QString QEImage::profileHoz2ThicknessVariable [read, write]**

EPICS variable name (CA PV). This variable is used to write the areadetector second horizontal profile thickness.

**9.64.6.88 QString QEImage::profileHoz2Variable [read, write]**

EPICS variable name (CA PV). This variable is used to write the areadetector second horizontal profile.

**9.64.6.89 QString QEImage::profileHoz3ThicknessVariable [read, write]**

EPICS variable name (CA PV). This variable is used to write the areadetector third horizontal profile thickness.

**9.64.6.90 QString QEImage::profileHoz3Variable [read, write]**

EPICS variable name (CA PV). This variable is used to write the areadetector third horizontal profile.

**9.64.6.91 QString QEImage::profileHoz4ThicknessVariable [read, write]**

EPICS variable name (CA PV). This variable is used to write the areadetector fourth horizontal profile thickness.

**9.64.6.92 QString QEImage::profileHoz4Variable [read, write]**

EPICS variable name (CA PV). This variable is used to write the areadetector fourth horizontal profile.

**9.64.6.93 QString QEImage::profileHoz5ThicknessVariable [read, write]**

EPICS variable name (CA PV). This variable is used to write the areadetector fifth horizontal profile thickness.

**9.64.6.94 QString QEImage::profileHoz5Variable [read, write]**

EPICS variable name (CA PV). This variable is used to write the areadetector fifth horizontal profile.

**9.64.6.95 QString QEImage::profileHozArrayVariable [read, write]**

EPICS variable name (CA PV). This variable is used to write the areadetector horizontal profile array.

**9.64.6.96 QString QEImage::profileVert1ThicknessVariable [read, write]**

EPICS variable name (CA PV). This variable is used to write the areadetector first vertical profile.

**9.64.6.97 QString QEImage::profileVert1Variable [read, write]**

EPICS variable name (CA PV). This variable is used to write the areadetector first vertical profile.

**9.64.6.98 QString QEImage::profileVert2ThicknessVariable [read, write]**

EPICS variable name (CA PV). This variable is used to write the areadetector second vertical profile.

**9.64.6.99 QString QEImage::profileVert2Variable [read, write]**

EPICS variable name (CA PV). This variable is used to write the areadetector second vertical profile.

**9.64.6.100 QString QEImage::profileVert3ThicknessVariable [read, write]**

EPICS variable name (CA PV). This variable is used to write the areadetector third vertical profile.

**9.64.6.101 QString QEImage::profileVert3Variable [read, write]**

EPICS variable name (CA PV). This variable is used to write the areadetector third vertical profile.

**9.64.6.102 QString QEImage::profileVert4ThicknessVariable [read, write]**

EPICS variable name (CA PV). This variable is used to write the areadetector fourth vertical profile.

**9.64.6.103 QString QEImage::profileVert4Variable [read, write]**

EPICS variable name (CA PV). This variable is used to write the areadetector fourth vertical profile.

**9.64.6.104 QString QEImage::profileVert5ThicknessVariable [read, write]**

EPICS variable name (CA PV). This variable is used to write the areadetector fifth vertical profile.

**9.64.6.105 QString QEImage::profileVert5Variable [read, write]**

EPICS variable name (CA PV). This variable is used to write the areadetector fifth vertical profile.

**9.64.6.106 QString QEImage::profileVertArrayVariable [read, write]**

EPICS variable name (CA PV). This variable is used to write the areadetector vertical profile array.

**9.64.6.107 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

**9.64.6.108 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

**9.64.6.109 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.

**9.64.6.110 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.

**9.64.6.111 QString QEImage::regionOfInterest1HVariable [read, write]**

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

**9.64.6.112 QString QEImage::regionOfInterest1WVariable [read, write]**

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

**9.64.6.113 QString QEImage::regionOfInterest1XVariable [read, write]**

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

**9.64.6.114 QString QEImage::regionOfInterest1YVariable [read, write]**

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

**9.64.6.124 QString QEImage::regionOfInterest4WVariable [read, write]**

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

**9.64.6.125 QString QEImage::regionOfInterest4XVariable [read, write]**

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

**9.64.6.126 QString QEImage::regionOfInterest4YVariable [read, write]**

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

**9.64.6.127 ResizeOptions QEImage::resizeOption [read, write]**

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

**9.64.6.128 RotationOptions QEImage::rotation [read, write]**

Image rotation option.

**9.64.6.129 bool QEImage::showTime [read, write]**

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

**9.64.6.130 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.

**9.64.6.131 QColor QEImage::targetColor [read, write]**

Used to select the color of the target marker.

**9.64.6.132 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.

**9.64.6.133 QString QEImage::targetXVariable [read, write]**

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

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

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

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

Used to select the color of the timestamp.

**9.64.6.136 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)

**9.64.6.137 bool QEImage::useFalseColour [read, write]**

If true, the apply false colour to the image.

**9.64.6.138 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 programmaticaly through setUserLevel(). Widgets that are always accessible should be visible at 'User'. Widgets that are only accessable to scientists managing the facility should be visible at 'Scientist'. Widgets that are only accessable to engineers maintaining the facility should be visible at 'Engineer'.

**9.64.6.139 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.

**9.64.6.140 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.

**9.64.6.141 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.

**9.64.6.142 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'.

**9.64.6.143 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.

**9.64.6.144 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.

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

If true, flip image vertically.

**9.64.6.146 QColor QEImage::vertSlice1Color [read, write]**

Used to select the color of the vertical slice 1 markup.

**9.64.6.147 QColor QEImage::vertSlice2Color [read, write]**

Used to select the color of the vertical slice 2 markup.

**9.64.6.148 QColor QEImage::vertSlice3Color [read, write]**

Used to select the color of the vertical slice 3 markup.

**9.64.6.149 QColor QEImage::vertSlice4Color [read, write]**

Used to select the color of the vertical slice 4 markup.

**9.64.6.150 QColor QEImage::vertSlice5Color [read, write]**

Used to select the color of the vertical slice 5 markup.

**9.64.6.151 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.

**9.64.6.152 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` = `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`, `Underline` = `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` }
- 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

- **QELLabel** (QWidget \*parent=0)
- **QELLabel** (const QString &variableName, QWidget \*parent=0)
- **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.*
- **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.*

## Properties

- `QString variable`
- `QString variableSubstitutions`
- `int arrayIndex`
- `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 `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.

### 9.67.2.2 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 [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.

### 9.67.2.4 enum QELabel::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.

### 9.67.2.5 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 int QELabel::arrayIndex [read, write]**

Index used to select a single item of data for processing. The default is 0.

**9.67.5.5 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.

**9.67.5.6 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.

**9.67.5.7 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.

**9.67.5.8 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).

**9.67.5.9 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.

**9.67.5.10 bool QELabel::leadingZero [read, write]**

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

### 9.67.5.11 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"

### 9.67.5.12 Notations QELabel::notation [read, write]

Notation used for numerical formatting. Default is fixed.

### 9.67.5.13 QPixmap QELabel:: pixmap0 [read, write]

Pixmap displayed when updateOption property is 'Picture' and data is interpreted as 0.

### 9.67.5.14 QPixmap QELabel:: pixmap1 [read, write]

Pixmap displayed when updateOption property is 'Picture' and data is interpreted as 1.

**9.67.5.15 QPixmap QELabel::pixmap2 [read, write]**

Pixmap displayed when updateOption property is 'Picture' and data is interpreted as 2.

**9.67.5.16 QPixmap QELabel::pixmap3 [read, write]**

Pixmap displayed when updateOption property is 'Picture' and data is interpreted as 3.

**9.67.5.17 QPixmap QELabel::pixmap4 [read, write]**

Pixmap displayed when updateOption property is 'Picture' and data is interpreted as 4.

**9.67.5.18 QPixmap QELabel::pixmap5 [read, write]**

Pixmap displayed when updateOption property is 'Picture' and data is interpreted as 5.

**9.67.5.19 QPixmap QELabel::pixmap6 [read, write]**

Pixmap displayed when updateOption property is 'Picture' and data is interpreted as 6.

**9.67.5.20 QPixmap QELabel::pixmap7 [read, write]**

Pixmap displayed when updateOption property is 'Picture' and data is interpreted as 7.

**9.67.5.21 int QELabel::precision [read, write]**

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

**9.67.5.22 int QELabel::radix [read, write]**

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

**9.67.5.23 Separators QELabel::separator [read, write]**

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

**9.67.5.24 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.

**9.67.5.25 bool QELabel::trailingZeros [read, write]**

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

**9.67.5.26 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.

**9.67.5.27 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.

**9.67.5.28 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'.

**9.67.5.29 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.

**9.67.5.30 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.

#### 9.67.5.31 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.

#### 9.67.5.32 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'.

#### 9.67.5.33 QString QELabel::variable [read, write]

EPICS variable name (CA PV)

#### 9.67.5.34 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.

#### 9.67.5.35 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.

#### 9.67.5.36 bool QELabel::visible [read, write]

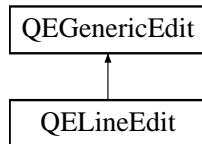
Display the widget. Default is true. Setting this property false is useful if widget is only used to provide a signal - for example, when supplying data to a [QELink](#) widget. Note, when false the widget will still be visible in Qt Designer.

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

- /tmp/epicsqt/trunk/framework/widgets/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 = QEStructFormatting::FORMAT\_DEFAULT,     Float-
   
    ing = QEStructFormatting::FORMAT\_FLOATING,     Integer =
   
    QEStructFormatting::FORMAT\_INTEGER,             UnsignedInteger =
   
    QEStructFormatting::FORMAT\_UNSIGNEDINTEGER,
   
    Time = QEStructFormatting::FORMAT\_TIME,        LocalEnumeration =
   
    QEStructFormatting::FORMAT\_LOCAL\_ENUMERATE }
- enum `Separators` { `NoSeparator` = QEStructFormatting::SEPARATOR\_-
   
    NONE, `Comma` = QEStructFormatting::SEPARATOR\_COMMA, `Under-
   
    score` = QEStructFormatting::SEPARATOR\_UNDERSCORE, `Space` =
   
    QEStructFormatting::SEPARATOR\_SPACE }
- enum `Notations` { `Fixed` = QEStructFormatting::NOTATION\_FIXED, `Sci-
   
    entific` = QEStructFormatting::NOTATION\_SCIENTIFIC, `Automatic` =
   
    QEStructFormatting::NOTATION\_AUTOMATIC }
- enum `ArrayActions` { `Append` = QEStructFormatting::APPEND, `Ascii` =
   
    QEStructFormatting::ASCII, `Index` = QEStructFormatting::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.*
- `QELLineEdit (QWidget *parent=0)`
- `QELLineEdit (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`

### 9.68.1 Member Enumeration Documentation

#### 9.68.1.1 enum QELineEdit::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.

#### 9.68.1.2 enum QELineEdit::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.

#### 9.68.1.3 enum QELineEdit::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.

#### 9.68.1.4 enum QELineEdit::Separators

User friendly enumerations for separator property - refer to `QQStringFormatting::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 QELineEdit::QELineEdit (QWidget \* *parent* = 0)

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

### 9.68.2.2 QELineEdit::QELineEdit (const QString & *variableName*, QWidget \* *parent* = 0)

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

## 9.68.3 Member Function Documentation

### 9.68.3.1 void QELineEdit::dbValueChanged (const QString & *out*) [signal]

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

## 9.68.4 Property Documentation

### 9.68.4.1 bool QELineEdit::addUnits [read, write]

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

### 9.68.4.2 ArrayActions QELineEdit::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 bool QELineEdit::leadingZero [read, write]

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

#### 9.68.4.5 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)  
>= 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"

**9.68.4.6 Notations QELineEdit::notation [read, write]**

Notation used for numerical formatting. Default is fixed.

**9.68.4.7 int QELineEdit::precision [read, write]**

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

**9.68.4.8 int QELineEdit::radix [read, write]**

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

**9.68.4.9 Separators QELineEdit::separator [read, write]**

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

**9.68.4.10 bool QELineEdit::trailingZeros [read, write]**

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

**9.68.4.11 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/QELineEdit/QELineEditManager.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**
- 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**

### 9.71.1 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 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/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 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 **userInfoSourceOptions** { **USER\_INFO\_SOURCE\_TEXT**, **USER\_-  
INFO\_SOURCE\_FILE** }
- 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 }
- enum **UserInfoSourceOptions** { **userInfoSourceText** = QEPeriodic::USER\_-  
INFO\_SOURCE\_TEXT, **userInfoSourceFile** = QEPeriodic::USER\_INFO\_-  
SOURCE\_FILE }

## 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** ()
- void **setUserInfoText** (QString userInfo)
- QString **getUserInfoText** ()
- void **setUserInfoFile** (QString userInfoFileIn)
- QString **getUserInfoFile** ()
- void **setUserInfoSourceOption** (userInfoSourceOptions userInfoSourceOptionIn)
- userInfoSourceOptions **getUserInfoSourceOption** ()
- void **updateUserInfoSource** ()
- 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 ()**

- **void setUserInfoSourceOptionProperty (UserInfoSourceOptions userInfoSourceOption)**

- **UserInfoSourceOptions getUserInfoSourceOptionProperty ()**

## 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

- `QEFloatingFormatting floatingFormatting`
- bool `localEnabled`
- bool `allowDrop`
- variableTypes `variableType1`
- variableTypes `variableType2`
- double `variableTolerance1`
- double `variableTolerance2`

## Properties

- `QString writeButtonVariable1`
- `QString writeButtonVariable2`
- `QString readbackLabelVariable1`
- `QString readbackLabelVariable2`
- `QString variableSubstitutions`
- bool `subscribe`
- bool `variableAsToolTip`
- bool `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`
- `UserInfoSourceOptions userInfoSourceOption`

### 9.74.1 Member Enumeration Documentation

#### 9.74.1.1 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.

#### 9.74.1.2 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.

### 9.74.2 Member Function Documentation

#### 9.74.2.1 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.

**9.74.2.2 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.

**9.74.3 Member Data Documentation****9.74.3.1 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.

**9.74.4 Property Documentation****9.74.4.1 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.

**9.74.4.2 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.

**9.74.4.3 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.

**9.74.4.4 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.

**9.74.4.5 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.

**9.74.4.6 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)

**9.74.4.7 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'.

**9.74.4.8 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.

**9.74.4.9 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.

**9.74.4.10 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.

**9.74.4.11 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'.

**9.74.4.12 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.

**9.74.4.13 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.

**9.74.4.14 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.

**9.74.4.15 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.

**9.74.4.16 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

## 9.75 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

## 9.76 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

## 9.77 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

## 9.78 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**

### 9.78.1 Member Enumeration Documentation

#### 9.78.1.1 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.

#### 9.78.1.2 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.

### 9.78.2 Member Function Documentation

#### 9.78.2.1 void QEPlot::dbValueChanged (const QVector< double > & out) [signal]

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

#### 9.78.2.2 void QEPlot::dbValueChanged (const double & out) [signal]

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

#### 9.78.2.3 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.

### 9.78.3 Member Data Documentation

#### 9.78.3.1 bool QEPlot::allowDrop [read, write, protected]

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

### 9.78.4 Property Documentation

#### 9.78.4.1 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.

#### 9.78.4.2 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.

#### 9.78.4.3 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.

#### 9.78.4.4 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.

#### 9.78.4.5 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.

**9.78.4.6 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'.

**9.78.4.7 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.

**9.78.4.8 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.

**9.78.4.9 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.

**9.78.4.10 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'.

**9.78.4.11 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](#).

**9.78.4.12 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](#).

**9.78.4.13 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](#).

**9.78.4.14 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](#).

**9.78.4.15 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.

**9.78.4.16 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.

**9.78.4.17 bool QEPlot::visible [read, write]**

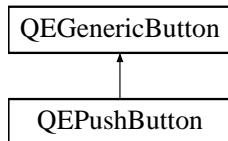
Display the widget. Default is true. Setting this property false is usefull if widget is only used to provide a signal - for example, when supplying data to a [QELink](#) widget. Note, when false the widget will still be visible in Qt Designer.

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

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

## 9.79 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` = QEStringFormatting::FORMAT\_DEFAULT, `Floating` = QEStringFormatting::FORMAT\_FLOATING, `Integer` = QEStringFormatting::FORMAT\_INTEGER, `UnsignedInteger` = QEStringFormatting::FORMAT\_UNSIGNEDINTEGER,
 `Time` = QEStringFormatting::FORMAT\_TIME, `LocalEnumeration` = QEStringFormatting::FORMAT\_LOCAL\_ENUMERATE
 }
  - enum `Notations` { `Fixed` = QEStringFormatting::NOTATION\_FIXED, `Scientific` = QEStringFormatting::NOTATION\_SCIENTIFIC, `Automatic` = QEStringFormatting::NOTATION\_AUTOMATIC }
  - enum `ArrayActions` { `Append` = QEStringFormatting::APPEND, `Ascii` = QEStringFormatting::ASCII, `Index` = QEStringFormatting::INDEX }
  - enum `UpdateOptions` { `Text` = 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\_STDOUPUT }
  - 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)
- **~QEPushButton** ()
 

*Destructor.*
- void **writeNow** ()

- void **setVariableNameSubstitutionsProperty** (const QString &substitutions)
- void **setAltReadbackProperty** (const QString &variableName)
- QString **getAltReadbackProperty** () const
- void **setAltReadbackArrayIndex** (const int arrayIndex)
- int **getAltReadbackArrayIndex** () const
- **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 variableSubstitutions`
- `int arrayIndex`
- `QString altReadbackVariable`
- `int altReadbackArrayIndex`
- `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`
- `QWidgetProperties::DisabledRecordPolicy disabledRecordPolicy`
- `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`

## 9.79.1 Member Enumeration Documentation

### 9.79.1.1 enum QEPushButton::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.

### 9.79.1.2 enum QEPushButton::CreationOptionNames

Creation options. Used to indicate how to present a GUI when requesting a new GUI be created. Open a new window, open a new tab, or replace the current window.

#### Enumerator:

*Open* Replace the current GUI with the new GUI.

*NewTab* Open new GUI in a new tab.

*NewWindow* Open new GUI in a new window.

*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.

#### 9.79.1.3 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.

#### 9.79.1.4 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.

#### 9.79.1.5 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.

### 9.79.1.6 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.

### 9.79.1.7 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).

### 9.79.1.8 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.

## 9.79.2 Constructor & Destructor Documentation

### 9.79.2.1 QEPushButton::QEPushButton (QWidget \* *parent* = 0)

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

**9.79.2.2 QEPushButton::QEPushButton (const QString & *variableName*, QWidget \* *parent* = 0)**

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

**9.79.3 Member Function Documentation****9.79.3.1 void QEPushButton::clicked (int *value*) [signal]**

Button has been Clicked. The value emitted is the integer interpretation of the clickText property (or the clickCheckedText property if the button was checked)

**9.79.3.2 void QEPushButton::dbValueChanged (const QString & *out*) [signal]**

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

**9.79.3.3 void QEPushButton::pressed (int *value*) [signal]**

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

**9.79.3.4 void QEPushButton::released (int *value*) [signal]**

Button has been Released The value emitted is the integer interpretation of the releaseText property

**9.79.3.5 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.

**9.79.3.6 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.

## 9.79.4 Property Documentation

**9.79.4.1 bool QEPushButton::addUnits [read, write]**

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

**9.79.4.2 Qt::Alignment QEPushButton::alignment [read, write]**

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

**9.79.4.3 bool QEPushButton::allowDrop [read, write]**

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

**9.79.4.4 QString QEPushButton::altReadbackVariable [read, write]**

EPICS variable name (CA PV). This variable is used to provide a readback value when different to the variable written to by a button press.

**9.79.4.5 QStringList QEPushButton::arguments [read, write]**

Arguments for program specified in the 'program' property.

**9.79.4.6 ArrayActions QEPushButton::arrayAction [read, write]**

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

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

**9.79.4.7 int QEPushButton::arrayIndex [read, write]**

Index used to select a single item of data for processing. The default is 0.

**9.79.4.8 QString QEPushButton::clickCheckedText [read, write]**

Text used to compare with text written or read to determine if push button should be marked as checked. Note, must be an exact match following formatting of data updates. When writing values, the 'pressText', 'ReleaseText', or 'clickedtext' must match this property to cause the button to be checked when the write occurs.

Good example: formatting set to display a data value of '1' as 'On', clickCheckedText is 'On', clickText is 'On'. In this example, the push button will be checked when a data update occurs with a value of 1 or when the button is clicked.

Bad example: formatting set to display a data value of '1' as 'On', clickCheckedText is 'On', clickText is '1'. In this example, the push button will be checked when a data update occurs with a value of 1 but, although a valid value will be written when clicked, the button will not be checked when clicked as '1' is not the same as 'On'.

Reimplemented from [QEGenericButton](#).

**9.79.4.9 QString QEPushButton::clickText [read, write]**

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

Reimplemented from [QEGenericButton](#).

**9.79.4.10 bool QEPushButton::confirmAction [read, write]**

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

**9.79.4.11 QString QEPushButton::confirmText [read, write]**

Text used to confirm action if confirmation dialog is presented

Reimplemented from [QEGenericButton](#).

**9.79.4.12 CreationOptionNames QEPushButton::creationOption [read, write]**

Creation options when opening a new GUI. Open a new window, open a new tab, or replace the current window. The creation option is supplied when the button generates a newGui signal. Application code connected to this signal should honour this request if possible. When used within the QEGui application, the QEGui application creates a new window, new tab, or replaces the current window as appropriate.

Reimplemented from [QEGenericButton](#).

**9.79.4.13 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](#).

**9.79.4.14 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.

**9.79.4.15 QEWidgetProperties::DisabledRecordPolicy  
QEPushButton::disabledRecordPolicy [read, write]**

Set the widget's disabled record policy, i.e. the action to be taken when the under lying record is disabled, i.e. when the assiociated record's DISA and DISV field values are equal. Note: this is only applicable IOC process variables. When the policy is ignore, then no special action is taken. This is the default policy. When the policy is grayout, the widget is style is set as if disconnected when the record is disabled.

Reimplemented from [QEGenericButton](#).

**9.79.4.16 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.

**9.79.4.17 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.

**9.79.4.18 QString QEPushButton::format [read, write]**

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

**9.79.4.19 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::openQEFile\(\)](#) in [QEWidget.cpp](#) for details.

**9.79.4.20 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!)

**9.79.4.21 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](#).

**9.79.4.22 bool QEPushButton::leadingZero [read, write]**

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

**9.79.4.23 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"

#### 9.79.4.24 Notations QEPushButton::notation [read, write]

Notation used for numerical formatting. Default is fixed.

#### 9.79.4.25 QString QEPushButton::password [read, write]

Password user will need to enter before any action is taken

Reimplemented from [QEGenericButton](#).

#### 9.79.4.26 QPixmap QEPushButton::pixmap0 [read, write]

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

#### 9.79.4.27 QPixmap QEPushButton::pixmap1 [read, write]

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

#### 9.79.4.28 QPixmap QEPushButton::pixmap2 [read, write]

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

**9.79.4.29 QPixmap QEPushButton:: pixmap3 [read, write]**

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

**9.79.4.30 QPixmap QEPushButton:: pixmap4 [read, write]**

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

**9.79.4.31 QPixmap QEPushButton:: pixmap5 [read, write]**

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

**9.79.4.32 QPixmap QEPushButton:: pixmap6 [read, write]**

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

**9.79.4.33 QPixmap QEPushButton:: pixmap7 [read, write]**

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

**9.79.4.34 int QEPushButton:: precision [read, write]**

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

**9.79.4.35 QString QEPushButton:: pressText [read, write]**

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

Reimplemented from [QEGenericButton](#).

**9.79.4.36 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 substitions applied to the new form, they are appended to the list and the existing macro substitutions take precedence.

Reimplemented from [QEGenericButton](#).

#### **9.79.4.37 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

#### **9.79.4.38 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.

#### **9.79.4.39 QString QEPushButton::releaseText [read, write]**

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

Reimplemented from [QEGenericButton](#).

#### **9.79.4.40 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.

#### **9.79.4.41 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)

#### **9.79.4.42 bool QEPushButton::trailingZeros [read, write]**

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

#### **9.79.4.43 UpdateOptions QEPushButton::updateOption [read, write]**

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

Reimplemented from [QEGenericButton](#).

#### **9.79.4.44 bool QEPushButton::useDbPrecision [read, write]**

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

**9.79.4.45 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'.

**9.79.4.46 QString QEPushButton::userLevelEngineerStyle [read, write]**

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

**9.79.4.47 QString QEPushButton::userLevelScientistStyle [read, write]**

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

**9.79.4.48 QString QEPushButton::userLevelUserStyle [read, write]**

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

**9.79.4.49 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'.

**9.79.4.50 QString QEPushButton::variable [read, write]**

EPICS variable name (CA PV)

**9.79.4.51 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.

**9.79.4.52 QString QEPushButton::variableSubstitutions [read, write]**

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

**9.79.4.53 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.

**9.79.4.54 bool QEPushButton::writeOnClick [read, write]**

If true, the 'clickText' property is written when the button is clicked. Default is true  
Reimplemented from [QEGenericButton](#).

**9.79.4.55 bool QEPushButton::writeOnPress [read, write]**

If true, the 'pressText' property is written when the button is pressed. Default is false  
Reimplemented from [QEGenericButton](#).

**9.79.4.56 bool QEPushButton::writeOnRelease [read, write]**

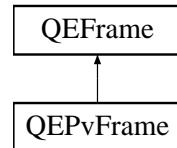
If true, the 'releaseText' property is written when the button is released. Default is false  
Reimplemented from [QEGenericButton](#).

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

## 9.80 QEPvFrame Class Reference

Inheritance diagram for QEPvFrame::



### 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

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

### Protected Member Functions

- qcaobject::QCaObject \* **createQcaItem** (unsigned int variableIndex)
- void **establishConnection** (unsigned int variableIndex)
- QString **copyVariable** ()
- QVariant **copyData** ()

### Properties

- QString **variable**
- QString **variableSubstitutions**
- int **arrayIndex**

#### 9.80.1 Member Function Documentation

##### 9.80.1.1 void QEPvFrame::dbConnectionChanged (const bool & isConnected) [signal]

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

**9.80.1.2 void QEPvFrame::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.

## 9.80.2 Property Documentation

**9.80.2.1 int QEPvFrame::arrayIndex [read, write]**

Index used to select a single item of data for processing. The default is 0.

Array Index element to use if variable is an array (waveform). Defaults to 0.

**9.80.2.2 QString QEPvFrame::variable [read, write]**

EPICS variable name (CA PV)

**9.80.2.3 QString QEPvFrame::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/QEFrame/QEPvFrame.h
- /tmp/epicsqt/trunk/framework/widgets/QEFrame/QEPvFrame.cpp

## 9.81 QEPvFrameManager Class Reference

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

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

## 9.82 QEPVNameLists Class Reference

### Public Member Functions

- void **prependOrMoveToFirst** (const QString &item)
- void **saveConfiguration** (PMElement &parentElement)
- void **restoreConfiguration** (PMElement &parentElement)

### Static Public Member Functions

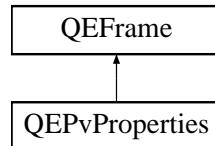
- static void **constructor** ()

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 \* `createQcaItem` (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

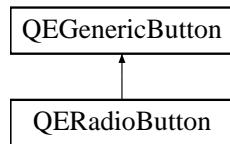
- **QEPvPropertiesManager** (QObject \*parent=0)
- bool **isContainer** () const
- bool **isInitialized** () const
- QIcon **icon** () const
- QString **group** () const
- QString **includeFile** () const
- QString **name** () const
- QString **toolTip** () const
- QString **whatsThis** () const
- QWidget \* **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, `Under_score` = 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\_STDOUPUT }

- 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 ()`
- `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
- int `arrayIndex`
- 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](#)
- `QWidgetProperties::DisabledRecordPolicy disabledRecordPolicy`
- `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 int QERadioButton::arrayIndex [read, write]**

Index used to select a single item of data for processing. The default is 0.

**9.85.4.7 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](#).

**9.85.4.8 QString QERadioButton::clickText [read, write]**

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

Reimplemented from [QEGenericButton](#).

**9.85.4.9 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

**9.85.4.10 QString QERadioButton::confirmText [read, write]**

Text used to confirm action if confirmation dialog is presented

Reimplemented from [QEGenericButton](#).

**9.85.4.11 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](#).

**9.85.4.12 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. Customisations are not applied if the GUI is opened as a dock.

Reimplemented from [QEGenericButton](#).

**9.85.4.13 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.

**9.85.4.14 QEWidgetProperties::DisabledRecordPolicy  
QERadioButton::disabledRecordPolicy [read, write]**

Set the widget's disabled record policy, i.e. the action to be taken when the underlying record is disabled, i.e. when the associated record's DISA and DISV field values are equal. Note: this is only applicable IOC process variables. When the policy is ignore, then no special action is taken. This is the default policy. When the policy is grayout, the widget is styled as if disconnected when the record is disabled.

Reimplemented from [QEGenericButton](#).

**9.85.4.15 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.

**9.85.4.16 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.

**9.85.4.17 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).

**9.85.4.18 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 [QWidget::openQEFfile\(\)](#) in [QWidget.cpp](#) for details.

**9.85.4.19 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.

**9.85.4.20 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](#).

#### **9.85.4.21 bool QERadioButton::leadingZero [read, write]**

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

#### **9.85.4.22 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"

#### **9.85.4.23 Notations QERadioButton::notation [read, write]**

Notation used for numerical formatting. Default is fixed.

**9.85.4.24 QString QERadioButton::password [read, write]**

Password user will need to enter before any action is taken

Reimplemented from [QEGenericButton](#).

**9.85.4.25 QPixmap QERadioButton:: pixmap0 [read, write]**

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

**9.85.4.26 QPixmap QERadioButton:: pixmap1 [read, write]**

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

**9.85.4.27 QPixmap QERadioButton:: pixmap2 [read, write]**

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

**9.85.4.28 QPixmap QERadioButton:: pixmap3 [read, write]**

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

**9.85.4.29 QPixmap QERadioButton:: pixmap4 [read, write]**

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

**9.85.4.30 QPixmap QERadioButton:: pixmap5 [read, write]**

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

**9.85.4.31 QPixmap QERadioButton:: pixmap6 [read, write]**

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

**9.85.4.32 QPixmap QERadioButton:: pixmap7 [read, write]**

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

**9.85.4.33 int QERadioButton::precision [read, write]**

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

**9.85.4.34 QString QERadioButton::pressText [read, write]**

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

Reimplemented from [QEGenericButton](#).

**9.85.4.35 QString QERadioButton::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](#).

**9.85.4.36 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

**9.85.4.37 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.

**9.85.4.38 int QERadioButton::radix [read, write]**

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

**9.85.4.39 QString QERadioButton::releaseText [read, write]**

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

Reimplemented from [QEGenericButton](#).

**9.85.4.40 Separators QERadioButton::separator [read, write]**

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

**9.85.4.41 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.

**9.85.4.42 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)

**9.85.4.43 bool QERadioButton::trailingZeros [read, write]**

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

**9.85.4.44 UpdateOptions QERadioButton::updateOption [read, write]**

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

Reimplemented from [QEGenericButton](#).

**9.85.4.45 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.

**9.85.4.46 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'.

**9.85.4.47 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.

#### **9.85.4.48 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.

#### **9.85.4.49 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.

#### **9.85.4.50 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'.

#### **9.85.4.51 QString QERadioButton::variable [read, write]**

EPICS variable name (CA PV)

#### **9.85.4.52 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.

#### **9.85.4.53 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 exam-

ple, '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.

#### **9.85.4.54 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.

#### **9.85.4.55 bool QERadioButton::writeOnClick [read, write]**

If true, the 'clickText' property is written when the button is clicked. Default is true  
Reimplemented from [QEGenericButton](#).

#### **9.85.4.56 bool QERadioButton::writeOnPress [read, write]**

If true, the 'pressText' property is written when the button is pressed. Default is false  
Reimplemented from [QEGenericButton](#).

#### **9.85.4.57 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 QEReipe 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 **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

## 9.88 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`

## 9.89 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 **showColumnNumber**  
*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](#)

## 9.89.1 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.

## 9.89.2 Member Enumeration Documentation

### 9.89.2.1 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.

### 9.89.2.2 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.

## 9.89.3 Member Function Documentation

### 9.89.3.1 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.

## 9.89.4 Property Documentation

### 9.89.4.1 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.

### 9.89.4.2 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.

**9.89.4.3 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.

**9.89.4.4 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.

**9.89.4.5 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.

**9.89.4.6 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.

**9.89.4.7 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 programmaticaly 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'.

**9.89.4.8 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.

#### **9.89.4.9 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.

#### **9.89.4.10 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.

#### **9.89.4.11 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'.

#### **9.89.4.12 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.

#### **9.89.4.13 bool QEScript::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/QEScript/QEScript.h

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

## 9.90 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](#)

### 9.90.1 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.

### 9.90.2 Member Enumeration Documentation

#### 9.90.2.1 enum QEShape::animationOptions

Options for how a variable will animate the shape.

#### 9.90.2.2 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.

#### 9.90.2.3 enum QEShape::shapeOptions

Options for the type of shape.

#### 9.90.2.4 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.

### 9.90.3 Constructor & Destructor Documentation

#### 9.90.3.1 QEShape::QEShape (QWidget \* *parent* = 0)

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

#### 9.90.3.2 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.

### 9.90.4 Member Function Documentation

#### 9.90.4.1 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.

#### 9.90.4.2 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.

#### 9.90.4.3 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.

**9.90.4.4 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.

**9.90.4.5 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.

**9.90.4.6 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.

**9.90.4.7 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.

## 9.90.5 Property Documentation

**9.90.5.1 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.

**9.90.5.2 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.

**9.90.5.3 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.

**9.90.5.4 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.

**9.90.5.5 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.

**9.90.5.6 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.

**9.90.5.7 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.

**9.90.5.8 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'.

**9.90.5.9 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'.

**9.90.5.10 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'.

**9.90.5.11 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'.

**9.90.5.12 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'.

**9.90.5.13 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'.

**9.90.5.14 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'.

**9.90.5.15 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'.

**9.90.5.16 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'.

**9.90.5.17 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.

**9.90.5.18 bool QEShape::displayAlarmState [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.

**9.90.5.19 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.

**9.90.5.20 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.

#### **9.90.5.21 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, RoundedRect, Ellipse, Arc, Chord, Pie, Path

#### **9.90.5.22 double QEShape::offset1 [read, write]**

Offset applied to data from the 1st variable before it is used to animate the shape

#### **9.90.5.23 double QEShape::offset2 [read, write]**

Offset applied to data from the 2nd variable before it is used to animate the shape

#### **9.90.5.24 double QEShape::offset3 [read, write]**

Offset applied to data from the 3rd variable before it is used to animate the shape

#### **9.90.5.25 double QEShape::offset4 [read, write]**

Offset applied to data from the 4th variable before it is used to animate the shape

#### **9.90.5.26 double QEShape::offset5 [read, write]**

Offset applied to data from the 5th variable before it is used to animate the shape

#### **9.90.5.27 double QEShape::offset6 [read, write]**

Offset applied to data from the 6th variable before it is used to animate the shape

**9.90.5.28 QPoint QEShape::point1 [read, write]**

1st coordinate used when drawing the shape. Used for the following shapes: Line, Points, Polyline, Polygon, Rect, RoundedRectangle, Ellipse, Arc, Chord, Pie, Path, Text,Pixmap

**9.90.5.29 QPoint QEShape::point10 [read, write]**

10th coordinate used when drawing the shape. Used for the following shapes: Points, Polyline, Polygon, Path

**9.90.5.30 QPoint QEShape::point2 [read, write]**

2nd coordinate used when drawing the shape. Used for the following shapes: Line, Points, Polyline, Polygon, Rect, RoundedRectangle, Ellipse, Arc, Chord, Pie, Path,Pixmap

**9.90.5.31 QPoint QEShape::point3 [read, write]**

3rd coordinate used when drawing the shape. Used for the following shapes: Points, Polyline, Polygon, Path

**9.90.5.32 QPoint QEShape::point4 [read, write]**

4th coordinate used when drawing the shape. Used for the following shapes: Points, Polyline, Polygon, Path

**9.90.5.33 QPoint QEShape::point5 [read, write]**

5th coordinate used when drawing the shape. Used for the following shapes: Points, Polyline, Polygon, Path

**9.90.5.34 QPoint QEShape::point6 [read, write]**

6th coordinate used when drawing the shape. Used for the following shapes: Points, Polyline, Polygon, Path

**9.90.5.35 QPoint QEShape::point7 [read, write]**

7th coordinate used when drawing the shape. Used for the following shapes: Points, Polyline, Polygon, Path

**9.90.5.36 QPoint QEShape::point8 [read, write]**

8th coordinate used when drawing the shape. Used for the following shapes: Points, Polyline, Polygon, Path

**9.90.5.37 QPoint QEShape::point9 [read, write]**

9th coordinate used when drawing the shape. Used for the following shapes: Points, Polyline, Polygon, Path

**9.90.5.38 double QEShape::scale2 [read, write]**

Scale factor applied to data from the 2nd variable before it is used to animate the shape

**9.90.5.39 double QEShape::scale3 [read, write]**

Scale factor applied to data from the 3rd variable before it is used to animate the shape

**9.90.5.40 double QEShape::scale4 [read, write]**

Scale factor applied to data from the 4th variable before it is used to animate the shape

**9.90.5.41 double QEShape::scale5 [read, write]**

Scale factor applied to data from the 5th variable before it is used to animate the shape

**9.90.5.42 double QEShape::scale6 [read, write]**

Scale factor applied to data from the 6th variable before it is used to animate the shape

**9.90.5.43 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.

**9.90.5.44 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'.

#### 9.90.5.45 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.

#### 9.90.5.46 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.

#### 9.90.5.47 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.

#### 9.90.5.48 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'.

**9.90.5.49 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.

**9.90.5.50 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.

**9.90.5.51 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.

**9.90.5.52 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.

**9.90.5.53 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.

**9.90.5.54 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.

**9.90.5.55 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.

**9.90.5.56 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 ex-

ample, 'SAMPLE=SAM1, NAME = "Ref foil"' These substitutions are applied to all the variable names.

#### 9.90.5.57 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

## 9.91 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 qlonglong &out)

### Public Member Functions

- `QESlider` (QWidget \*parent=0)
- `QESlider` (const QString &variableName, QWidget \*parent=0)
- void `setWriteOnChange` (bool `writeOnChange`)
- bool `getWriteOnChange` () const
- void `setSubscribe` (bool subscribe)
- bool `getSubscribe` () const
- void `setScale` (double scaleIn)
- double `getScale` () const
- void `setOffset` (double offsetIn)
- double `getOffset` () const
- void `setAllowFocusUpdate` (bool allowFocusUpdate)
- bool `getAllowFocusUpdate` () const
- void `writeNow` ()
- `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**
- **int arrayIndex**
- **bool subscribe**
- **bool allowFocusUpdate**
- **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`

### 9.91.1 Member Enumeration Documentation

#### 9.91.1.1 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.

#### 9.91.1.2 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.

### 9.91.2 Member Function Documentation

#### 9.91.2.1 void QESlider::dbValueChanged (const qlonglong & out) [signal]

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

**9.91.2.2 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.

### 9.91.3 Member Data Documentation

**9.91.3.1 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).

### 9.91.4 Property Documentation

**9.91.4.1 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.

**9.91.4.2 bool QESlider::allowFocusUpdate [read, write]**

Allow updated while widget has focus - defaults to false

**9.91.4.3 int QESlider::arrayIndex [read, write]**

Index used to select a single item of data for processing. The default is 0.

**9.91.4.4 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.

**9.91.4.5 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.

**9.91.4.6 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.

**9.91.4.7 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.

**9.91.4.8 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.

**9.91.4.9 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)

**9.91.4.10 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'.

**9.91.4.11 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.

**9.91.4.12 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.

**9.91.4.13 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.

**9.91.4.14 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'.

**9.91.4.15 QString QESlider::variable [read, write]**

EPICS variable name (CA PV)

**9.91.4.16 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.

**9.91.4.17 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.

**9.91.4.18 bool QESlider::visible [read, write]**

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

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

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

## 9.92 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 writeOnChangeIn)
- bool `getWriteOnChange` () const
- void `setSubscribe` (bool subscribe)
- bool `getSubscribe` () const
- void `setAddUnitsAsSuffix` (bool addUnitsAsSuffixIn)
- bool `getAddUnitsAsSuffix` () const
- void `setUseDbPrecisionForDecimals` (bool useDbPrecisionForDecimalIn)
- bool `getUseDbPrecisionForDecimals` () const
- void `setAllowFocusUpdate` (bool allowFocusUpdate)
- bool `getAllowFocusUpdate` () const
- void `writeNow` ()
- `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**
- int **arrayIndex**

- 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 `allowFocusUpdate`
- bool `useDbPrecision`
- bool `addUnits`
- double `value`

### 9.92.1 Member Enumeration Documentation

#### 9.92.1.1 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.

#### 9.92.1.2 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.

## 9.92.2 Member Function Documentation

### 9.92.2.1 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.

### 9.92.2.2 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.

## 9.92.3 Property Documentation

### 9.92.3.1 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.

### 9.92.3.2 bool QESpinBox::allowFocusUpdate [read, write]

Allow updated while widget has focus - defaults to false

### 9.92.3.3 int QESpinBox::arrayIndex [read, write]

Index used to select a single item of data for processing. The default is 0.

### 9.92.3.4 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.

### 9.92.3.5 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.

**9.92.3.6 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.

**9.92.3.7 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.

**9.92.3.8 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.

**9.92.3.9 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)

**9.92.3.10 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'.

**9.92.3.11 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.

**9.92.3.12 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.

**9.92.3.13 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.

**9.92.3.14 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'.

**9.92.3.15 QString QESpinBox::variable [read, write]**

EPICS variable name (CA PV)

**9.92.3.16 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.

**9.92.3.17 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.

**9.92.3.18 bool QESpinBox::visible [read, write]**

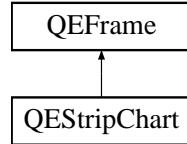
Display the widget. Default is true. Setting this property false is usefull if widget is only used to provide a signal - for example, when supplying data to a [QELink](#) widget. Note, when false the widget will still be visible in Qt Designer.

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

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

## 9.93 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 endTimeIn)
- 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
- void **setDefaultDir** (const QString &defaultDir)
- QString **getDefaultDir** () 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 \* **createQcaItem** (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 **defaultDir**
- 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](#)

### 9.93.1 Property Documentation

#### 9.93.1.1 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

## 9.94 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

## 9.95 QEStripChartContextMenu Class Reference

### Signals

- void **contextMenuSelected** (const QEStripChartNames::ContextMenuOptions)

### Public Member Functions

- **QEStripChartContextMenu** (bool inUse, QWidget \*parent=0)
- void **setPredefinedNames** (const QStringList &pvList)
- void **setUseReceiveTime** (const bool useReceiveTime)
- void **setArchiveReadHow** (const QEArcInterface::How how)
- void **setLineDrawMode** (const QEStripChartNames::LineDrawModes mode)

#### 9.95.1 Constructor & Destructor Documentation

##### 9.95.1.1 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

## 9.96 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

## 9.97 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** () const
- bool **isCalculation** () const
- void **setPvName** (QString pvName, QString substitutions)
- QString **getPvName** () const
- bool **isScaled** () const
- bool **getUseReceiveTime** () const
- QEArchiveInterface::How **getArchiveReadHow** () const
- QEStripChartNames::LineDrawModes **getLineDrawMode** () const
- 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

- QCVariableNamePropertyManager **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

## 9.98 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 Member Functions

- static QString **chartTimeModeStatus** (const ChartTimeModes mode)
- static QString **chartYRangeStatus** (const ChartYRanges yRange)

## 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 files:

- /tmp/epicsqt/trunk/framework/widgets/QEStripChart/QEStripChartNames.h
- /tmp/epicsqt/trunk/framework/widgets/QEStripChart/QEStripChartNames.cpp

## 9.99 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

## 9.100 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

## 9.101 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

## 9.102 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

## 9.103 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

## 9.104 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

## 9.105 QEStripChartToolBar Class Reference

This class holds all the StripChart tool bar widgets.

```
#include <QEStripChartToolBar.h>
```

### Classes

- class [OwnTabWidget](#)

### 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** ()
- void **loadSelected** ()
- void **saveAsSelected** ()
- void **loadSelectedFile** (const QString &filename)

### Public Member Functions

- **QEStripChartToolBar** (QWidget \*parent=0)
- void **setYRangeStatus** (const QEStripChartNames::ChartYRanges yRange)
- void  **setTimeStatus** (const QString &timeStatus)
- void **setDurationStatus** (const QString &durationStatus)
- void **setNOARStatus** (const int noar)
- void  **setTimeModeStatus** (const QEStripChartNames::ChartTimeModes timeMode)
- void  **setStateSelectionEnabled** (const QEStripChartNames::StateModes mode, const bool enabled)
- void  **setTimeRefs** (const QDateTime &t1, const QDateTime &t2)
- void  **setValue1Refs** (const double v1, const double v2)
- void  **setValue2Refs** (const double v1, const double v2)

### Static Public Member Functions

- static int **designHeight** ()

## Friends

- class **OwnTabWidget**

### 9.105.1 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

## 9.106 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**

#### 9.106.1 Member Data Documentation

##### 9.106.1.1 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

#### 9.106.2 Property Documentation

##### 9.106.2.1 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

## 9.107 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/QEImage/recording.h
- /tmp/epicsqt/trunk/framework/widgets/QEImage/recording.cpp

## 9.108 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/QEImage/brightnessContrast.h

## 9.109 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/QEImage/screenSelectDialog.h
- /tmp/epicsqt/trunk/framework/widgets/QEImage/screenSelectDialog.cpp

## 9.110 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/QEImage/selectMenu.h
- /tmp/epicsqt/trunk/framework/widgets/QEImage/selectMenu.cpp

## 9.111 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**
- bool **hasCurrentPoint**

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

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

## 9.112 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

## 9.113 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`

## 9.114 ValueScaling Class Reference

### Public Member Functions

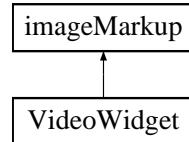
- void **reset** ()
- void **assign** (const ValueScaling &s)
- void **set** (const double dIn, const double mIn, const double cIn)
- void **get** (double &dOut, double &mOut, double &cOut) const
- void **map** (const double fromLower, const double fromUpper, const double toLower, const double toUpper)
- 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

## 9.115 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** (markupIds 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/QEImage/videowidget.h
- /tmp/epicsqt/trunk/framework/widgets/QEImage/videowidget.cpp

## 9.116 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/QEImage/zoomMenu.h
- /tmp/epicsqt/trunk/framework/widgets/QEImage/zoomMenu.cpp

# Index

\_CopyPaste, 29  
\_Field, 30  
\_Item, 31  
\_QDialogItem, 32  
\_QPushButtonGroup, 33  
\_QTableWidgetFileBrowser, 34  
\_QTableWidgetLog, 35  
\_QTableWidgetScript, 36

addUnits  
QEAnalogProgressBar, 103  
QECheckBox, 123  
QELabel, 246  
QELineEdit, 256  
QEPushButton, 297  
QERadioButton, 321

alarmSeverityDisplayMode  
QEAnalogProgressBar, 103

alignment  
QECheckBox, 123  
QEPushButton, 297  
QERadioButton, 321

allowDrop  
QEAnalogProgressBar, 103  
QEBitStatus, 112  
QECheckBox, 123  
QEComboBox, 137  
QEConfiguredLayout, 144  
QEFileBrowser, 152  
QEForm, 157  
QEFrame, 162  
QEGenericEdit, 175  
QEGroupBox, 181  
QEImage, 218  
QELabel, 246  
QELog, 265  
QEPeriodic, 275  
QEPlot, 286  
QEPushButton, 297  
QERadioButton, 321  
QEScript, 340

QEShape, 351  
QESlider, 363  
QESpinBox, 370

allowFocusUpdate  
QEComboBox, 137  
QESlider, 363  
QESpinBox, 370

altReadbackVariable  
QEPushButton, 297

Always  
QEAnalogProgressBar, 101  
QEBitStatus, 111  
QECheckBox, 120  
QEComboBox, 136  
QEConfiguredLayout, 143  
QEFileBrowser, 151  
QEFrame, 162  
QEGenericEdit, 172  
QEGroupBox, 180  
QEImage, 214  
QELabel, 244  
QELog, 264  
QEPeriodic, 274

QEPlot, 285  
QEPushButton, 294  
QERadioButton, 318  
QEScript, 340  
QEShape, 349  
QESlider, 362  
QESpinBox, 369

animation1  
QEShape, 351

animation2  
QEShape, 351

animation3  
QEShape, 351

animation4  
QEShape, 352

animation5  
QEShape, 352

animation6

QEShape, 352  
 animationOptions  
     QEShape, 349  
 Append  
     QEAnalogProgressBar, 100  
     QECheckBox, 119  
     QELabel, 243  
     QELineEdit, 255  
     QEPushButton, 293  
     QERadioButton, 317  
 areaColor  
     QEImage, 218  
 areaInfo, 37  
 arguments  
     QECheckBox, 123  
     QEPushButton, 297  
     QERadioButton, 321  
 arguments1  
     QEImage, 218  
 arguments2  
     QEImage, 218  
 arrayAction  
     QEAnalogProgressBar, 103  
     QECheckBox, 124  
     QELabel, 246  
     QELineEdit, 256  
     QEPushButton, 297  
     QERadioButton, 322  
 ArrayActions  
     QEAnalogProgressBar, 100  
     QECheckBox, 119  
     QELabel, 243  
     QELineEdit, 255  
     QEPushButton, 293  
     QERadioButton, 317  
 arrayIndex  
     QEAnalogProgressBar, 103  
     QEBitStatus, 112  
     QECheckBox, 124  
     QEComboBox, 137  
     QEGenericEdit, 175  
     QELabel, 246  
     QEPushButton, 297  
     QE PvFrame, 307  
     QERadioButton, 322  
     QESlider, 363  
     QESpinBox, 370  
 Ascii  
     QEAnalogProgressBar, 100  
     QECheckBox, 119  
 QELabel, 243  
 QELineEdit, 255  
 QEPushButton, 293  
 QERadioButton, 317  
 autoBrightnessContrast  
 QEImage, 218  
 Automatic  
     QEAnalogProgressBar, 101  
     QECheckBox, 121  
     QELabel, 244  
     QELineEdit, 255  
     QEPushButton, 294  
     QERadioButton, 319  
 backgroundColour  
     QEAnalogIndicator, 94  
 Bar  
     QEAnalogIndicator, 94  
 Bayer  
     QEImage, 215  
 BayerBG  
     QEImage, 215  
 BayerGB  
     QEImage, 215  
 BayerGR  
     QEImage, 215  
 BayerRG  
     QEImage, 215  
 beamColor  
     QEImage, 219  
 beamXVariable  
     QEImage, 219  
 beamYVariable  
     QEImage, 219  
 bitDepthVariable  
     QEImage, 219  
 borderColour  
     QEAnalogIndicator, 94  
 Bottom\_To\_Top  
     QEAnalogIndicator, 94  
 BOUNDING\_RECTANGLE  
     QEImage, 214  
 BoundingRectangle  
     QEImage, 214  
 briefInfoArea  
     QEImage, 219  
 buildImageCore  
     imagePropertiesCore, 64  
 CenterAndSize

QEImage, 214  
centreAngle  
QEAnalogIndicator, 94  
clickCheckedText  
QECheckBox, 124  
QEPushButton, 298  
QERadioButton, 322  
clicked  
QECheckBox, 122  
QEPushButton, 296  
QERadioButton, 320  
clickText  
QECheckBox, 124  
QEPushButton, 298  
QERadioButton, 322  
clippingHighVariable  
QEImage, 219  
clippingLowVariable  
QEImage, 219  
clippingOnOffVariable  
QEImage, 219  
color1  
QEShape, 352  
color10  
QEShape, 352  
color2  
QEShape, 352  
color3  
QEShape, 352  
color4  
QEShape, 352  
color5  
QEShape, 352  
color6  
QEShape, 353  
color7  
QEShape, 353  
color8  
QEShape, 353  
color9  
QEShape, 353  
Comma  
QEAnalogProgressBar, 101  
QECheckBox, 121  
QELabel, 244  
QELineEdit, 256  
QERadioButton, 319  
confirmAction  
QECheckBox, 124  
QEPushButton, 298  
QERadioButton, 322  
confirmText  
QECheckBox, 125  
QEPushButton, 298  
QERadioButton, 323  
confirmWrite  
QEGenericEdit, 175  
contrastReversal  
QEImage, 219  
creationOption  
QECheckBox, 125  
QEPushButton, 298  
QERadioButton, 323  
CreationOptionNames  
QECheckBox, 119  
QEPushButton, 293  
QERadioButton, 317  
customisationName  
QECheckBox, 125  
QEPushButton, 298  
QERadioButton, 323  
dataTypeVariable  
QEImage, 220  
dbConnectionChanged  
QEAnalogProgressBar, 102  
QEBitStatus, 111  
QEPvFrame, 306  
dbElementChanged  
QEPeriodic, 274  
dbValueChanged  
QEAnalogProgressBar, 102  
QEBitStatus, 111  
QECheckBox, 122  
QEComboBox, 137  
QEImage, 217  
QELabel, 246  
QELineEdit, 256  
QEPeriodic, 274  
QEPlot, 285  
QEPushButton, 296  
QEPvFrame, 306  
QERadioButton, 320  
QESlider, 362  
QESpinBox, 370  
dbValueChanged1  
QEShape, 350  
dbValueChanged2  
QEShape, 350  
dbValueChanged3

QEShape, 350  
 dbValueChanged4  
     QEShape, 350  
 dbValueChanged5  
     QEShape, 351  
 dbValueChanged6  
     QEShape, 351  
 Default  
     QEAnalogProgressBar, 101  
     QECheckBox, 120  
     QELabel, 244  
     QELineEdit, 255  
     QEPushButton, 294  
     QERadioButton, 318  
 defaultStyle  
     QEAnalogProgressBar, 103  
     QEBitStatus, 112  
     QECheckBox, 125  
     QEComboBox, 137  
     QEConfiguredLayout, 144  
     QEFileBrowser, 152  
     QEFrame, 162  
     QEGenericEdit, 175  
     QEGroupBox, 181  
     QEImage, 220  
     QELabel, 247  
     QELog, 265  
     QEPlot, 286  
     QEPushButton, 299  
     QERadioButton, 323  
     QEScript, 340  
     QEShape, 353  
     QESlider, 363  
     QESpinBox, 370  
 dimension1Variable  
     QEImage, 220  
 dimension2Variable  
     QEImage, 220  
 dimension3Variable  
     QEImage, 220  
 dimensionsVariable  
     QEImage, 220  
 disabledRecordPolicy  
     QECheckBox, 125  
     QEPushButton, 299  
     QERadioButton, 323  
 displayAlarmState  
     QEAnalogProgressBar, 104  
     QEBitStatus, 112  
     QECheckBox, 125  
     QEComboBox, 138  
     QEConfiguredLayout, 144  
     QEFileBrowser, 152  
     QEFrame, 162  
     QEGenericEdit, 175  
     QEGroupBox, 181  
     QEImage, 220  
     QELabel, 247  
     QELog, 265  
     QEPlot, 286  
     QEPushButton, 299  
     QERadioButton, 323  
     QEScript, 340  
     QEShape, 353  
     QESlider, 363  
     QESpinBox, 370  
     DisplayAlarmStateOptions  
         QEAnalogProgressBar, 100  
         QEBitStatus, 111  
         QECheckBox, 120  
         QEComboBox, 136  
         QEConfiguredLayout, 143  
         QEFileBrowser, 151  
         QEFrame, 162  
         QEGenericEdit, 172  
         QEGroupBox, 180  
         QEImage, 214

QELabel, 243  
QELog, 264  
QEPeriodic, 274  
QEPlot, 285  
QEPushButton, 294  
QERadioButton, 318  
QEScript, 340  
QEShape, 349  
QESlider, 362  
QESpinBox, 369  
displayArea1Selection  
    QEImage, 221  
displayArea2Selection  
    QEImage, 221  
displayArea3Selection  
    QEImage, 221  
displayArea4Selection  
    QEImage, 221  
displayBeamSelection  
    QEImage, 221  
displayButtonBar  
    QEImage, 218  
displayCursorPixelInfo  
    QEImage, 221  
displayEllipse  
    QEImage, 221  
displayHozSlice1Selection  
    QEImage, 222  
displayHozSlice2Selection  
    QEImage, 222  
displayHozSlice3Selection  
    QEImage, 222  
displayHozSlice4Selection  
    QEImage, 222  
displayHozSlice5Selection  
    QEImage, 222  
displayProfileSelection  
    QEImage, 222  
displayTargetSelection  
    QEImage, 222  
displayVertSlice1Selection  
    QEImage, 222  
displayVertSlice2Selection  
    QEImage, 222  
displayVertSlice3Selection  
    QEImage, 223  
displayVertSlice4Selection  
    QEImage, 223  
displayVertSlice5Selection  
    QEImage, 223

DockBottom  
    QECheckBox, 119  
    QEPushButton, 293  
    QERadioButton, 317  
DockBottomTabbed  
    QECheckBox, 120  
    QEPushButton, 293  
    QERadioButton, 318  
DockFloating  
    QECheckBox, 120  
    QEPushButton, 294  
    QERadioButton, 318  
DockLeft  
    QECheckBox, 120  
    QEPushButton, 293  
    QERadioButton, 318  
DockLeftTabbed  
    QECheckBox, 120  
    QEPushButton, 293  
    QERadioButton, 318  
DockRight  
    QECheckBox, 120  
    QEPushButton, 293  
    QERadioButton, 318  
DockRightTabbed  
    QECheckBox, 120  
    QEPushButton, 294  
    QERadioButton, 318  
DockTop  
    QECheckBox, 119  
    QEPushButton, 293  
    QERadioButton, 317  
DockTopTabbed  
    QECheckBox, 120  
    QEPushButton, 293  
    QERadioButton, 318  
DottedFullCrosshair  
    QEImage, 217  
drawMarkup  
    markupHLine, 71  
    markupVLine, 78

ellipseColor  
    QEImage, 223  
ellipseHVariable  
    QEImage, 223  
EllipseVariableDefinitions  
    QEImage, 214  
ellipseVariableDefinitions  
    QEImage, 214

ellipseWVariable  
     QEImage, 223  
 ellipseXVariable  
     QEImage, 223  
 ellipseYVariable  
     QEImage, 223  
 enableArea1Selection  
     QEImage, 223  
 enableArea2Selection  
     QEImage, 224  
 enableArea3Selection  
     QEImage, 224  
 enableArea4Selection  
     QEImage, 224  
 enableBeamSelection  
     QEImage, 224  
 enableHozSlice1Selection  
     QEImage, 224  
 enableHozSlice2Selection  
     QEImage, 224  
 enableHozSlice3Selection  
     QEImage, 224  
 enableHozSlice4Selection  
     QEImage, 224  
 enableHozSlice5Selection  
     QEImage, 225  
 enableProfileSelection  
     QEImage, 225  
 enableTargetSelection  
     QEImage, 225  
 enableVertSlice1Selection  
     QEImage, 225  
 enableVertSlice2Selection  
     QEImage, 225  
 enableVertSlice3Selection  
     QEImage, 225  
 enableVertSlice4Selection  
     QEImage, 225  
 enableVertSlice5Selection  
     QEImage, 226  
 Engineer  
     QEAnalogProgressBar, 102  
     QEBitStatus, 111  
     QECheckBox, 122  
     QEComboBox, 136  
     QEConfiguredLayout, 143  
     QEFileBrowser, 151  
     QEFrame, 162  
     QEGenericEdit, 173  
     QEGroupBox, 180  
 QEImage, 217  
 QELabel, 245  
 QELog, 265  
 QEPeriodic, 274  
 QEPlot, 285  
 QEPushButton, 295  
 QERadioButton, 320  
 QEScript, 340  
 QEShape, 350  
 QESlider, 362  
 QESpinBox, 369  
 externalControls  
     QEImage, 226  
 FFBuffer, 41  
 FFThread, 42  
 Fit  
     QEImage, 215  
 Fixed  
     QEAnalogProgressBar, 101  
     QECheckBox, 121  
     QELabel, 244  
     QELineEdit, 255  
     QEPushButton, 294  
     QERadioButton, 319  
 flipRotateMenu, 43  
 Floating  
     QEAnalogProgressBar, 101  
     QECheckBox, 120  
     QELabel, 244  
     QELineEdit, 255  
     QEPushButton, 294  
     QERadioButton, 318  
 fontColour  
     QEAnalogIndicator, 94  
 foregroundColour  
     QEAnalogIndicator, 94  
 format  
     QEAnalogProgressBar, 104  
     QECheckBox, 126  
     QELabel, 247  
     QELineEdit, 257  
     QEPushButton, 299  
     QERadioButton, 324  
 formatOption  
     QEImage, 226  
 FormatOptions  
     QEImage, 214  
 Formats  
     QEAnalogProgressBar, 101

QECheckBox, 120  
QELabel, 244  
QELineEdit, 255  
QEPushButton, 294  
QERadioButton, 318  
formatVariable  
    QEImage, 226  
fullScreenWindow, 44  
  
getConfirmWrite  
    QEGenericEdit, 173  
getPixelValueFromData  
    imageProcessor, 60  
getSubscribe  
    QEGenericEdit, 173  
getWriteOnEnter  
    QEGenericEdit, 173  
getWriteOnFinish  
    QEGenericEdit, 173  
getWriteOnLoseFocus  
    QEGenericEdit, 174  
guiFile  
    QECheckBox, 126  
    QEPushButton, 300  
    QERadioButton, 324  
  
handleGuiLaunchRequests  
    QEForm, 157  
heightVariable  
    QEImage, 226  
histogram, 45  
histogramScroll, 46  
historicImage, 47  
horizontalFlip  
    QEImage, 226  
hozSlice1Color  
    QEImage, 226  
hozSlice2Color  
    QEImage, 226  
hozSlice3Color  
    QEImage, 226  
hozSlice4Color  
    QEImage, 227  
hozSlice5Color  
    QEImage, 227  
  
Icon  
    QECheckBox, 121  
    QEPushButton, 295  
    QERadioButton, 319  
  
imageContextMenu, 48  
imageDisplayProperties, 50  
imageDisplayProperties::rgbPixel, 393  
imageInfo, 52  
imageMarkup, 53  
imageMarkupLegendSetText, 56  
imageProcessor, 57  
    getPixelValueFromData, 60  
imageProperties, 61  
    imageProperties, 63  
    ROTATION\_0, 63  
    ROTATION\_180, 63  
    ROTATION\_90\_LEFT, 63  
    ROTATION\_90\_RIGHT, 63  
    rotationOptions, 63  
imagePropertiesCore, 64  
    buildImageCore, 64  
imageUpdateIndicator, 65  
imageVariable  
    QEImage, 227  
Index  
    QEAnalogProgressBar, 100  
    QECheckBox, 119  
    QELabel, 243  
    QELineEdit, 255  
    QEPushButton, 293  
    QERadioButton, 317  
initialHosScrollPos  
    QEImage, 227  
initialVertScrollPos  
    QEImage, 218  
int  
    QEAnalogProgressBar, 104  
    QEBitStatus, 112  
    QECheckBox, 126  
    QEComboBox, 138  
    QEConfiguredLayout, 144  
    QEFileBrowser, 152  
    QEForm, 158  
    QEFrame, 163  
    QEGenericEdit, 176  
    QEGroupBox, 181  
    QEImage, 227  
    QELabel, 247  
    QELog, 266  
    QEPeriodic, 275  
    QEPlot, 286  
    QEPushButton, 300  
    QERadioButton, 324  
    QEScript, 341

QEShape, 354  
 QESlider, 364  
 QESpinBox, 371  
**Integer**  
     QEAnalogProgressBar, 101  
     QECheckBox, 120  
     QELabel, 244  
     QELineEdit, 255  
     QEPushButton, 294  
     QERadioButton, 318  
**labelText**  
     QECheckBox, 126  
     QEPushButton, 300  
     QERadioButton, 324  
     QESubstitutedLabel, 391  
**leadingZero**  
     QEAnalogProgressBar, 104  
     QECheckBox, 127  
     QELabel, 247  
     QELineEdit, 257  
     QEPushButton, 300  
     QERadioButton, 325  
**Left\_To\_Right**  
     QEAnalogIndicator, 94  
**lineProfileArrayVariable**  
     QEImage, 227  
**lineProfileThicknessVariable**  
     QEImage, 227  
**lineProfileX1Variable**  
     QEImage, 227  
**lineProfileX2Variable**  
     QEImage, 228  
**lineProfileY1Variable**  
     QEImage, 228  
**lineProfileY2Variable**  
     QEImage, 228  
**LocalEnumeration**  
     QEAnalogProgressBar, 101  
     QECheckBox, 120  
     QELabel, 244  
     QELineEdit, 255  
     QEPushButton, 294  
     QERadioButton, 318  
**localEnumeration**  
     QEAnalogProgressBar, 104  
     QECheckBox, 127  
     QEComboBox, 138  
     QELabel, 247  
     QELineEdit, 257  
**QEPushButton**, 300  
**QERadioButton**, 325  
**logBrightness**  
     QEImage, 228  
**loginWidget**, 66  
**LogOutput**  
     QECheckBox, 121  
     QEImage, 215  
     QEPushButton, 295  
     QERadioButton, 319  
**logScale**  
     QEAnalogIndicator, 94  
**logScaleInterval**  
     QEAnalogIndicator, 95  
**majorInterval**  
     QEAnalogIndicator, 95  
**markupCrosshair1**, 67  
**markupCrosshair2**, 68  
**markupDisplayMenu**, 69  
**markupEllipse**, 70  
**markupHLine**, 71  
     drawMarkup, 71  
**markupItem**, 72  
**markupLine**, 75  
**markupRegion**, 76  
**markupText**, 77  
**markupVLine**, 78  
     drawMarkup, 78  
**maximum**  
     QEAnalogIndicator, 95  
**messageFormFilter**  
     QEForm, 158  
**messageSourceFilter**  
     QEForm, 158  
**Meter**  
     QEAnalogIndicator, 94  
**minimum**  
     QEAnalogIndicator, 95  
**minorInterval**  
     QEAnalogIndicator, 95  
**mode**  
     QEAnalogIndicator, 95  
**Modes**  
     QEAnalogIndicator, 94  
**Mono**  
     QEImage, 215  
**mpegSource**, 79  
     updateImage, 79  
**mpegSourceObject**, 80

Never  
QEAnalogProgressBar, 101  
QEBitStatus, 111  
QECheckBox, 120  
QEComboBox, 136  
QEConfiguredLayout, 143  
QEFileBrowser, 151  
QEFrame, 162  
QEGenericEdit, 172  
QEGroupBox, 180  
QEImage, 214  
QELabel, 244  
QELog, 264  
QEPeriodic, 274  
QEPlot, 285  
QEPushButton, 294  
QERadioButton, 318  
QEScript, 340  
QEShape, 349  
QESlider, 362  
QESpinBox, 369  
NewTab  
QECheckBox, 119  
QEPushButton, 293  
QERadioButton, 317  
NewWindow  
QECheckBox, 119  
QEPushButton, 293  
QERadioButton, 317  
None  
QECheckBox, 121  
QEImage, 215  
QEPushButton, 295  
QERadioButton, 319  
NoRotation  
QEImage, 216  
NoSeparator  
QEAnalogProgressBar, 101  
QECheckBox, 121  
QELabel, 244  
QELineEdit, 256  
QERadioButton, 319  
notation  
QEAnalogProgressBar, 105  
QECheckBox, 127  
QELabel, 248  
QELineEdit, 257  
QEPushButton, 301  
QERadioButton, 325  
Notations  
QEAnalogProgressBar, 101  
QECheckBox, 120  
QELabel, 244  
QELineEdit, 255  
QEPushButton, 294  
QERadioButton, 318  
offset1  
QEShape, 354  
offset2  
QEShape, 354  
offset3  
QEShape, 354  
offset4  
QEShape, 354  
offset5  
QEShape, 354  
offset6  
QEShape, 354  
Open  
QECheckBox, 119  
QEPushButton, 293  
QERadioButton, 317  
orientation  
QEAnalogIndicator, 95  
Orientations  
QEAnalogIndicator, 94  
password  
QECheckBox, 127  
QEPushButton, 301  
QERadioButton, 325  
PeriodicDialog, 82  
PeriodicElementSetupForm, 83  
PeriodicSetupDialog, 84  
Picture  
QELabel, 245  
pixmap  
QEFrame, 163  
pixmap0  
QECheckBox, 128  
QEFrame, 163  
QELabel, 248  
QEPushButton, 301  
QERadioButton, 326  
pixmap1  
QECheckBox, 128  
QEFrame, 163  
QELabel, 248  
QEPushButton, 301

QERadioButton, 326  
 pixmap2  
     QECheckBox, 128  
     QEFrame, 163  
     QELabel, 248  
     QEPushButton, 301  
     QERadioButton, 326  
 pixmap3  
     QECheckBox, 128  
     QEFrame, 163  
     QELabel, 249  
     QEPushButton, 301  
     QERadioButton, 326  
 pixmap4  
     QECheckBox, 128  
     QEFrame, 164  
     QELabel, 249  
     QEPushButton, 302  
     QERadioButton, 326  
 pixmap5  
     QECheckBox, 128  
     QEFrame, 164  
     QELabel, 249  
     QEPushButton, 302  
     QERadioButton, 326  
 pixmap6  
     QECheckBox, 128  
     QEFrame, 164  
     QELabel, 249  
     QEPushButton, 302  
     QERadioButton, 326  
 pixmap7  
     QECheckBox, 128  
     QEFrame, 164  
     QELabel, 249  
     QEPushButton, 302  
     QERadioButton, 326  
 playbackTimer, 85  
 point1  
     QEShape, 354  
 point10  
     QEShape, 355  
 point2  
     QEShape, 355  
 point3  
     QEShape, 355  
 point4  
     QEShape, 355  
 point5  
     QEShape, 355  
 point6  
     QEShape, 355  
 point7  
     QEShape, 355  
 point8  
     QEShape, 355  
 point9  
     QEShape, 356  
 pointInfo, 86  
 precision  
     QEAnalogProgressBar, 105  
     QECheckBox, 128  
     QELabel, 249  
     QLineEdit, 258  
     QEPushButton, 302  
     QERadioButton, 326  
 pressed  
     QECheckBox, 122  
     QEPushButton, 296  
     QERadioButton, 320  
 pressText  
     QECheckBox, 129  
     QEPushButton, 302  
     QERadioButton, 327  
 prioritySubstitutions  
     QECheckBox, 129  
     QEPushButton, 302  
     QERadioButton, 327  
 profileColor  
     QEImage, 228  
 profileHoz1ThicknessVariable  
     QEImage, 228  
 profileHoz1Variable  
     QEImage, 228  
 profileHoz2ThicknessVariable  
     QEImage, 228  
 profileHoz2Variable  
     QEImage, 228  
 profileHoz3ThicknessVariable  
     QEImage, 229  
 profileHoz3Variable  
     QEImage, 229  
 profileHoz4ThicknessVariable  
     QEImage, 229  
 profileHoz4Variable  
     QEImage, 229  
 profileHoz5ThicknessVariable  
     QEImage, 229  
 profileHoz5Variable  
     QEImage, 229

profileHozArrayVariable  
QEImage, 229

profilePlot, 87

profileVert1ThicknessVariable  
QEImage, 229

profileVert1Variable  
QEImage, 230

profileVert2ThicknessVariable  
QEImage, 230

profileVert2Variable  
QEImage, 230

profileVert3ThicknessVariable  
QEImage, 230

profileVert3Variable  
QEImage, 230

profileVert4ThicknessVariable  
QEImage, 230

profileVert4Variable  
QEImage, 230

profileVert5ThicknessVariable  
QEImage, 230

profileVert5Variable  
QEImage, 231

profileVertArrayListVariable  
QEImage, 231

program  
QECheckBox, 129  
QEPushButton, 303  
QERadioButton, 327

program1  
QEImage, 231

program2  
QEImage, 231

programStartupOption  
QECheckBox, 129  
QEPushButton, 303  
QERadioButton, 327

programStartupOption1  
QEImage, 231

programStartupOption2  
QEImage, 231

ProgramStartupOptionNames  
QECheckBox, 121  
QEImage, 215  
QEPushButton, 294  
QERadioButton, 319

QBitStatus, 88

QEAnalogIndicator, 90  
backgroundColour, 94

Bar, 94

borderColour, 94

Bottom\_To\_Top, 94

centreAngle, 94

fontColour, 94

foregroundColour, 94

Left\_To\_Right, 94

logScale, 94

logScaleInterval, 95

majorInterval, 95

maximum, 95

Meter, 94

minimum, 95

minorInterval, 95

mode, 95

Modes, 94

orientation, 95

Orientations, 94

Right\_To\_Left, 94

Scale, 94

showScale, 95

showText, 95

spanAngle, 95

Top\_To\_Bottom, 94

value, 95

QEAnalogIndicator::Band, 38

QEAnalogIndicator::BandList, 39

QEAnalogProgressBar, 97  
addUnits, 103  
alarmSeverityDisplayStyle, 103  
allowDrop, 103  
Always, 101  
Append, 100  
arrayAction, 103  
ArrayActions, 100  
arrayIndex, 103  
Ascii, 100  
Automatic, 101  
Comma, 101  
dbConnectionChanged, 102  
dbValueChanged, 102  
Default, 101  
defaultStyle, 103  
displayAlarmState, 104  
displayAlarmStateOption, 104  
DisplayAlarmStateOptions, 100  
Engineer, 102  
Fixed, 101  
Floating, 101  
format, 104

Formats, 101  
Index, 100  
int, 104  
Integer, 101  
leadingZero, 104  
LocalEnumeration, 101  
localEnumeration, 104  
Never, 101  
NoSeparator, 101  
notation, 105  
Notations, 101  
precision, 105  
QEAnalogProgressBar, 102  
radix, 105  
Scientific, 101  
Scientist, 102  
separator, 105  
Separators, 101  
setManagedVisible, 102  
Space, 102  
styleSheet, 105  
Time, 101  
trailingZeros, 106  
Underscore, 102  
UnsignedInteger, 101  
useDbDisplayLimits, 106  
useDbPrecision, 106  
User, 102  
userLevelEnabled, 106  
userLevelEngineerStyle, 106  
UserLevels, 102  
userLevelScientistStyle, 106  
userLevelUserStyle, 107  
userLevelVisibility, 107  
value, 107  
variable, 107  
variableAsToolTip, 107  
variableSubstitutions, 107  
visible, 108  
WhenInAlarm, 101  
QEBitStatus, 109  
allowDrop, 112  
Always, 111  
arrayIndex, 112  
dbConnectionChanged, 111  
dbValueChanged, 111  
defaultStyle, 112  
displayAlarmState, 112  
displayAlarmStateOption, 112  
DisplayAlarmStateOptions, 111  
Engineer, 111  
int, 112  
Never, 111  
Scientist, 111  
setManagedVisible, 111  
styleSheet, 113  
User, 111  
userLevelEnabled, 113  
userLevelEngineerStyle, 113  
UserLevels, 111  
userLevelScientistStyle, 113  
userLevelUserStyle, 113  
userLevelVisibility, 114  
variable, 114  
variableAsToolTip, 114  
variableSubstitutions, 114  
visible, 114  
WhenInAlarm, 111  
QECheckBox, 115  
addUnits, 123  
alignment, 123  
allowDrop, 123  
Always, 120  
Append, 119  
arguments, 123  
arrayAction, 124  
ArrayActions, 119  
arrayIndex, 124  
Ascii, 119  
Automatic, 121  
clickCheckedText, 124  
clicked, 122  
clickText, 124  
Comma, 121  
confirmAction, 124  
confirmText, 125  
creationOption, 125  
CreationOptionNames, 119  
customisationName, 125  
dbValueChanged, 122  
Default, 120  
defaultStyle, 125  
disabledRecordPolicy, 125  
displayAlarmState, 125  
displayAlarmStateOption, 126  
DisplayAlarmStateOptions, 120  
DockBottom, 119  
DockBottomTabbed, 120  
DockFloating, 120  
DockLeft, 120

DockLeftTabbed, 120  
DockRight, 120  
DockRightTabbed, 120  
DockTop, 119  
DockTopTabbed, 120  
Engineer, 122  
Fixed, 121  
Floating, 120  
format, 126  
Formats, 120  
guiFile, 126  
Icon, 121  
Index, 119  
int, 126  
Integer, 120  
labelText, 126  
leadingZero, 127  
LocalEnumeration, 120  
localEnumeration, 127  
LogOutput, 121  
Never, 120  
NewTab, 119  
NewWindow, 119  
None, 121  
NoSeparator, 121  
notation, 127  
Notations, 120  
Open, 119  
password, 127  
pixmap0, 128  
pixmap1, 128  
pixmap2, 128  
pixmap3, 128  
pixmap4, 128  
pixmap5, 128  
pixmap6, 128  
pixmap7, 128  
precision, 128  
pressed, 122  
pressText, 129  
prioritySubstitutions, 129  
program, 129  
programStartupOption, 129  
ProgramStartupOptionNames, 121  
QECheckBox, 122  
radix, 129  
released, 123  
releaseText, 129  
requestAction, 123  
Scientific, 121  
Scientist, 122  
separator, 129  
Separators, 121  
setManagedVisible, 123  
Space, 121  
State, 122  
StdOutput, 121  
styleSheet, 130  
subscribe, 130  
Terminal, 121  
Text, 121  
TextAndIcon, 122  
Time, 120  
trailingZeros, 130  
Underscore, 121  
UnsignedInteger, 120  
updateOption, 130  
UpdateOptions, 121  
useDbPrecision, 130  
User, 122  
userLevelEnabled, 130  
userLevelEngineerStyle, 130  
UserLevels, 122  
userLevelScientistStyle, 131  
userLevelUserStyle, 131  
userLevelVisibility, 131  
variable, 131  
variableAsToolTip, 131  
variableSubstitutions, 131  
visible, 131  
WhenInAlarm, 120  
writeOnClick, 132  
writeOnPress, 132  
writeOnRelease, 132  
QECheckBoxManager, 133  
QEComboBox, 134  
allowDrop, 137  
allowFocusUpdate, 137  
Always, 136  
arrayIndex, 137  
dbValueChanged, 137  
defaultStyle, 137  
displayAlarmState, 138  
displayAlarmStateOption, 138  
DisplayAlarmStateOptions, 136  
Engineer, 136  
int, 138  
localEnumeration, 138  
Never, 136  
Scientist, 136

setManagedVisible, 137  
styleSheet, 138  
subscribe, 138  
useDbEnumerations, 137  
User, 136  
userLevelEnabled, 138  
userLevelEngineerStyle, 139  
UserLevels, 136  
userLevelScientistStyle, 139  
userLevelUserStyle, 139  
userLevelVisibility, 139  
variable, 139  
variableAsToolTip, 140  
variableSubstitutions, 140  
visible, 140  
WhenInAlarm, 136  
writeOnChange, 137  
QEConfiguredLayout, 141  
allowDrop, 144  
Always, 143  
defaultStyle, 144  
displayAlarmState, 144  
displayAlarmStateOption, 144  
DisplayAlarmStateOptions, 143  
Engineer, 143  
int, 144  
Never, 143  
Scientist, 143  
setManagedVisible, 144  
styleSheet, 145  
User, 143  
userLevelEnabled, 145  
userLevelEngineerStyle, 145  
UserLevels, 143  
userLevelScientistStyle, 145  
userLevelUserStyle, 145  
userLevelVisibility, 146  
variableAsToolTip, 146  
visible, 146  
WhenInAlarm, 143  
QEConfiguredLayoutManager, 147  
QEFileBrowser, 148  
allowDrop, 152  
Always, 151  
defaultStyle, 152  
displayAlarmState, 152  
displayAlarmStateOption, 152  
DisplayAlarmStateOptions, 151  
Engineer, 151  
int, 152  
Never, 151  
Scientist, 151  
selected, 152  
setManagedVisible, 152  
styleSheet, 153  
User, 151  
userLevelEnabled, 153  
userLevelEngineerStyle, 153  
UserLevels, 151  
userLevelScientistStyle, 153  
userLevelUserStyle, 153  
userLevelVisibility, 154  
variable, 154  
variableAsToolTip, 154  
variableSubstitutions, 154  
visible, 154  
WhenInAlarm, 151  
QEForm, 155  
allowDrop, 157  
displayAlarmStateOption, 157  
handleGuiLaunchRequests, 157  
int, 158  
messageFormFilter, 158  
messageSourceFilter, 158  
resizeContents, 157  
uiFile, 158  
variableAsToolTip, 158  
variableSubstitutions, 158  
QEFrame, 160  
allowDrop, 162  
Always, 162  
defaultStyle, 162  
displayAlarmState, 162  
displayAlarmStateOption, 163  
DisplayAlarmStateOptions, 162  
Engineer, 162  
int, 163  
Never, 162  
 pixmap, 163  
 pixmap0, 163  
 pixmap1, 163  
 pixmap2, 163  
 pixmap3, 163  
 pixmap4, 164  
 pixmap5, 164  
 pixmap6, 164  
 pixmap7, 164  
 scaledContents, 164  
Scientist, 162  
setManagedVisible, 162

styleSheet, 164  
User, 162  
userLevelEnabled, 164  
userLevelEngineerStyle, 164  
UserLevels, 162  
userLevelScientistStyle, 165  
userLevelUserStyle, 165  
userLevelVisibility, 165  
variableAsToolTip, 165  
visible, 165  
WhenInAlarm, 162  
QEGenericButton, 167  
QEGenericEdit, 170  
    allowDrop, 175  
    Always, 172  
    arrayIndex, 175  
    confirmWrite, 175  
    defaultStyle, 175  
    displayAlarmState, 175  
    displayAlarmStateOption, 175  
    DisplayAlarmStateOptions, 172  
    Engineer, 173  
    getConfirmWrite, 173  
    getSubscribe, 173  
    getWriteOnEnter, 173  
    getWriteOnFinish, 173  
    getWriteOnLoseFocus, 174  
    int, 176  
    Never, 172  
    QEGenericEdit, 173  
    Scientist, 173  
    setAllowFocusUpdate, 174  
    setConfirmWrite, 174  
    setManagedVisible, 174  
    setSubscribe, 174  
    setWriteOnEnter, 174  
    setWriteOnFinish, 174  
    setWriteOnLoseFocus, 174  
    styleSheet, 176  
    subscribe, 176  
    User, 173  
    userLevelEnabled, 176  
    userLevelEngineerStyle, 176  
    UserLevels, 173  
    userLevelScientistStyle, 176  
    userLevelUserStyle, 177  
    userLevelVisibility, 177  
    variable, 177  
    variableAsToolTip, 177  
    variableSubstitutions, 177  
        visible, 177  
        WhenInAlarm, 173  
        writeOnEnter, 177  
        writeOnFinish, 178  
        writeOnLoseFocus, 178  
QEGroupBox, 179  
    allowDrop, 181  
    Always, 180  
    defaultStyle, 181  
    displayAlarmState, 181  
    displayAlarmStateOption, 181  
    DisplayAlarmStateOptions, 180  
    Engineer, 180  
    int, 181  
    Never, 180  
    Scientist, 180  
    setManagedVisible, 181  
    styleSheet, 181  
    substitutedTitle, 182  
    textSubstitutions, 182  
    User, 180  
    userLevelEnabled, 182  
    userLevelEngineerStyle, 182  
    UserLevels, 180  
    userLevelScientistStyle, 182  
    userLevelUserStyle, 182  
    userLevelVisibility, 183  
    variableAsToolTip, 183  
    visible, 183  
    WhenInAlarm, 180  
QEImage, 184  
    allowDrop, 218  
    Always, 214  
    areaColor, 218  
    arguments1, 218  
    arguments2, 218  
    autoBrightnessContrast, 218  
    Bayer, 215  
    BayerBG, 215  
    BayerGB, 215  
    BayerGR, 215  
    BayerRG, 215  
    beamColor, 219  
    beamXVariable, 219  
    beamYVariable, 219  
    bitDepthVariable, 219  
    BOUNDING\_RECTANGLE, 214  
    BoundingRectangle, 214  
    briefInfoArea, 219  
    CenterAndSize, 214

clippingHighVariable, 219  
clippingLowVariable, 219  
clippingOnOffVariable, 219  
contrastReversal, 219  
dataTypeVariable, 220  
dbValueChanged, 217  
defaultStyle, 220  
dimension1Variable, 220  
dimension2Variable, 220  
dimension3Variable, 220  
dimensionsVariable, 220  
displayAlarmState, 220  
displayAlarmStateOption, 221  
DisplayAlarmStateOptions, 214  
displayArea1Selection, 221  
displayArea2Selection, 221  
displayArea3Selection, 221  
displayArea4Selection, 221  
displayBeamSelection, 221  
displayButtonBar, 218  
displayCursorPixelInfo, 221  
displayEllipse, 221  
displayHozSlice1Selection, 222  
displayHozSlice2Selection, 222  
displayHozSlice3Selection, 222  
displayHozSlice4Selection, 222  
displayHozSlice5Selection, 222  
displayProfileSelection, 222  
displayTargetSelection, 222  
displayVertSlice1Selection, 222  
displayVertSlice2Selection, 222  
displayVertSlice3Selection, 223  
displayVertSlice4Selection, 223  
displayVertSlice5Selection, 223  
DottedFullCrosshair, 217  
ellipseColor, 223  
ellipseHVariable, 223  
EllipseVariableDefinitions, 214  
ellipseVariableDefinitions, 214  
ellipseWVariable, 223  
ellipseXVariable, 223  
ellipseYVariable, 223  
enableArea1Selection, 223  
enableArea2Selection, 224  
enableArea3Selection, 224  
enableArea4Selection, 224  
enableBeamSelection, 224  
enableHozSlice1Selection, 224  
enableHozSlice2Selection, 224  
enableHozSlice3Selection, 224  
enableHozSlice4Selection, 224  
enableHozSlice5Selection, 225  
enableProfileSelection, 225  
enableTargetSelection, 225  
enableVertSlice1Selection, 225  
enableVertSlice2Selection, 225  
enableVertSlice3Selection, 225  
enableVertSlice4Selection, 225  
enableVertSlice5Selection, 226  
Engineer, 217  
externalControls, 226  
Fit, 215  
formatOption, 226  
FormatOptions, 214  
formatVariable, 226  
heightVariable, 226  
horizontalFlip, 226  
hozSlice1Color, 226  
hozSlice2Color, 226  
hozSlice3Color, 226  
hozSlice4Color, 227  
hozSlice5Color, 227  
imageVariable, 227  
initialHosScrollPos, 227  
initialVertScrollPos, 218  
int, 227  
lineProfileArrayVariable, 227  
lineProfileThicknessVariable, 227  
lineProfileX1Variable, 227  
lineProfileX2Variable, 228  
lineProfileY1Variable, 228  
lineProfileY2Variable, 228  
logBrightness, 228  
LogOutput, 215  
Mono, 215  
Never, 214  
None, 215  
NoRotation, 216  
profileColor, 228  
profileHoz1ThicknessVariable, 228  
profileHoz1Variable, 228  
profileHoz2ThicknessVariable, 228  
profileHoz2Variable, 228  
profileHoz3ThicknessVariable, 229  
profileHoz3Variable, 229  
profileHoz4ThicknessVariable, 229  
profileHoz4Variable, 229  
profileHoz5ThicknessVariable, 229  
profileHoz5Variable, 229  
profileHozArrayVariable, 229

profileVert1ThicknessVariable, 229  
profileVert1Variable, 230  
profileVert2ThicknessVariable, 230  
profileVert2Variable, 230  
profileVert3ThicknessVariable, 230  
profileVert3Variable, 230  
profileVert4ThicknessVariable, 230  
profileVert4Variable, 230  
profileVert5ThicknessVariable, 230  
profileVert5Variable, 231  
profileVertArrayVariable, 231  
program1, 231  
program2, 231  
programStartupOption1, 231  
programStartupOption2, 231  
ProgramStartupOptionNames, 215  
QEImage, 217  
regionOfInterest1HVariable, 231  
regionOfInterest1WVariable, 232  
regionOfInterest1XVariable, 232  
regionOfInterest1YVariable, 232  
regionOfInterest2HVariable, 232  
regionOfInterest2WVariable, 232  
regionOfInterest2XVariable, 232  
regionOfInterest2YVariable, 232  
regionOfInterest3HVariable, 232  
regionOfInterest3WVariable, 233  
regionOfInterest3XVariable, 233  
regionOfInterest3YVariable, 233  
regionOfInterest4HVariable, 233  
regionOfInterest4WVariable, 233  
regionOfInterest4XVariable, 233  
regionOfInterest4YVariable, 233  
RESIZE\_OPTION\_FIT, 216  
RESIZE\_OPTION\_ZOOM, 216  
resizeOption, 233  
ResizeOptions, 215  
resizeOptions, 215  
rgb1, 215  
rgb2, 215  
rgb3, 215  
Rotate180, 216  
Rotate90Left, 216  
Rotate90Right, 216  
rotation, 234  
RotationOptions, 216  
Scientist, 217  
selectOptions, 216  
setImageFile, 217  
setManagedVisible, 218  
showTime, 234  
SO\_AREA4, 216  
SO\_BEAM, 216  
SO\_HSLICE1, 216  
SO\_HSLICE2, 216  
SO\_HSLICE3, 216  
SO\_HSLICE4, 216  
SO\_HSLICE5, 216  
SO\_NONE, 216  
SO\_PANNING, 216  
SO\_PROFILE, 216  
SO\_TARGET, 216  
SO\_VSLICE1, 216  
SO\_VSLICE2, 216  
SO\_VSLICE3, 216  
SO\_VSLICE4, 216  
SO\_VSLICE5, 216  
SolidSmallCrosshair, 217  
StdOutput, 215  
styleSheet, 234  
targetColor, 234  
TargetOptions, 216  
targetTriggerVariable, 234  
targetXVariable, 234  
targetYVariable, 234  
Terminal, 215  
timeColor, 234  
URL, 234  
useFalseColour, 235  
User, 217  
userLevelEnabled, 235  
userLevelEngineerStyle, 235  
UserLevels, 217  
userLevelScientistStyle, 235  
userLevelUserStyle, 235  
userLeveIVisibility, 235  
variableAsToolTip, 236  
variableSubstitutions, 236  
verticalFlip, 236  
vertSlice1Color, 236  
vertSlice2Color, 236  
vertSlice3Color, 236  
vertSlice4Color, 236  
vertSlice5Color, 236  
visible, 237  
WhenInAlarm, 214  
widthVariable, 237  
yuv422, 215  
yuv444, 215  
Zoom, 215

QEImageMarkupThickness, 238  
QEImageOptionsDialog, 239  
QELabel, 240  
    addUnits, 246  
    allowDrop, 246  
    Always, 244  
    Append, 243  
    arrayAction, 246  
    ArrayActions, 243  
    arrayIndex, 246  
    Ascii, 243  
    Automatic, 244  
    Comma, 244  
    dbValueChanged, 246  
    Default, 244  
    defaultStyle, 247  
    displayAlarmState, 247  
    displayAlarmStateOption, 247  
    DisplayAlarmStateOptions, 243  
    Engineer, 245  
    Fixed, 244  
    Floating, 244  
    format, 247  
    Formats, 244  
    Index, 243  
    int, 247  
    Integer, 244  
    leadingZero, 247  
    LocalEnumeration, 244  
    localEnumeration, 247  
    Never, 244  
    NoSeparator, 244  
    notation, 248  
    Notations, 244  
    Picture, 245  
    pixmap0, 248  
    pixmap1, 248  
    pixmap2, 248  
    pixmap3, 249  
    pixmap4, 249  
    pixmap5, 249  
    pixmap6, 249  
    pixmap7, 249  
    precision, 249  
    QELabel, 245  
    radix, 249  
    Scientific, 244  
    Scientist, 245  
    separator, 249  
    Separators, 244  
setManagedVisible, 246  
Space, 245  
styleSheet, 249  
Text, 245  
Time, 244  
trailingZeros, 250  
Underscore, 245  
UnsignedInteger, 244  
UPDATE\_PIXMAP, 245  
UPDATE\_TEXT, 245  
updateOption, 250  
UpdateOptions, 245  
updateOptions, 245  
useDbPrecision, 250  
User, 245  
userLevelEnabled, 250  
userLevelEngineerStyle, 250  
UserLevels, 245  
userLevelScientistStyle, 250  
userLevelUserStyle, 251  
userLevelVisibility, 251  
variable, 251  
variableAsToolTip, 251  
variableSubstitutions, 251  
visible, 251  
    WhenInAlarm, 244  
QLineEdit, 253  
    addUnits, 256  
    Append, 255  
    arrayAction, 256  
    ArrayActions, 255  
    Ascii, 255  
    Automatic, 255  
    Comma, 256  
    dbValueChanged, 256  
    Default, 255  
    Fixed, 255  
    Floating, 255  
    format, 257  
    Formats, 255  
    Index, 255  
    Integer, 255  
    leadingZero, 257  
    LocalEnumeration, 255  
    localEnumeration, 257  
    NoSeparator, 256  
    notation, 257  
    Notations, 255  
    precision, 258  
    QLineEdit, 256

radix, 258  
Scientific, 255  
separator, 258  
Separators, 255  
Space, 256  
Time, 255  
trailingZeros, 258  
Underscore, 256  
UnsignedInteger, 255  
useDbPrecision, 258  
QELineEditManager, 259  
QELink, 260  
QELog, 262  
    allowDrop, 265  
    Always, 264  
    defaultStyle, 265  
    displayAlarmState, 265  
    displayAlarmStateOption, 265  
    DisplayAlarmStateOptions, 264  
    Engineer, 265  
    int, 266  
    Never, 264  
    Scientist, 265  
    setManagedVisible, 265  
    styleSheet, 266  
    User, 265  
    userLevelEnabled, 266  
    userLevelEngineerStyle, 266  
    UserLevels, 264  
    userLevelScientistStyle, 266  
    userLevelUserStyle, 266  
    userLevelVisibility, 267  
    variableAsToolTip, 267  
    visible, 267  
    WhenInAlarm, 264  
QELogin, 268  
QELoginDialog, 269  
QEPeriodic, 270  
    allowDrop, 275  
    Always, 274  
    dbElementChanged, 274  
    dbValueChanged, 274  
    displayAlarmState, 275  
    displayAlarmStateOption, 275  
    DisplayAlarmStateOptions, 274  
    Engineer, 274  
    int, 275  
    Never, 274  
    readbackLabelVariable1, 275  
    readbackLabelVariable2, 275  
    Scientist, 274  
    subscribe, 276  
    User, 274  
    userLevelEnabled, 276  
    userLevelEngineerStyle, 276  
    UserLevels, 274  
    userLevelScientistStyle, 276  
    userLevelUserStyle, 276  
    userLevelVisibility, 276  
    variableAsToolTip, 277  
    variableSubstitutions, 277  
    visible, 277  
    WhenInAlarm, 274  
    writeButtonVariable1, 277  
    writeButtonVariable2, 277  
QEPeriodic::elementInfoStruct, 40  
QEPeriodic::userInfoStructArray, 398  
QEPeriodicComponentData, 278  
QEPeriodicTaskMenu, 279  
QEPeriodicTaskMenuFactory, 280  
QEPlot, 281  
    allowDrop, 286  
    Always, 285  
    dbValueChanged, 285  
    defaultStyle, 286  
    displayAlarmState, 286  
    displayAlarmStateOption, 286  
    DisplayAlarmStateOptions, 285  
    Engineer, 285  
    int, 286  
    Never, 285  
    Scientist, 285  
    setManagedVisible, 285  
    styleSheet, 286  
    User, 285  
    userLevelEnabled, 286  
    userLevelEngineerStyle, 287  
    UserLevels, 285  
    userLevelScientistStyle, 287  
    userLevelUserStyle, 287  
    userLevelVisibility, 287  
    variable1, 287  
    variable2, 288  
    variable3, 288  
    variable4, 288  
    variableAsToolTip, 288  
    variableSubstitutions, 288  
    visible, 288  
    WhenInAlarm, 285  
    QEPushButton, 289

addUnits, 297  
alignment, 297  
allowDrop, 297  
altReadbackVariable, 297  
Always, 294  
Append, 293  
arguments, 297  
arrayAction, 297  
ArrayActions, 293  
arrayIndex, 297  
Ascii, 293  
Automatic, 294  
clickCheckedText, 298  
clicked, 296  
clickText, 298  
confirmAction, 298  
confirmText, 298  
creationOption, 298  
CreationOptionNames, 293  
customisationName, 298  
dbValueChanged, 296  
Default, 294  
defaultStyle, 299  
disabledRecordPolicy, 299  
displayAlarmState, 299  
displayAlarmStateOption, 299  
DisplayAlarmStateOptions, 294  
DockBottom, 293  
DockBottomTabbed, 293  
DockFloating, 294  
DockLeft, 293  
DockLeftTabbed, 293  
DockRight, 293  
DockRightTabbed, 294  
DockTop, 293  
DockTopTabbed, 293  
Engineer, 295  
Fixed, 294  
Floating, 294  
format, 299  
Formats, 294  
guiFile, 300  
Icon, 295  
Index, 293  
int, 300  
Integer, 294  
labelText, 300  
leadingZero, 300  
LocalEnumeration, 294  
localEnumeration, 300  
LogOutput, 295  
Never, 294  
NewTab, 293  
NewWindow, 293  
None, 295  
notation, 301  
Notations, 294  
Open, 293  
password, 301  
 pixmap0, 301  
 pixmap1, 301  
 pixmap2, 301  
 pixmap3, 301  
 pixmap4, 302  
 pixmap5, 302  
 pixmap6, 302  
 pixmap7, 302  
precision, 302  
pressed, 296  
pressText, 302  
prioritySubstitutions, 302  
program, 303  
programStartupOption, 303  
ProgramStartupOptionNames, 294  
QEPushButton, 295  
released, 296  
releaseText, 303  
requestAction, 296  
Scientific, 294  
Scientist, 295  
setManagedVisible, 296  
State, 295  
StdOutput, 295  
styleSheet, 303  
subscribe, 303  
Terminal, 295  
Text, 295  
TextAndIcon, 295  
Time, 294  
trailingZeros, 303  
UnsignedInteger, 294  
updateOption, 303  
UpdateOptions, 295  
useDbPrecision, 303  
User, 295  
userLevelEnabled, 303  
userLevelEngineerStyle, 304  
UserLevels, 295  
userLevelScientistStyle, 304  
userLevelUserStyle, 304

userLevelVisibility, 304  
variable, 304  
variableAsToolTip, 305  
variableSubstitutions, 305  
visible, 305  
WhenInAlarm, 294  
writeOnClick, 305  
writeOnPress, 305  
writeOnRelease, 305  
QEPvFrame, 306  
arrayIndex, 307  
dbConnectionChanged, 306  
dbValueChanged, 306  
variable, 307  
variableSubstitutions, 307  
QEPvFrameManager, 308  
QEPVNameLists, 309  
QEPvProperties, 310  
variable, 311  
variableSubstitutions, 311  
QEPvPropertiesManager, 312  
QERadioButton, 313  
addUnits, 321  
alignment, 321  
allowDrop, 321  
Always, 318  
Append, 317  
arguments, 321  
arrayAction, 322  
ArrayActions, 317  
arrayIndex, 322  
Ascii, 317  
Automatic, 319  
clickCheckedText, 322  
clicked, 320  
clickText, 322  
Comma, 319  
confirmAction, 322  
confirmText, 323  
creationOption, 323  
CreationOptionNames, 317  
customisationName, 323  
dbValueChanged, 320  
Default, 318  
defaultStyle, 323  
disabledRecordPolicy, 323  
displayAlarmState, 323  
displayAlarmStateOption, 324  
DisplayAlarmStateOptions, 318  
DockBottom, 317  
DockBottomTabbed, 318  
DockFloating, 318  
DockLeft, 318  
DockLeftTabbed, 318  
DockRight, 318  
DockRightTabbed, 318  
DockTop, 317  
DockTopTabbed, 318  
Engineer, 320  
Fixed, 319  
Floating, 318  
format, 324  
Formats, 318  
guiFile, 324  
Icon, 319  
Index, 317  
int, 324  
Integer, 318  
labelText, 324  
leadingZero, 325  
LocalEnumeration, 318  
localEnumeration, 325  
LogOutput, 319  
Never, 318  
NewTab, 317  
NewWindow, 317  
None, 319  
NoSeparator, 319  
notation, 325  
Notations, 318  
Open, 317  
password, 325  
 pixmap0, 326  
 pixmap1, 326  
 pixmap2, 326  
 pixmap3, 326  
 pixmap4, 326  
 pixmap5, 326  
 pixmap6, 326  
 pixmap7, 326  
precision, 326  
pressed, 320  
pressText, 327  
prioritySubstitutions, 327  
program, 327  
programStartupOption, 327  
ProgramStartupOptionNames, 319  
QERadioButton, 320  
radix, 327  
released, 321

releaseText, 327  
 requestAction, 321  
 Scientific, 319  
 Scientist, 320  
 separator, 327  
 Separators, 319  
 setManagedVisible, 321  
 Space, 319  
 State, 320  
 StdOutput, 319  
 styleSheet, 328  
 subscribe, 328  
 Terminal, 319  
 Text, 319  
 TextAndIcon, 320  
 Time, 318  
 trailingZeros, 328  
 Underscore, 319  
 UnsignedInteger, 318  
 updateOption, 328  
 UpdateOptions, 319  
 useDbPrecision, 328  
 User, 320  
 userLevelEnabled, 328  
 userLevelEngineerStyle, 328  
 UserLevels, 320  
 userLevelScientistStyle, 329  
 userLevelUserStyle, 329  
 userLevelVisibility, 329  
 variable, 329  
 variableAsToolTip, 329  
 variableSubstitutions, 329  
 visible, 330  
 WhenInAlarm, 318  
 writeOnClick, 330  
 writeOnPress, 330  
 writeOnRelease, 330  
 QERecipe, 331  
 QERecordSpec, 333  
 QERecordSpecList, 334  
 QEScript, 335  
     allowDrop, 340  
     Always, 340  
     defaultStyle, 340  
     displayAlarmState, 340  
     displayAlarmStateOption, 341  
     DisplayAlarmStateOptions, 340  
     Engineer, 340  
     int, 341  
     Never, 340  
     Scientist, 340  
     setManagedVisible, 340  
     styleSheet, 341  
     User, 340  
     userLevelEnabled, 341  
     userLevelEngineerStyle, 341  
     UserLevels, 340  
     userLevelScientistStyle, 342  
     userLevelUserStyle, 342  
     userLevelVisibility, 342  
     variableAsToolTip, 342  
     visible, 342  
     WhenInAlarm, 340  
 QEShape, 344  
     allowDrop, 351  
     Always, 349  
     animation1, 351  
     animation2, 351  
     animation3, 351  
     animation4, 352  
     animation5, 352  
     animation6, 352  
     animationOptions, 349  
     color1, 352  
     color10, 352  
     color2, 352  
     color3, 352  
     color4, 352  
     color5, 352  
     color6, 353  
     color7, 353  
     color8, 353  
     color9, 353  
     dbValueChanged1, 350  
     dbValueChanged2, 350  
     dbValueChanged3, 350  
     dbValueChanged4, 350  
     dbValueChanged5, 351  
     dbValueChanged6, 351  
     defaultStyle, 353  
     displayAlarmState, 353  
     displayAlarmStateOption, 353  
     DisplayAlarmStateOptions, 349  
     Engineer, 350  
     int, 354  
     Never, 349  
     offset1, 354  
     offset2, 354  
     offset3, 354  
     offset4, 354

offset5, 354  
offset6, 354  
point1, 354  
point10, 355  
point2, 355  
point3, 355  
point4, 355  
point5, 355  
point6, 355  
point7, 355  
point8, 355  
point9, 356  
QEShape, 350  
scale2, 356  
scale3, 356  
scale4, 356  
scale5, 356  
scale6, 356  
Scientist, 350  
setManagedVisible, 351  
shapeOptions, 349  
styleSheet, 356  
User, 350  
userLevelEnabled, 356  
userLevelEngineerStyle, 357  
UserLevels, 349  
userLevelScientistStyle, 357  
userLevelUserStyle, 357  
userLevelVisibility, 357  
variable1, 357  
variable2, 358  
variable3, 358  
variable4, 358  
variable5, 358  
variable6, 358  
variableAsToolTip, 358  
variableSubstitutions, 358  
visible, 359  
WhenInAlarm, 349  
QESlider, 360  
allowDrop, 363  
allowFocusUpdate, 363  
Always, 362  
arrayIndex, 363  
dbValueChanged, 362  
defaultStyle, 363  
displayAlarmState, 363  
displayAlarmStateOption, 363  
DisplayAlarmStateOptions, 362  
Engineer, 362  
int, 364  
Never, 362  
Scientist, 362  
setManagedVisible, 362  
styleSheet, 364  
subscribe, 364  
User, 362  
userLevelEnabled, 364  
userLevelEngineerStyle, 364  
UserLevels, 362  
userLevelScientistStyle, 364  
userLevelUserStyle, 365  
userLevelVisibility, 365  
variable, 365  
variableAsToolTip, 365  
variableSubstitutions, 365  
visible, 365  
WhenInAlarm, 362  
writeOnChange, 363  
QESpinBox, 367  
allowDrop, 370  
allowFocusUpdate, 370  
Always, 369  
arrayIndex, 370  
dbValueChanged, 370  
defaultStyle, 370  
displayAlarmState, 370  
displayAlarmStateOption, 370  
DisplayAlarmStateOptions, 369  
Engineer, 369  
int, 371  
Never, 369  
Scientist, 369  
setManagedVisible, 370  
styleSheet, 371  
subscribe, 371  
User, 369  
userLevelEnabled, 371  
userLevelEngineerStyle, 371  
UserLevels, 369  
userLevelScientistStyle, 371  
userLevelUserStyle, 372  
userLevelVisibility, 372  
variable, 372  
variableAsToolTip, 372  
variableSubstitutions, 372  
visible, 372  
WhenInAlarm, 369  
QEStripChart, 374  
variableSubstitutions, 376

QEStripChartAdjustPVDialog, 377  
QEStripChartContextMenu, 378  
    QEStripChartContextMenu, 378  
QEStripChartDurationDialog, 379  
QEStripChartItem, 380  
QEStripChartNames, 381  
QEStripChartPushButtonSpecifications,  
    383  
QEStripChartRangeDialog, 384  
QEStripChartState, 385  
QEStripChartStateList, 386  
QEStripChartStatistics, 387  
QEStripChartTimeDialog, 388  
QEStripChartToolBar, 389  
QEStripChartToolBar::OwnTabWidget,  
    81  
QESubstitutedLabel, 391  
    labelText, 391  
    textSubstitutions, 391  
  
radix  
    QEAnalogProgressBar, 105  
    QECheckBox, 129  
    QELabel, 249  
    QELlineEdit, 258  
    QERadioButton, 327  
readbackLabelVariable1  
    QEPeriodic, 275  
readbackLabelVariable2  
    QEPeriodic, 275  
recording, 392  
regionOfInterest1HVariable  
    QEImage, 231  
regionOfInterest1WVariable  
    QEImage, 232  
regionOfInterest1XVariable  
    QEImage, 232  
regionOfInterest1YVariable  
    QEImage, 232  
regionOfInterest2HVariable  
    QEImage, 232  
regionOfInterest2WVariable  
    QEImage, 232  
regionOfInterest2XVariable  
    QEImage, 232  
regionOfInterest2YVariable  
    QEImage, 232  
regionOfInterest3HVariable  
    QEImage, 232  
regionOfInterest3WVariable  
    QEImage, 232

QEImage, 216  
rotation  
    QEImage, 234  
ROTATION\_0  
    imageProperties, 63  
ROTATION\_180  
    imageProperties, 63  
ROTATION\_90\_LEFT  
    imageProperties, 63  
ROTATION\_90\_RIGHT  
    imageProperties, 63  
RotationOptions  
    QEImage, 216  
rotationOptions  
    imageProperties, 63  
  
Scale  
    QEAnalogIndicator, 94  
scale2  
    QEShape, 356  
scale3  
    QEShape, 356  
scale4  
    QEShape, 356  
scale5  
    QEShape, 356  
scale6  
    QEShape, 356  
scaledContents  
    QEFrame, 164  
Scientific  
    QEAnalogProgressBar, 101  
    QECheckBox, 121  
    QELabel, 244  
    QELineEdit, 255  
    QEPushButton, 294  
    QERadioButton, 319  
Scientist  
    QEAnalogProgressBar, 102  
    QEBitStatus, 111  
    QECheckBox, 122  
    QEComboBox, 136  
    QEConfiguredLayout, 143  
    QEFileBrowser, 151  
    QEFrame, 162  
    QEGenericEdit, 173  
    QEGroupBox, 180  
    QEImage, 217  
    QELabel, 245  
    QELog, 265  
QEPeriodic, 274  
QEPlot, 285  
QEPushButton, 295  
QERadioButton, 320  
QEScript, 340  
QEShape, 350  
QESlider, 362  
QESpinBox, 369  
screenSelectDialog, 394  
selected  
    QEFileBrowser, 152  
selectMenu, 395  
selectOptions  
    QEImage, 216  
separator  
    QEAnalogProgressBar, 105  
    QECheckBox, 129  
    QELabel, 249  
    QELineEdit, 258  
    QERadioButton, 327  
Separators  
    QEAnalogProgressBar, 101  
    QECheckBox, 121  
    QELabel, 244  
    QELineEdit, 255  
    QERadioButton, 319  
setAllowFocusUpdate  
    QEGenericEdit, 174  
setConfirmWrite  
    QEGenericEdit, 174  
setImageFile  
    QEImage, 217  
setManagedVisible  
    QEAnalogProgressBar, 102  
    QEBitStatus, 111  
    QECheckBox, 123  
    QEComboBox, 137  
    QEConfiguredLayout, 144  
    QEFileBrowser, 152  
    QEFrame, 162  
    QEGenericEdit, 174  
    QEGroupBox, 181  
    QEImage, 218  
    QELabel, 246  
    QELog, 265  
    QEPlot, 285  
    QEPushButton, 296  
    QERadioButton, 321  
    QEScript, 340  
    QEShape, 351

QESlider, 362  
 QESpinBox, 370  
 setSubscribe  
     QEGenericEdit, 174  
 setWriteOnEnter  
     QEGenericEdit, 174  
 setWriteOnFinish  
     QEGenericEdit, 174  
 setWriteOnLoseFocus  
     QEGenericEdit, 174  
 shapeOptions  
     QEShape, 349  
 showScale  
     QEAnalogIndicator, 95  
 showText  
     QEAnalogIndicator, 95  
 showTime  
     QEImage, 234  
 SO\_AREA4  
     QEImage, 216  
 SO\_BEAM  
     QEImage, 216  
 SO\_HSLICE1  
     QEImage, 216  
 SO\_HSLICE2  
     QEImage, 216  
 SO\_HSLICE3  
     QEImage, 216  
 SO\_HSLICE4  
     QEImage, 216  
 SO\_HSLICE5  
     QEImage, 216  
 SO\_NONE  
     QEImage, 216  
 SO\_PANNING  
     QEImage, 216  
 SO\_PROFILE  
     QEImage, 216  
 SO\_TARGET  
     QEImage, 216  
 SO\_VSLICE1  
     QEImage, 216  
 SO\_VSLICE2  
     QEImage, 216  
 SO\_VSLICE3  
     QEImage, 216  
 SO\_VSLICE4  
     QEImage, 216  
 SO\_VSLICE5  
     QEImage, 216

SolidSmallCrosshair  
 QEImage, 217  
 Space  
     QEAnalogProgressBar, 102  
 QECheckBox, 121  
 QELabel, 245  
 QELineEdit, 256  
 QERadioButton, 319  
 spanAngle  
     QEAnalogIndicator, 95  
 State  
     QECheckBox, 122  
     QEPushButton, 295  
     QERadioButton, 320  
 StdOutput  
     QECheckBox, 121  
     QEImage, 215  
     QEPushButton, 295  
     QERadioButton, 319  
 styleSheet  
     QEAnalogProgressBar, 105  
     QEBitStatus, 113  
     QECheckBox, 130  
     QEComboBox, 138  
     QEConfiguredLayout, 145  
     QEFileBrowser, 153  
     QEFrame, 164  
     QEGenericEdit, 176  
     QEGroupBox, 181  
     QEImage, 234  
     QELabel, 249  
     QELog, 266  
     QEPlot, 286  
     QEPushButton, 303  
     QERadioButton, 328  
     QEScript, 341  
     QEShape, 356  
     QESlider, 364  
     QESpinBox, 371  
 subscribe  
     QECheckBox, 130  
     QEComboBox, 138  
     QEGenericEdit, 176  
     QEPeriodic, 276  
     QEPushButton, 303  
     QERadioButton, 328  
     QESlider, 364  
     QESpinBox, 371  
 substitutedTitle  
     QEGroupBox, 182

targetColor  
QEImage, 234

TargetOptions  
QEImage, 216

targetTriggerVariable  
QEImage, 234

targetXVariable  
QEImage, 234

targetYVariable  
QEImage, 234

Terminal  
QECheckBox, 121  
QEImage, 215  
QEPushButton, 295  
QERadioButton, 319

Text  
QECheckBox, 121  
QELabel, 245  
QEPushButton, 295  
QERadioButton, 319

TextAndIcon  
QECheckBox, 122  
QEPushButton, 295  
QERadioButton, 320

textSubstitutions  
QEGroupBox, 182  
QESubstitutedLabel, 391

Time  
QEAnalogProgressBar, 101  
QECheckBox, 120  
QELabel, 244  
QELlineEdit, 255  
QEPushButton, 294  
QERadioButton, 318

timeColor  
QEImage, 234

Top\_To\_Bottom  
QEAnalogIndicator, 94

trace, 396

trailingZeros  
QEAnalogProgressBar, 106  
QECheckBox, 130  
QELabel, 250  
QELlineEdit, 258  
QEPushButton, 303  
QERadioButton, 328

uiFile  
QEForm, 158

Underscore

QEAnalogProgressBar, 102  
QECheckBox, 121  
QELabel, 245  
QELlineEdit, 256  
QERadioButton, 319

UnsignedInteger  
QEAnalogProgressBar, 101  
QECheckBox, 120  
QELabel, 244  
QELlineEdit, 255  
QEPushButton, 294  
QERadioButton, 318

UPDATE\_PIXMAP  
QELabel, 245

UPDATE\_TEXT  
QELabel, 245

updateImage  
mpegSource, 79

updateOption  
QECheckBox, 130  
QELabel, 250  
QEPushButton, 303  
QERadioButton, 328

UpdateOptions  
QECheckBox, 121  
QELabel, 245  
QEPushButton, 295  
QERadioButton, 319

updateOptions  
QELabel, 245

URL  
QEImage, 234

useDbDisplayLimits  
QEAnalogProgressBar, 106

useDbEnumerations  
QEComboBox, 137

useDbPrecision  
QEAnalogProgressBar, 106  
QECheckBox, 130  
QELabel, 250  
QELlineEdit, 258  
QEPushButton, 303  
QERadioButton, 328

useFalseColour  
QEImage, 235

User  
QEAnalogProgressBar, 102  
QEBitStatus, 111  
QECheckBox, 122  
QEComboBox, 136

QEConfiguredLayout, 143  
QEFileBrowser, 151  
QEFrame, 162  
QEGenericEdit, 173  
QEGroupBox, 180  
QEImage, 217  
QELabel, 245  
QELog, 265  
QEPeriodic, 274  
QEPlot, 285  
QEPushButton, 295  
QERadioButton, 320  
QEScript, 340  
QEShape, 350  
QESlider, 362  
QESpinBox, 369  
userInfoStruct, 397  
userLevelEnabled  
    QEAnalogProgressBar, 106  
    QEBitStatus, 113  
    QECheckBox, 130  
    QEComboBox, 138  
    QEConfiguredLayout, 145  
    QEFileBrowser, 153  
    QEFrame, 164  
    QEGenericEdit, 176  
    QEGroupBox, 182  
    QEImage, 235  
    QELabel, 250  
    QELog, 266  
    QEPeriodic, 276  
    QEPlot, 286  
    QEPushButton, 303  
    QERadioButton, 328  
    QEScript, 341  
    QEShape, 356  
    QESlider, 364  
    QESpinBox, 371  
userLevelEngineerStyle  
    QEAnalogProgressBar, 106  
    QEBitStatus, 113  
    QECheckBox, 130  
    QEComboBox, 139  
    QEConfiguredLayout, 145  
    QEFileBrowser, 153  
    QEFrame, 164  
    QEGenericEdit, 176  
    QEGroupBox, 182  
    QEImage, 235  
    QELabel, 250  
    QELog, 266  
    QEPeriodic, 276  
    QEPlot, 287  
    QEPushButton, 304  
    QERadioButton, 329  
    QEScript, 342  
    QEShape, 357  
    QESlider, 364  
UserLevels  
    QEAnalogProgressBar, 102  
    QEBitStatus, 111  
    QECheckBox, 122  
    QEComboBox, 136  
    QEConfiguredLayout, 143  
    QEFileBrowser, 151  
    QEFrame, 162  
    QEGenericEdit, 173  
    QEGroupBox, 180  
    QEImage, 217  
    QELabel, 245  
    QELog, 264  
    QEPeriodic, 274  
    QEPlot, 285  
    QEPushButton, 295  
    QERadioButton, 320  
    QEScript, 340  
    QEShape, 349  
    QESlider, 362  
    QESpinBox, 369  
userLevelScientistStyle  
    QEAnalogProgressBar, 106  
    QEBitStatus, 113  
    QECheckBox, 131  
    QEComboBox, 139  
    QEConfiguredLayout, 145  
    QEFileBrowser, 153  
    QEFrame, 165  
    QEGenericEdit, 176  
    QEGroupBox, 182  
    QEImage, 235  
    QELabel, 250  
    QELog, 266  
    QEPeriodic, 276  
    QEPlot, 287  
    QEPushButton, 304  
    QERadioButton, 329  
    QEScript, 342  
    QEShape, 357  
    QESlider, 364

QESpinBox, 371  
userLevelUserStyle  
QEAnalogProgressBar, 107  
QEBitStatus, 113  
QECheckBox, 131  
QEComboBox, 139  
QEConfiguredLayout, 145  
QEFileBrowser, 153  
QEFrame, 165  
QEGenericEdit, 177  
QEGroupBox, 182  
QEImage, 235  
QELabel, 251  
QELog, 266  
QEPeriodic, 276  
QEPlot, 287  
QEPushButton, 304  
QERadioButton, 329  
QEScript, 342  
QEShape, 357  
QESlider, 365  
QESpinBox, 372  
userLevelVisibility  
QEAnalogProgressBar, 107  
QEBitStatus, 114  
QECheckBox, 131  
QEComboBox, 139  
QEConfiguredLayout, 146  
QEFileBrowser, 154  
QEFrame, 165  
QEGenericEdit, 177  
QEGroupBox, 183  
QEImage, 235  
QELabel, 251  
QELog, 267  
QEPeriodic, 276  
QEPlot, 287  
QEPushButton, 304  
QERadioButton, 329  
QEScript, 342  
QEShape, 357  
QESlider, 365  
QESpinBox, 372  
value  
QEAnalogIndicator, 95  
QEAnalogProgressBar, 107  
ValueScaling, 399  
variable  
QEAnalogProgressBar, 107  
QEBitStatus, 114  
QECheckBox, 131  
QEComboBox, 139  
QEConfiguredLayout, 145  
QEFileBrowser, 153  
QEFrame, 165  
QEGenericEdit, 177  
QEGroupBox, 182  
QEImage, 235  
QELabel, 251  
QELog, 266  
QEPeriodic, 276  
QEPlot, 287  
QEPushButton, 304  
QERadioButton, 329  
QEScript, 342  
QEShape, 357  
QESlider, 365  
QESpinBox, 372  
variable1  
QEPlot, 287  
QEShape, 357  
variable2  
QEPlot, 288  
QEShape, 358  
variable3  
QEPlot, 288  
QEShape, 358  
variable4  
QEPlot, 288  
QEShape, 358  
variable5  
QEShape, 358  
variable6  
QEShape, 358  
variableAsToolTip  
QEAnalogProgressBar, 107  
QEBitStatus, 114  
QECheckBox, 131  
QEComboBox, 140  
QEConfiguredLayout, 146  
QEFileBrowser, 154  
QEForm, 158  
QEFrame, 165  
QEGenericEdit, 177  
QEGroupBox, 183  
QEImage, 236  
QELabel, 251  
QELog, 267  
QEPeriodic, 277  
QEPlot, 288  
QEPushButton, 305  
QERadioButton, 329  
QEScript, 342  
QEShape, 358  
QESlider, 365  
QESpinBox, 372

variableSubstitutions  
 QEAnalogProgressBar, 107  
 QEBitStatus, 114  
 QECheckBox, 131  
 QEComboBox, 140  
 QEFileBrowser, 154  
 QEForm, 158  
 QEGenericEdit, 177  
 QEImage, 236  
 QELabel, 251  
 QEPeriodic, 277  
 QEPlot, 288  
 QEPushButton, 305  
 QEPvFrame, 307  
 QEPvProperties, 311  
 QERadioButton, 329  
 QEShape, 358  
 QESlider, 365  
 QESpinBox, 372  
 QEStripChart, 376  
 verticalFlip  
 QEImage, 236  
 vertSlice1Color  
 QEImage, 236  
 vertSlice2Color  
 QEImage, 236  
 vertSlice3Color  
 QEImage, 236  
 vertSlice4Color  
 QEImage, 236  
 vertSlice5Color  
 QEImage, 236  
 VideoWidget, 400  
 visible  
 QEAnalogProgressBar, 108  
 QEBitStatus, 114  
 QECheckBox, 131  
 QEComboBox, 140  
 QEConfiguredLayout, 146  
 QEFileBrowser, 154  
 QEFrame, 165  
 QEGenericEdit, 177  
 QEGroupBox, 183  
 QEImage, 237  
 QELabel, 251  
 QELog, 267  
 QEPeriodic, 277  
 QEPlot, 288  
 QEPushButton, 305  
 QERadioButton, 330  
 QEScript, 342  
 QEShape, 359  
 QESlider, 365  
 QESpinBox, 372  
 WhenInAlarm  
 QEAnalogProgressBar, 101  
 QEBitStatus, 111  
 QECheckBox, 120  
 QEComboBox, 136  
 QEConfiguredLayout, 143  
 QEFileBrowser, 151  
 QEFrame, 162  
 QEGenericEdit, 173  
 QEGroupBox, 180  
 QEImage, 214  
 QELabel, 244  
 QELog, 264  
 QEPeriodic, 274  
 QEPlot, 285  
 QEPushButton, 294  
 QERadioButton, 318  
 QEScript, 340  
 QEShape, 349  
 QESlider, 362  
 QESpinBox, 369  
 widthVariable  
 QEImage, 237  
 writeButtonVariable1  
 QEPeriodic, 277  
 writeButtonVariable2  
 QEPeriodic, 277  
 writeOnChange  
 QEComboBox, 137  
 QESlider, 363  
 writeOnClick  
 QECheckBox, 132  
 QEPushButton, 305  
 QERadioButton, 330  
 writeOnEnter  
 QEGenericEdit, 177  
 writeOnFinish  
 QEGenericEdit, 178  
 writeOnLoseFocus  
 QEGenericEdit, 178  
 writeOnPress  
 QECheckBox, 132  
 QEPushButton, 305  
 QERadioButton, 330  
 writeOnRelease

QECheckBox, [132](#)  
QEPushButton, [305](#)  
QERadioButton, [330](#)

yuv422  
    QEImage, [215](#)  
yuv444  
    QEImage, [215](#)

Zoom  
    QEImage, [215](#)  
zoomMenu, [402](#)