

# EPICS QT Framework

## 2.9.0

Generated by Doxygen 1.7.4

Wed Sep 10 2014 15:19:49



# Contents

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

9.2	<a href="#">_Field Class Reference</a>	27
9.3	<a href="#">_Item Class Reference</a>	28
9.4	<a href="#">_QDialogItem Class Reference</a>	29
9.5	<a href="#">_QPushButtonGroup Class Reference</a>	29
9.6	<a href="#">_QTableWidgetFileBrowser Class Reference</a>	29
9.7	<a href="#">_QTableWidgetLog Class Reference</a>	30
9.8	<a href="#">_QTableWidgetScript Class Reference</a>	30
9.9	<a href="#">areaInfo Class Reference</a>	30
9.10	<a href="#">QEAnalogIndicator::Band Struct Reference</a>	31
9.11	<a href="#">QEAnalogIndicator::BandList Class Reference</a>	31
9.12	<a href="#">QEPeriodic::elementInfoStruct Struct Reference</a>	31
9.13	<a href="#">FFBuffer Class Reference</a>	32
9.14	<a href="#">FFThread Class Reference</a>	32
9.15	<a href="#">flipRotateMenu Class Reference</a>	33
9.16	<a href="#">fullScreenWindow Class Reference</a>	33
9.17	<a href="#">histogram Class Reference</a>	34
9.18	<a href="#">histogramScroll Class Reference</a>	34
9.19	<a href="#">historicImage Class Reference</a>	34
9.20	<a href="#">imageContextMenu Class Reference</a>	35
9.21	<a href="#">imageDisplayProperties Class Reference</a>	36
9.22	<a href="#">imageInfo Class Reference</a>	37
9.23	<a href="#">imageMarkup Class Reference</a>	38
9.24	<a href="#">imageUpdateIndicator Class Reference</a>	40
9.25	<a href="#">loginWidget Class Reference</a>	40
9.26	<a href="#">markupCrosshair1 Class Reference</a>	40
9.27	<a href="#">markupCrosshair2 Class Reference</a>	41
9.28	<a href="#">markupDisplayMenu Class Reference</a>	42
9.29	<a href="#">markupEllipse Class Reference</a>	42
9.30	<a href="#">markupHLine Class Reference</a>	43
9.30.1	<a href="#">Member Function Documentation</a>	44
9.30.1.1	<a href="#">drawMarkup</a>	44
9.31	<a href="#">markupItem Class Reference</a>	44
9.32	<a href="#">markupLine Class Reference</a>	46
9.33	<a href="#">markupRegion Class Reference</a>	47

9.34	markupText Class Reference	48
9.35	markupVLine Class Reference	48
9.35.1	Member Function Documentation	49
9.35.1.1	drawMarkup	49
9.36	mpegSource Class Reference	49
9.36.1	Member Function Documentation	50
9.36.1.1	updateImage	50
9.37	mpegSourceObject Class Reference	50
9.38	QEStripChartToolBar::OwnWidgets Class Reference	51
9.39	PeriodicDialog Class Reference	51
9.40	PeriodicElementSetupForm Class Reference	51
9.41	PeriodicSetupDialog Class Reference	52
9.42	playbackTimer Class Reference	52
9.43	pointInfo Class Reference	52
9.44	profilePlot Class Reference	53
9.45	QBitStatus Class Reference	53
9.46	QEAAnalogIndicator Class Reference	55
9.46.1	Detailed Description	58
9.46.2	Member Enumeration Documentation	58
9.46.2.1	Modes	58
9.46.2.2	Orientations	59
9.46.3	Property Documentation	59
9.46.3.1	backgroundColour	59
9.46.3.2	borderColour	59
9.46.3.3	centreAngle	59
9.46.3.4	fontColour	59
9.46.3.5	foregroundColour	59
9.46.3.6	logScale	59
9.46.3.7	logScaleInterval	59
9.46.3.8	majorInterval	60
9.46.3.9	maximum	60
9.46.3.10	minimum	60
9.46.3.11	minorInterval	60
9.46.3.12	mode	60

9.46.3.13 orientation . . . . .	60
9.46.3.14 showScale . . . . .	60
9.46.3.15 showText . . . . .	60
9.46.3.16 spanAngle . . . . .	60
9.46.3.17 value . . . . .	60
9.47 QEAnalogProgressBar Class Reference . . . . .	61
9.47.1 Member Enumeration Documentation . . . . .	64
9.47.1.1 ArrayActions . . . . .	64
9.47.1.2 Formats . . . . .	64
9.47.1.3 Notations . . . . .	64
9.47.1.4 UserLevels . . . . .	64
9.47.2 Constructor & Destructor Documentation . . . . .	65
9.47.2.1 QEAnalogProgressBar . . . . .	65
9.47.2.2 QEAnalogProgressBar . . . . .	65
9.47.3 Member Function Documentation . . . . .	65
9.47.3.1 dbValueChanged . . . . .	65
9.47.4 Property Documentation . . . . .	65
9.47.4.1 addUnits . . . . .	65
9.47.4.2 alarmSeverityDisplayMode . . . . .	65
9.47.4.3 allowDrop . . . . .	65
9.47.4.4 arrayAction . . . . .	66
9.47.4.5 displayAlarmState . . . . .	66
9.47.4.6 format . . . . .	66
9.47.4.7 int . . . . .	66
9.47.4.8 leadingZero . . . . .	66
9.47.4.9 localEnumeration . . . . .	67
9.47.4.10 notation . . . . .	67
9.47.4.11 precision . . . . .	67
9.47.4.12 trailingZeros . . . . .	67
9.47.4.13 useDbDisplayLimits . . . . .	67
9.47.4.14 useDbPrecision . . . . .	68
9.47.4.15 userLevelEnabled . . . . .	68
9.47.4.16 userLevelEngineerStyle . . . . .	68
9.47.4.17 userLevelScientistStyle . . . . .	68

9.47.4.18	userLevelUserStyle	68
9.47.4.19	userLevelVisibility	68
9.47.4.20	value	69
9.47.4.21	variable	69
9.47.4.22	variableAsToolTip	69
9.47.4.23	variableSubstitutions	69
9.47.4.24	visible	69
9.48	QEBitStatus Class Reference	69
9.48.1	Member Enumeration Documentation	71
9.48.1.1	UserLevels	71
9.48.2	Member Function Documentation	71
9.48.2.1	dbValueChanged	71
9.48.3	Property Documentation	72
9.48.3.1	allowDrop	72
9.48.3.2	displayAlarmState	72
9.48.3.3	int	72
9.48.3.4	userLevelEnabled	72
9.48.3.5	userLevelEngineerStyle	72
9.48.3.6	userLevelScientistStyle	72
9.48.3.7	userLevelUserStyle	73
9.48.3.8	userLevelVisibility	73
9.48.3.9	variable	73
9.48.3.10	variableAsToolTip	73
9.48.3.11	variableSubstitutions	73
9.48.3.12	visible	73
9.49	QECheckBox Class Reference	74
9.49.1	Member Enumeration Documentation	77
9.49.1.1	ArrayActions	77
9.49.1.2	CreationOptionNames	77
9.49.1.3	Formats	78
9.49.1.4	Notations	78
9.49.1.5	ProgramStartupOptionNames	78
9.49.1.6	UpdateOptions	79
9.49.1.7	UserLevels	79

9.49.2	Constructor & Destructor Documentation . . . . .	79
9.49.2.1	QECheckBox . . . . .	79
9.49.2.2	QECheckBox . . . . .	79
9.49.3	Member Function Documentation . . . . .	80
9.49.3.1	clicked . . . . .	80
9.49.3.2	dbValueChanged . . . . .	80
9.49.3.3	pressed . . . . .	80
9.49.3.4	released . . . . .	80
9.49.3.5	requestAction . . . . .	80
9.49.4	Property Documentation . . . . .	80
9.49.4.1	addUnits . . . . .	80
9.49.4.2	alignment . . . . .	81
9.49.4.3	allowDrop . . . . .	81
9.49.4.4	arguments . . . . .	81
9.49.4.5	arrayAction . . . . .	81
9.49.4.6	clickCheckedText . . . . .	81
9.49.4.7	clickText . . . . .	82
9.49.4.8	confirmAction . . . . .	82
9.49.4.9	confirmText . . . . .	82
9.49.4.10	creationOption . . . . .	82
9.49.4.11	customisationName . . . . .	82
9.49.4.12	displayAlarmState . . . . .	82
9.49.4.13	format . . . . .	83
9.49.4.14	guiFile . . . . .	83
9.49.4.15	int . . . . .	83
9.49.4.16	labelText . . . . .	83
9.49.4.17	leadingZero . . . . .	83
9.49.4.18	localEnumeration . . . . .	83
9.49.4.19	notation . . . . .	84
9.49.4.20	password . . . . .	84
9.49.4.21	pixmap0 . . . . .	84
9.49.4.22	pixmap1 . . . . .	84
9.49.4.23	pixmap2 . . . . .	85
9.49.4.24	pixmap3 . . . . .	85



9.49.4.25 pixmap4 . . . . .	85
9.49.4.26 pixmap5 . . . . .	85
9.49.4.27 pixmap6 . . . . .	85
9.49.4.28 pixmap7 . . . . .	85
9.49.4.29 precision . . . . .	85
9.49.4.30 pressText . . . . .	85
9.49.4.31 prioritySubstitutions . . . . .	86
9.49.4.32 program . . . . .	86
9.49.4.33 programStartupOption . . . . .	86
9.49.4.34 releaseText . . . . .	86
9.49.4.35 subscribe . . . . .	86
9.49.4.36 trailingZeros . . . . .	86
9.49.4.37 updateOption . . . . .	86
9.49.4.38 useDbPrecision . . . . .	87
9.49.4.39 userLevelEnabled . . . . .	87
9.49.4.40 userLevelEngineerStyle . . . . .	87
9.49.4.41 userLevelScientistStyle . . . . .	87
9.49.4.42 userLevelUserStyle . . . . .	87
9.49.4.43 userLevelVisibility . . . . .	87
9.49.4.44 variable . . . . .	88
9.49.4.45 variableAsToolTip . . . . .	88
9.49.4.46 variableSubstitutions . . . . .	88
9.49.4.47 visible . . . . .	88
9.49.4.48 writeOnClick . . . . .	88
9.49.4.49 writeOnPress . . . . .	88
9.49.4.50 writeOnRelease . . . . .	88
9.50 QECheckBoxManager Class Reference . . . . .	89
9.51 QEComboBox Class Reference . . . . .	89
9.51.1 Member Enumeration Documentation . . . . .	91
9.51.1.1 UserLevels . . . . .	91
9.51.2 Member Function Documentation . . . . .	91
9.51.2.1 dbValueChanged . . . . .	91
9.51.3 Member Data Documentation . . . . .	91
9.51.3.1 useDbEnumerations . . . . .	91

9.51.3.2	<a href="#">writeOnChange</a>	92
9.51.4	<a href="#">Property Documentation</a>	92
9.51.4.1	<a href="#">allowDrop</a>	92
9.51.4.2	<a href="#">displayAlarmState</a>	92
9.51.4.3	<a href="#">int</a>	92
9.51.4.4	<a href="#">localEnumeration</a>	92
9.51.4.5	<a href="#">subscribe</a>	92
9.51.4.6	<a href="#">userLevelEnabled</a>	92
9.51.4.7	<a href="#">userLevelEngineerStyle</a>	93
9.51.4.8	<a href="#">userLevelScientistStyle</a>	93
9.51.4.9	<a href="#">userLevelUserStyle</a>	93
9.51.4.10	<a href="#">userLevelVisibility</a>	93
9.51.4.11	<a href="#">variable</a>	93
9.51.4.12	<a href="#">variableAsToolTip</a>	93
9.51.4.13	<a href="#">variableSubstitutions</a>	94
9.51.4.14	<a href="#">visible</a>	94
9.52	<a href="#">QEConfiguredLayout Class Reference</a>	94
9.52.1	<a href="#">Member Enumeration Documentation</a>	96
9.52.1.1	<a href="#">UserLevels</a>	96
9.52.2	<a href="#">Property Documentation</a>	96
9.52.2.1	<a href="#">allowDrop</a>	96
9.52.2.2	<a href="#">displayAlarmState</a>	96
9.52.2.3	<a href="#">int</a>	96
9.52.2.4	<a href="#">userLevelEnabled</a>	97
9.52.2.5	<a href="#">userLevelEngineerStyle</a>	97
9.52.2.6	<a href="#">userLevelScientistStyle</a>	97
9.52.2.7	<a href="#">userLevelUserStyle</a>	97
9.52.2.8	<a href="#">userLevelVisibility</a>	97
9.52.2.9	<a href="#">variableAsToolTip</a>	98
9.52.2.10	<a href="#">visible</a>	98
9.53	<a href="#">QEConfiguredLayoutManager Class Reference</a>	98
9.54	<a href="#">QEFileBrowser Class Reference</a>	98
9.54.1	<a href="#">Detailed Description</a>	101
9.54.2	<a href="#">Member Enumeration Documentation</a>	101

9.54.2.1	UserLevels	101
9.54.3	Member Function Documentation	101
9.54.3.1	selected	101
9.54.4	Property Documentation	102
9.54.4.1	allowDrop	102
9.54.4.2	displayAlarmState	102
9.54.4.3	int	102
9.54.4.4	userLevelEnabled	102
9.54.4.5	userLevelEngineerStyle	102
9.54.4.6	userLevelScientistStyle	102
9.54.4.7	userLevelUserStyle	103
9.54.4.8	userLevelVisibility	103
9.54.4.9	variable	103
9.54.4.10	variableAsToolTip	103
9.54.4.11	variableSubstitutions	103
9.54.4.12	visible	103
9.55	QForm Class Reference	104
9.56	QFrame Class Reference	105
9.56.1	Member Enumeration Documentation	106
9.56.1.1	UserLevels	106
9.56.2	Property Documentation	106
9.56.2.1	allowDrop	106
9.56.2.2	displayAlarmState	107
9.56.2.3	int	107
9.56.2.4	userLevelEnabled	107
9.56.2.5	userLevelEngineerStyle	107
9.56.2.6	userLevelScientistStyle	107
9.56.2.7	userLevelUserStyle	107
9.56.2.8	userLevelVisibility	108
9.56.2.9	variableAsToolTip	108
9.56.2.10	visible	108
9.57	QGenericButton Class Reference	108
9.58	QGenericEdit Class Reference	110
9.58.1	Member Enumeration Documentation	112

9.58.1.1	UserLevels	112
9.58.2	Constructor & Destructor Documentation	113
9.58.2.1	QEGenericEdit	113
9.58.2.2	QEGenericEdit	113
9.58.3	Member Function Documentation	113
9.58.3.1	getConfirmWrite	113
9.58.3.2	getSubscribe	113
9.58.3.3	getWriteOnEnter	113
9.58.3.4	getWriteOnFinish	113
9.58.3.5	getWriteOnLoseFocus	113
9.58.3.6	setConfirmWrite	114
9.58.3.7	setSubscribe	114
9.58.3.8	setWriteOnEnter	114
9.58.3.9	setWriteOnFinish	114
9.58.3.10	setWriteOnLoseFocus	114
9.58.4	Property Documentation	114
9.58.4.1	allowDrop	114
9.58.4.2	confirmWrite	114
9.58.4.3	displayAlarmState	114
9.58.4.4	int	115
9.58.4.5	subscribe	115
9.58.4.6	userLevelEnabled	115
9.58.4.7	userLevelEngineerStyle	115
9.58.4.8	userLevelScientistStyle	115
9.58.4.9	userLevelUserStyle	116
9.58.4.10	userLevelVisibility	116
9.58.4.11	variable	116
9.58.4.12	variableAsToolTip	116
9.58.4.13	variableSubstitutions	116
9.58.4.14	visible	116
9.58.4.15	writeOnEnter	116
9.58.4.16	writeOnFinish	117
9.58.4.17	writeOnLoseFocus	117
9.59	QEGroupBox Class Reference	117

9.59.1	Member Enumeration Documentation	118
9.59.1.1	UserLevels	118
9.59.2	Property Documentation	118
9.59.2.1	allowDrop	118
9.59.2.2	displayAlarmState	118
9.59.2.3	int	119
9.59.2.4	substitutedTitle	119
9.59.2.5	textSubstitutions	119
9.59.2.6	userLevelEnabled	119
9.59.2.7	userLevelEngineerStyle	119
9.59.2.8	userLevelScientistStyle	119
9.59.2.9	userLevelUserStyle	120
9.59.2.10	userLevelVisibility	120
9.59.2.11	variableAsToolTip	120
9.59.2.12	visible	120
9.60	QEImage Class Reference	120
9.60.1	Member Enumeration Documentation	137
9.60.1.1	ellipseVariableDefinitions	137
9.60.1.2	EllipseVariableDefinitions	137
9.60.1.3	FormatOptions	137
9.60.1.4	ProgramStartupOptionNames	137
9.60.1.5	ResizeOptions	138
9.60.1.6	resizeOptions	138
9.60.1.7	RotationOptions	138
9.60.1.8	rotationOptions	138
9.60.1.9	selectOptions	139
9.60.1.10	TargetOptions	139
9.60.1.11	UserLevels	139
9.60.2	Constructor & Destructor Documentation	140
9.60.2.1	QEImage	140
9.60.2.2	QEImage	140
9.60.3	Member Function Documentation	140
9.60.3.1	dbValueChanged	140
9.60.4	Member Data Documentation	140

9.60.4.1	displayButtonBar	140
9.60.4.2	initialVertScrollPos	140
9.60.5	Property Documentation	140
9.60.5.1	allowDrop	140
9.60.5.2	areaColor	141
9.60.5.3	arguments1	141
9.60.5.4	arguments2	141
9.60.5.5	autoBrightnessContrast	141
9.60.5.6	beamColor	141
9.60.5.7	beamXVariable	141
9.60.5.8	beamYVariable	141
9.60.5.9	bitDepthVariable	141
9.60.5.10	briefInfoArea	141
9.60.5.11	clippingHighVariable	142
9.60.5.12	clippingLowVariable	142
9.60.5.13	clippingOnOffVariable	142
9.60.5.14	contrastReversal	142
9.60.5.15	dimension1Variable	142
9.60.5.16	dimension2Variable	142
9.60.5.17	dimension3Variable	142
9.60.5.18	dimensionsVariable	142
9.60.5.19	displayAlarmState	143
9.60.5.20	displayArea1Selection	143
9.60.5.21	displayArea2Selection	143
9.60.5.22	displayArea3Selection	143
9.60.5.23	displayArea4Selection	143
9.60.5.24	displayBeamSelection	143
9.60.5.25	displayCursorPixelInfo	143
9.60.5.26	displayEllipse	143
9.60.5.27	displayHozSliceSelection	144
9.60.5.28	displayProfileSelection	144
9.60.5.29	displayTargetSelection	144
9.60.5.30	displayVertSliceSelection	144
9.60.5.31	ellipseColor	144

9.60.5.32 ellipseHVariable . . . . .	144
9.60.5.33 ellipseWVariable . . . . .	144
9.60.5.34 ellipseXVariable . . . . .	144
9.60.5.35 ellipseYVariable . . . . .	144
9.60.5.36 enableArea1Selection . . . . .	145
9.60.5.37 enableArea2Selection . . . . .	145
9.60.5.38 enableArea3Selection . . . . .	145
9.60.5.39 enableArea4Selection . . . . .	145
9.60.5.40 enableBeamSelection . . . . .	145
9.60.5.41 enableHozSliceSelection . . . . .	145
9.60.5.42 enableProfileSelection . . . . .	145
9.60.5.43 enableTargetSelection . . . . .	145
9.60.5.44 enableVertSliceSelection . . . . .	146
9.60.5.45 externalControls . . . . .	146
9.60.5.46 formatOption . . . . .	146
9.60.5.47 formatVariable . . . . .	146
9.60.5.48 heightVariable . . . . .	146
9.60.5.49 horizontalFlip . . . . .	146
9.60.5.50 hozSliceColor . . . . .	146
9.60.5.51 imageVariable . . . . .	146
9.60.5.52 initialHosScrollPos . . . . .	147
9.60.5.53 int . . . . .	147
9.60.5.54 lineProfileArrayVariable . . . . .	147
9.60.5.55 lineProfileThicknessVariable . . . . .	147
9.60.5.56 lineProfileX1Variable . . . . .	147
9.60.5.57 lineProfileX2Variable . . . . .	147
9.60.5.58 lineProfileY1Variable . . . . .	147
9.60.5.59 lineProfileY2Variable . . . . .	147
9.60.5.60 logBrightness . . . . .	148
9.60.5.61 profileColor . . . . .	148
9.60.5.62 profileHozArrayVariable . . . . .	148
9.60.5.63 profileHozThicknessVariable . . . . .	148
9.60.5.64 profileHozVariable . . . . .	148
9.60.5.65 profileVertArrayVariable . . . . .	148

9.60.5.66 profileVertThicknessVariable . . . . .	148
9.60.5.67 profileVertVariable . . . . .	148
9.60.5.68 program1 . . . . .	148
9.60.5.69 program2 . . . . .	149
9.60.5.70 programStartupOption1 . . . . .	149
9.60.5.71 programStartupOption2 . . . . .	149
9.60.5.72 regionOfInterest1HVariable . . . . .	149
9.60.5.73 regionOfInterest1WVariable . . . . .	149
9.60.5.74 regionOfInterest1XVariable . . . . .	149
9.60.5.75 regionOfInterest1YVariable . . . . .	149
9.60.5.76 regionOfInterest2HVariable . . . . .	149
9.60.5.77 regionOfInterest2WVariable . . . . .	150
9.60.5.78 regionOfInterest2XVariable . . . . .	150
9.60.5.79 regionOfInterest2YVariable . . . . .	150
9.60.5.80 regionOfInterest3HVariable . . . . .	150
9.60.5.81 regionOfInterest3WVariable . . . . .	150
9.60.5.82 regionOfInterest3XVariable . . . . .	150
9.60.5.83 regionOfInterest3YVariable . . . . .	150
9.60.5.84 regionOfInterest4HVariable . . . . .	150
9.60.5.85 regionOfInterest4WVariable . . . . .	150
9.60.5.86 regionOfInterest4XVariable . . . . .	151
9.60.5.87 regionOfInterest4YVariable . . . . .	151
9.60.5.88 resizeOption . . . . .	151
9.60.5.89 rotation . . . . .	151
9.60.5.90 showTime . . . . .	151
9.60.5.91 targetColor . . . . .	151
9.60.5.92 targetTriggerVariable . . . . .	151
9.60.5.93 targetXVariable . . . . .	151
9.60.5.94 targetYVariable . . . . .	151
9.60.5.95 timeColor . . . . .	152
9.60.5.96 URL . . . . .	152
9.60.5.97 useFalseColour . . . . .	152
9.60.5.98 userLevelEnabled . . . . .	152
9.60.5.99 userLevelEngineerStyle . . . . .	152



9.60.5.100	<a href="#">userLevelScientistStyle</a>	152
9.60.5.101	<a href="#">userLevelUserStyle</a>	153
9.60.5.102	<a href="#">userLevelVisibility</a>	153
9.60.5.103	<a href="#">variableAsToolTip</a>	153
9.60.5.104	<a href="#">variableSubstitutions</a>	153
9.60.5.105	<a href="#">verticalFlip</a>	153
9.60.5.106	<a href="#">vertSliceColor</a>	153
9.60.5.107	<a href="#">visible</a>	153
9.60.5.108	<a href="#">widthVariable</a>	154
9.61	<a href="#">QEImageMarkupThickness Class Reference</a>	154
9.62	<a href="#">QEImageOptionsDialog Class Reference</a>	154
9.63	<a href="#">QELabel Class Reference</a>	155
9.63.1	<a href="#">Detailed Description</a>	158
9.63.2	<a href="#">Member Enumeration Documentation</a>	158
9.63.2.1	<a href="#">ArrayActions</a>	158
9.63.2.2	<a href="#">Formats</a>	158
9.63.2.3	<a href="#">Notations</a>	158
9.63.2.4	<a href="#">UpdateOptions</a>	159
9.63.2.5	<a href="#">updateOptions</a>	159
9.63.2.6	<a href="#">UserLevels</a>	159
9.63.3	<a href="#">Constructor &amp; Destructor Documentation</a>	159
9.63.3.1	<a href="#">QELabel</a>	159
9.63.3.2	<a href="#">QELabel</a>	159
9.63.4	<a href="#">Member Function Documentation</a>	160
9.63.4.1	<a href="#">dbValueChanged</a>	160
9.63.5	<a href="#">Property Documentation</a>	160
9.63.5.1	<a href="#">addUnits</a>	160
9.63.5.2	<a href="#">allowDrop</a>	160
9.63.5.3	<a href="#">arrayAction</a>	160
9.63.5.4	<a href="#">displayAlarmState</a>	160
9.63.5.5	<a href="#">format</a>	161
9.63.5.6	<a href="#">int</a>	161
9.63.5.7	<a href="#">leadingZero</a>	161
9.63.5.8	<a href="#">localEnumeration</a>	161

9.63.5.9 notation	162
9.63.5.10 pixmap0	162
9.63.5.11 pixmap1	162
9.63.5.12 pixmap2	162
9.63.5.13 pixmap3	162
9.63.5.14 pixmap4	162
9.63.5.15 pixmap5	162
9.63.5.16 pixmap6	162
9.63.5.17 pixmap7	163
9.63.5.18 precision	163
9.63.5.19 trailingZeros	163
9.63.5.20 updateOption	163
9.63.5.21 useDbPrecision	163
9.63.5.22 userLevelEnabled	163
9.63.5.23 userLevelEngineerStyle	163
9.63.5.24 userLevelScientistStyle	164
9.63.5.25 userLevelUserStyle	164
9.63.5.26 userLevelVisibility	164
9.63.5.27 variable	164
9.63.5.28 variableAsToolTip	164
9.63.5.29 variableSubstitutions	164
9.63.5.30 visible	165
9.64 QLineEdit Class Reference	165
9.64.1 Member Enumeration Documentation	166
9.64.1.1 ArrayActions	166
9.64.1.2 Formats	167
9.64.1.3 Notations	167
9.64.2 Constructor & Destructor Documentation	167
9.64.2.1 QLineEdit	167
9.64.2.2 QLineEdit	167
9.64.3 Member Function Documentation	167
9.64.3.1 dbValueChanged	167
9.64.4 Property Documentation	168
9.64.4.1 addUnits	168

9.64.4.2	<a href="#">arrayAction</a>	168
9.64.4.3	<a href="#">format</a>	168
9.64.4.4	<a href="#">int</a>	168
9.64.4.5	<a href="#">leadingZero</a>	168
9.64.4.6	<a href="#">localEnumeration</a>	168
9.64.4.7	<a href="#">notation</a>	169
9.64.4.8	<a href="#">precision</a>	169
9.64.4.9	<a href="#">trailingZeros</a>	169
9.64.4.10	<a href="#">useDbPrecision</a>	169
9.65	<a href="#">QELineEditManager Class Reference</a>	170
9.66	<a href="#">QELink Class Reference</a>	170
9.67	<a href="#">QELog Class Reference</a>	172
9.67.1	<a href="#">Member Enumeration Documentation</a>	174
9.67.1.1	<a href="#">UserLevels</a>	174
9.67.2	<a href="#">Property Documentation</a>	174
9.67.2.1	<a href="#">allowDrop</a>	174
9.67.2.2	<a href="#">displayAlarmState</a>	174
9.67.2.3	<a href="#">int</a>	174
9.67.2.4	<a href="#">userLevelEnabled</a>	175
9.67.2.5	<a href="#">userLevelEngineerStyle</a>	175
9.67.2.6	<a href="#">userLevelScientistStyle</a>	175
9.67.2.7	<a href="#">userLevelUserStyle</a>	175
9.67.2.8	<a href="#">userLevelVisibility</a>	175
9.67.2.9	<a href="#">variableAsToolTip</a>	176
9.67.2.10	<a href="#">visible</a>	176
9.68	<a href="#">QELogin Class Reference</a>	176
9.69	<a href="#">QELoginDialog Class Reference</a>	177
9.70	<a href="#">QENumericEdit Class Reference</a>	177
9.70.1	<a href="#">Detailed Description</a>	179
9.70.2	<a href="#">Constructor &amp; Destructor Documentation</a>	179
9.70.2.1	<a href="#">QENumericEdit</a>	179
9.70.2.2	<a href="#">QENumericEdit</a>	179
9.70.3	<a href="#">Member Function Documentation</a>	179
9.70.3.1	<a href="#">dbValueChanged</a>	179

9.70.4	Property Documentation	179
9.70.4.1	addUnits	179
9.70.4.2	autoScale	179
9.70.4.3	leadingZeros	180
9.70.4.4	maximum	180
9.70.4.5	minimum	180
9.70.4.6	precision	180
9.71	QENumericEditManager Class Reference	180
9.72	QEPeriodic Class Reference	181
9.72.1	Member Enumeration Documentation	184
9.72.1.1	UserLevels	184
9.72.2	Member Function Documentation	184
9.72.2.1	dbElementChanged	184
9.72.2.2	dbValueChanged	184
9.72.3	Member Data Documentation	184
9.72.3.1	allowDrop	184
9.72.4	Property Documentation	184
9.72.4.1	displayAlarmState	184
9.72.4.2	int	185
9.72.4.3	readbackLabelVariable1	185
9.72.4.4	readbackLabelVariable2	185
9.72.4.5	subscribe	185
9.72.4.6	userLevelEnabled	185
9.72.4.7	userLevelEngineerStyle	185
9.72.4.8	userLevelScientistStyle	185
9.72.4.9	userLevelUserStyle	186
9.72.4.10	userLevelVisibility	186
9.72.4.11	variableAsToolTip	186
9.72.4.12	variableSubstitutions	186
9.72.4.13	visible	186
9.72.4.14	writeButtonVariable1	186
9.72.4.15	writeButtonVariable2	187
9.73	QEPeriodicComponentData Class Reference	187
9.74	QEPeriodicTaskMenu Class Reference	187

9.75 QEP periodicTaskMenuFactory Class Reference . . . . .	187
9.76 QEPlot Class Reference . . . . .	188
9.76.1 Member Enumeration Documentation . . . . .	191
9.76.1.1 UserLevels . . . . .	191
9.76.2 Member Function Documentation . . . . .	191
9.76.2.1 dbValueChanged . . . . .	191
9.76.2.2 dbValueChanged . . . . .	192
9.76.3 Member Data Documentation . . . . .	192
9.76.3.1 allowDrop . . . . .	192
9.76.4 Property Documentation . . . . .	192
9.76.4.1 displayAlarmState . . . . .	192
9.76.4.2 int . . . . .	192
9.76.4.3 userLevelEnabled . . . . .	192
9.76.4.4 userLevelEngineerStyle . . . . .	192
9.76.4.5 userLevelScientistStyle . . . . .	193
9.76.4.6 userLevelUserStyle . . . . .	193
9.76.4.7 userLevelVisibility . . . . .	193
9.76.4.8 variable1 . . . . .	193
9.76.4.9 variable2 . . . . .	193
9.76.4.10 variable3 . . . . .	193
9.76.4.11 variable4 . . . . .	194
9.76.4.12 variableAsToolTip . . . . .	194
9.76.4.13 variableSubstitutions . . . . .	194
9.76.4.14 visible . . . . .	194
9.77 QEPushButton Class Reference . . . . .	194
9.77.1 Member Enumeration Documentation . . . . .	198
9.77.1.1 ArrayActions . . . . .	198
9.77.1.2 CreationOptionNames . . . . .	198
9.77.1.3 Formats . . . . .	198
9.77.1.4 Notations . . . . .	199
9.77.1.5 ProgramStartupOptionNames . . . . .	199
9.77.1.6 UpdateOptions . . . . .	199
9.77.1.7 UserLevels . . . . .	200
9.77.2 Constructor & Destructor Documentation . . . . .	200

9.77.2.1	QEPushButton	200
9.77.2.2	QEPushButton	200
9.77.3	Member Function Documentation	200
9.77.3.1	clicked	200
9.77.3.2	dbValueChanged	200
9.77.3.3	pressed	200
9.77.3.4	released	201
9.77.3.5	requestAction	201
9.77.4	Property Documentation	201
9.77.4.1	addUnits	201
9.77.4.2	alignment	201
9.77.4.3	allowDrop	201
9.77.4.4	altReadbackVariable	201
9.77.4.5	arguments	201
9.77.4.6	arrayAction	202
9.77.4.7	clickCheckedText	202
9.77.4.8	clickText	202
9.77.4.9	confirmAction	202
9.77.4.10	confirmText	202
9.77.4.11	creationOption	203
9.77.4.12	customisationName	203
9.77.4.13	displayAlarmState	203
9.77.4.14	format	203
9.77.4.15	guiFile	203
9.77.4.16	int	203
9.77.4.17	labelText	204
9.77.4.18	leadingZero	204
9.77.4.19	localEnumeration	204
9.77.4.20	notation	205
9.77.4.21	password	205
9.77.4.22	pixmap0	205
9.77.4.23	pixmap1	205
9.77.4.24	pixmap2	205
9.77.4.25	pixmap3	205

9.77.4.26 pixmap4 . . . . .	205
9.77.4.27 pixmap5 . . . . .	205
9.77.4.28 pixmap6 . . . . .	206
9.77.4.29 pixmap7 . . . . .	206
9.77.4.30 precision . . . . .	206
9.77.4.31 pressText . . . . .	206
9.77.4.32 prioritySubstitutions . . . . .	206
9.77.4.33 program . . . . .	206
9.77.4.34 programStartupOption . . . . .	206
9.77.4.35 releaseText . . . . .	207
9.77.4.36 subscribe . . . . .	207
9.77.4.37 trailingZeros . . . . .	207
9.77.4.38 updateOption . . . . .	207
9.77.4.39 useDbPrecision . . . . .	207
9.77.4.40 userLevelEnabled . . . . .	207
9.77.4.41 userLevelEngineerStyle . . . . .	207
9.77.4.42 userLevelScientistStyle . . . . .	208
9.77.4.43 userLevelUserStyle . . . . .	208
9.77.4.44 userLevelVisibility . . . . .	208
9.77.4.45 variable . . . . .	208
9.77.4.46 variableAsToolTip . . . . .	208
9.77.4.47 variableSubstitutions . . . . .	208
9.77.4.48 visible . . . . .	209
9.77.4.49 writeOnClick . . . . .	209
9.77.4.50 writeOnPress . . . . .	209
9.77.4.51 writeOnRelease . . . . .	209
9.78 QEPVNameLists Class Reference . . . . .	209
9.79 QEPvProperties Class Reference . . . . .	209
9.79.1 Property Documentation . . . . .	210
9.79.1.1 variable . . . . .	210
9.79.1.2 variableSubstitutions . . . . .	211
9.80 QEPvPropertiesManager Class Reference . . . . .	211
9.81 QERadioButton Class Reference . . . . .	211
9.81.1 Member Enumeration Documentation . . . . .	215

9.81.1.1	ArrayActions	215
9.81.1.2	CreationOptionNames	215
9.81.1.3	Formats	216
9.81.1.4	Notations	216
9.81.1.5	ProgramStartupOptionNames	216
9.81.1.6	UpdateOptions	216
9.81.1.7	UserLevels	217
9.81.2	Constructor & Destructor Documentation	217
9.81.2.1	QERadioButton	217
9.81.2.2	QERadioButton	217
9.81.3	Member Function Documentation	217
9.81.3.1	clicked	217
9.81.3.2	dbValueChanged	217
9.81.3.3	pressed	218
9.81.3.4	released	218
9.81.3.5	requestAction	218
9.81.4	Property Documentation	218
9.81.4.1	addUnits	218
9.81.4.2	alignment	218
9.81.4.3	allowDrop	218
9.81.4.4	arguments	218
9.81.4.5	arrayAction	219
9.81.4.6	clickCheckedText	219
9.81.4.7	clickText	219
9.81.4.8	confirmAction	219
9.81.4.9	confirmText	219
9.81.4.10	creationOption	220
9.81.4.11	customisationName	220
9.81.4.12	displayAlarmState	220
9.81.4.13	format	220
9.81.4.14	guiFile	220
9.81.4.15	int	220
9.81.4.16	labelText	221
9.81.4.17	leadingZero	221



9.81.4.18 localEnumeration . . . . .	221
9.81.4.19 notation . . . . .	222
9.81.4.20 password . . . . .	222
9.81.4.21 pixmap0 . . . . .	222
9.81.4.22 pixmap1 . . . . .	222
9.81.4.23 pixmap2 . . . . .	222
9.81.4.24 pixmap3 . . . . .	222
9.81.4.25 pixmap4 . . . . .	222
9.81.4.26 pixmap5 . . . . .	222
9.81.4.27 pixmap6 . . . . .	223
9.81.4.28 pixmap7 . . . . .	223
9.81.4.29 precision . . . . .	223
9.81.4.30 pressText . . . . .	223
9.81.4.31 prioritySubstitutions . . . . .	223
9.81.4.32 program . . . . .	223
9.81.4.33 programStartupOption . . . . .	223
9.81.4.34 releaseText . . . . .	224
9.81.4.35 subscribe . . . . .	224
9.81.4.36 trailingZeros . . . . .	224
9.81.4.37 updateOption . . . . .	224
9.81.4.38 useDbPrecision . . . . .	224
9.81.4.39 userLevelEnabled . . . . .	224
9.81.4.40 userLevelEngineerStyle . . . . .	224
9.81.4.41 userLevelScientistStyle . . . . .	225
9.81.4.42 userLevelUserStyle . . . . .	225
9.81.4.43 userLevelVisibility . . . . .	225
9.81.4.44 variable . . . . .	225
9.81.4.45 variableAsToolTip . . . . .	225
9.81.4.46 variableSubstitutions . . . . .	225
9.81.4.47 visible . . . . .	226
9.81.4.48 writeOnClick . . . . .	226
9.81.4.49 writeOnPress . . . . .	226
9.81.4.50 writeOnRelease . . . . .	226
9.82 QERecipe Class Reference . . . . .	226

9.83	QERecordFieldName Class Reference	228
9.84	QERecordSpec Class Reference	229
9.85	QERecordSpecList Class Reference	229
9.86	QEScript Class Reference	229
9.86.1	Detailed Description	233
9.86.2	Member Enumeration Documentation	233
9.86.2.1	UserLevels	233
9.86.3	Property Documentation	234
9.86.3.1	allowDrop	234
9.86.3.2	displayAlarmState	234
9.86.3.3	int	234
9.86.3.4	userLevelEnabled	234
9.86.3.5	userLevelEngineerStyle	234
9.86.3.6	userLevelScientistStyle	234
9.86.3.7	userLevelUserStyle	235
9.86.3.8	userLevelVisibility	235
9.86.3.9	variableAsToolTip	235
9.86.3.10	visible	235
9.87	QEShape Class Reference	235
9.87.1	Detailed Description	239
9.87.2	Member Enumeration Documentation	239
9.87.2.1	animationOptions	239
9.87.2.2	shapeOptions	240
9.87.2.3	UserLevels	240
9.87.3	Constructor & Destructor Documentation	240
9.87.3.1	QEShape	240
9.87.3.2	QEShape	240
9.87.4	Member Function Documentation	240
9.87.4.1	dbValueChanged1	240
9.87.4.2	dbValueChanged2	240
9.87.4.3	dbValueChanged3	241
9.87.4.4	dbValueChanged4	241
9.87.4.5	dbValueChanged5	241
9.87.4.6	dbValueChanged6	241

9.87.5	Property Documentation	241
9.87.5.1	allowDrop	241
9.87.5.2	animation1	241
9.87.5.3	animation2	241
9.87.5.4	animation3	242
9.87.5.5	animation4	242
9.87.5.6	animation5	242
9.87.5.7	animation6	242
9.87.5.8	color1	242
9.87.5.9	color10	242
9.87.5.10	color2	242
9.87.5.11	color3	242
9.87.5.12	color4	242
9.87.5.13	color5	243
9.87.5.14	color6	243
9.87.5.15	color7	243
9.87.5.16	color8	243
9.87.5.17	color9	243
9.87.5.18	displayAlarmState	243
9.87.5.19	int	243
9.87.5.20	offset1	244
9.87.5.21	offset2	244
9.87.5.22	offset3	244
9.87.5.23	offset4	244
9.87.5.24	offset5	244
9.87.5.25	offset6	244
9.87.5.26	point1	244
9.87.5.27	point10	244
9.87.5.28	point2	244
9.87.5.29	point3	245
9.87.5.30	point4	245
9.87.5.31	point5	245
9.87.5.32	point6	245
9.87.5.33	point7	245

9.87.5.34 point8 . . . . .	245
9.87.5.35 point9 . . . . .	245
9.87.5.36 scale2 . . . . .	245
9.87.5.37 scale3 . . . . .	245
9.87.5.38 scale4 . . . . .	246
9.87.5.39 scale5 . . . . .	246
9.87.5.40 scale6 . . . . .	246
9.87.5.41 userLevelEnabled . . . . .	246
9.87.5.42 userLevelEngineerStyle . . . . .	246
9.87.5.43 userLevelScientistStyle . . . . .	246
9.87.5.44 userLevelUserStyle . . . . .	247
9.87.5.45 userLevelVisibility . . . . .	247
9.87.5.46 variable1 . . . . .	247
9.87.5.47 variable2 . . . . .	247
9.87.5.48 variable3 . . . . .	247
9.87.5.49 variable4 . . . . .	247
9.87.5.50 variable5 . . . . .	248
9.87.5.51 variable6 . . . . .	248
9.87.5.52 variableAsToolTip . . . . .	248
9.87.5.53 variableSubstitutions . . . . .	248
9.87.5.54 visible . . . . .	248
9.88 QESlider Class Reference . . . . .	248
9.88.1 Member Enumeration Documentation . . . . .	250
9.88.1.1 UserLevels . . . . .	250
9.88.2 Member Function Documentation . . . . .	250
9.88.2.1 dbValueChanged . . . . .	250
9.88.3 Member Data Documentation . . . . .	250
9.88.3.1 writeOnChange . . . . .	250
9.88.4 Property Documentation . . . . .	251
9.88.4.1 allowDrop . . . . .	251
9.88.4.2 displayAlarmState . . . . .	251
9.88.4.3 int . . . . .	251
9.88.4.4 subscribe . . . . .	251
9.88.4.5 userLevelEnabled . . . . .	251

9.88.4.6	userLevelEngineerStyle	251
9.88.4.7	userLevelScientistStyle	252
9.88.4.8	userLevelUserStyle	252
9.88.4.9	userLevelVisibility	252
9.88.4.10	variable	252
9.88.4.11	variableAsToolTip	252
9.88.4.12	variableSubstitutions	252
9.88.4.13	visible	253
9.89	QESpinBox Class Reference	253
9.89.1	Member Enumeration Documentation	255
9.89.1.1	UserLevels	255
9.89.2	Member Function Documentation	255
9.89.2.1	dbValueChanged	255
9.89.3	Property Documentation	255
9.89.3.1	allowDrop	255
9.89.3.2	displayAlarmState	255
9.89.3.3	int	255
9.89.3.4	subscribe	256
9.89.3.5	userLevelEnabled	256
9.89.3.6	userLevelEngineerStyle	256
9.89.3.7	userLevelScientistStyle	256
9.89.3.8	userLevelUserStyle	256
9.89.3.9	userLevelVisibility	256
9.89.3.10	variable	257
9.89.3.11	variableAsToolTip	257
9.89.3.12	variableSubstitutions	257
9.89.3.13	visible	257
9.90	QEStriptChart Class Reference	257
9.90.1	Property Documentation	259
9.90.1.1	variableSubstitutions	259
9.91	QEStriptChartAdjustPVDialo Class Reference	260
9.92	QEStriptChartContextMenu Class Reference	260
9.92.1	Constructor & Destructor Documentation	260
9.92.1.1	QEStriptChartContextMenu	260

9.93 QEStripChartItem Class Reference . . . . .	261
9.94 QEStripChartNames Class Reference . . . . .	262
9.95 QEStripChartPushButtonSpecifications Struct Reference . . . . .	263
9.96 QEStripChartRangeDialog Class Reference . . . . .	263
9.97 QEStripChartState Class Reference . . . . .	263
9.98 QEStripChartStateList Class Reference . . . . .	264
9.99 QEStripChartStatistics Class Reference . . . . .	264
9.100QEStripChartTimeDialog Class Reference . . . . .	265
9.101QEStripChartToolBar Class Reference . . . . .	265
9.101.1 Detailed Description . . . . .	266
9.102QESubstitutedLabel Class Reference . . . . .	266
9.102.1 Member Data Documentation . . . . .	266
9.102.1.1 labelText . . . . .	266
9.102.2 Property Documentation . . . . .	267
9.102.2.1 textSubstitutions . . . . .	267
9.103recording Class Reference . . . . .	267
9.104imageDisplayProperties::rgbPixel Struct Reference . . . . .	267
9.105screenSelectDialog Class Reference . . . . .	268
9.106selectMenu Class Reference . . . . .	268
9.107trace Class Reference . . . . .	268
9.108userInfoStruct Class Reference . . . . .	269
9.109QEPeriodic::userInfoStructArray Struct Reference . . . . .	269
9.110ValueScaling Class Reference . . . . .	269
9.111VideoWidget Class Reference . . . . .	270
9.112zoomMenu Class Reference . . . . .	271

## Chapter 1

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

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

### 1.1 Documentation

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

## 1.2 License

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

## 1.3 Platforms

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

## 1.4 Screenshots

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

Screenshots are only available in the HTML docs.

## 1.5 Downloads

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

For getting a development snapshot from the SVN repository:

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

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

## 1.6 Installation

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

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

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

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



## 1.7 Support

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

## 1.8 Related Projects

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

## 1.9 Credits:

### Authors:

Andrew Rhyder, Anthony Owen, Glenn Jackson

### Project admin:

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



## Chapter 2

# GNU General Public License

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

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

See the GNU General Public License for more details.

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

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



## Chapter 3

### ASgui screen shots

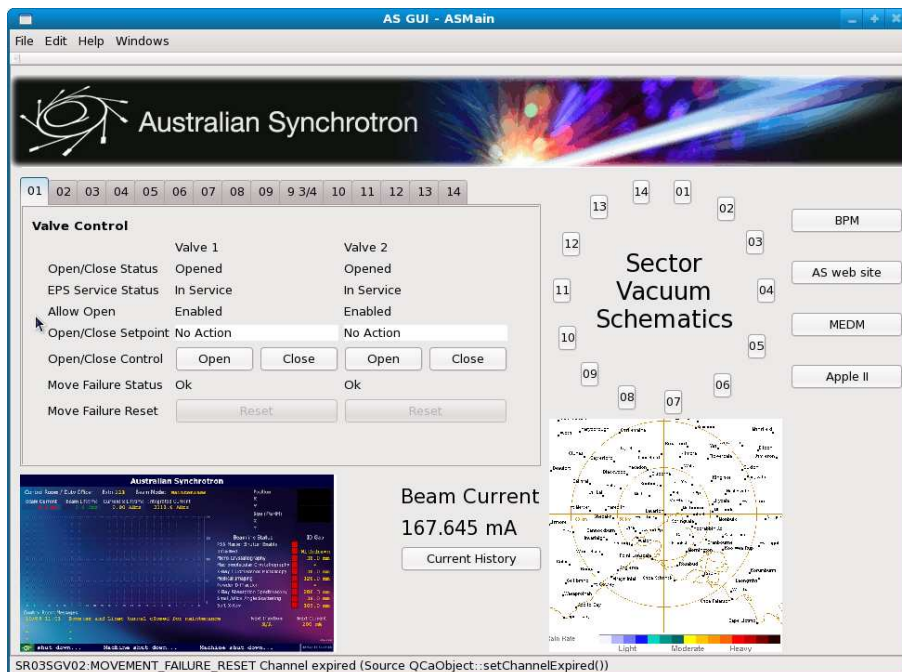


Figure 3.1: Australian Synchrotron mock up

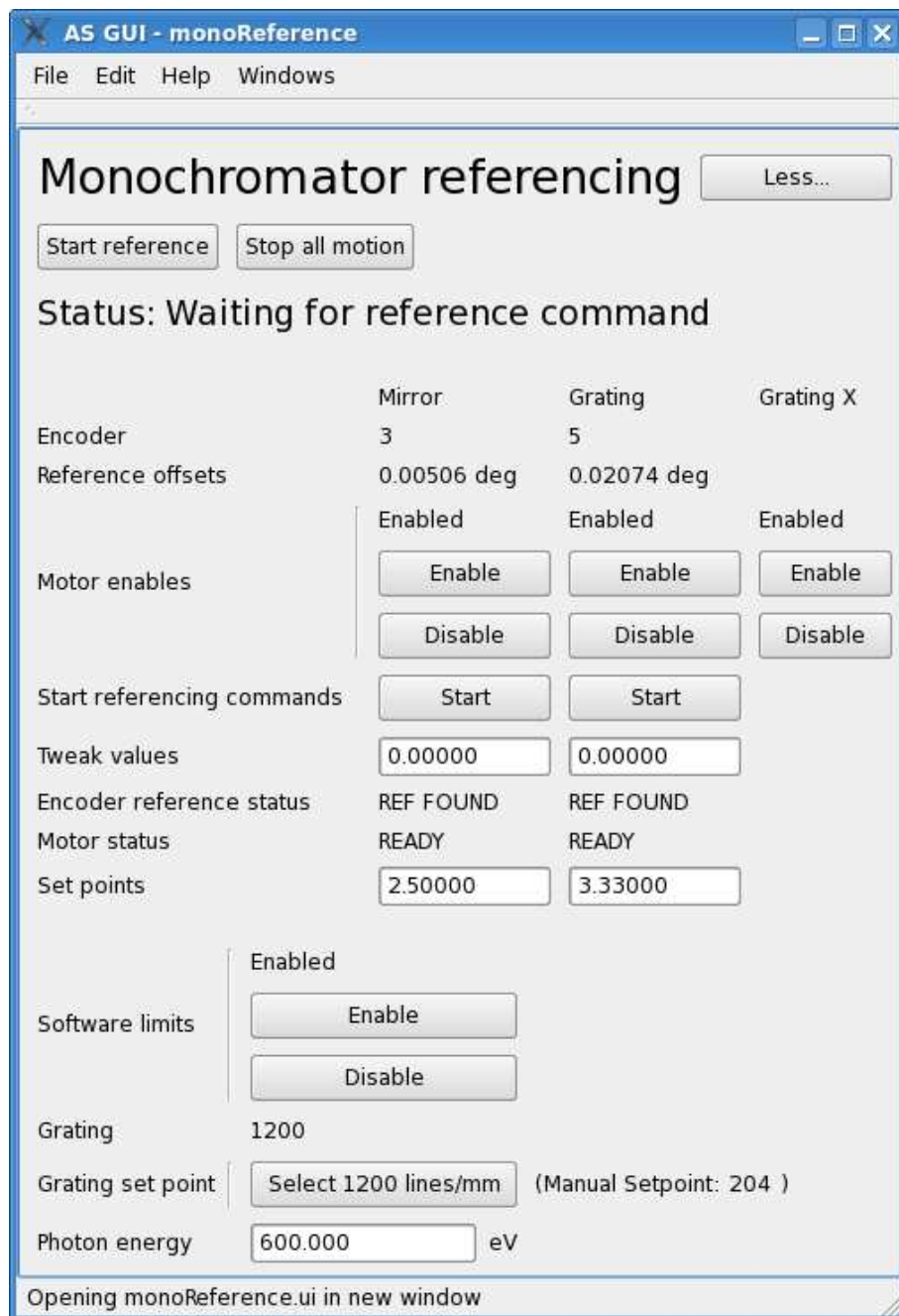


Figure 3.2: Monochromator referencing

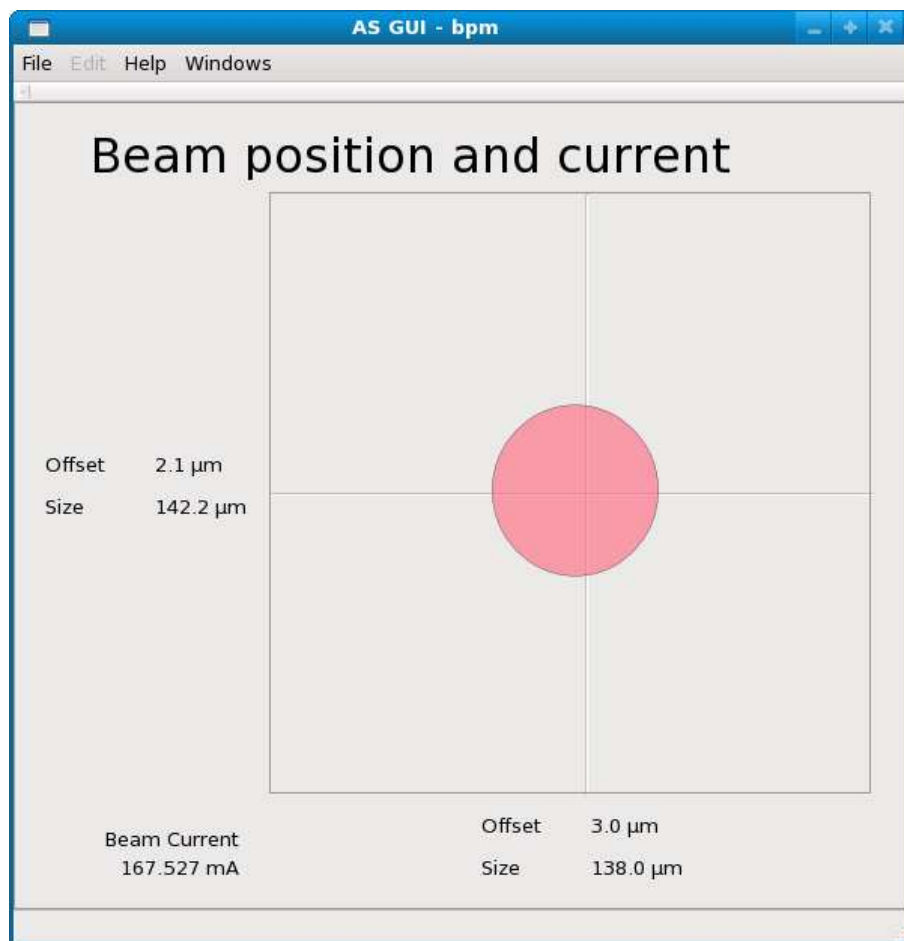


Figure 3.3: Beam position monitor

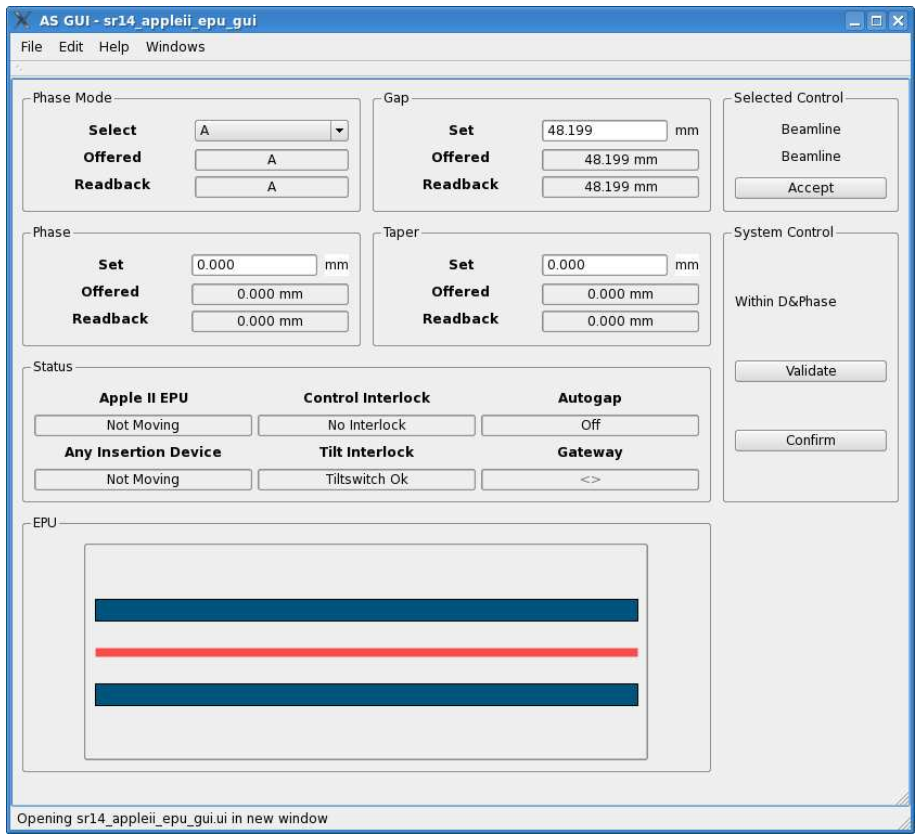


Figure 3.4: Insertion device



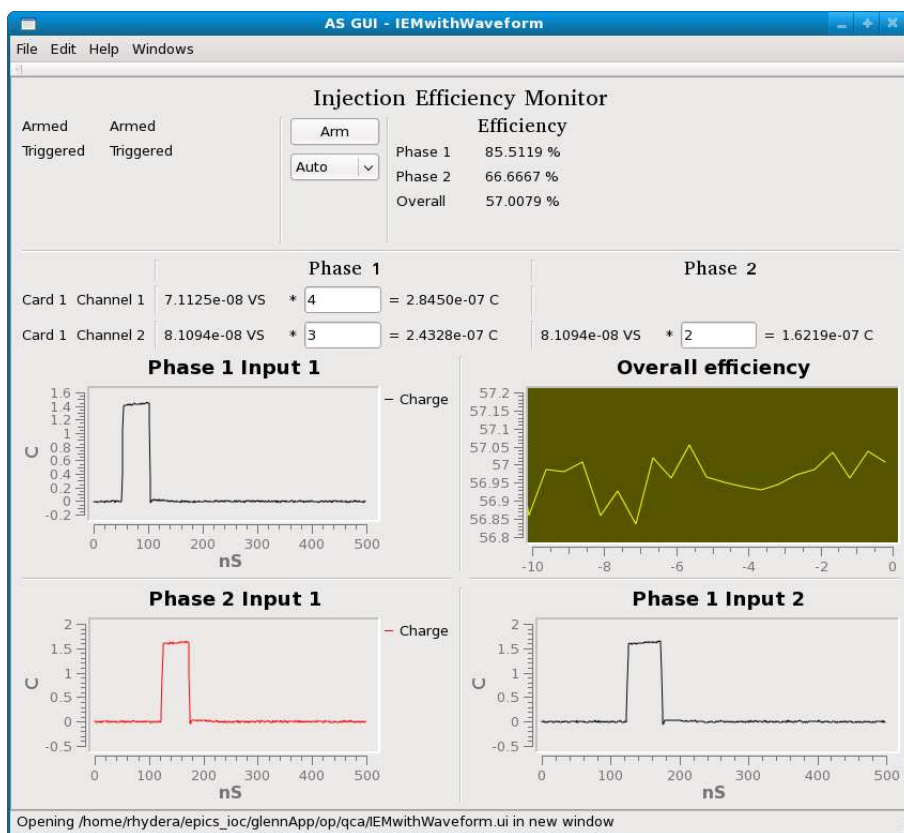


Figure 3.5: Injection efficiency monitor



## Chapter 4

# other applications using epicsqt widgets

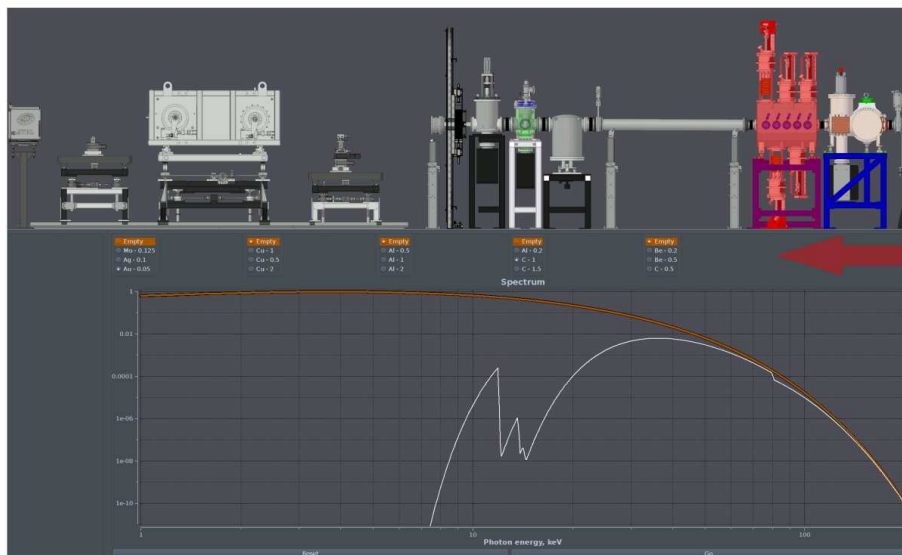


Figure 4.1: Medical Imaging beamline

2B SampleTable Z <@SR08ID01OPI01>

PV name: SR08ID01:MTR32B View mode: Macro

Description: 2B SampleTable Z

Precision: 5 Units: mm

Message: Connection established. clean

User: 6mm Move absolutely Raw: -80932

JOG< UNDO >JOG

LIMIT< Move relatively >LIMIT

< 1mm >

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

User: 6mm = Hi limit

Resolution: 0.0001mm/step \* 42271.2068mm

Raw: -80932 + Lo Limit

Offset: -2.0932mm -42275.3932mm

Speed Acceleration

Maximum: 1.2mm/s

Normal: 0.8mm/s 1s

Backlash: 0mm/s 1s

log: 1mm/s 10s

Backlash: 0mm

Figure 4.2: Motor controller

MotorMx <@SR08ID01OPI01>

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

Add motor

Figure 4.3: Motor controller

## Chapter 5

# Qt Designer

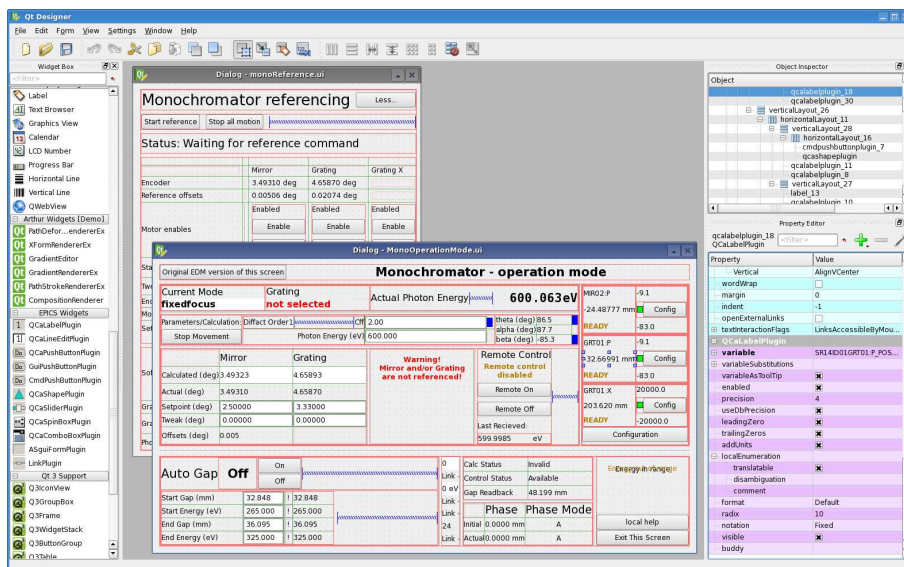


Figure 5.1: Editing multiple GUIs

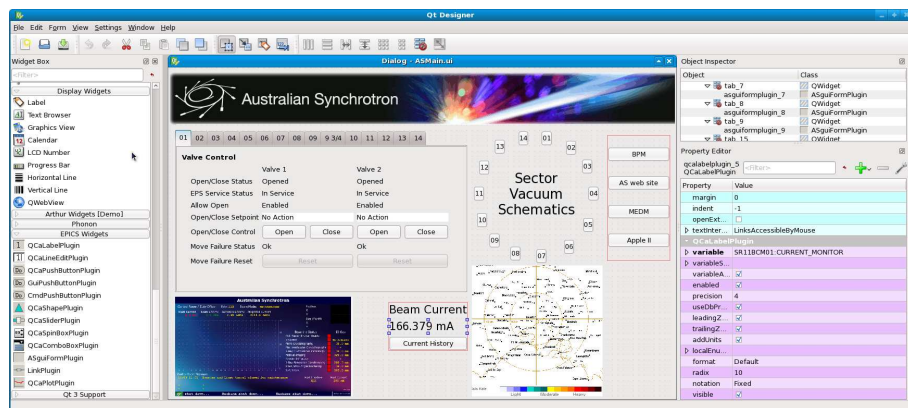


Figure 5.2: Editing a GUI

## Chapter 6

# Qt Creator

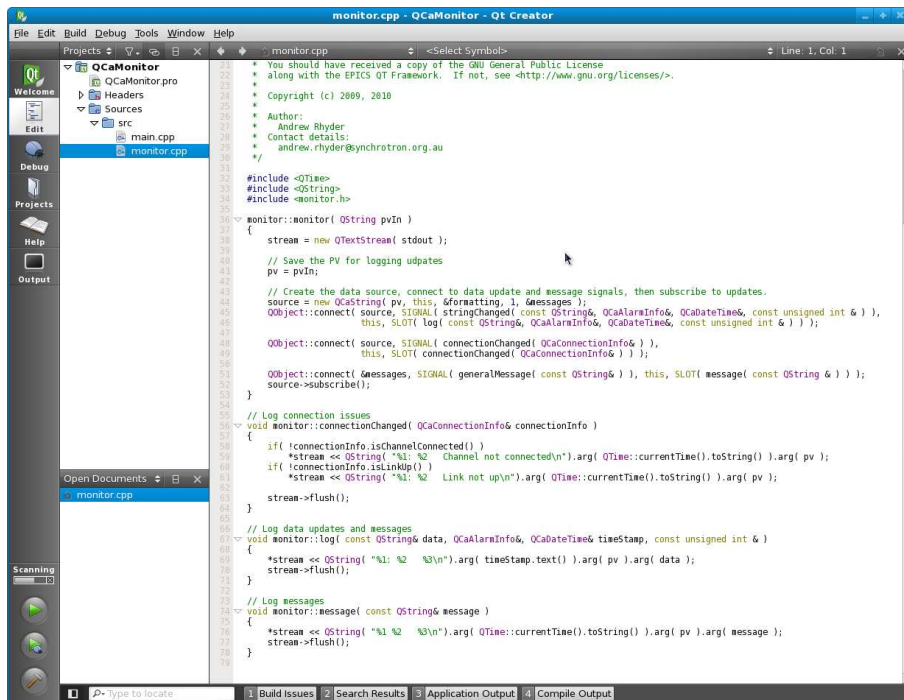


Figure 6.1: Application using epicsqt data source classes





## Chapter 7

# Class Index

### 7.1 Class Hierarchy

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

_CopyPaste	27
_Field	27
_Item	28
_QDialogItem	29
_QPushButtonGroup	29
_QTableWidgetFileBrowser	29
_QTableWidgetLog	30
_QTableWidgetScript	30
areaInfo	30
QEAAnalogIndicator::Band	31
QEAAnalogIndicator::BandList	31
QEPeiodic::elementInfoStruct	31
FFBuffer	32
FFTThread	32
flipRotateMenu	33
fullScreenWindow	33
histogram	34
histogramScroll	34
historicImage	34
imageContextMenu	35
imageDisplayProperties	36
imageInfo	37
QEImage	120
imageMarkup	38
VideoWidget	270
imageUpdateIndicator	40
loginWidget	40
markupDisplayMenu	42
markupItem	44

markupCrosshair1 . . . . .	40
markupCrosshair2 . . . . .	41
markupEllipse . . . . .	42
markupHLine . . . . .	43
markupLine . . . . .	46
markupRegion . . . . .	47
markupText . . . . .	48
markupVLine . . . . .	48
mpegSource . . . . .	49
QElImage . . . . .	120
mpegSourceObject . . . . .	50
QEStripChartToolBar::OwnWidgets . . . . .	51
PeriodicDialog . . . . .	51
PeriodicElementSetupForm . . . . .	51
PeriodicSetupDialog . . . . .	52
playbackTimer . . . . .	52
pointInfo . . . . .	52
profilePlot . . . . .	53
QBitStatus . . . . .	53
QEBitStatus . . . . .	69
QEAnalogIndicator . . . . .	55
QEAnalogProgressBar . . . . .	61
QECheckBoxManager . . . . .	89
QEComboBox . . . . .	89
QEConfiguredLayout . . . . .	94
QEConfiguredLayoutManager . . . . .	98
QEFileBrowser . . . . .	98
QEForm . . . . .	104
QEFrame . . . . .	105
QEPvProperties . . . . .	209
QEStripChart . . . . .	257
QEGenericButton . . . . .	108
QECheckBox . . . . .	74
QEPushButton . . . . .	194
QERadioButton . . . . .	211
QEGenericEdit . . . . .	110
QELineEdit . . . . .	165
QENumericEdit . . . . .	177
QEGroupBox . . . . .	117
QEImageMarkupThickness . . . . .	154
QEImageOptionsDialog . . . . .	154
QELabel . . . . .	155
QELineEditManager . . . . .	170
QELink . . . . .	170
QELog . . . . .	172
QELogin . . . . .	176
QELoginDialog . . . . .	177
QENumericEditManager . . . . .	180

QEPeiodic . . . . .	181
QEPeiodicComponentData . . . . .	187
QEPeiodicTaskMenu . . . . .	187
QEPeiodicTaskMenuFactory . . . . .	187
QEPlot . . . . .	188
QEPVNameLists . . . . .	209
QEPvPropertiesManager . . . . .	211
QERecipe . . . . .	226
QERecordFieldName . . . . .	228
QERecordSpec . . . . .	229
QERecordSpecList . . . . .	229
QEScript . . . . .	229
QEShape . . . . .	235
QESlider . . . . .	248
QESpinBox . . . . .	253
QEStripChartAdjustPVDialog . . . . .	260
QEStripChartContextMenu . . . . .	260
QEStripChartItem . . . . .	261
QEStripChartNames . . . . .	262
QEStripChartPushButtonSpecifications . . . . .	263
QEStripChartRangeDialog . . . . .	263
QEStripChartState . . . . .	263
QEStripChartStateList . . . . .	264
QEStripChartStatistics . . . . .	264
QEStripChartTimeDialog . . . . .	265
QEStripChartToolBar . . . . .	265
QESubstitutedLabel . . . . .	266
recording . . . . .	267
imageDisplayProperties::rgbPixel . . . . .	267
screenSelectDialog . . . . .	268
selectMenu . . . . .	268
trace . . . . .	268
userInfoStruct . . . . .	269
QEPeiodic::userInfoStructArray . . . . .	269
ValueScaling . . . . .	269
zoomMenu . . . . .	271



## Chapter 8

# Class Index

### 8.1 Class List

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

<a href="#">_CopyPaste</a>	27
<a href="#">_Field</a>	27
<a href="#">_Item</a>	28
<a href="#">_QDialogItem</a>	29
<a href="#">_QPushButtonGroup</a>	29
<a href="#">_QTableWidgetFileBrowser</a>	29
<a href="#">_QTableWidgetLog</a>	30
<a href="#">_QTableWidgetScript</a>	30
<a href="#">areaInfo</a>	30
<a href="#">QEAAnalogIndicator::Band</a>	31
<a href="#">QEAAnalogIndicator::BandList</a>	31
<a href="#">QEPeriodic::elementInfoStruct</a>	31
<a href="#">FFBuffer</a>	32
<a href="#">FFThread</a>	32
<a href="#">flipRotateMenu</a>	33
<a href="#">fullScreenWindow</a>	33
<a href="#">histogram</a>	34
<a href="#">histogramScroll</a>	34
<a href="#">historicImage</a>	34
<a href="#">imageContextMenu</a>	35
<a href="#">imageDisplayProperties</a>	36
<a href="#">imageInfo</a>	37
<a href="#">imageMarkup</a>	38
<a href="#">imageUpdateIndicator</a>	40
<a href="#">loginWidget</a>	40
<a href="#">markupCrosshair1</a>	40
<a href="#">markupCrosshair2</a>	41
<a href="#">markupDisplayMenu</a>	42
<a href="#">markupEllipse</a>	42

<a href="#">markupHLine</a>	43
<a href="#">markupItem</a>	44
<a href="#">markupLine</a>	46
<a href="#">markupRegion</a>	47
<a href="#">markupText</a>	48
<a href="#">markupVLine</a>	48
<a href="#">mpegSource</a>	49
<a href="#">mpegSourceObject</a>	50
<a href="#">QEStripChartToolBar::OwnWidgets</a>	51
<a href="#">PeriodicDialog</a>	51
<a href="#">PeriodicElementSetupForm</a>	51
<a href="#">PeriodicSetupDialog</a>	52
<a href="#">playbackTimer</a>	52
<a href="#">pointInfo</a>	52
<a href="#">profilePlot</a>	53
<a href="#">QBitStatus</a>	53
<a href="#">QEAAnalogIndicator</a>	55
<a href="#">QEAAnalogProgressBar</a>	61
<a href="#">QEBitStatus</a>	69
<a href="#">QECheckBox</a>	74
<a href="#">QECheckBoxManager</a>	89
<a href="#">QEComboBox</a>	89
<a href="#">QEConfiguredLayout</a>	94
<a href="#">QEConfiguredLayoutManager</a>	98
<a href="#">QEFileBrowser</a>	98
<a href="#">QEForm</a>	104
<a href="#">QEFrame</a>	105
<a href="#">QEGenericButton</a>	108
<a href="#">QEGenericEdit</a>	110
<a href="#">QEGroupBox</a>	117
<a href="#">QEImage</a>	120
<a href="#">QEImageMarkupThickness</a>	154
<a href="#">QEImageOptionsDialog</a>	154
<a href="#">QELabel</a>	155
<a href="#">QELineEdit</a>	165
<a href="#">QELineEditManager</a>	170
<a href="#">QELink</a>	170
<a href="#">QELog</a>	172
<a href="#">QELogin</a>	176
<a href="#">QELoginDialog</a>	177
<a href="#">QENumericEdit</a> (The <a href="#">QENumericEdit</a> class This class is similar to <a href="#">QELineEdit</a> (both of which are derived from <a href="#">QLineEdit</a> ). However this class is tailored specficcially for editing numerical values )	177
<a href="#">QENumericEditManager</a>	180
<a href="#">QEPeriodic</a>	181
<a href="#">QEPeriodicComponentData</a>	187
<a href="#">QEPeriodicTaskMenu</a>	187
<a href="#">QEPeriodicTaskMenuFactory</a>	187
<a href="#">QEPlot</a>	188
<a href="#">QEPushButton</a>	194

QEPVNameLists	209
QEPvProperties	209
QEPvPropertiesManager	211
QERadioButton	211
QERecipe	226
QERecordFieldName	228
QERecordSpec	229
QERecordSpecList	229
QEScript	229
QEShape	235
QESlider	248
QESpinBox	253
QEStripChart	257
QEStripChartAdjustPVDDialog	260
QEStripChartContextMenu	260
QEStripChartItem	261
QEStripChartNames	262
QEStripChartPushButtonSpecifications	263
QEStripChartRangeDialog	263
QEStripChartState	263
QEStripChartStateList	264
QEStripChartStatistics	264
QEStripChartTimeDialog	265
QEStripChartToolBar (This class holds all the StripChart tool bar widgets )	265
QESubstitutedLabel	266
recording	267
imageDisplayProperties::rgbPixel	267
screenSelectDialog	268
selectMenu	268
trace	268
userInfoStruct	269
QEPeriodic::userInfoStructArray	269
ValueScaling	269
VideoWidget	270
zoomMenu	271





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

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

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

### 9.2 `_Field` Class Reference

#### Public Member Functions

- QEWidget \* **`getWidget`** ()

- void **setWidget** (QString \*pValue)
- QString **getName** ()
- void **setName** (QString pValue)
- QString **getProcessVariable** ()
- void **setProcessVariable** (QString pValue)
- void **setJoin** (bool pValue)
- bool **getJoin** ()
- int **getType** ()
- void **setType** (int pValue)
- QString **getGroup** ()
- void **setGroup** (QString pValue)
- QString **getVisible** ()
- void **setVisible** (QString pValue)
- QString **getEditable** ()
- void **setEditable** (QString pValue)
- bool **getVisibility** ()
- void **setVisibility** (bool pValue)

### Public Attributes

- QEWidget \* **qeWidget**

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

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

## 9.3 \_Item Class Reference

### Public Member Functions

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

### Public Attributes

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

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

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

## 9.4 `_QDialogItem` Class Reference

### Public Member Functions

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

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

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

## 9.5 `_QPushButtonGroup` Class Reference

### Public Slots

- void **`buttonGroupClicked`** ()

### Public Member Functions

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

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

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

## 9.6 `_QTableWidgetFileBrowser` Class Reference

### Public Member Functions

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

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

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

## 9.7 \_QTableWidgetLog Class Reference

### Public Member Functions

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

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

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

## 9.8 \_QTableWidgetScript Class Reference

### Public Member Functions

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

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

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

## 9.9 arealInfo Class Reference

### Public Member Functions

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

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

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

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

## 9.10 QEAnalogIndicator::Band Struct Reference

### Public Attributes

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

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

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

## 9.11 QEAnalogIndicator::BandList Class Reference

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

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

## 9.12 QEPeriodic::elementInfoStruct Struct Reference

### Public Attributes

- unsigned int **number**
- double **atomicWeight**
- QString **name**
- QString **symbol**
- double **meltingPoint**

- double **boilingPoint**
- double **density**
- unsigned int **group**
- double **ionizationEnergy**
- unsigned int **tableRow**
- unsigned int **tableCol**

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

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

## 9.13 FFBuffer Class Reference

### Public Member Functions

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

### Public Attributes

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

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

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

## 9.14 FFThread Class Reference

### Public Slots

- void **stopGracefully** ()

### Signals

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

### Public Member Functions

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

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

- /tmp/epicsqt/trunk/framework/widgets/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\_VERT**,  
**ICM\_ENABLE\_HOZ**,  
**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\_HSLICE**, **ICM\_SELECT\_**  
**VSlice**,  
**ICM\_SELECT\_AREA1**, **ICM\_SELECT\_AREA2**, **ICM\_SELECT\_AREA3**, **ICM\_**  
**SELECT\_AREA4**,  
**ICM\_SELECT\_PROFILE**, **ICM\_SELECT\_TARGET**, **ICM\_SELECT\_BEAM**, **ICM\_**  
**CLEAR\_MARKUP**,  
**ICM\_THICKNESS\_ONE\_MARKUP**, **ICM\_THICKNESS\_SELECT\_MARKUP**, **ICM\_**  
**COPY\_PLOT\_DATA**, **ICM\_FULL\_SCREEN**,  
**ICM\_DISPLAY\_HSLICE**, **ICM\_DISPLAY\_VSLICE**, **ICM\_DISPLAY\_AREA1**, **ICM\_**  
**DISPLAY\_AREA2**,  
**ICM\_DISPLAY\_AREA3**, **ICM\_DISPLAY\_AREA4**, **ICM\_DISPLAY\_PROFILE**, **ICM\_**  
**DISPLAY\_TARGET**,  
**ICM\_DISPLAY\_BEAM**, **ICM\_DISPLAY\_TIMESTAMP**, **ICM\_DISPLAY\_ELLIPSE**,  
**ICM\_OPTIONS** }

### Public Member Functions

- **imageContextMenu** (QWidget \*parent=0)
- void **getContextMenuOption** (const QPoint &, imageContextMenuOptions \*option, bool \*checked)
- void **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 bit-Depth, unsigned int binsIn[HISTOGRAM\_BINS], [rgbPixel](#) pixelLookup[256])
- 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**
- bool **statisticsSet**

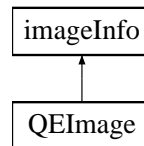
- [rgbPixel](#) \* **pixelLookup**
- QLabel \* **histXLabel**

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

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

## 9.22 imageInfo Class Reference

Inheritance diagram for imageInfo:



### Public Member Functions

- void **showInfo** (bool show)
- QLayout \* **getInfoWidget** ()
- void **infoShow** (const bool show)
- void **infoUpdateTarget** ()
- void **infoUpdateTarget** (const int x, const int y)
- void **infoUpdateBeam** ()
- void **infoUpdateBeam** (const int x, const int y)
- void **infoUpdateVertProfile** ()
- void **infoUpdateVertProfile** (const int x, const unsigned int thickness)
- void **infoUpdateHozProfile** ()
- void **infoUpdateHozProfile** (const int y, const unsigned int thickness)
- void **infoUpdateProfile** ()
- void **infoUpdateProfile** (const QPoint start, const QPoint end, const unsigned int thickness)
- void **infoUpdateRegion** (const unsigned int region)
- void **infoUpdateRegion** (const unsigned int region, const int x1, const int y1, const int x2, const int y2)
- void **infoUpdatePixel** ()
- void **infoUpdatePixel** (const QPoint pos, int value)
- void **infoUpdateZoom** ()
- void **infoUpdateZoom** (int value)
- void **infoUpdatePaused** ()
- void **infoUpdatePaused** (bool paused)
- void **setBriefInfoArea** (const bool briefIn)

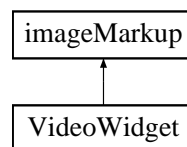
- bool **getBriefInfoArea** ()
- void **freshImage** (QDateTime &time)

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

- /tmp/epicsqt/trunk/framework/widgets/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\_H\_SLICE**, **MARKUP\_ID\_V\_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 displayMarkups)
- void **markupHProfileChange** (int y, bool displayMarkups)
- void **markupVProfileChange** (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 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

## 9.25 loginWidget Class Reference

### Public Member Functions

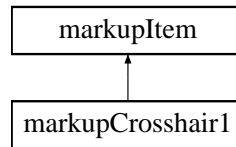
- **loginWidget** (QELogin \*ownerIn)
- userLevelTypes::userLevels **getUserType** ()
- QString **getPassword** ()
- void **clearPassword** ()

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

## 9.26 markupCrosshair1 Class Reference

Inheritance diagram for markupCrosshair1:



### Public Member Functions

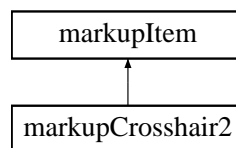
- **markupCrosshair1** ([imageMarkup](#) \*ownerIn, const bool interactiveIn, const bool reportOnMoveIn, const QString legendIn)
- void **startDrawing** (const QPoint pos)
- void **setArea** ()
- void **drawMarkup** (QPainter &p)
- void **moveTo** (const QPoint pos)
- bool **isOver** (const QPoint point, 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

## 9.27 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

## 9.28 markupDisplayMenu Class Reference

### Public Member Functions

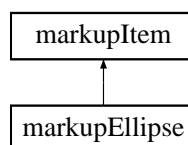
- **markupDisplayMenu** (QWidget \*parent=0)
- void **setDisplay** (imageContextMenu::imageContextMenuOptions option, bool state)
- void **setItemText** (imageContextMenu::imageContextMenuOptions option, QString title)
- bool **isDisplayed** (imageContextMenu::imageContextMenuOptions option)
- void **enable**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

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

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

## 9.29 markupEllipse Class Reference

Inheritance diagram for markupEllipse:





### Public Member Functions

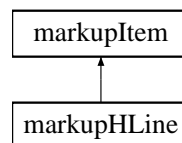
- **markupEllipse** ([imageMarkup](#) \*ownerIn, const bool interactiveIn, const bool reportOnMoveIn, const QString legendIn)
- void **startDrawing** (const QPoint pos)
- void **setArea** ()
- void **drawMarkup** (QPainter &p)
- void **moveTo** (const QPoint pos)
- bool **isOver** (const QPoint point, 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

## 9.30 markupHLine Class Reference

Inheritance diagram for markupHLine:



### Public Member Functions

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

### 9.30.1 Member Function Documentation

9.30.1.1 `void markupHLine::drawMarkup ( QPainter & p )` [virtual]

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

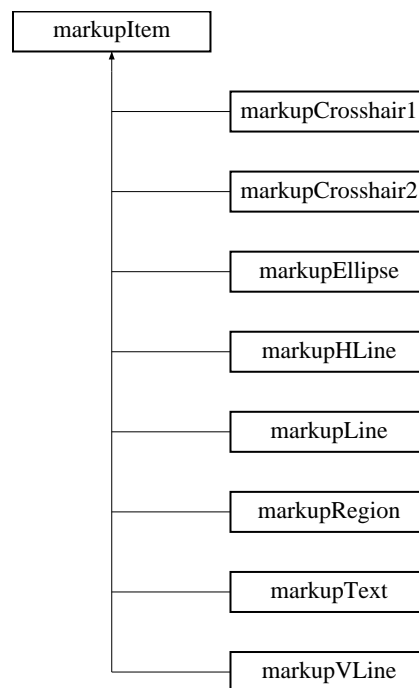
Implements [markupItem](#).

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

## 9.31 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)
- void **scale** (const double xScale, const double yScale, const double zoomScale)
- 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 QCursor **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

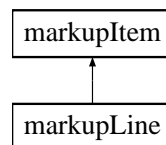
- markupHandles **activeHandle**
- [imageMarkup](#) \* **owner**
- unsigned int **thickness**
- unsigned int **maxThickness**

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

## 9.32 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** ()

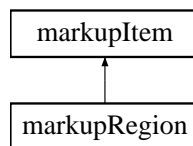
- void **drawMarkup** (QPainter &p)
- void **moveTo** (const QPoint pos)
- bool **isOver** (const QPoint point, QCursor \*cursor)
- QPoint **origin** ()
- QCursor **cursorForHandle** (const markupItem::markupHandles handle)
- QPoint **getPoint1** ()
- QPoint **getPoint2** ()
- QCursor **defaultCursor**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

## 9.33 markupRegion Class Reference

Inheritance diagram for markupRegion:



### Public Member Functions

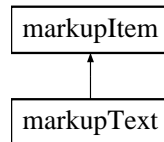
- **markupRegion** ([imageMarkup](#) \*ownerIn, const bool interactiveIn, const bool reportOnMoveIn, const QString legendIn)
- void **startDrawing** (const QPoint pos)
- void **setArea** ()
- void **drawMarkup** (QPainter &p)
- void **moveTo** (const QPoint pos)
- bool **isOver** (const QPoint point, 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

### 9.34 markupText Class Reference

Inheritance diagram for markupText:



#### Public Member Functions

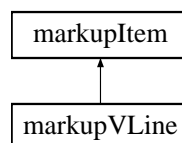
- **markupText** ([imageMarkup](#) \*ownerIn, const bool interactiveIn, const bool reportOnMoveIn, const QString legendIn)
- void **setText** (QString textIn)
- void **startDrawing** (const QPoint pos)
- void **setArea** ()
- void **drawMarkup** (QPainter &p)
- void **moveTo** (const QPoint pos)
- bool **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

### 9.35 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** ()
- QCursor **cursorForHandle** (const markupItem::markupHandles handle)
- QPoint **getPoint1** ()
- QPoint **getPoint2** ()
- QCursor **defaultCursor** ()
- void **nonInteractiveUpdate** (QPoint p1, QPoint p2)

### 9.35.1 Member Function Documentation

9.35.1.1 void markupVLine::drawMarkup ( QPainter & p ) [virtual]

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

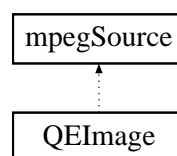
Implements [markupItem](#).

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

## 9.36 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** ()

### 9.36.1 Member Function Documentation

#### 9.36.1.1 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

## 9.37 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



## 9.38 QEStripChartToolBar::OwnWidgets Class Reference

### Public Member Functions

- **OwnWidgets** (QEStripChartToolBar \*parent)

### Public Attributes

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

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

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

## 9.39 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

## 9.40 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

## 9.41 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

## 9.42 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

## 9.43 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

## 9.44 profilePlot Class Reference

### Public Types

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

### Public Member Functions

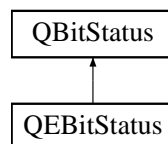
- **profilePlot** (plotDirections plotDirectionIn)
- void **setProfile** (QVector< QPointF > \*profile, double minX, double maxX, double minY, double maxY, QString title, QPoint start, QPoint end, unsigned int thicknessIn)
- void **clearProfile** ()

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

## 9.45 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**

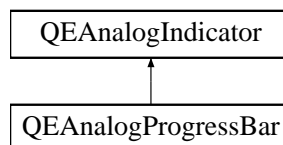
The documentation for this class was generated from the following files:

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

## 9.46 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.*

### 9.46.1 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.

### 9.46.2 Member Enumeration Documentation

#### 9.46.2.1 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)



### 9.46.2.2 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.

### 9.46.3 Property Documentation

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

Background colour

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

Border colour

#### 9.46.3.3 int QEAnalogIndicator::centreAngle [read, write]

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

#### 9.46.3.4 QColor QEAnalogIndicator::fontColour [read, write]

Font colour

#### 9.46.3.5 QColor QEAnalogIndicator::foregroundColour [read, write]

Foreground colour

#### 9.46.3.6 bool QEAnalogIndicator::logScale [read, write]

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

#### 9.46.3.7 int QEAnalogIndicator::logScaleInterval [read, write]

Log scale interval.

**9.46.3.8** `double QEAnalogIndicator::majorInterval` [read, write]

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

**9.46.3.9** `double QEAnalogIndicator::maximum` [read, write]

Maximum indicated value.

**9.46.3.10** `double QEAnalogIndicator::minimum` [read, write]

Minimum indicated value.

**9.46.3.11** `double QEAnalogIndicator::minorInterval` [read, write]

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

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

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

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

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

**9.46.3.14** `bool QEAnalogIndicator::showScale` [read, write]

If set, show the scale

**9.46.3.15** `bool QEAnalogIndicator::showText` [read, write]

If set, show textual representation of value on the indicator

**9.46.3.16** `int QEAnalogIndicator::spanAngle` [read, write]

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

**9.46.3.17** `double QEAnalogIndicator::value` [read, write]

Current indicated value.

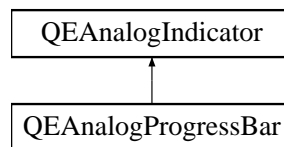
Reimplemented in [QEAnalogProgressBar](#).

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

## 9.47 QEAnalogProgressBar Class Reference

Inheritance diagram for QEAnalogProgressBar:



### Public Types

- enum **UserLevels** { **User** = userLevelTypes::USERLEVEL\_USER, **Scientist** = userLevelTypes::USERLEVEL\_SCIENTIST, **Engineer** = userLevelTypes::USERLEVEL\_ENGINEER }
- enum **AlarmSeverityDisplayModes** { **foreground**, **background** }
- enum **Formats** {  
**Default** = QStringFormatting::FORMAT\_DEFAULT, **Floating** = QStringFormatting::FORMAT\_FLOATING, **Integer** = QStringFormatting::FORMAT\_INTEGER, **UnsignedInteger** = QStringFormatting::FORMAT\_UNSIGNEDINTEGER,  
**Time** = QStringFormatting::FORMAT\_TIME, **LocalEnumeration** = QStringFormatting::FORMAT\_LOCAL\_ENUMERATE }
- enum **Notations** { **Fixed** = QStringFormatting::NOTATION\_FIXED, **Scientific** = QStringFormatting::NOTATION\_SCIENTIFIC, **Automatic** = QStringFormatting::NOTATION\_AUTOMATIC }
- enum **ArrayActions** { **Append** = QStringFormatting::APPEND, **Ascii** = QStringFormatting::ASCII, **Index** = QStringFormatting::INDEX }

### Signals

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

### Public Member Functions

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

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

- [UserLevels](#) [getUserLevelEnabledProperty](#) ()  
Access function for [userLevelEnabled](#) property - refer to [userLevelEnabled](#) property for details.
- void [setUserLevelEnabledProperty](#) ([UserLevels](#) level)  
Access function for [userLevelEnabled](#) property - refer to [userLevelEnabled](#) property for details.
- void [setFormatProperty](#) ([Formats](#) format)  
Access function for [format](#) property - refer to [format](#) property for details.
- [Formats](#) [getFormatProperty](#) ()  
Access function for [format](#) property - refer to [format](#) property for details.
- void [setNotationProperty](#) ([Notations](#) notation)  
Access function for [notation](#) property - refer to [notation](#) property for details.
- [Notations](#) [getNotationProperty](#) ()  
Access function for [notation](#) property - refer to [notation](#) property for details.
- void [setArrayActionProperty](#) ([ArrayActions](#) arrayAction)  
Access function for [arrayAction](#) property - refer to [arrayAction](#) property for details.
- [ArrayActions](#) [getArrayActionProperty](#) ()  
Access function for [arrayAction](#) property - refer to [arrayAction](#) property for details.
- [QEAnalogProgressBar](#) (QWidget \*parent=0)
- [QEAnalogProgressBar](#) (const QString &variableName, QWidget \*parent=0)
- virtual [~QEAnalogProgressBar](#) ()  
Destruction.
- void [setUseDbDisplayLimits](#) (bool useDbDisplayLimitsIn)  
Access function for [useDbDisplayLimits](#) property - refer to [useDbDisplayLimits](#) property for details.
- bool [getUseDbDisplayLimits](#) ()  
Access function for [useDbDisplayLimits](#) property - refer to [useDbDisplayLimits](#) property for details.
- void [setAlarmSeverityDisplayMode](#) ([AlarmSeverityDisplayModes](#) value)  
Access function for [#AlarmSeverityDisplayModes](#) property - refer to [#AlarmSeverityDisplayModes](#) property for details.
- [AlarmSeverityDisplayModes](#) [getAlarmSeverityDisplayMode](#) ()  
Access function for [#AlarmSeverityDisplayModes](#) property - refer to [#AlarmSeverityDisplayModes](#) property for details.

## Protected Member Functions

- QString [getTextImage](#) ()
- [BandList](#) [getBandList](#) ()
- void [establishConnection](#) (unsigned int variableIndex)
- void [stringFormattingChange](#) ()
- void [dragEnterEvent](#) (QDragEnterEvent \*event)
- void [dropEvent](#) (QDropEvent \*event)

- void **mousePressEvent** (QMouseEvent \*event)
- void **setDrop** (QVariant drop)
- QVariant **getDrop** ()
- QString **copyVariable** ()
- QVariant **copyData** ()

### Protected Attributes

- QEFloatingFormatting **floatingFormatting**

### Properties

- QString [variable](#)
- QString [variableSubstitutions](#)
- bool [variableAsToolTip](#)
- bool [allowDrop](#)
- bool [visible](#)
- unsigned [int](#)
- QString [userLevelUserStyle](#)
- QString [userLevelScientistStyle](#)
- QString [userLevelEngineerStyle](#)
- [UserLevels](#) [userLevelVisibility](#)
- [UserLevels](#) [userLevelEnabled](#)
- bool [displayAlarmState](#)
- AlarmSeverityDisplayModes [alarmSeverityDisplayMode](#)
- 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](#)
- [Notations](#) [notation](#)
- [ArrayActions](#) [arrayAction](#)

### 9.47.1 Member Enumeration Documentation

#### 9.47.1.1 enum `QEAnalogProgressBar::ArrayActions`

User friendly enumerations for arrayAction property - refer to `QStringFormatting::arrayActions` for details.

**Enumerator:**

***Append*** Refer to `QStringFormatting::APPEND` for details.

***Ascii*** Refer to `QStringFormatting::ASCII` for details.

***Index*** Refer to `QStringFormatting::INDEX` for details.

#### 9.47.1.2 enum `QEAnalogProgressBar::Formats`

User friendly enumerations for format property - refer to `QStringFormatting::formats` for details.

**Enumerator:**

***Default*** Format as best appropriate for the data type.

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

***Integer*** Format as an integer.

***UnsignedInteger*** Format as an unsigned integer.

***Time*** Format as a time.

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

#### 9.47.1.3 enum `QEAnalogProgressBar::Notations`

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

**Enumerator:**

***Fixed*** Refer to `QStringFormatting::NOTATION_FIXED` for details.

***Scientific*** Refer to `QStringFormatting::NOTATION_SCIENTIFIC` for details.

***Automatic*** Refer to `QStringFormatting::NOTATION_AUTOMATIC` for details.

#### 9.47.1.4 enum `QEAnalogProgressBar::UserLevels`

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

**Enumerator:**

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

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

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

**9.47.2 Constructor & Destructor Documentation****9.47.2.1 QEAnalogProgressBar::QEAnalogProgressBar ( QWidget \* parent = 0 )**

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

**9.47.2.2 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.

**9.47.3 Member Function Documentation****9.47.3.1 void QEAnalogProgressBar::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.

**9.47.4 Property Documentation****9.47.4.1 bool QEAnalogProgressBar::addUnits [read, write]**

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

**9.47.4.2 AlarmSeverityDisplayModes QEAnalogProgressBar::alarmSeverityDisplayMode [read, write]**

Visualise the EPICS alarm severity

**9.47.4.3 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.

#### 9.47.4.4 **ArrayActions** `QAnalogProgressBar::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'.

#### 9.47.4.5 **bool** `QAnalogProgressBar::displayAlarmState` [read, write]

If set (default) widget will indicate the alarm state of any variable data 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.

#### 9.47.4.6 **Formats** `QAnalogProgressBar::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).

#### 9.47.4.7 **unsigned** `QAnalogProgressBar::int` [read, write]

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

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

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

#### 9.47.4.8 **bool** `QAnalogProgressBar::leadingZero` [read, write]

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



**9.47.4.9 QString QEAnalogProgressBar::localEnumeration** [read, write]

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

Format is:

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

Where: < Less than <= Less than or equal = Equal (default if no operator specified)

>= Greather than or equal > Greater than Always match (used to specify default text)

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

may be included in a string to indicate a line break

Examples are:

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

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

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

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

Notation used for numerical formatting. Default is fixed.

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

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

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

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

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

Use the EPICS database display limits

#### 9.47.4.14 `bool QEAnalogProgressBar::useDbPrecision` [read, write]

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

#### 9.47.4.15 `UserLevels QEAnalogProgressBar::userLevelEnabled` [read, write]

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

#### 9.47.4.16 `QString QEAnalogProgressBar::userLevelEngineerStyle` [read, write]

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

#### 9.47.4.17 `QString QEAnalogProgressBar::userLevelScientistStyle` [read, write]

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

#### 9.47.4.18 `QString QEAnalogProgressBar::userLevelUserStyle` [read, write]

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

#### 9.47.4.19 `UserLevels QEAnalogProgressBar::userLevelVisibility` [read, write]

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

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

**9.47.4.20** `int QEAnalogProgressBar::value` [read, write]

Current indicated value.

Reimplemented from [QEAnalogIndicator](#).

**9.47.4.21** `QString QEAnalogProgressBar::variable` [read, write]

EPICS variable name (CA PV)

**9.47.4.22** `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.

**9.47.4.23** `QString QEAnalogProgressBar::variableSubstitutions` [read, write]

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

**9.47.4.24** `bool QEAnalogProgressBar::visible` [read, write]

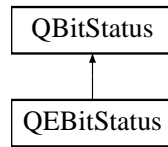
Display the widget. Default is true. Setting this property false is usefull if widget is only used to provide a signal - for example, when supplying data to a [QELink](#) widget. Note, when false the widget will still be visible in Qt Designer.

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

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

## 9.48 QEBitStatus Class Reference

Inheritance diagram for QEBitStatus:



## Public Types

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

## Signals

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

## 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.*
- QEBitStatus** (QWidget \*parent=0)
- QEBitStatus** (const QString &variableName, QWidget \*parent=0)
- void **setVariableNameAndSubstitutions** (QString variableNameIn, QString variableNameSubstitutionsIn, unsigned int variableIndex)

## Protected Member Functions

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

### Protected Attributes

- QEIntegerFormatting **integerFormatting**

### Properties

- QString [variable](#)
- QString [variableSubstitutions](#)
- bool [variableAsToolTip](#)
- bool [allowDrop](#)
- bool [visible](#)
- unsigned [int](#)
- QString [userLevelUserStyle](#)
- QString [userLevelScientistStyle](#)
- QString [userLevelEngineerStyle](#)
- [UserLevels](#) [userLevelVisibility](#)
- [UserLevels](#) [userLevelEnabled](#)
- bool [displayAlarmState](#)
- double **value**
- bool **isActive**

## 9.48.1 Member Enumeration Documentation

### 9.48.1.1 enum QEBitStatus::UserLevels

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

#### Enumerator:

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

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

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

## 9.48.2 Member Function Documentation

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

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

### 9.48.3 Property Documentation

#### 9.48.3.1 `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.

#### 9.48.3.2 `bool QEBitStatus::displayAlarmState` [read, write]

If set (default) widget will indicate the alarm state of any variable data 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.

#### 9.48.3.3 `unsigned QEBitStatus::int` [read, write]

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

#### 9.48.3.4 `UserLevels QEBitStatus::userLevelEnabled` [read, write]

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

#### 9.48.3.5 `QString QEBitStatus::userLevelEngineerStyle` [read, write]

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

#### 9.48.3.6 `QString QEBitStatus::userLevelScientistStyle` [read, write]

Style Sheet string to be applied when the widget is displayed in 'Scientist' mode. Default is an empty string. The syntax is the standard Qt Style Sheet syntax. For example, 'background-color: red' This Style Sheet string will be applied by the styleManager

class. Refer to the `styleManager` class for details about how this Style Sheet string will be merged with any pre-existing Style Sheet string and any Style Sheet strings generated during the display of data.

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

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

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

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

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

EPICS variable name (CA PV)

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

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

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

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

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

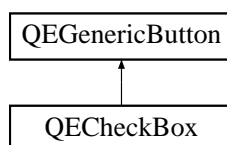
Display the widget. Default is true. Setting this property false is usefull if widget is only used to provide a signal - for example, when supplying data to a [QELink](#) widget. Note, when false the widget will still be visible in Qt Designer.

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

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

## 9.49 QECheckBox Class Reference

Inheritance diagram for QECheckBox:



### Public Types

- enum [UserLevels](#) { [User](#) = userLevelTypes::USERLEVEL\_USER, [Scientist](#) = userLevelTypes::USERLEVEL\_SCIENTIST, [Engineer](#) = userLevelTypes::USERLEVEL\_ENGINEER }
- enum [Formats](#) {  
[Default](#) = QQStringFormatting::FORMAT\_DEFAULT, [Floating](#) = QQStringFormatting::FORMAT\_FLOATING, [Integer](#) = QQStringFormatting::FORMAT\_INTEGER, [UnsignedInteger](#) = QQStringFormatting::FORMAT\_UNSIGNEDINTEGER,  
[Time](#) = QQStringFormatting::FORMAT\_TIME, [LocalEnumeration](#) = QQStringFormatting::FORMAT\_LOCAL\_ENUMERATE }
- enum [Notations](#) { [Fixed](#) = QQStringFormatting::NOTATION\_FIXED, [Scientific](#) = QQStringFormatting::NOTATION\_SCIENTIFIC, [Automatic](#) = QQStringFormatting::NOTATION\_AUTOMATIC }
- enum [ArrayActions](#) { [Append](#) = QQStringFormatting::APPEND, [Ascii](#) = QQStringFormatting::ASCII, [Index](#) = QQStringFormatting::INDEX }
- enum [UpdateOptions](#) { [Text](#) = QEGenericButton::UPDATE\_TEXT, [Icon](#) = QEGenericButton::UPDATE\_ICON, [TextAndIcon](#) = QEGenericButton::UPDATE\_TEXT\_AND\_ICON, [State](#) = QEGenericButton::UPDATE\_STATE }  
*User friendly enumerations for updateOption property - refer to QEGenericButton::updateOptions for details.*
- enum [ProgramStartupOptionNames](#) { [None](#) = applicationLauncher::PSO\_NONE, [Terminal](#) = applicationLauncher::PSO\_TERMINAL, [LogOutput](#) = applicationLauncher::PSO\_LOGOUTPUT, [StdOutput](#) = applicationLauncher::PSO\_STDOUTPUT }
- enum [CreationOptionNames](#) {  
[Open](#) = QEActionRequests::OptionOpen, [NewTab](#) = QEActionRequests::OptionNewTab, [NewWindow](#) = QEActionRequests::OptionNewWindow, [DockTop](#) = QEActionRequests::OptionTopDockWindow,  
[DockBottom](#) = QEActionRequests::OptionBottomDockWindow, [DockLeft](#) = QEActionRequests::OptionLeftDockWindow, [DockRight](#) = QEActionRequests::OptionRightDockWindow, [DockTopTabbed](#) = QEActionRequests::OptionTopDockWindowTabbed,



`DockBottomTabbed` = `QActionRequests::OptionBottomDockWindowTabbed`, `DockLeftTabbed` = `QActionRequests::OptionLeftDockWindowTabbed`, `DockRightTabbed` = `QActionRequests::OptionRightDockWindowTabbed`, `DockFloating` = `QActionRequests::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 `QActionRequests` &request)

## 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 `QActionRequests` &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)
- `UserLevels` `getUserLevelVisibilityProperty` ()  
*Access function for `userLevelVisibility` property - refer to `userLevelVisibility` property for details.*
- void `setUserLevelVisibilityProperty` (`UserLevels` level)  
*Access function for `userLevelVisibility` property - refer to `userLevelVisibility` property for details.*
- `UserLevels` `getUserLevelEnabledProperty` ()  
*Access function for `userLevelEnabled` property - refer to `userLevelEnabled` property for details.*
- void `setUserLevelEnabledProperty` (`UserLevels` level)  
*Access function for `userLevelEnabled` property - refer to `userLevelEnabled` property for details.*
- void `setFormatProperty` (`Formats` format)  
*Access function for `format` property - refer to `format` property for details.*
- `Formats` `getFormatProperty` ()

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

- void [setNotationProperty](#) ([Notations](#) notation)

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

- [Notations](#) [getNotationProperty](#) ()

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

- void [setArrayActionProperty](#) ([ArrayActions](#) arrayAction)

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

- [ArrayActions](#) [getArrayActionProperty](#) ()

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

## Properties

- QString [variable](#)
- QString [variableSubstitutions](#)
- bool [subscribe](#)
- bool [variableAsToolTip](#)
- bool [allowDrop](#)
- bool [visible](#)
- unsigned [int](#)
- QString [userLevelUserStyle](#)
- QString [userLevelScientistStyle](#)
- QString [userLevelEngineerStyle](#)
- [UserLevels](#) [userLevelVisibility](#)
- [UserLevels](#) [userLevelEnabled](#)
- bool [displayAlarmState](#)
- int [precision](#)
- bool [useDbPrecision](#)
- bool [leadingZero](#)
- bool [trailingZeros](#)
- bool [addUnits](#)
- QString [localEnumeration](#)
- [Formats](#) [format](#)
- [Notations](#) [notation](#)
- [ArrayActions](#) [arrayAction](#)
- Qt::Alignment [alignment](#)
- [UpdateOptions](#) [updateOption](#)
- QPixmap [pixmap0](#)
- QPixmap [pixmap1](#)
- QPixmap [pixmap2](#)
- QPixmap [pixmap3](#)
- QPixmap [pixmap4](#)
- QPixmap [pixmap5](#)
- QPixmap [pixmap6](#)
- QPixmap [pixmap7](#)
- QString [password](#)

- bool [confirmAction](#)
- QString [confirmText](#)
- bool [writeOnPress](#)
- bool [writeOnRelease](#)
- bool [writeOnClick](#)
- QString [pressText](#)
- QString [releaseText](#)
- QString [clickText](#)
- QString [clickCheckedText](#)
- QString [labelText](#)
- QString [program](#)
- QStringList [arguments](#)
- [ProgramStartupOptionNames](#) [programStartupOption](#)
- QString [guiFile](#)
- [CreationOptionNames](#) [creationOption](#)
- QString [prioritySubstitutions](#)
- QString [customisationName](#)

### 9.49.1 Member Enumeration Documentation

#### 9.49.1.1 enum QCheckBox::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.

#### 9.49.1.2 enum QCheckBox::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.

#### 9.49.1.3 enum `QCheckBox::Formats`

User friendly enumerations for format property - refer to `QStringFormatting::formats` for details.

##### Enumerator:

***Default*** Format as best appropriate for the data type.

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

***Integer*** Format as an integer.

***UnsignedInteger*** Format as an unsigned integer.

***Time*** Format as a time.

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

#### 9.49.1.4 enum `QCheckBox::Notations`

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

##### Enumerator:

***Fixed*** Refer to `QStringFormatting::NOTATION_FIXED` for details.

***Scientific*** Refer to `QStringFormatting::NOTATION_SCIENTIFIC` for details.

***Automatic*** Refer to `QStringFormatting::NOTATION_AUTOMATIC` for details.

#### 9.49.1.5 enum `QCheckBox::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.

#### 9.49.1.6 enum QCheckBox::UpdateOptions

User friendly enumerations for updateOption property - refer to QGenericButton::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)

#### 9.49.1.7 enum QCheckBox::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.

## 9.49.2 Constructor & Destructor Documentation

### 9.49.2.1 QCheckBox::QCheckBox ( QWidget \* parent = 0 )

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

### 9.49.2.2 QCheckBox::QCheckBox ( 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.

### 9.49.3 Member Function Documentation

#### 9.49.3.1 void QCheckBox::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)

#### 9.49.3.2 void QCheckBox::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.

#### 9.49.3.3 void QCheckBox::pressed ( int *value* ) [signal]

Button has been Pressed. The value emitted is the integer interpretation of the press-Text property

#### 9.49.3.4 void QCheckBox::released ( int *value* ) [signal]

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

#### 9.49.3.5 void QCheckBox::requestAction ( const QActionRequests & *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.

### 9.49.4 Property Documentation

#### 9.49.4.1 bool QCheckBox::addUnits [read, write]

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

#### 9.49.4.2 Qt::Alignment QCheckBox::alignment [read, write]

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

#### 9.49.4.3 bool QCheckBox::allowDrop [read, write]

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

#### 9.49.4.4 QStringList QCheckBox::arguments [read, write]

Arguments for program specified in the 'program' property.

#### 9.49.4.5 ArrayActions QCheckBox::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'.

#### 9.49.4.6 QString QCheckBox::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](#).

#### 9.49.4.7 QString QCheckBox::clickText [read, write]

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

Reimplemented from [QEGenericButton](#).

#### 9.49.4.8 bool QCheckBox::confirmAction [read, write]

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

#### 9.49.4.9 QString QCheckBox::confirmText [read, write]

Text used to confirm action if confirmation dialog is presented

Reimplemented from [QEGenericButton](#).

#### 9.49.4.10 CreationOptionNames QCheckBox::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](#).

#### 9.49.4.11 QString QCheckBox::customisationName [read, write]

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

Reimplemented from [QEGenericButton](#).

#### 9.49.4.12 bool QCheckBox::displayAlarmState [read, write]

If set (default) widget will indicate the alarm state of any variable data 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.



#### 9.49.4.13 Formats QCheckBox::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).

#### 9.49.4.14 QString QCheckBox::guiFile [read, write]

File name of GUI to be presented on button click. File name can be absolute, relative to the path of the QForm in which the [QEPushButton](#) is located, relative to the any path in the path list published in the ContainerProfile class, or relative to the current path. See [QEWWidget::openQEFile\(\)](#) in [QEWWidget.cpp](#) for details.

#### 9.49.4.15 unsigned QCheckBox::int [read, write]

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

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

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

#### 9.49.4.16 QString QCheckBox::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](#).

#### 9.49.4.17 bool QCheckBox::leadingZero [read, write]

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

#### 9.49.4.18 QString QCheckBox::localEnumeration [read, write]

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

Format is:

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

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

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

may be included in a string to indicate a line break

Examples are:

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

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

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

#### 9.49.4.19 Notations QCheckBox::notation [read, write]

Notation used for numerical formatting. Default is fixed.

#### 9.49.4.20 QString QCheckBox::password [read, write]

Password user will need to enter before any action is taken

Reimplemented from [QGenericButton](#).

#### 9.49.4.21 QPixmap QCheckBox::pixmap0 [read, write]

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

#### 9.49.4.22 QPixmap QCheckBox::pixmap1 [read, write]

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

**9.49.4.23 QPixmap QCheckBox::pixmap2** [read, write]

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

**9.49.4.24 QPixmap QCheckBox::pixmap3** [read, write]

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

**9.49.4.25 QPixmap QCheckBox::pixmap4** [read, write]

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

**9.49.4.26 QPixmap QCheckBox::pixmap5** [read, write]

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

**9.49.4.27 QPixmap QCheckBox::pixmap6** [read, write]

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

**9.49.4.28 QPixmap QCheckBox::pixmap7** [read, write]

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

**9.49.4.29 int QCheckBox::precision** [read, write]

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

**9.49.4.30 QString QCheckBox::pressText** [read, write]

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

Reimplemented from [QEGenericButton](#).

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

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

Reimplemented from [QEGenericButton](#).

**9.49.4.32 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

**9.49.4.33 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.

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

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

Reimplemented from [QEGenericButton](#).

**9.49.4.35 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)

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

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

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

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

Reimplemented from [QEGenericButton](#).

**9.49.4.38** `bool QCheckBox::useDbPrecision` [read, write]

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

**9.49.4.39** `UserLevels QCheckBox::userLevelEnabled` [read, write]

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

**9.49.4.40** `QString QCheckBox::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.

**9.49.4.41** `QString QCheckBox::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.

**9.49.4.42** `QString QCheckBox::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.

**9.49.4.43** `UserLevels QCheckBox::userLevelVisibility` [read, write]

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

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

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

EPICS variable name (CA PV)

#### 9.49.4.45 `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.

#### 9.49.4.46 `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 are also used for other purposes.

#### 9.49.4.47 `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.

#### 9.49.4.48 `bool QECheckBox::writeOnClick` [read, write]

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

Reimplemented from [QEGenericButton](#).

#### 9.49.4.49 `bool QECheckBox::writeOnPress` [read, write]

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

Reimplemented from [QEGenericButton](#).

#### 9.49.4.50 `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

## 9.50 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

## 9.51 QEComboBox Class Reference

### Public Types

- enum **UserLevels** { **User** = userLevelTypes::USERLEVEL\_USER, **Scientist** = userLevelTypes::USERLEVEL\_SCIENTIST, **Engineer** = userLevelTypes::USERLEVEL\_ENGINEER }

### Signals

- void **dbValueChanged** (const qlonglong &out)
- void **userChange** (const QString &oldValue, const QString &newValue, const QString &lastValue)

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

## Public Member Functions

- **QEComboBox** (QWidget \*parent=0)
- **QEComboBox** (const QString &variableName, QWidget \*parent=0)
- void **setWriteOnChange** (bool writeOnChangeIn)
- bool **getWriteOnChange** ()
- void **setSubscribe** (bool subscribe)
- bool **getSubscribe** ()
- void **setUseDbEnumerations** (bool [useDbEnumerations](#))
- bool **getUseDbEnumerations** ()
- void **setLocalEnumerations** (const QString &localEnumerations)
- QString **getLocalEnumerations** ()
- [UserLevels](#) **getUserLevelVisibilityProperty** ()  
*Access function for [userLevelVisibility](#) property - refer to [userLevelVisibility](#) property for details.*
- void **setUserLevelVisibilityProperty** ([UserLevels](#) level)  
*Access function for [userLevelVisibility](#) property - refer to [userLevelVisibility](#) property for details.*
- [UserLevels](#) **getUserLevelEnabledProperty** ()  
*Access function for [userLevelEnabled](#) property - refer to [userLevelEnabled](#) property for details.*
- void **setUserLevelEnabledProperty** ([UserLevels](#) level)  
*Access function for [userLevelEnabled](#) property - refer to [userLevelEnabled](#) property for details.*

## Protected Member Functions

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

## Protected Attributes

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



## Properties

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

### 9.51.1 Member Enumeration Documentation

#### 9.51.1.1 enum QComboBox::UserLevels

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

#### Enumerator:

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

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

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

### 9.51.2 Member Function Documentation

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

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

### 9.51.3 Member Data Documentation

#### 9.51.3.1 bool QComboBox::useDbEnumerations [read, write, protected]

Use database enumerations - defaults to true

### 9.51.3.2 `bool QComboBox::writeOnChange` [read, write, protected]

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

## 9.51.4 Property Documentation

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

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

### 9.51.4.2 `bool QComboBox::displayAlarmState` [read, write]

If set (default) widget will indicate the alarm state of any variable data 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.

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

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

### 9.51.4.4 `QString QComboBox::localEnumeration` [read, write]

Enumerations values used when `useDbEnumerations` is false.

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

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

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

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

#### 9.51.4.7 QString QComboBox::userLevelEngineerStyle [read, write]

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

#### 9.51.4.8 QString QComboBox::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.

#### 9.51.4.9 QString QComboBox::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.

#### 9.51.4.10 UserLevels QComboBox::userLevelVisibility [read, write]

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

#### 9.51.4.11 QString QComboBox::variable [read, write]

EPICS variable name (CA PV)

#### 9.51.4.12 bool QComboBox::variableAsToolTip [read, write]

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

#### 9.51.4.13 QString QComboBox::variableSubstitutions [read, write]

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

#### 9.51.4.14 bool QComboBox::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/QComboBox/QComboBox.h
- /tmp/epicsqt/trunk/framework/widgets/QComboBox/QComboBox.cpp

## 9.52 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 }

### 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.*

### Public Attributes

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

### Protected Attributes

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

### Properties

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

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

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

### 9.52.1 Member Enumeration Documentation

#### 9.52.1.1 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.

### 9.52.2 Property Documentation

#### 9.52.2.1 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.

#### 9.52.2.2 bool `QEConfiguredLayout::displayAlarmState` [[read](#), [write](#)]

If set (default) widget will indicate the alarm state of any variable data 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.

#### 9.52.2.3 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.

#### 9.52.2.4 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 programatically through `setUserLevel()` Widgets that are always accessible should be visible at 'User'. Widgets that are only accessible to scientists managing the facility should be visible at 'Scientist'. Widgets that are only accessible to engineers maintaining the facility should be visible at 'Engineer'.

#### 9.52.2.5 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.

#### 9.52.2.6 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.

#### 9.52.2.7 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.

#### 9.52.2.8 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 programatically through `setUserLevel()` Widgets that are always visible should be visible at 'User'. Widgets that are only used by scientists managing the facility should be visible at 'Scientist'. Widgets that are only used by engineers maintaining the facility should be visible at 'Engineer'.

### 9.52.2.9 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.

### 9.52.2.10 bool QEConfiguredLayout::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/QEConfiguredLayout/QEConfiguredLayout.h
- /tmp/epicsqt/trunk/framework/widgets/QEConfiguredLayout/QEConfiguredLayout.cpp

## 9.53 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

## 9.54 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 }

## Signals

- void **selected** (QString pFilename)

## Public Member Functions

- **QFileBrowser** (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.

## Protected Attributes

- [QLineEdit](#) \* [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 [userLevelUserStyle](#)
- QString [userLevelScientistStyle](#)
- QString [userLevelEngineerStyle](#)
- [UserLevels](#) [userLevelVisibility](#)
- [UserLevels](#) [userLevelEnabled](#)
- bool [displayAlarmState](#)

### 9.54.1 Detailed Description

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

### 9.54.2 Member Enumeration Documentation

#### 9.54.2.1 enum QFileBrowser::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.

### 9.54.3 Member Function Documentation

#### 9.54.3.1 void QFileBrowser::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).

### 9.54.4 Property Documentation

#### 9.54.4.1 `bool QFileBrowser::allowDrop` [read, write]

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

#### 9.54.4.2 `bool QFileBrowser::displayAlarmState` [read, write]

If set (default) widget will indicate the alarm state of any variable data 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.

#### 9.54.4.3 `unsigned QFileBrowser::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.

#### 9.54.4.4 `UserLevels QFileBrowser::userLevelEnabled` [read, write]

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

#### 9.54.4.5 `QString QFileBrowser::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.

#### 9.54.4.6 `QString QFileBrowser::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.

#### 9.54.4.7 `QString QFileBrowser::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.

#### 9.54.4.8 `UserLevels QFileBrowser::userLevelVisibility` [read, write]

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

#### 9.54.4.9 `QString QFileBrowser::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.

#### 9.54.4.10 `bool QFileBrowser::variableAsToolTip` [read, write]

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

#### 9.54.4.11 `QString QFileBrowser::variableSubstitutions` [read, write]

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

#### 9.54.4.12 `bool QFileBrowser::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/QEFileBrowser/QEFileBrowser.h
- /tmp/epicsqt/trunk/framework/widgets/QEFileBrowser/QEFileBrowser.cpp

## 9.55 QEForm Class Reference

### Public Types

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

### Public Slots

- bool **readUiFile** ()

### Public Member Functions

- **QEForm** (QWidget \*parent=0)
- **QEForm** (const QString &uifileName, QWidget \*parent=0)
- void **commonInit** (const bool alertIfUINotFound, const bool loadManually)
- void **setQEGuiTitle** (const QString title)
- QString **getQEGuiTitle** ()
- QString **getFullFileName** ()
- QString **getUiFileName** ()
- void **setFileMonitoringIsEnabled** (bool fileMonitoringIsEnabled)
- bool **getFileMonitoringIsEnabled** ()
- void **setHandleGuiLaunchRequests** (bool handleGuiLaunchRequests)
- bool **getHandleGuiLaunchRequests** ()
- void **setResizeContents** (bool resizeContents)
- bool **getResizeContents** ()
- QString **getContainedFrameworkVersion** ()
- QString **getUniqueIdentifier** ()
- void **setUniqueIdentifier** (QString name)
- int **getDisconnectedCount** ()
- int **getConnectedCount** ()
- void **setUiFileNameProperty** (QString uiFileName)
- QString **getUiFileNameProperty** ()
- void **setVariableNameSubstitutionsProperty** (QString variableNameSubstitutions)
- QString **getVariableNameSubstitutionsProperty** ()
- MessageFilterOptions **getMessageFormFilter** ()
- void **setMessageFormFilter** (MessageFilterOptions messageFormFilter)
- MessageFilterOptions **getMessageSourceFilter** ()
- void **setMessageSourceFilter** (MessageFilterOptions messageSourceFilter)

### Protected Attributes

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

### Properties

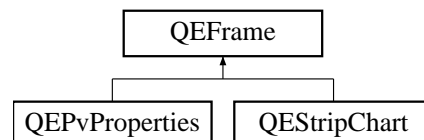
- QString **uiFile**
- QString **variableSubstitutions**
- unsigned int
- MessageFilterOptions **messageFormFilter**
- MessageFilterOptions **messageSourceFilter**

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

## 9.56 QEFrame Class Reference

Inheritance diagram for QEFrame:



### Public Types

- enum **UserLevels** { **User** = userLevelTypes::USERLEVEL\_USER, **Scientist** = userLevelTypes::USERLEVEL\_SCIENTIST, **Engineer** = userLevelTypes::USERLEVEL\_ENGINEER }

### 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.
- **QEFrame** (QWidget \*parent=0)
- QSize **sizeHint** () const

## Properties

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

## 9.56.1 Member Enumeration Documentation

### 9.56.1.1 enum QEFrame::UserLevels

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

#### Enumerator:

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

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

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

## 9.56.2 Property Documentation

### 9.56.2.1 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.



#### 9.56.2.2 `bool QEFrame::displayAlarmState` [read, write]

If set (default) widget will indicate the alarm state of any variable data 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.

#### 9.56.2.3 `unsigned QEFrame::int` [read, write]

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

#### 9.56.2.4 `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 programatically through `setUserLevel()` Widgets that are always accessible should be visible at 'User'. Widgets that are only accessible to scientists managing the facility should be visible at 'Scientist'. Widgets that are only accessible to engineers maintaining the facility should be visible at 'Engineer'.

#### 9.56.2.5 `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.

#### 9.56.2.6 `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.

#### 9.56.2.7 `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.

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

Lowest user level at which the widget is visible. Default is 'User'. Used when designing GUIs that display more and more detail according to the user mode. The user mode is set application wide through the [QELogin](#) widget, or 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'.

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

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

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

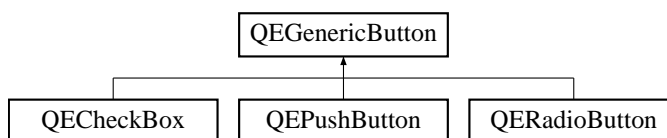
Display the widget. Default is true. Setting this property false is usefull if widget is only used to provide a signal - for example, when supplying data to a [QELink](#) widget. Note, when false the widget will still be visible in Qt Designer.

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

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

## 9.57 QEGenericButton Class Reference

Inheritance diagram for QEGenericButton:



### Public Types

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

## Public Member Functions

- **QEGenericButton** (QWidget \*owner)
- void **setSubscribe** (bool subscribe)
- bool **getSubscribe** ()
- void **setUpdateOption** (updateOptions updateOptionIn)
- updateOptions **getUpdateOption** ()
- void **setTextAlignment** (Qt::Alignment alignment)
- Qt::Alignment **getTextAlignment** ()
- void **setPassword** (QString password)
- QString **getPassword** ()
- void **setConfirmAction** (bool confirmRequiredIn)
- bool **getConfirmAction** ()
- void **setConfirmText** (QString confirmTextIn)
- QString **getConfirmText** ()
- void **setWriteOnPress** (bool writeOnPress)
- bool **getWriteOnPress** ()
- void **setWriteOnRelease** (bool writeOnRelease)
- bool **getWriteOnRelease** ()
- void **setWriteOnClick** (bool writeOnClick)
- bool **getWriteOnClick** ()
- void **setPressText** (QString pressText)
- QString **getPressText** ()
- void **setReleaseText** (QString releaseTextIn)
- QString **getReleaseText** ()
- void **setClickText** (QString clickTextIn)
- QString **getClickText** ()
- void **setClickCheckedText** (QString clickCheckedTextIn)
- QString **getClickCheckedText** ()
- void **setProgram** (QString program)
- QString **getProgram** ()
- void **setArguments** (QStringList arguments)
- QStringList **getArguments** ()
- void **setProgramStartupOption** (applicationLauncher::programStartupOptions programStartupOptionIn)
- applicationLauncher::programStartupOptions **getProgramStartupOption** ()
- void **setGuiName** (QString guiName)
- QString **getGuiName** ()
- void **setPrioritySubstitutions** (QString prioritySubstitutionsIn)
- QString **getPrioritySubstitutions** ()
- void **setCustomisationName** (QString customisationNameIn)
- QString **getCustomisationName** ()
- void **setCreationOption** (QEActionRequests::Options creationOption)
- QEActionRequests::Options **getCreationOption** ()
- void **setLabelTextProperty** (QString labelTextIn)
- QString **getLabelTextProperty** ()

### Protected Member Functions

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

### Protected Attributes

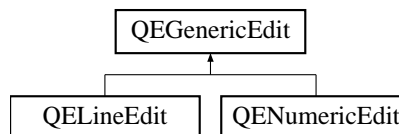
- applicationLauncher **programLauncher**

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

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

## 9.58 QEGenericEdit Class Reference

Inheritance diagram for QEGenericEdit:



### Public Types

- enum **UserLevels** { **User** = userLevelTypes::USERLEVEL\_USER, **Scientist** = userLevelTypes::USERLEVEL\_SCIENTIST, **Engineer** = userLevelTypes::USERLEVEL\_ENGINEER }

### Signals

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

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

- void [requestResend](#) ()

*Internal use only. Used when changing a property value to force a re-display to reflect the new property value.*

## Public Member Functions

- void [setVariableNameProperty](#) (QString variableName)  
*Access function for [variable](#) property - refer to [variable](#) property for details.*
- QString [getVariableNameProperty](#) ()  
*Access function for [variable](#) property - refer to [variable](#) property for details.*
- void [setVariableNameSubstitutionsProperty](#) (QString variableNameSubstitutions)  
  
*Access function for [variableSubstitutions](#) property - refer to [variableSubstitutions](#) property for details.*
- QString [getVariableNameSubstitutionsProperty](#) ()  
*Access function for [variableSubstitutions](#) property - refer to [variableSubstitutions](#) property for details.*
- [UserLevels](#) [getUserLevelVisibilityProperty](#) ()  
*Access function for [userLevelVisibility](#) property - refer to [userLevelVisibility](#) property for details.*
- void [setUserLevelVisibilityProperty](#) ([UserLevels](#) level)  
*Access function for [userLevelVisibility](#) property - refer to [userLevelVisibility](#) property for details.*
- [UserLevels](#) [getUserLevelEnabledProperty](#) ()  
*Access function for [userLevelEnabled](#) property - refer to [userLevelEnabled](#) property for details.*
- void [setUserLevelEnabledProperty](#) ([UserLevels](#) level)  
*Access function for [userLevelEnabled](#) property - refer to [userLevelEnabled](#) property for details.*
- [QEGenericEdit](#) (QWidget \*parent=0)
- [QEGenericEdit](#) (const QString &variableName, QWidget \*parent=0)
- void [setWriteOnLoseFocus](#) (bool writeOnLoseFocus)
- bool [getWriteOnLoseFocus](#) ()
- void [setWriteOnEnter](#) (bool writeOnEnter)
- bool [getWriteOnEnter](#) ()
- void [setWriteOnFinish](#) (bool writeOnFinish)
- bool [getWriteOnFinish](#) ()
- void [setConfirmWrite](#) (bool confirmWrite)
- bool [getConfirmWrite](#) ()
- void [setSubscribe](#) (bool subscribe)
- bool [getSubscribe](#) ()
- void [writeValue](#) (qcaobject::QCaObject \*qca, QVariant newValue)
- void [writeNow](#) ()

### Protected Member Functions

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

### Protected Attributes

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

### Properties

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

## 9.58.1 Member Enumeration Documentation

### 9.58.1.1 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.

**9.58.2 Constructor & Destructor Documentation****9.58.2.1 QEGenericEdit::QEGenericEdit ( QWidget \* parent = 0 )**

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

**9.58.2.2 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.

**9.58.3 Member Function Documentation****9.58.3.1 bool QEGenericEdit::getConfirmWrite ( )**

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

**9.58.3.2 bool QEGenericEdit::getSubscribe ( )**

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

**9.58.3.3 bool QEGenericEdit::getWriteOnEnter ( )**

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

**9.58.3.4 bool QEGenericEdit::getWriteOnFinish ( )**

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

**9.58.3.5 bool QEGenericEdit::getWriteOnLoseFocus ( )**

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

**9.58.3.6 void QEGenericEdit::setConfirmWrite ( bool *confirmWrite* )**

Sets if this widget will ask for confirmation (using a dialog box) prior to writing data. Default is 'false' (will not ask for confirmation (using a dialog box) prior to writing data).

**9.58.3.7 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)

**9.58.3.8 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').

**9.58.3.9 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).

**9.58.3.10 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).

**9.58.4 Property Documentation****9.58.4.1 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.

**9.58.4.2 bool QEGenericEdit::confirmWrite [read, write]**

Sets if this widget will ask for confirmation (using a dialog box) prior to writing data. Default is 'false' (will not ask for confirmation (using a dialog box) prior to writing data).

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

If set (default) widget will indicate the alarm state of any variable data 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 `unsigned QEGenericEdit::int` [read, write]

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

Reimplemented in [QELineEdit](#).

#### 9.58.4.5 `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)

#### 9.58.4.6 `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 programatically through `setUserLevel()` Widgets that are always accessible should be visible at 'User'. Widgets that are only accessible to scientists managing the facility should be visible at 'Scientist'. Widgets that are only accessible to engineers maintaining the facility should be visible at 'Engineer'.

#### 9.58.4.7 `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.

#### 9.58.4.8 `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.

#### 9.58.4.9 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.

#### 9.58.4.10 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 programatically through setUserLevel(). Widgets that are always visible should be visible at 'User'. Widgets that are only used by scientists managing the facility should be visible at 'Scientist'. Widgets that are only used by engineers maintaining the facility should be visible at 'Engineer'.

#### 9.58.4.11 QString QEGenericEdit::variable [read, write]

EPICS variable name (CA PV)

#### 9.58.4.12 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.

#### 9.58.4.13 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.

#### 9.58.4.14 bool QEGenericEdit::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.

#### 9.58.4.15 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').

## 9.58.4.16 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).

## 9.58.4.17 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/QLineEdit/QEGenericEdit.h
- /tmp/epicsqt/trunk/framework/widgets/QLineEdit/QEGenericEdit.cpp

## 9.59 QEGroupBox Class Reference

### Public Types

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

### 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.*
- **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 [userLevelUserStyle](#)
- QString [userLevelScientistStyle](#)
- QString [userLevelEngineerStyle](#)
- [UserLevels](#) [userLevelVisibility](#)
- [UserLevels](#) [userLevelEnabled](#)
- bool [displayAlarmState](#)
- QString [substitutedTitle](#)
- QString [textSubstitutions](#)

## 9.59.1 Member Enumeration Documentation

### 9.59.1.1 enum QEGroupBox::UserLevels

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

#### Enumerator:

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

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

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

## 9.59.2 Property Documentation

### 9.59.2.1 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.

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

If set (default) widget will indicate the alarm state of any variable data 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.

### 9.59.2.3 unsigned QEGroupBox::int [read, write]

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

### 9.59.2.4 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

### 9.59.2.5 QString QEGroupBox::textSubstitutions [read, write]

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

### 9.59.2.6 UserLevels QEGroupBox::userLevelEnabled [read, write]

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

### 9.59.2.7 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.

### 9.59.2.8 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.

### 9.59.2.9 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.

### 9.59.2.10 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'.

### 9.59.2.11 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.

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

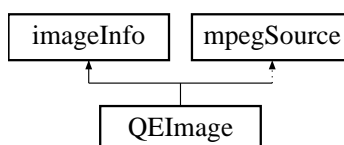
Display the widget. Default is true. Setting this property false is usefull if widget is only used to provide a signal - for example, when supplying data to a [QELink](#) widget. Note, when false the widget will still be visible in Qt Designer.

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

## 9.60 QEImage Class Reference

Inheritance diagram for QEImage:



## Public Types

- enum `selectOptions` {  
`SO_NONE`, `SO_PANNING`, `SO_VSLICE`, `SO_HSLICE`,  
`SO_AREA1`, `SO_AREA2`, `SO_AREA3`, `SO_AREA4`,  
`SO_PROFILE`, `SO_TARGET`, `SO_BEAM` }
- enum `imageUses` { `IMAGE_USE_DISPLAY`, `IMAGE_USE_SAVE`, `IMAGE_USE_DISPLAY_AND_SAVE` }
- enum `resizeOptions` { `RESIZE_OPTION_ZOOM`, `RESIZE_OPTION_FIT` }
- enum `rotationOptions` { `ROTATION_0`, `ROTATION_90_RIGHT`, `ROTATION_90_LEFT`, `ROTATION_180` }
- enum `ellipseVariableDefinitions` { `BOUNDING_RECTANGLE`, `CENTRE_AND_SIZE` }
- enum `UserLevels` { `User` = `userLevelTypes::USERLEVEL_USER`, `Scientist` = `userLevelTypes::USERLEVEL_SCIENTIST`, `Engineer` = `userLevelTypes::USERLEVEL_ENGINEER` }
- enum `FormatOptions` {  
`Mono` = `imageDataFormats::MONO`, `Bayer` = `imageDataFormats::BAYERRG`, `BayerGB` = `imageDataFormats::BAYERGB`, `BayerBG` = `imageDataFormats::BAYERBG`,  
`BayerGR` = `imageDataFormats::BAYERGR`, `BayerRG` = `imageDataFormats::BAYERRG`,  
`rgb1` = `imageDataFormats::RGB1`, `rgb2` = `imageDataFormats::RGB2`,  
`rgb3` = `imageDataFormats::RGB3`, `yuv444` = `imageDataFormats::YUV444`, `yuv422` = `imageDataFormats::YUV422`, `yuv421` = `imageDataFormats::YUV421` }
- enum `EllipseVariableDefinitions` { `BoundingBoxRectangle` = `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` = `QEImage::ROTATION_0`, `Rotate90Right` = `QEImage::ROTATION_90_RIGHT`, `Rotate90Left` = `QEImage::ROTATION_90_LEFT`, `Rotate180` = `QEImage::ROTATION_180` }
- enum `ProgramStartupOptionNames` { `None` = `applicationLauncher::PSO_NONE`, `Terminal` = `applicationLauncher::PSO_TERMINAL`, `LogOutput` = `applicationLauncher::PSO_LOGOUTPUT`, `StdOutput` = `applicationLauncher::PSO_STDOUTPUT` }

## 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 [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 \*)

## 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

- [QImage](#) (QWidget \*parent=0)
- [QImage](#) (const QString &variableName, QWidget \*parent=0)
- [~QImage](#) ()  
*Destructor.*
- [selectOptions](#) [getSelectionOption](#) ()
- void [setBitDepth](#) (unsigned int bitDepthIn)  
*Access function for #bitDepth property - refer to #bitDepth property for details.*
- unsigned int [getBitDepth](#) ()  
*Access function for #bitDepth property - refer to #bitDepth property for details.*
- void [setFormatOption](#) (imageDataFormats::formatOptions formatOption)  
*Access function for [formatOption](#) property - refer to [formatOption](#) property for details.*
- imageDataFormats::formatOptions [getFormatOption](#) ()  
*Access function for [formatOption](#) property - refer to [formatOption](#) property for details.*
- void [setResizeOption](#) (resizeOptions resizeOptionIn)  
*Access function for #resizeOption property - refer to #resizeOption property for details.*
- [resizeOptions](#) [getResizeOption](#) ()  
*Access function for #resizeOption property - refer to #resizeOption property for details.*
- void [setZoom](#) (int zoomIn)  
*Access function for [zoom](#) property - refer to [zoom](#) property for details.*
- int [getZoom](#) ()  
*Access function for [zoom](#) property - refer to [zoom](#) property for details.*
- void [setRotation](#) (rotationOptions rotationIn)  
*Access function for #rotation property - refer to #rotation property for details.*
- [rotationOptions](#) [getRotation](#) ()  
*Access function for #rotation property - refer to #rotation property for details.*
- void [setHorizontalFlip](#) (bool flipHozIn)  
*Access function for [horizontalFlip](#) property - refer to [horizontalFlip](#) property for details.*
- bool [getHorizontalFlip](#) ()  
*Access function for [horizontalFlip](#) property - refer to [horizontalFlip](#) property for details.*
- void [setVerticalFlip](#) (bool flipVertIn)  
*Access function for [verticalFlip](#) property - refer to [verticalFlip](#) property for details.*
- bool [getVerticalFlip](#) ()  
*Access function for [verticalFlip](#) property - refer to [verticalFlip](#) property for details.*
- void [setInitialHozScrollPos](#) (int initialHosScrollPosIn)  
*Access function for [initialHosScrollPos](#) property - refer to [initialHosScrollPos](#) property for details.*
- int [getInitialHozScrollPos](#) ()  
*Access function for [initialHosScrollPos](#) property - refer to [initialHosScrollPos](#) property for details.*
- void [setInitialVertScrollPos](#) (int initialVertScrollPosIn)  
*Access function for [initialVertScrollPos](#) property - refer to [initialVertScrollPos](#) property for details.*

- int [getInitialVertScrollPos](#) ()  
Access function for [initialVertScrollPos](#) property - refer to [initialVertScrollPos](#) property for details.
- void [setDisplayButtonBar](#) (bool displayButtonBarIn)  
Access function for [displayButtonBar](#) property - refer to [displayButtonBar](#) property for details.
- bool [getDisplayButtonBar](#) ()  
Access function for [displayButtonBar](#) property - refer to [displayButtonBar](#) property for details.
- void [setShowTime](#) (bool pValue)  
Access function for [showTime](#) property - refer to [showTime](#) property for details.
- bool [getShowTime](#) ()  
Access function for [showTime](#) property - refer to [showTime](#) property for details.
- void [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 [setVertSliceMarkupColor](#) (QColor pValue)  
Access function for [vertSliceColor](#) property - refer to [vertSliceColor](#) property for details.
- QColor [getVertSliceMarkupColor](#) ()  
Access function for [vertSliceColor](#) property - refer to [vertSliceColor](#) property for details.
- void [setHozSliceMarkupColor](#) (QColor pValue)  
Access function for [hozSliceColor](#) property - refer to [hozSliceColor](#) property for details.
- QColor [getHozSliceMarkupColor](#) ()  
Access function for [hozSliceColor](#) property - refer to [hozSliceColor](#) property for details.
- void [setProfileMarkupColor](#) (QColor pValue)  
Access function for [profileColor](#) property - refer to [profileColor](#) property for details.
- QColor [getProfileMarkupColor](#) ()  
Access function for [profileColor](#) property - refer to [profileColor](#) property for details.
- void [setAreaMarkupColor](#) (QColor pValue)  
Access function for [areaColor](#) property - refer to [areaColor](#) property for details.
- QColor [getAreaMarkupColor](#) ()  
Access function for [areaColor](#) property - refer to [areaColor](#) property for details.
- void [setTargetMarkupColor](#) (QColor pValue)  
Access function for [targetColor](#) property - refer to [targetColor](#) property for details.
- QColor [getTargetMarkupColor](#) ()  
Access function for [targetColor](#) property - refer to [targetColor](#) property for details.
- void [setBeamMarkupColor](#) (QColor pValue)  
Access function for [beamColor](#) property - refer to [beamColor](#) property for details.
- QColor [getBeamMarkupColor](#) ()  
Access function for [beamColor](#) property - refer to [beamColor](#) property for details.
- void [setTimeMarkupColor](#) (QColor pValue)

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

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

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

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

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

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

  - void [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 [setDisplayVertSliceSelection](#) (bool displayVSLiceSelection)  
Access function for [displayVertSliceSelection](#) property - refer to [displayVertSliceSelection](#) property for details.
- bool [getDisplayVertSliceSelection](#) ()  
Access function for [displayVertSliceSelection](#) property - refer to [displayVertSliceSelection](#) property for details.
- void [setDisplayHozSliceSelection](#) (bool displayHSLiceSelection)  
Access function for [displayHozSliceSelection](#) property - refer to [displayHozSliceSelection](#) property for details.
- bool [getDisplayHozSliceSelection](#) ()  
Access function for [displayHozSliceSelection](#) property - refer to [displayHozSliceSelection](#) 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 [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 [getHozSliceLegend](#) ()
 

Access function for [hozSliceLegend](#) property - refer to [hozSliceLegend](#) property for details.
- void [setHozSliceLegend](#) (QString legend)

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

- [QString getVertSliceLegend \(\)](#)

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

- [void setVertSliceLegend \(QString legend\)](#)

Access function for [vertSliceLegend](#) property - refer to [vertSliceLegend](#) 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 deta.
- QString [getSubstitutedUrl](#) ()

Access function for [URL](#) property - refer to [URL](#) property for deta.
- [UserLevels](#) [getUserLevelVisibilityProperty](#) ()

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

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

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

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

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

Access function for [formatOption](#) property - refer to [formatOption](#) property for details.
- void [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](#) program-StartupOption)  
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](#) program-StartupOption)  
Access function for [#ProgramStartupOptionNames2](#) property - refer to [#ProgramStartupOptionNames2](#) property for details.
- [ProgramStartupOptionNames](#) [getProgramStartupOptionProperty2](#) ()  
Access function for [#ProgramStartupOptionNames2](#) property - refer to [#ProgramStartupOptionNames2](#) property for details.

## Protected Types

- enum **variableIndexes** {  
**IMAGE\_VARIABLE, FORMAT\_VARIABLE, BIT\_DEPTH\_VARIABLE, WIDTH\_VARIABLE,**  
**HEIGHT\_VARIABLE, NUM\_DIMENSIONS\_VARIABLE, DIMENSION\_0\_VARIABLE, DIMENSION\_1\_VARIABLE,**  
**DIMENSION\_2\_VARIABLE, ROI1\_X\_VARIABLE, ROI1\_Y\_VARIABLE, ROI1\_W\_VARIABLE,**  
**ROI1\_H\_VARIABLE, ROI2\_X\_VARIABLE, ROI2\_Y\_VARIABLE, ROI2\_W\_VARIABLE,**  
**ROI2\_H\_VARIABLE, ROI3\_X\_VARIABLE, ROI3\_Y\_VARIABLE, ROI3\_W\_VARIABLE,**  
**ROI3\_H\_VARIABLE, ROI4\_X\_VARIABLE, ROI4\_Y\_VARIABLE, ROI4\_W\_VARIABLE,**  
**ROI4\_H\_VARIABLE, TARGET\_X\_VARIABLE, TARGET\_Y\_VARIABLE, BEAM\_X\_VARIABLE,**  
**BEAM\_Y\_VARIABLE, TARGET\_TRIGGER\_VARIABLE, CLIPPING\_ONOFF\_VARIABLE, CLIPPING\_LOW\_VARIABLE,**  
**CLIPPING\_HIGH\_VARIABLE, PROFILE\_H\_VARIABLE, PROFILE\_H\_THICKNESS\_VARIABLE, PROFILE\_V\_VARIABLE,**

```

    PROFILE_V_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)
- QImage **copyImage** ()
- 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

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

### Properties

- QString [imageVariable](#)
- QString [formatVariable](#)
- QString [bitDepthVariable](#)
- QString [widthVariable](#)
- QString [heightVariable](#)

- QString [dimensionsVariable](#)
- QString [dimension1Variable](#)
- QString [dimension2Variable](#)
- QString [dimension3Variable](#)
- QString [regionOfInterest1XVariable](#)
- QString [regionOfInterest1YVariable](#)
- QString [regionOfInterest1WVariable](#)
- QString [regionOfInterest1HVariable](#)
- QString [regionOfInterest2XVariable](#)
- QString [regionOfInterest2YVariable](#)
- QString [regionOfInterest2WVariable](#)
- QString [regionOfInterest2HVariable](#)
- QString [regionOfInterest3XVariable](#)
- QString [regionOfInterest3YVariable](#)
- QString [regionOfInterest3WVariable](#)
- QString [regionOfInterest3HVariable](#)
- QString [regionOfInterest4XVariable](#)
- QString [regionOfInterest4YVariable](#)
- QString [regionOfInterest4WVariable](#)
- QString [regionOfInterest4HVariable](#)
- QString [targetXVariable](#)
- QString [targetYVariable](#)
- QString [beamXVariable](#)
- QString [beamYVariable](#)
- QString [targetTriggerVariable](#)
- QString [clippingOnOffVariable](#)
- QString [clippingLowVariable](#)
- QString [clippingHighVariable](#)
- QString [profileHozVariable](#)
- QString [profileHozThicknessVariable](#)
- QString [profileVertVariable](#)
- QString [profileVertThicknessVariable](#)
- 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 [userLevelUserStyle](#)
- QString [userLevelScientistStyle](#)
- QString [userLevelEngineerStyle](#)
- [UserLevels](#) [userLevelVisibility](#)
- [UserLevels](#) [userLevelEnabled](#)
- bool [displayAlarmState](#)
- [FormatOptions](#) [formatOption](#)
- bool [enableVertSliceSelection](#)
- bool [enableHozSliceSelection](#)
- bool [enableProfileSelection](#)
- bool [enableArea1Selection](#)
- bool [enableArea2Selection](#)
- bool [enableArea3Selection](#)
- bool [enableArea4Selection](#)
- bool [enableTargetSelection](#)
- bool [enableBeamSelection](#)
- QString [hozSliceLegend](#)  
*Name of horizontal slice profile markup.*
- QString [vertSliceLegend](#)  
*Name of vertical slice profile 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 [displayHozSliceSelection](#)
- 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 [hozSliceColor](#)
  - 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](#)

### 9.60.1 Member Enumeration Documentation

#### 9.60.1.1 enum QImage::ellipseVariableDefinitions

Options for the use of ellipse markup variables.

**Enumerator:**

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

#### 9.60.1.2 enum QImage::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.

#### 9.60.1.3 enum QImage::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 (???)

#### 9.60.1.4 enum QImage::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.

**9.60.1.5 enum QImage::ResizeOptions**

User friendly enumerations for #resizeOption property

**Enumerator:**

***Zoom*** Zoom to selected percentage.

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

**9.60.1.6 enum QImage::resizeOptions**

Image resize options

**Enumerator:**

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

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

**9.60.1.7 enum QImage::RotationOptions**

User friendly enumerations for #rotation property

**Enumerator:**

***NoRotation*** No image rotation.

***Rotate90Right*** Rotate image 90 degrees clockwise.

***Rotate90Left*** Rotate image 90 degrees anticlockwise.

***Rotate180*** Rotate image 180 degrees.

**9.60.1.8 enum QImage::rotationOptions**

Image rotation options

**Enumerator:**

***ROTATION\_0*** No image rotation.



***ROTATION\_90\_RIGHT*** Rotate image 90 degrees clockwise.

***ROTATION\_90\_LEFT*** Rotate image 90 degrees anticlockwise.

***ROTATION\_180*** Rotate image 180 degrees.

#### 9.60.1.9 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\_VSLICE*** Select the vertical slice point.

***SO\_HSLICE*** Select the horizontal slice point.

***SO\_AREA4*** User is selecting an area (for region of interest)

***SO\_PROFILE*** Select an arbitrary line across the image (to determine a profile)

***SO\_TARGET*** Mark the target point.

***SO\_BEAM*** Mark the current beam location.

#### 9.60.1.10 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.

#### 9.60.1.11 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.

## 9.60.2 Constructor & Destructor Documentation

### 9.60.2.1 QImage::QImage ( 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.

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

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

## 9.60.3 Member Function Documentation

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

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

## 9.60.4 Member Data Documentation

### 9.60.4.1 bool QImage::displayButtonBar [read, write, protected]

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

### 9.60.4.2 int QImage::initialVertScrollPos [read, write, protected]

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

## 9.60.5 Property Documentation

### 9.60.5.1 bool QImage::allowDrop [read, write]

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

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

Used to select the color of the area selection markups.

**9.60.5.3 QStringList QImage::arguments1** [read, write]

Arguments for program specified in the 'program1' property.

**9.60.5.4 QStringList QImage::arguments2** [read, write]

Arguments for program specified in the 'program2' property.

**9.60.5.5 bool QImage::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.

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

Used to select the color of the beam marker.

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

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

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

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

**9.60.5.9 QString QImage::bitDepthVariable** [read, write]

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

**9.60.5.10 bool QImage::briefInfoArea** [read, write]

If true, the information area will be brief (one row)

**9.60.5.11 QString QEImage::clippingHighVariable** [read, write]

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

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

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

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

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

**9.60.5.14 bool QEImage::contrastReversal** [read, write]

If true, the image will undergo contrast reversal.

**9.60.5.15 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.

**9.60.5.16 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.

**9.60.5.17 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.

**9.60.5.18 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)

**9.60.5.19** `bool QImage::displayAlarmState` `[read, write]`

If set (default) widget will indicate the alarm state of any variable data 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.

**9.60.5.20** `bool QImage::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.

**9.60.5.21** `bool QImage::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.

**9.60.5.22** `bool QImage::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.

**9.60.5.23** `bool QImage::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.

**9.60.5.24** `bool QImage::displayBeamSelection` `[read, write]`

If true, beam selection will be displayed on the image. Note, this property is ignored unless the [enableBeamSelection](#) property is true.

**9.60.5.25** `bool QImage::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.

**9.60.5.26** `bool QImage::displayEllipse` `[read, write]`

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

**9.60.5.27 bool QImage::displayHozSliceSelection** [read, write]

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

**9.60.5.28 bool QImage::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.

**9.60.5.29 bool QImage::displayTargetSelection** [read, write]

If true, target selection will be displayed on the image. Note, this property is ignored unless the [enableTargetSelection](#) property is true.

**9.60.5.30 bool QImage::displayVertSliceSelection** [read, write]

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

**9.60.5.31 QColor QImage::ellipseColor** [read, write]

Used to select the color of the ellipse marker.

**9.60.5.32 QString QImage::ellipseHVariable** [read, write]

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

**9.60.5.33 QString QImage::ellipseWVariable** [read, write]

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

**9.60.5.34 QString QImage::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`).

**9.60.5.35 QString QImage::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`).

**9.60.5.36** `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

**9.60.5.37** `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

**9.60.5.38** `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

**9.60.5.39** `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

**9.60.5.40** `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.

**9.60.5.41** `bool QEImage::enableHozSliceSelection` `[read, write]`

If true, the option to select a horizontal slice through the image will be available to the user. This will be used to generate a horizontal pixel profile, and write the position of the slice to the optional variable specified by the [profileHozVariable](#) property. The profile will only be presented to the user if `#enableHozSlicePresentation` property is true.

**9.60.5.42** `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.

**9.60.5.43** `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.

**9.60.5.44 bool QImage::enableVertSliceSelection** [read, write]

If true, the option to select a vertical slice through the image will be available to the user. This will be used to generate a horizontal pixel profile, and write the position of the slice to the optional variable specified by the [profileVertVariable](#) property. The profile will only be presented to the user if `#enableVertSlicePresentation` property is true.

**9.60.5.45 bool QImage::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.

**9.60.5.46 FormatOptions QImage::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.)

**9.60.5.47 QString QImage::formatVariable** [read, write]

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

**9.60.5.48 QString QImage::heightVariable** [read, write]

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

**9.60.5.49 bool QImage::horizontalFlip** [read, write]

If true, flip image horizontally.

**9.60.5.50 QColor QImage::hozSliceColor** [read, write]

Used to select the color of the horizontal slice markup.

**9.60.5.51 QString QImage::imageVariable** [read, write]

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



**9.60.5.52** `int QImage::initialHosScrollPos` [read, write]

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

**9.60.5.53** `unsigned QImage::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).

**9.60.5.54** `QString QImage::lineProfileArrayVariable` [read, write]

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

**9.60.5.55** `QString QImage::lineProfileThicknessVariable` [read, write]

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

**9.60.5.56** `QString QImage::lineProfileX1Variable` [read, write]

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

**9.60.5.57** `QString QImage::lineProfileX2Variable` [read, write]

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

**9.60.5.58** `QString QImage::lineProfileY1Variable` [read, write]

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

**9.60.5.59** `QString QImage::lineProfileY2Variable` [read, write]

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

**9.60.5.60** `bool QImage::logBrightness` `[read, write]`

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

**9.60.5.61** `QColor QImage::profileColor` `[read, write]`

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

**9.60.5.62** `QString QImage::profileHozArrayVariable` `[read, write]`

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

**9.60.5.63** `QString QImage::profileHozThicknessVariable` `[read, write]`

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

**9.60.5.64** `QString QImage::profileHozVariable` `[read, write]`

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

**9.60.5.65** `QString QImage::profileVertArrayVariable` `[read, write]`

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

**9.60.5.66** `QString QImage::profileVertThicknessVariable` `[read, write]`

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

**9.60.5.67** `QString QImage::profileVertVariable` `[read, write]`

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

**9.60.5.68** `QString QImage::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

**9.60.5.69 QString QImage::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

**9.60.5.70 ProgramStartupOptionNames QImage::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.

**9.60.5.71 ProgramStartupOptionNames QImage::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.

**9.60.5.72 QString QImage::regionOfInterest1HVariable** [read, write]

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

**9.60.5.73 QString QImage::regionOfInterest1WVariable** [read, write]

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

**9.60.5.74 QString QImage::regionOfInterest1XVariable** [read, write]

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

**9.60.5.75 QString QImage::regionOfInterest1YVariable** [read, write]

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

**9.60.5.76 QString QImage::regionOfInterest2HVariable** [read, write]

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

**9.60.5.86** `QString QEImage::regionOfInterest4XVariable` [read, write]

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

**9.60.5.87** `QString QEImage::regionOfInterest4YVariable` [read, write]

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

**9.60.5.88** `ResizeOptions QEImage::resizeOption` [read, write]

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

**9.60.5.89** `RotationOptions QEImage::rotation` [read, write]

Image rotation option.

**9.60.5.90** `bool QEImage::showTime` [read, write]

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

**9.60.5.91** `QColor QEImage::targetColor` [read, write]

Used to select the color of the target marker.

**9.60.5.92** `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.

**9.60.5.93** `QString QEImage::targetXVariable` [read, write]

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

**9.60.5.94** `QString QEImage::targetYVariable` [read, write]

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

**9.60.5.95 QColor QImage::timeColor** [read, write]

Used to select the color of the timestamp.

**9.60.5.96 QString QImage::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)

**9.60.5.97 bool QImage::useFalseColour** [read, write]

If true, the apply false colour to the image.

**9.60.5.98 UserLevels QImage::userLevelEnabled** [read, write]

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

**9.60.5.99 QString QImage::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.

**9.60.5.100 QString QImage::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.

**9.60.5.101 QString QImage::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.

**9.60.5.102 UserLevels QImage::userLevelVisibility** [read, write]

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

**9.60.5.103 bool QImage::variableAsToolTip** [read, write]

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

**9.60.5.104 QString QImage::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.

**9.60.5.105 bool QImage::verticalFlip** [read, write]

If true, flip image vertically.

**9.60.5.106 QColor QImage::vertSliceColor** [read, write]

Used to select the color of the vertical slice markup.

**9.60.5.107 bool QImage::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.

#### 9.60.5.108 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

### 9.61 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

### 9.62 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



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

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

### Signals

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

### Public Member Functions

- [QELabel](#) ([QWidget](#) \*parent=0)
- [QELabel](#) (const [QString](#) &variableName, [QWidget](#) \*parent=0)
- [UserLevels](#) [getUserLevelVisibilityProperty](#) ()  
*Access function for [userLevelVisibility](#) property - refer to [userLevelVisibility](#) property for details.*
- void [setUserLevelVisibilityProperty](#) ([UserLevels](#) level)  
*Access function for [userLevelVisibility](#) property - refer to [userLevelVisibility](#) property for details.*
- [UserLevels](#) [getUserLevelEnabledProperty](#) ()  
*Access function for [userLevelEnabled](#) property - refer to [userLevelEnabled](#) property for details.*

- void [setUserLevelEnabledProperty](#) ([UserLevels](#) level)  
Access function for [userLevelEnabled](#) property - refer to [userLevelEnabled](#) property for details.
- void [setFormatProperty](#) ([Formats](#) format)  
Access function for [format](#) property - refer to [format](#) property for details.
- [Formats](#) [getFormatProperty](#) ()  
Access function for [format](#) property - refer to [format](#) property for details.
- void [setNotationProperty](#) ([Notations](#) notation)  
Access function for [notation](#) property - refer to [notation](#) property for details.
- [Notations](#) [getNotationProperty](#) ()  
Access function for [notation](#) property - refer to [notation](#) property for details.
- void [setArrayActionProperty](#) ([ArrayActions](#) arrayAction)  
Access function for [arrayAction](#) property - refer to [arrayAction](#) property for details.
- [ArrayActions](#) [getArrayActionProperty](#) ()  
Access function for [arrayAction](#) property - refer to [arrayAction](#) property for details.
- void [setUpdateOptionProperty](#) ([UpdateOptions](#) updateOption)  
Access function for [#updateOption](#) property - refer to [#updateOption](#) property for details.
- [UpdateOptions](#) [getUpdateOptionProperty](#) ()  
Access function for [#updateOption](#) property - refer to [#updateOption](#) property for details.
- void [setPixmap0Property](#) (QPixmap pixmap)  
'Set' access function for [pixmap0](#) properties. Refer to [pixmap0](#) property for details
- void [setPixmap1Property](#) (QPixmap pixmap)  
'Set' access function for [pixmap1](#) properties. Refer to [pixmap1](#) property for details
- void [setPixmap2Property](#) (QPixmap pixmap)  
'Set' access function for [pixmap2](#) properties. Refer to [pixmap2](#) property for details
- void [setPixmap3Property](#) (QPixmap pixmap)  
'Set' access function for [pixmap3](#) properties. Refer to [pixmap3](#) property for details
- void [setPixmap4Property](#) (QPixmap pixmap)  
'Set' access function for [pixmap4](#) properties. Refer to [pixmap4](#) property for details
- void [setPixmap5Property](#) (QPixmap pixmap)  
'Set' access function for [pixmap5](#) properties. Refer to [pixmap5](#) property for details
- void [setPixmap6Property](#) (QPixmap pixmap)  
'Set' access function for [pixmap6](#) properties. Refer to [pixmap6](#) property for details
- void [setPixmap7Property](#) (QPixmap pixmap)  
'Set' access function for [pixmap7](#) properties. Refer to [pixmap7](#) property for details
- QPixmap [getPixmap0Property](#) ()  
'Get' access function for [pixmap0](#) properties. Refer to [pixmap0](#) property for details
- QPixmap [getPixmap1Property](#) ()  
'Get' access function for [pixmap1](#) properties. Refer to [pixmap1](#) property for details
- QPixmap [getPixmap2Property](#) ()  
'Get' access function for [pixmap2](#) properties. Refer to [pixmap2](#) property for details
- QPixmap [getPixmap3Property](#) ()

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

- QPixmap [getPixmap4Property](#) ()

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

- QPixmap [getPixmap5Property](#) ()

*'Get' access function for [pixmap5](#) properties. Refer to [pixmap5](#) property for details*

- QPixmap [getPixmap6Property](#) ()

*'Get' access function for [pixmap6](#) properties. Refer to [pixmap6](#) property for details*

- QPixmap [getPixmap7Property](#) ()

*'Get' access function for [pixmap7](#) properties. Refer to [pixmap7](#) property for details*

## Properties

- QString [variable](#)
- QString [variableSubstitutions](#)
- bool [variableAsToolTip](#)
- bool [allowDrop](#)
- bool [visible](#)
- unsigned int
- QString [userLevelUserStyle](#)
- QString [userLevelScientistStyle](#)
- QString [userLevelEngineerStyle](#)
- [UserLevels](#) [userLevelVisibility](#)
- [UserLevels](#) [userLevelEnabled](#)
- bool [displayAlarmState](#)
- int [precision](#)
- bool [useDbPrecision](#)
- bool [leadingZero](#)
- bool [trailingZeros](#)
- bool [addUnits](#)
- QString [localEnumeration](#)
- [Formats](#) [format](#)
- [Notations](#) [notation](#)
- [ArrayActions](#) [arrayAction](#)
- QString [text](#)
- [UpdateOptions](#) [updateOption](#)
- QPixmap [pixmap0](#)
- QPixmap [pixmap1](#)
- QPixmap [pixmap2](#)
- QPixmap [pixmap3](#)
- QPixmap [pixmap4](#)
- QPixmap [pixmap5](#)
- QPixmap [pixmap6](#)
- QPixmap [pixmap7](#)

### 9.63.1 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.

### 9.63.2 Member Enumeration Documentation

#### 9.63.2.1 enum QELabel::ArrayActions

User friendly enumerations for arrayAction property - refer to QEStrngFormatting::arrayActions for details.

##### Enumerator:

**Append** Refer to QEStrngFormatting::APPEND for details.

**Ascii** Refer to QEStrngFormatting::ASCII for details.

**Index** Refer to QEStrngFormatting::INDEX for details.

#### 9.63.2.2 enum QELabel::Formats

User friendly enumerations for format property - refer to QEStrngFormatting::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.

#### 9.63.2.3 enum QELabel::Notations

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

##### Enumerator:

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

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

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

#### 9.63.2.4 enum QELabel::UpdateOptions

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

##### Enumerator:

**Text** Data updates will update the label text.

**Picture** Data updates will update the label icon.

#### 9.63.2.5 enum QELabel::updateOptions

Options for updating the label. The formatted text is used to update the label text, or select a background pixmap.

##### Enumerator:

**UPDATE\_TEXT** Update the label text.

**UPDATE\_PIXMAP** Update the label background pixmap.

#### 9.63.2.6 enum QELabel::UserLevels

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

##### Enumerator:

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

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

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

### 9.63.3 Constructor & Destructor Documentation

#### 9.63.3.1 QELabel::QELabel ( QWidget \* *parent* = 0 )

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

#### 9.63.3.2 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.

### 9.63.4 Member Function Documentation

#### 9.63.4.1 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.

### 9.63.5 Property Documentation

#### 9.63.5.1 bool QELabel::addUnits [read, write]

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

#### 9.63.5.2 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.

#### 9.63.5.3 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'.

#### 9.63.5.4 bool QELabel::displayAlarmState [read, write]

If set (default) widget will indicate the alarm state of any variable data 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.

**9.63.5.5 Formats QELabel::format** [read, write]

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

**9.63.5.6 unsigned QELabel::int** [read, write]

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

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

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

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

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

**9.63.5.8 QString QELabel::localEnumeration** [read, write]

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

Format is:

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

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

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

may be included in a string to indicate a line break

Examples are:

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

The data value is converted to a string if no enumeration for that value is available. For example, if the local enumeration is '0:off,1:on', and a value of 10 is processed, the

text generated is '10'. If a blank string is required, this should be explicit. for example, '0:off,1:on,10:'''

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

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

Notation used for numerical formatting. Default is fixed.

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

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

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

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

#### 9.63.5.12 QPixmap QELabel::pixmap2 [read, write]

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

#### 9.63.5.13 QPixmap QELabel::pixmap3 [read, write]

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

#### 9.63.5.14 QPixmap QELabel::pixmap4 [read, write]

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

#### 9.63.5.15 QPixmap QELabel::pixmap5 [read, write]

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

#### 9.63.5.16 QPixmap QELabel::pixmap6 [read, write]

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



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

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

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

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

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

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

**9.63.5.20 UpdateOptions QELabel::updateOption** [read, write]

Determines if data updates the label text, or the label pixmap. For both options all normal string formatting is applied. If Text, the formatted text is simply presented as the label text. If Picture, the FORMATTED text is then interpreted as an integer and used to select one of the pixmaps specified by properties pixmap0 through to pixmap7.

**9.63.5.21 bool QELabel::useDbPrecision** [read, write]

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

**9.63.5.22 UserLevels QELabel::userLevelEnabled** [read, write]

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

**9.63.5.23 QString QELabel::userLevelEngineerStyle** [read, write]

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

#### 9.63.5.24 QString QELabel::userLevelScientistStyle [read, write]

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

#### 9.63.5.25 QString QELabel::userLevelUserStyle [read, write]

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

#### 9.63.5.26 UserLevels QELabel::userLevelVisibility [read, write]

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

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

EPICS variable name (CA PV)

#### 9.63.5.28 bool QELabel::variableAsToolTip [read, write]

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

#### 9.63.5.29 QString QELabel::variableSubstitutions [read, write]

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

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

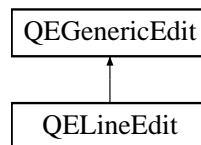
Display the widget. Default is true. Setting this property false is useful if widget is only used to provide a signal - for example, when supplying data to a [QELink](#) widget. Note, when false the widget will still be visible in Qt Designer.

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

## 9.64 QELineEdit Class Reference

Inheritance diagram for QELineEdit:



### Public Types

- 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 }

### Signals

- void [dbValueChanged](#) (const QString &out)
- void [userChange](#) (const QString &oldValue, const QString &newValue, const QString &lastValue)  
*Internal use only. Used by [QEConfiguredLayout](#) to be notified when one of its widgets has written something.*
- void [requestResend](#) ()  
*Internal use only. Used when changing a property value to force a re-display to reflect the new property value.*

## Public Member Functions

- 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.*
- [QLineEdit](#) (QWidget \*parent=0)
- [QLineEdit](#) (const QString &variableName, QWidget \*parent=0)

## Properties

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

### 9.64.1 Member Enumeration Documentation

#### 9.64.1.1 enum [QLineEdit::ArrayActions](#)

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

#### Enumerator:

- Append*** Refer to [QStringFormatting::APPEND](#) for details.
- Ascii*** Refer to [QStringFormatting::ASCII](#) for details.
- Index*** Refer to [QStringFormatting::INDEX](#) for details.

## 9.64.1.2 enum QLEdit::Formats

User friendly enumerations for format property - refer to QStringFormatting::formats for details.

**Enumerator:**

**Default** Format as best appropriate for the data type.

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

**Integer** Format as an integer.

**UnsignedInteger** Format as an unsigned integer.

**Time** Format as a time.

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

## 9.64.1.3 enum QLEdit::Notations

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

**Enumerator:**

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

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

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

## 9.64.2 Constructor &amp; Destructor Documentation

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

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

## 9.64.2.2 QLEdit::QLEdit ( const QString &amp; 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.

## 9.64.3 Member Function Documentation

## 9.64.3.1 void QLEdit::dbValueChanged ( const QString &amp; 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 Property Documentation

#### 9.64.4.1 `bool QLEdit::addUnits` [read, write]

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

#### 9.64.4.2 `ArrayActions QLEdit::arrayAction` [read, write]

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

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

#### 9.64.4.3 `Formats QLEdit::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).

#### 9.64.4.4 `unsigned QLEdit::int` [read, write]

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

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

Reimplemented from [QEGenericEdit](#).

#### 9.64.4.5 `bool QLEdit::leadingZero` [read, write]

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

#### 9.64.4.6 `QString QLEdit::localEnumeration` [read, write]

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

Format is:

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

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

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

may be included in a string to indicate a line break

Examples are:

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

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

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

#### 9.64.4.7 Notations QLEdit::notation [read, write]

Notation used for numerical formatting. Default is fixed.

#### 9.64.4.8 int QLEdit::precision [read, write]

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

#### 9.64.4.9 bool QLEdit::trailingZeros [read, write]

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

#### 9.64.4.10 bool QLEdit::useDbPrecision [read, write]

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

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

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

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

## 9.65 QLineEditManager Class Reference

### Public Member Functions

- **QLineEditManager** (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/QLineEdit/QLineEditManager.h

## 9.66 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 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 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

## 9.67 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 }

### Public Member Functions

- **QELog** (QWidget \*pParent=0)
- void **setShowColumnTime** (bool pValue)
- bool **getShowColumnTime** ()
- void **setShowColumnType** (bool pValue)
- bool **getShowColumnType** ()
- void **setShowColumnMessage** (bool pValue)
- bool **getShowColumnMessage** ()
- void **setShowMessageFilter** (bool pValue)
- bool **getShowMessageFilter** ()
- void **setShowClear** (bool pValue)
- bool **getShowClear** ()
- void **setShowSave** (bool pValue)
- bool **getShowSave** ()
- void **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.*

### Protected Attributes

- [\\_QTableWidgetLog](#) \* [qTableWidgetLog](#)
- [QCheckBox](#) \* [qCheckBoxInfoMessage](#)
- [QCheckBox](#) \* [qCheckBoxWarningMessage](#)
- [QCheckBox](#) \* [qCheckBoxErrorMessage](#)
- [QPushButton](#) \* [qPushButtonClear](#)
- [QPushButton](#) \* [qPushButtonSave](#)
- [QColor](#) [qColorInfo](#)
- [QColor](#) [qColorWarning](#)
- [QColor](#) [qColorError](#)
- bool [scrollToBottom](#)
- int [optionsLayout](#)

### Properties

- bool [showColumnTime](#)
- bool [showColumnType](#)
- bool [showColumnMessage](#)
- bool [showMessageFilter](#)
- bool [showClear](#)
- bool [showSave](#)
- [optionsLayoutProperty](#) [optionsLayout](#)
- [QColor](#) [infoColor](#)
- [QColor](#) [warningColor](#)
- [QColor](#) [errorColor](#)
- [MessageFilterOptions](#) [messageFormFilter](#)
- [MessageFilterOptions](#) [messageSourceFilter](#)
- unsigned [int](#)
- bool [variableAsToolTip](#)
- bool [allowDrop](#)
- bool [visible](#)

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

## 9.67.1 Member Enumeration Documentation

### 9.67.1.1 enum `QLog::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.

## 9.67.2 Property Documentation

### 9.67.2.1 bool `QLog::allowDrop` [read, write]

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

### 9.67.2.2 bool `QLog::displayAlarmState` [read, write]

If set (default) widget will indicate the alarm state of any variable data 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.

### 9.67.2.3 unsigned `QLog::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 [QLog](#) widget may be set up to only log messages from a select set of widgets.

#### 9.67.2.4 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 programatically through `setUserLevel()` Widgets that are always accessible should be visible at 'User'. Widgets that are only accessible to scientists managing the facility should be visible at 'Scientist'. Widgets that are only accessible to engineers maintaining the facility should be visible at 'Engineer'.

#### 9.67.2.5 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.

#### 9.67.2.6 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.

#### 9.67.2.7 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.

#### 9.67.2.8 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 programatically through `setUserLevel()` Widgets that are always visible should be visible at 'User'. Widgets that are only used by scientists managing the facility should be visible at 'Scientist'. Widgets that are only used by engineers maintaining the facility should be visible at 'Engineer'.

### 9.67.2.9 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.

### 9.67.2.10 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

## 9.68 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

## 9.69 QELoginDialog Class Reference

### Public Member Functions

- **QELoginDialog** ([QELogin](#) \*ownerIn)

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

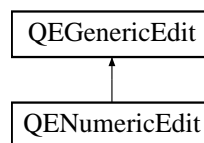
- /tmp/epicsqt/trunk/framework/widgets/QELogin/QELogin.h
- /tmp/epicsqt/trunk/framework/widgets/QELogin/QELogin.cpp

## 9.70 QENumericEdit Class Reference

The [QENumericEdit](#) class This class is similar to [QELineEdit](#) (both of which are derived from QLineEdit). However this class is tailored specficially for editing numerical values.

```
#include <QENumericEdit.h>
```

Inheritance diagram for QENumericEdit:



### Signals

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

### Public Member Functions

- [QENumericEdit](#) (QWidget \*parent=0)
- [QENumericEdit](#) (const QString &variableName, QWidget \*parent=0)
- virtual [~QENumericEdit](#) ()

*Destruction.*

- double **getNumericValue** ()
- void **setAutoScale** (const bool value)
- bool **getAutoScale** ()
- void **setPropertyPrecision** (const int value)
- int **getPropertyPrecision** ()
- void **setPropertyLeadingZeros** (const int value)
- int **getPropertyLeadingZeros** ()
- void **setPropertyMinimum** (const double value)
- double **getPropertyMinimum** ()

- void **setPropertyMaximum** (const double value)
- double **getPropertyMaximum** ()
- void **setAddUnits** (bool addUnits)
- bool **getAddUnits** ()
- void **setRadix** (const QFixedPointRadix::Radices value)
- QFixedPointRadix::Radices **getRadix** ()
- void **setSeparator** (const QFixedPointRadix::Separators value)
- QFixedPointRadix::Separators **getSeparator** ()

### Protected Member Functions

- void **keyPressEvent** (QKeyEvent \*event)
- void **focusInEvent** (QFocusEvent \*event)
- void **mouseReleaseEvent** (QMouseEvent \*event)
- void **establishConnection** (unsigned int variableIndex)
- qcaobject::QCaObject \* **createQcaltem** (unsigned int variableIndex)
- int **getPrecision** ()
- int **getLeadingZeros** ()
- double **getMinimum** ()
- double **getMaximum** ()
- int **maximumSignificance** ()
- int **getRadixValue** ()
- void **setValue** (const QVariant &value)  
*Sets the undelying QLineEdit widget to the given value.*
- QVariant **getValue** ()  
*Gets the undelying value.*
- bool **writeData** (const QVariant &value, QString &message)  
*Write the data to the channel.*

### Protected Attributes

- QEFloatingFormatting **floatingFormatting**

### Properties

- bool **autoScale**
- QFixedPointRadix::Radices **radix**  
*Specify radix, default is Decimal.*
- QFixedPointRadix::Separators **separator**  
*Specify digit 'thousands' separator character, default is none.*
- int **precision**
- int **leadingZeros**
- double **minimum**
- double **maximum**
- bool **addUnits**



## Friends

- class **NumericValidator**

### 9.70.1 Detailed Description

The [QENumericEdit](#) class This class is similar to [QELineEdit](#) (both of which are derived from [QLineEdit](#)). However this class is tailored specficially for editing numerical values.

Note: this class based on thumb\_wheel\_edits.pas by same author.

### 9.70.2 Constructor & Destructor Documentation

#### 9.70.2.1 QENumericEdit::QENumericEdit ( QWidget \* *parent* = 0 )

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

#### 9.70.2.2 QENumericEdit::QENumericEdit ( const QString & *variableName*, QWidget \* *parent* = 0 )

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

### 9.70.3 Member Function Documentation

#### 9.70.3.1 void QENumericEdit::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.

### 9.70.4 Property Documentation

#### 9.70.4.1 bool QENumericEdit::addUnits [read, write]

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

#### 9.70.4.2 bool QENumericEdit::autoScale [read, write]

If true (default), display and editing of numbers using the precision, and control limits supplied with the data. If false, the precision, leadingZeros, minimum and maximum properties are used.

#### 9.70.4.3 `int QNumericEdit::leadingZeros` [read, write]

Specifies the number of leading zeros. This is only used if `autoScale` is false. Strictly speaking, this should be an unsigned int, but designer properties editor much 'nicer' with integers.

#### 9.70.4.4 `double QNumericEdit::maximum` [read, write]

Specifies the maximum allowed value. This is only used if `autoScale` is false.

#### 9.70.4.5 `double QNumericEdit::minimum` [read, write]

Specifies the minimum allowed value. This is only used if `autoScale` is false.

#### 9.70.4.6 `int QNumericEdit::precision` [read, write]

Precision used for the display and editing of numbers. The default is 4. This is only used if `autoScale` is false. Strictly speaking, this should be an unsigned int, but designer properties editor much 'nicer' with integers.

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

- `/tmp/epicsqt/trunk/framework/widgets/QLineEdit/QNumericEdit.h`
- `/tmp/epicsqt/trunk/framework/widgets/QLineEdit/QNumericEdit.cpp`

## 9.71 QNumericEditManager Class Reference

### Public Member Functions

- **QNumericEditManager** (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/QLineEdit/QNumericEditManager.h`
- `/tmp/epicsqt/trunk/framework/widgets/QLineEdit/QNumericEditManager.cpp`

## 9.72 QEPeiodic Class Reference

### Classes

- struct [elementInfoStruct](#)
- struct [userInfoStructArray](#)

### Public Types

- enum **variableTypes** {  
**VARIABLE\_TYPE\_NUMBER**, **VARIABLE\_TYPE\_ATOMIC\_WEIGHT**, **VARIABLE\_TYPE\_MELTING\_POINT**, **VARIABLE\_TYPE\_BOILING\_POINT**,  
**VARIABLE\_TYPE\_DENSITY**, **VARIABLE\_TYPE\_GROUP**, **VARIABLE\_TYPE\_IONIZATION\_ENERGY**, **VARIABLE\_TYPE\_USER\_VALUE\_1**,  
**VARIABLE\_TYPE\_USER\_VALUE\_2** }
- enum **presentationOptions** { **PRESENTATION\_BUTTON\_AND\_LABEL**, **PRESENTATION\_BUTTON\_ONLY**, **PRESENTATION\_LABEL\_ONLY** }
- enum **UserLevels** { **User** = `userLevelTypes::USERLEVEL_USER`, **Scientist** = `userLevelTypes::USERLEVEL_SCIENTIST`, **Engineer** = `userLevelTypes::USERLEVEL_ENGINEER` }
- enum **PresentationOptions** { **buttonAndLabel** = `QEPeiodic::PRESENTATION_BUTTON_AND_LABEL`, **buttonOnly** = `QEPeiodic::PRESENTATION_BUTTON_ONLY`, **labelOnly** = `QEPeiodic::PRESENTATION_LABEL_ONLY` }
- enum **VariableTypes** {  
**Number** = `QEPeiodic::VARIABLE_TYPE_NUMBER`, **atomicWeight** = `QEPeiodic::VARIABLE_TYPE_ATOMIC_WEIGHT`, **meltingPoint** = `QEPeiodic::VARIABLE_TYPE_MELTING_POINT`, **boilingPoint** = `QEPeiodic::VARIABLE_TYPE_BOILING_POINT`,  
**density** = `QEPeiodic::VARIABLE_TYPE_DENSITY`, **group** = `QEPeiodic::VARIABLE_TYPE_GROUP`, **ionizationEnergy** = `QEPeiodic::VARIABLE_TYPE_IONIZATION_ENERGY`, **userValue1** = `QEPeiodic::VARIABLE_TYPE_USER_VALUE_1`,  
**userValue2** = `QEPeiodic::VARIABLE_TYPE_USER_VALUE_2` }

### Signals

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

### Public Member Functions

- **QEPeiodic** (QWidget \*parent=0)
- **QEPeiodic** (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** ()
- [UserLevels](#) **getUserLevelVisibilityProperty** ()  
*Access function for [userLevel/Visibility](#) property - refer to [userLevel/Visibility](#) property for details.*
- void [setUserLevelVisibilityProperty](#) ([UserLevels](#) level)  
*Access function for [userLevel/Visibility](#) property - refer to [userLevel/Visibility](#) property for details.*
- [UserLevels](#) **getUserLevelEnabledProperty** ()  
*Access function for [userLevel/Enabled](#) property - refer to [userLevel/Enabled](#) property for details.*
- void [setUserLevelEnabledProperty](#) ([UserLevels](#) level)  
*Access function for [userLevel/Enabled](#) property - refer to [userLevel/Enabled](#) property for details.*
- void **setPresentationOptionProperty** (PresentationOptions presentationOption)
- PresentationOptions **getPresentationOptionProperty** ()
- void **setVariableType1Property** (VariableTypes variableType)
- void **setVariableType2Property** (VariableTypes variableType)
- VariableTypes **getVariableType1Property** ()
- VariableTypes **getVariableType2Property** ()

## Public Attributes

- [userInfoStruct](#) **userInfo** [NUM\_ELEMENTS]

## Static Public Attributes

- static [elementInfoStruct](#) **elementInfo** [NUM\_ELEMENTS]

### Protected Member Functions

- void **establishConnection** (unsigned int variableIndex)
- void **dragEnterEvent** (QDragEnterEvent \*event)
- void **dropEvent** (QDropEvent \*event)
- void **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](#)
- PresentationOptions **presentationOption**
- VariableTypes **variableType1**
- VariableTypes **variableType2**
- QString **userInfo**

## 9.72.1 Member Enumeration Documentation

### 9.72.1.1 enum QEPeriodic::UserLevels

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

#### Enumerator:

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

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

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

## 9.72.2 Member Function Documentation

### 9.72.2.1 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.

### 9.72.2.2 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.

## 9.72.3 Member Data Documentation

### 9.72.3.1 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.

## 9.72.4 Property Documentation

### 9.72.4.1 bool QEPeriodic::displayAlarmState [read, write]

If set (default) widget will indicate the alarm state of any variable data 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.

**9.72.4.2 unsigned QEPeiodic::int** [read, write]

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

**9.72.4.3 QString QEPeiodic::readbackLabelVariable1** [read, write]

EPICS variable name (CA PV). This variable is used to read the value to the first of two positioners to determine which (if any) element is currently selected.

**9.72.4.4 QString QEPeiodic::readbackLabelVariable2** [read, write]

EPICS variable name (CA PV). This variable is used to read the value to the second of two positioners to determine which (if any) element is currently selected.

**9.72.4.5 bool QEPeiodic::subscribe** [read, write]

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

**9.72.4.6 UserLevels QEPeiodic::userLevelEnabled** [read, write]

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

**9.72.4.7 QString QEPeiodic::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.

**9.72.4.8 QString QEPeiodic::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.

#### 9.72.4.9 QString QEPeiodic::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.

#### 9.72.4.10 UserLevels QEPeiodic::userLevelVisibility [read, write]

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

#### 9.72.4.11 bool QEPeiodic::variableAsToolTip [read, write]

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

#### 9.72.4.12 QString QEPeiodic::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.

#### 9.72.4.13 bool QEPeiodic::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.

#### 9.72.4.14 QString QEPeiodic::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.



## 9.72.4.15 QString QEPERiodic::writeButtonVariable2 [read, write]

EPICS variable name (CA PV). This variable is used to write a value to the second of two positioners that will position the select element.

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

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

## 9.73 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

## 9.74 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

## 9.75 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`

## 9.76 QEPlot Class Reference

### Public Types

- enum `UserLevels` { `User` = `userLevelTypes::USERLEVEL_USER`, `Scientist` = `userLevelTypes::USERLEVEL_SCIENTIST`, `Engineer` = `userLevelTypes::USERLEVEL_ENGINEER` }
- enum `TraceStyles` { `Lines` = `QwtPlotCurve::Lines`, `Sticks` = `QwtPlotCurve::Sticks`, `Steps` = `QwtPlotCurve::Steps`, `Dots` = `QwtPlotCurve::Dots` }

### 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** ()
- [UserLevels](#) **getUserLevelVisibilityProperty** ()

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

- void [setUserLevelVisibilityProperty](#) (UserLevels level)  
*Access function for [userLevelVisibility](#) property - refer to [userLevelVisibility](#) property for details.*
- [UserLevels](#) [getUserLevelEnabledProperty](#) ()  
*Access function for [userLevelEnabled](#) property - refer to [userLevelEnabled](#) property for details.*
- void [setUserLevelEnabledProperty](#) (UserLevels level)  
*Access function for [userLevelEnabled](#) property - refer to [userLevelEnabled](#) property for details.*
- void **setTraceStyle1** (TraceStyles traceStyle)
- void **setTraceStyle2** (TraceStyles traceStyle)
- void **setTraceStyle3** (TraceStyles traceStyle)
- void **setTraceStyle4** (TraceStyles traceStyle)
- TraceStyles **getTraceStyle1** ()
- TraceStyles **getTraceStyle2** ()
- TraceStyles **getTraceStyle3** ()
- TraceStyles **getTraceStyle4** ()

### Protected Member Functions

- void **establishConnection** (unsigned int variableIndex)
- void **dragEnterEvent** (QDragEnterEvent \*event)
- void **dropEvent** (QDropEvent \*event)
- void **mousePressEvent** (QMouseEvent \*event)
- void **setDrop** (QVariant drop)
- QVariant **getDrop** ()
- 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 [userLevelUserStyle](#)
- QString [userLevelScientistStyle](#)
- QString [userLevelEngineerStyle](#)
- [UserLevels](#) [userLevelVisibility](#)
- [UserLevels](#) [userLevelEnabled](#)
- bool [displayAlarmState](#)
- 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**

### 9.76.1 Member Enumeration Documentation

#### 9.76.1.1 enum QEPlot::UserLevels

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

#### Enumerator:

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

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

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

### 9.76.2 Member Function Documentation

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

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

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

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

## 9.76.3 Member Data Documentation

### 9.76.3.1 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.

## 9.76.4 Property Documentation

### 9.76.4.1 bool QEPlot::displayAlarmState [read, write]

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

### 9.76.4.2 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.

### 9.76.4.3 UserLevels QEPlot::userLevelEnabled [read, write]

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

### 9.76.4.4 QString QEPlot::userLevelEngineerStyle [read, write]

Style Sheet string to be applied when the widget is displayed in 'Engineer' mode. Default is an empty string. The syntax is the standard Qt Style Sheet syntax. For example, 'background-color: red' This Style Sheet string will be applied by the `styleManager` class. Refer to the `styleManager` class for details about how this Style Sheet string

will be merged with any pre-existing Style Sheet string and any Style Sheet strings generated during the display of data.

#### 9.76.4.5 QString QEPlot::userLevelScientistStyle [read, write]

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

#### 9.76.4.6 QString QEPlot::userLevelUserStyle [read, write]

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

#### 9.76.4.7 UserLevels QEPlot::userLevelVisibility [read, write]

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

#### 9.76.4.8 QString QEPlot::variable1 [read, write]

EPICS variable name (CA PV). This variable is used to read updating values or waveforms for plotting in the first trace.

#### 9.76.4.9 QString QEPlot::variable2 [read, write]

EPICS variable name (CA PV). This variable is used to read updating values or waveforms for plotting in the second trace.

#### 9.76.4.10 QString QEPlot::variable3 [read, write]

EPICS variable name (CA PV). This variable is used to read updating values or waveforms for plotting in the third trace.

#### 9.76.4.11 QString QEPlot::variable4 [read, write]

EPICS variable name (CA PV). This variable is used to read updating values or waveforms for plotting in the fourth trace.

#### 9.76.4.12 bool QEPlot::variableAsToolTip [read, write]

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

#### 9.76.4.13 QString QEPlot::variableSubstitutions [read, write]

Macro substitutions. The default is no substitutions. The format is NAME1=VALUE1[, NAME2=VALUE2... Values may be quoted strings. For example, 'SAMPLE=SAM1, NAME = "Ref foil"' These substitutions are applied to all the variable names.

#### 9.76.4.14 bool QEPlot::visible [read, write]

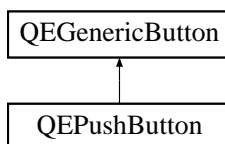
Display the widget. Default is true. Setting this property false is usefull if widget is only used to provide a signal - for example, when supplying data to a [QELink](#) widget. Note, when false the widget will still be visible in Qt Designer.

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

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

## 9.77 QEPushButton Class Reference

Inheritance diagram for QEPushButton:



### Public Types

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



- `Default` = `QStringFormatting::FORMAT_DEFAULT`, `Floating` = `QStringFormatting::FORMAT_FLOATING`, `Integer` = `QStringFormatting::FORMAT_INTEGER`, `UnsignedInteger` = `QStringFormatting::FORMAT_UNSIGNEDINTEGER`,  
`Time` = `QStringFormatting::FORMAT_TIME`, `LocalEnumeration` = `QStringFormatting::FORMAT_LOCAL_ENUMERATE` }
- enum `Notations` { `Fixed` = `QStringFormatting::NOTATION_FIXED`, `Scientific` = `QStringFormatting::NOTATION_SCIENTIFIC`, `Automatic` = `QStringFormatting::NOTATION_AUTOMATIC` }
  - enum `ArrayActions` { `Append` = `QStringFormatting::APPEND`, `Ascii` = `QStringFormatting::ASCII`, `Index` = `QStringFormatting::INDEX` }
  - enum `UpdateOptions` { `Text` = `QEGenericButton::UPDATE_TEXT`, `Icon` = `QEGenericButton::UPDATE_ICON`, `TextAndIcon` = `QEGenericButton::UPDATE_TEXT_AND_ICON`, `State` = `QEGenericButton::UPDATE_STATE` }
- User friendly enumerations for updateOption property - refer to QEGenericButton::updateOptions for details.*
- enum `ProgramStartupOptionNames` { `None` = `applicationLauncher::PSO_NONE`, `Terminal` = `applicationLauncher::PSO_TERMINAL`, `LogOutput` = `applicationLauncher::PSO_LOGOUTPUT`, `StdOutput` = `applicationLauncher::PSO_STDOUTPUT` }
  - enum `CreationOptionNames` {  
`Open` = `QEAActionRequests::OptionOpen`, `NewTab` = `QEAActionRequests::OptionNewTab`,  
`NewWindow` = `QEAActionRequests::OptionNewWindow`, `DockTop` = `QEAActionRequests::OptionTopDockWindow`,  
`DockBottom` = `QEAActionRequests::OptionBottomDockWindow`, `DockLeft` = `QEAActionRequests::OptionLeftDockWindow`, `DockRight` = `QEAActionRequests::OptionRightDockWindow`,  
`DockTopTabbed` = `QEAActionRequests::OptionTopDockWindowTabbed`,  
`DockBottomTabbed` = `QEAActionRequests::OptionBottomDockWindowTabbed`, `DockLeftTabbed` = `QEAActionRequests::OptionLeftDockWindowTabbed`, `DockRightTabbed` = `QEAActionRequests::OptionRightDockWindowTabbed`, `DockFloating` = `QEAActionRequests::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 `QEAActionRequests` &request)

## 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 `QEAActionRequests` &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 compelled.*

## Public Member Functions

- [QEPushButton](#) (QWidget \*parent=0)
- [QEPushButton](#) (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.*
- void [setFormatProperty](#) ([Formats](#) format)  
*Access function for [format](#) property - refer to [format](#) property for details.*
- [Formats](#) [getFormatProperty](#) ()  
*Access function for [format](#) property - refer to [format](#) property for details.*
- void [setNotationProperty](#) ([Notations](#) notation)  
*Access function for [notation](#) property - refer to [notation](#) property for details.*
- [Notations](#) [getNotationProperty](#) ()  
*Access function for [notation](#) property - refer to [notation](#) property for details.*
- void [setArrayActionProperty](#) ([ArrayActions](#) arrayAction)  
*Access function for [arrayAction](#) property - refer to [arrayAction](#) property for details.*
- [ArrayActions](#) [getArrayActionProperty](#) ()  
*Access function for [arrayAction](#) property - refer to [arrayAction](#) property for details.*

## Properties

- QString [variable](#)
- QString [altReadbackVariable](#)
- QString [variableSubstitutions](#)
- bool [subscribe](#)
- bool [variableAsToolTip](#)
- bool [allowDrop](#)
- bool [visible](#)

- unsigned [int](#)
- [QString](#) [userLevelUserStyle](#)
- [QString](#) [userLevelScientistStyle](#)
- [QString](#) [userLevelEngineerStyle](#)
- [UserLevels](#) [userLevelVisibility](#)
- [UserLevels](#) [userLevelEnabled](#)
- [bool](#) [displayAlarmState](#)
- [int](#) [precision](#)
- [bool](#) [useDbPrecision](#)
- [bool](#) [leadingZero](#)
- [bool](#) [trailingZeros](#)
- [bool](#) [addUnits](#)
- [QString](#) [localEnumeration](#)
- [Formats](#) [format](#)
- [Notations](#) [notation](#)
- [ArrayActions](#) [arrayAction](#)
- [Qt::Alignment](#) [alignment](#)
- [UpdateOptions](#) [updateOption](#)
- [QPixmap](#) [pixmap0](#)
- [QPixmap](#) [pixmap1](#)
- [QPixmap](#) [pixmap2](#)
- [QPixmap](#) [pixmap3](#)
- [QPixmap](#) [pixmap4](#)
- [QPixmap](#) [pixmap5](#)
- [QPixmap](#) [pixmap6](#)
- [QPixmap](#) [pixmap7](#)
- [QString](#) [password](#)
- [bool](#) [confirmAction](#)
- [QString](#) [confirmText](#)
- [bool](#) [writeOnPress](#)
- [bool](#) [writeOnRelease](#)
- [bool](#) [writeOnClick](#)
- [QString](#) [pressText](#)
- [QString](#) [releaseText](#)
- [QString](#) [clickText](#)
- [QString](#) [clickCheckedText](#)
- [QString](#) [labelText](#)
- [QString](#) [program](#)
- [QStringList](#) [arguments](#)
- [ProgramStartupOptionNames](#) [programStartupOption](#)
- [QString](#) [guiFile](#)
- [CreationOptionNames](#) [creationOption](#)
- [QString](#) [prioritySubstitutions](#)
- [QString](#) [customisationName](#)

### 9.77.1 Member Enumeration Documentation

#### 9.77.1.1 enum `QEPushButton::ArrayActions`

User friendly enumerations for arrayAction property - refer to `QStringFormatting::arrayActions` for details.

**Enumerator:**

***Append*** Refer to `QStringFormatting::APPEND` for details.

***Ascii*** Refer to `QStringFormatting::ASCII` for details.

***Index*** Refer to `QStringFormatting::INDEX` for details.

#### 9.77.1.2 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.

#### 9.77.1.3 enum `QEPushButton::Formats`

User friendly enumerations for format property - refer to `QStringFormatting::formats` for details.

**Enumerator:**

***Default*** Format as best appropriate for the data type.

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

**Integer** Format as an integer.

**UnsignedInteger** Format as an unsigned integer.

**Time** Format as a time.

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

#### 9.77.1.4 enum QEPushButton::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.

#### 9.77.1.5 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 doutput to standard output and standard error.

#### 9.77.1.6 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)

### 9.77.1.7 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.

### 9.77.2 Constructor & Destructor Documentation

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

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

#### 9.77.2.2 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.

### 9.77.3 Member Function Documentation

#### 9.77.3.1 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)

#### 9.77.3.2 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.

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

Button has been Pressed. The value emitted is the integer interpretation of the `press-Text` property

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

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

#### 9.77.3.5 void QEPushButton::requestAction ( const QActionRequests & *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.

### 9.77.4 Property Documentation

#### 9.77.4.1 bool QEPushButton::addUnits [read, write]

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

#### 9.77.4.2 Qt::Alignment QEPushButton::alignment [read, write]

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

#### 9.77.4.3 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.

#### 9.77.4.4 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.

#### 9.77.4.5 QStringList QEPushButton::arguments [read, write]

Arguments for program specified in the 'program' property.

#### 9.77.4.6 **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'.

#### 9.77.4.7 **QString QEPushButton::clickCheckedText** [read, write]

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

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

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

Reimplemented from [QEGenericButton](#).

#### 9.77.4.8 **QString QEPushButton::clickText** [read, write]

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

Reimplemented from [QEGenericButton](#).

#### 9.77.4.9 **bool QEPushButton::confirmAction** [read, write]

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

#### 9.77.4.10 **QString QEPushButton::confirmText** [read, write]

Text used to confirm action if confirmation dialog is presented

Reimplemented from [QEGenericButton](#).



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

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

Reimplemented from [QEGenericButton](#).

**9.77.4.12 QString QEPushButton::customisationName** [read, write]

Window customisation name. This name will be used to select a set of window customisations including menu items and tool bar buttons. Applications such as QEGui can load .xml files containing named sets of window customisations. This property is used to select a set loaded from these files. The selected set of customisations will be applied to the main window containing the new GUI. Customisations are not applied if the GUI is opened as a dock.

Reimplemented from [QEGenericButton](#).

**9.77.4.13 bool QEPushButton::displayAlarmState** [read, write]

If set (default) widget will indicate the alarm state of any variable data 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.

**9.77.4.14 Formats QEPushButton::format** [read, write]

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

**9.77.4.15 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.

**9.77.4.16 unsigned QEPushButton::int** [read, write]

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

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

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

#### 9.77.4.17 `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](#).

#### 9.77.4.18 `bool QEPushButton::leadingZero` [read, write]

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

#### 9.77.4.19 `QString QEPushButton::localEnumeration` [read, write]

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

Format is:

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

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

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

may be included in a string to indicate a line break

Examples are:

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

The data value is converted to a string if no enumeration for that value is available. For example, if the local enumeration is '0:off,1:on', and a value of 10 is processed, the

text generated is '10'. If a blank string is required, this should be explicit. for example, '0:off,1:on,10:'"

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

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

Notation used for numerical formatting. Default is fixed.

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

Password user will need to enter before any action is taken

Reimplemented from [QEGenericButton](#).

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

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

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

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

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

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

#### 9.77.4.25 QPixmap QEPushButton::pixmap3 [read, write]

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

#### 9.77.4.26 QPixmap QEPushButton::pixmap4 [read, write]

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

#### 9.77.4.27 QPixmap QEPushButton::pixmap5 [read, write]

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

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

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

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

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

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

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

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

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

Reimplemented from [QEGenericButton](#).

**9.77.4.32 QString QEPushButton::prioritySubstitutions** [read, write]

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

Reimplemented from [QEGenericButton](#).

**9.77.4.33 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

**9.77.4.34 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.

**9.77.4.35** `QString QEPushButton::releaseText` [read, write]

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

Reimplemented from [QEGenericButton](#).

**9.77.4.36** `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)

**9.77.4.37** `bool QEPushButton::trailingZeros` [read, write]

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

**9.77.4.38** `UpdateOptions QEPushButton::updateOption` [read, write]

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

Reimplemented from [QEGenericButton](#).

**9.77.4.39** `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.

**9.77.4.40** `UserLevels QEPushButton::userLevelEnabled` [read, write]

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

**9.77.4.41** `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.

**9.77.4.42 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.

**9.77.4.43 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.

**9.77.4.44 UserLevels QEPushButton::userLevelVisibility** [read, write]

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

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

EPICS variable name (CA PV). This variable is used for both writing (on button press), and reading if subscribed and no alternate readback variable is provided.

**9.77.4.46 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.

**9.77.4.47 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 are also used for other purposes.

**9.77.4.48** `bool QEPushButton::visible` [read, write]

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

**9.77.4.49** `bool QEPushButton::writeOnClick` [read, write]

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

**9.77.4.50** `bool QEPushButton::writeOnPress` [read, write]

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

**9.77.4.51** `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

## 9.78 QEPVNameLists Class Reference

### Public Member Functions

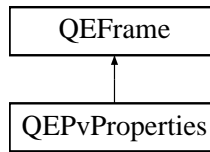
- void **prependOrMoveToFirst** (const QString &item)
- void **saveConfiguration** (PMElement &parentElement)
- void **restoreConfiguration** (PMElement &parentElement)

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

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

## 9.79 QEPvProperties Class Reference

Inheritance diagram for QEPvProperties:



## Signals

- void **setCurrentBoxIndex** (int index)

## Public Member Functions

- **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 \* **createQcaltem** (unsigned int variableIndex)
- void **mousePressEvent** (QMouseEvent \*event)
- void **dragEnterEvent** (QDragEnterEvent \*event)
- void **dropEvent** (QDropEvent \*event)
- void **saveConfiguration** (PersistenceManager \*pm)
- void **restoreConfiguration** (PersistenceManager \*pm, restorePhases restorePhase)
- QString **copyVariable** ()
- QVariant **copyData** ()
- void **paste** (QVariant s)

## Properties

- QString [variable](#)
- QString [variableSubstitutions](#)

### 9.79.1 Property Documentation

#### 9.79.1.1 QString QEPvProperties::variable [read, write]

EPICS variable name (CA PV)



## 9.79.1.2 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

## 9.80 QEPvPropertiesManager Class Reference

### Public Member Functions

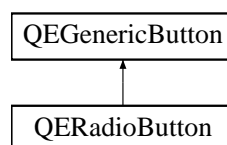
- **QEPvPropertiesManager** (QObject \*parent=0)
- bool **isContainer** () const
- bool **isInitialized** () const
- QIcon **icon** () const
- QString **group** () const
- QString **includeFile** () const
- QString **name** () const
- QString **toolTip** () const
- QString **whatsThis** () const
- QWidget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

## 9.81 QERadioButton Class Reference

Inheritance diagram for QERadioButton:



## Public Types

- enum [UserLevels](#) { [User](#) = userLevelTypes::USERLEVEL\_USER, [Scientist](#) = userLevelTypes::USERLEVEL\_SCIENTIST, [Engineer](#) = userLevelTypes::USERLEVEL\_ENGINEER }
  - enum [Formats](#) {  
[Default](#) = QQStringFormatting::FORMAT\_DEFAULT, [Floating](#) = QQStringFormatting::FORMAT\_FLOATING, [Integer](#) = QQStringFormatting::FORMAT\_INTEGER, [UnsignedInteger](#) = QQStringFormatting::FORMAT\_UNSIGNEDINTEGER,  
[Time](#) = QQStringFormatting::FORMAT\_TIME, [LocalEnumeration](#) = QQStringFormatting::FORMAT\_LOCAL\_ENUMERATE }
  - enum [Notations](#) { [Fixed](#) = QQStringFormatting::NOTATION\_FIXED, [Scientific](#) = QQStringFormatting::NOTATION\_SCIENTIFIC, [Automatic](#) = QQStringFormatting::NOTATION\_AUTOMATIC }
  - enum [ArrayActions](#) { [Append](#) = QQStringFormatting::APPEND, [Ascii](#) = QQStringFormatting::ASCII, [Index](#) = QQStringFormatting::INDEX }
  - enum [UpdateOptions](#) { [Text](#) = QEGenericButton::UPDATE\_TEXT, [Icon](#) = QEGenericButton::UPDATE\_ICON, [TextAndIcon](#) = QEGenericButton::UPDATE\_TEXT\_AND\_ICON, [State](#) = QEGenericButton::UPDATE\_STATE }
- User friendly enumerations for updateOption property - refer to QEGenericButton::updateOptions for details.*
- enum [ProgramStartupOptionNames](#) { [None](#) = applicationLauncher::PSO\_NONE, [Terminal](#) = applicationLauncher::PSO\_TERMINAL, [LogOutput](#) = applicationLauncher::PSO\_LOGOUTPUT, [StdOutput](#) = applicationLauncher::PSO\_STDOUTPUT }
  - enum [CreationOptionNames](#) {  
[Open](#) = QEActionRequests::OptionOpen, [NewTab](#) = QEActionRequests::OptionNewTab,  
[NewWindow](#) = QEActionRequests::OptionNewWindow, [DockTop](#) = QEActionRequests::OptionTopDockWindow,  
[DockBottom](#) = QEActionRequests::OptionBottomDockWindow, [DockLeft](#) = QEActionRequests::OptionLeftDockWindow, [DockRight](#) = QEActionRequests::OptionRightDockWindow,  
[DockTopTabbed](#) = QEActionRequests::OptionTopDockWindowTabbed,  
[DockBottomTabbed](#) = QEActionRequests::OptionBottomDockWindowTabbed, [DockLeftTabbed](#) = QEActionRequests::OptionLeftDockWindowTabbed, [DockRightTabbed](#) = QEActionRequests::OptionRightDockWindowTabbed, [DockFloating](#) = QEActionRequests::OptionFloatingDockWindow }

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

## Public Slots

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

## 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 QActionRequests &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)
- [UserLevels](#) [getUserLevelVisibilityProperty](#) ()  
*Access function for [userLevelVisibility](#) property - refer to [userLevelVisibility](#) property for details.*
- void [setUserLevelVisibilityProperty](#) ([UserLevels](#) level)  
*Access function for [userLevelVisibility](#) property - refer to [userLevelVisibility](#) property for details.*
- [UserLevels](#) [getUserLevelEnabledProperty](#) ()  
*Access function for [userLevelEnabled](#) property - refer to [userLevelEnabled](#) property for details.*
- void [setUserLevelEnabledProperty](#) ([UserLevels](#) level)  
*Access function for [userLevelEnabled](#) property - refer to [userLevelEnabled](#) property for details.*
- void [setFormatProperty](#) ([Formats](#) format)  
*Access function for [format](#) property - refer to [format](#) property for details.*
- [Formats](#) [getFormatProperty](#) ()  
*Access function for [format](#) property - refer to [format](#) property for details.*
- void [setNotationProperty](#) ([Notations](#) notation)  
*Access function for [notation](#) property - refer to [notation](#) property for details.*
- [Notations](#) [getNotationProperty](#) ()  
*Access function for [notation](#) property - refer to [notation](#) property for details.*
- void [setArrayActionProperty](#) ([ArrayActions](#) arrayAction)  
*Access function for [arrayAction](#) property - refer to [arrayAction](#) property for details.*
- [ArrayActions](#) [getArrayActionProperty](#) ()  
*Access function for [arrayAction](#) property - refer to [arrayAction](#) property for details.*

## Properties

- QString [variable](#)
- QString [variableSubstitutions](#)
- bool [subscribe](#)
- bool [variableAsToolTip](#)
- bool [allowDrop](#)
- bool [visible](#)
- unsigned [int](#)
- QString [userLevelUserStyle](#)
- QString [userLevelScientistStyle](#)
- QString [userLevelEngineerStyle](#)
- [UserLevels](#) [userLevelVisibility](#)
- [UserLevels](#) [userLevelEnabled](#)
- bool [displayAlarmState](#)
- int [precision](#)
- bool [useDbPrecision](#)
- bool [leadingZero](#)
- bool [trailingZeros](#)
- bool [addUnits](#)
- QString [localEnumeration](#)
- [Formats](#) [format](#)
- [Notations](#) [notation](#)
- [ArrayActions](#) [arrayAction](#)
- Qt::Alignment [alignment](#)
- [UpdateOptions](#) [updateOption](#)
- QPixmap [pixmap0](#)
- QPixmap [pixmap1](#)
- QPixmap [pixmap2](#)
- QPixmap [pixmap3](#)
- QPixmap [pixmap4](#)
- QPixmap [pixmap5](#)
- QPixmap [pixmap6](#)
- QPixmap [pixmap7](#)
- QString [password](#)
- bool [confirmAction](#)
- QString [confirmText](#)
- bool [writeOnPress](#)
- bool [writeOnRelease](#)
- bool [writeOnClick](#)
- QString [pressText](#)
- QString [releaseText](#)
- QString [clickText](#)
- QString [clickCheckedText](#)
- QString [labelText](#)
- QString [program](#)
- QStringList [arguments](#)

- [ProgramStartupOptionNames](#) `programStartupOption`
- `QString` `guiFile`
- [CreationOptionNames](#) `creationOption`
- `QString` `prioritySubstitutions`
- `QString` `customisationName`

### 9.81.1 Member Enumeration Documentation

#### 9.81.1.1 enum `QERadioButton::ArrayActions`

User friendly enumerations for `arrayAction` property - refer to `QStringFormatting::arrayActions` for details.

##### Enumerator:

**Append** Refer to `QStringFormatting::APPEND` for details.

**Ascii** Refer to `QStringFormatting::ASCII` for details.

**Index** Refer to `QStringFormatting::INDEX` for details.

#### 9.81.1.2 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.

### 9.81.1.3 enum `QERadioButton::Formats`

User friendly enumerations for format property - refer to `QStringFormatting::formats` for details.

#### Enumerator:

- Default** Format as best appropriate for the data type.
- Floating** Format as a floating point number.
- Integer** Format as an integer.
- UnsignedInteger** Format as an unsigned integer.
- Time** Format as a time.
- LocalEnumeration** Format as a selection from the [localEnumeration](#) property.

### 9.81.1.4 enum `QERadioButton::Notations`

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

#### Enumerator:

- Fixed** Refer to `QStringFormatting::NOTATION_FIXED` for details.
- Scientific** Refer to `QStringFormatting::NOTATION_SCIENTIFIC` for details.
- Automatic** Refer to `QStringFormatting::NOTATION_AUTOMATIC` for details.

### 9.81.1.5 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.

### 9.81.1.6 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)

**9.81.1.7 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.

**9.81.2 Constructor & Destructor Documentation****9.81.2.1 QERadioButton::QERadioButton ( QWidget \* *parent* = 0 )**

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

**9.81.2.2 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.

**9.81.3 Member Function Documentation****9.81.3.1 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)

**9.81.3.2 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.

### 9.81.3.3 void QERadioButton::pressed ( int *value* ) [signal]

Button has been Pressed. The value emitted is the integer interpretation of the press-Text property

### 9.81.3.4 void QERadioButton::released ( int *value* ) [signal]

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

### 9.81.3.5 void 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.

## 9.81.4 Property Documentation

### 9.81.4.1 bool QERadioButton::addUnits [read, write]

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

### 9.81.4.2 Qt::Alignment QERadioButton::alignment [read, write]

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

### 9.81.4.3 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.

### 9.81.4.4 QStringList QERadioButton::arguments [read, write]

Arguments for program specified in the 'program' property.



#### 9.81.4.5 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'.

#### 9.81.4.6 QString QERadioButton::clickCheckedText [read, write]

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

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

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

Reimplemented from [QEGenericButton](#).

#### 9.81.4.7 QString QERadioButton::clickText [read, write]

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

Reimplemented from [QEGenericButton](#).

#### 9.81.4.8 bool QERadioButton::confirmAction [read, write]

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

#### 9.81.4.9 QString QERadioButton::confirmText [read, write]

Text used to confirm action if confirmation dialog is presented

Reimplemented from [QEGenericButton](#).

#### 9.81.4.10 **CreationOptionNames** `QERadioButton::creationOption` [read, write]

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

Reimplemented from [QEGenericButton](#).

#### 9.81.4.11 **QString** `QERadioButton::customisationName` [read, write]

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

Reimplemented from [QEGenericButton](#).

#### 9.81.4.12 **bool** `QERadioButton::displayAlarmState` [read, write]

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

#### 9.81.4.13 **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).

#### 9.81.4.14 **QString** `QERadioButton::guiFile` [read, write]

File name of GUI to be presented on button click. File name can be absolute, relative to the path of the QEform in which the [QEPushButton](#) is located, relative to the any path in the path list published in the ContainerProfile class, or relative to the current path. See `QEWidget::openQEFile()` in `QEWidget.cpp` for details.

#### 9.81.4.15 **unsigned** `QERadioButton::int` [read, write]

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

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

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

#### 9.81.4.16 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](#).

#### 9.81.4.17 bool QERadioButton::leadingZero [read, write]

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

#### 9.81.4.18 QString QERadioButton::localEnumeration [read, write]

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

Format is:

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

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

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

may be included in a string to indicate a line break

Examples are:

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

The data value is converted to a string if no enumeration for that value is available. For example, if the local enumeration is '0:off,1:on', and a value of 10 is processed, the

text generated is '10'. If a blank string is required, this should be explicit. for example, '0:off,1:on,10:'''

A range of numbers can be covered by a pair of values as in the following example:  $\geq 4$ :"Between 4 and 8",  $\leq 8$ :"Between 4 and 8"

#### 9.81.4.19 Notations QERadioButton::notation [read, write]

Notation used for numerical formatting. Default is fixed.

#### 9.81.4.20 QString QERadioButton::password [read, write]

Password user will need to enter before any action is taken

Reimplemented from [QEGenericButton](#).

#### 9.81.4.21 QPixmap QERadioButton::pixmap0 [read, write]

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

#### 9.81.4.22 QPixmap QERadioButton::pixmap1 [read, write]

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

#### 9.81.4.23 QPixmap QERadioButton::pixmap2 [read, write]

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

#### 9.81.4.24 QPixmap QERadioButton::pixmap3 [read, write]

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

#### 9.81.4.25 QPixmap QERadioButton::pixmap4 [read, write]

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

#### 9.81.4.26 QPixmap QERadioButton::pixmap5 [read, write]

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

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

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

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

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

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

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

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

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

Reimplemented from [QEGenericButton](#).

**9.81.4.31 QString QERadioButton::prioritySubstitutions** [read, write]

Overriding macro substitutions. These macro substitutions take precedence over any existing macro substitutions defined by the variableSubstitutions property, any parent forms, or the application containing the button. These macro substitutions are particularly 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 substittions applied to the new form, they are appended to the list and the existing macro substitutions take precedence.

Reimplemented from [QEGenericButton](#).

**9.81.4.32 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

**9.81.4.33 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.

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

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

Reimplemented from [QEGenericButton](#).

**9.81.4.35 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)

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

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

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

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

Reimplemented from [QEGenericButton](#).

**9.81.4.38 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.

**9.81.4.39 UserLevels QERadioButton::userLevelEnabled** [read, write]

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

**9.81.4.40 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.

**9.81.4.41** `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.

**9.81.4.42** `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.

**9.81.4.43** `UserLevels QERadioButton::userLevelVisibility` [read, write]

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

**9.81.4.44** `QString QERadioButton::variable` [read, write]

EPICS variable name (CA PV)

**9.81.4.45** `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.

**9.81.4.46** `QString QERadioButton::variableSubstitutions` [read, write]

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

#### 9.81.4.47 `bool QERadioButton::visible` [read, write]

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

#### 9.81.4.48 `bool QERadioButton::writeOnClick` [read, write]

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

Reimplemented from [QEGenericButton](#).

#### 9.81.4.49 `bool QERadioButton::writeOnPress` [read, write]

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

Reimplemented from [QEGenericButton](#).

#### 9.81.4.50 `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

## 9.82 QERecipe Class Reference

### Public Types

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

### Public Member Functions

- **QERecipe** (QWidget \*pParent=0)
- void **setRecipeDescription** (QString pValue)
- QString **getRecipeDescription** ()
- void **setShowRecipeList** (bool pValue)



- bool **getShowRecipeList** ()
- void **setShowNew** (bool pValue)
- bool **getShowNew** ()
- void **setShowSave** (bool pValue)
- bool **getShowSave** ()
- void **setShowDelete** (bool pValue)
- bool **getShowDelete** ()
- void **setShowApply** (bool pValue)
- bool **getShowApply** ()
- void **setShowRead** (bool pValue)
- bool **getShowRead** ()
- void **setShowFields** (bool pValue)
- bool **getShowFields** ()
- void **setConfigurationType** (int pValue)
- int **getConfigurationType** ()
- void **setConfigurationFile** (QString pValue)
- QString **getConfigurationFile** ()
- void **setRecipeFile** (QString pValue)
- QString **getRecipeFile** ()
- void **setConfigurationText** (QString pValue)
- QString **getConfigurationText** ()
- void **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/QERecipe/QERecipe.h
- /tmp/epicsqt/trunk/framework/widgets/QERecipe/QERecipe.cpp

## 9.83 QERecordFieldName Class Reference

### Static Public Member Functions

- static QString **recordName** (const QString &pvName)
- static QString **fieldName** (const QString &pvName)
- static QString **fieldPvName** (const QString &pvName, const QString &field)
- static QString **rtypePvName** (const QString &pvName)
- static bool **pvNamelsValid** (const QString &pvName)
- static bool **extractPvName** (const QString &item, QString &pvName)

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

- /tmp/epicsqt/trunk/framework/widgets/QEPvProperties/QEPvPropertiesUtilities.h
- /tmp/epicsqt/trunk/framework/widgets/QEPvProperties/QEPvPropertiesUtilities.cpp

## 9.84 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

## 9.85 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

## 9.86 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 }

### 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.*

### 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 [scriptIdFile](#)
  - Define the file where to save the scripts (if not defined then the scripts will be saved in a file named "QEScript.xml")*
- QString [scriptIdText](#)
  - Define the XML text that contains the scripts.*
- QString [scriptIdDefault](#)

*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 RIG.

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

### 9.86.1 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.

### 9.86.2 Member Enumeration Documentation

#### 9.86.2.1 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.

### 9.86.3 Property Documentation

#### 9.86.3.1 `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.

#### 9.86.3.2 `bool QEScript::displayAlarmState` [read, write]

If set (default) widget will indicate the alarm state of any variable data 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.

#### 9.86.3.3 `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.

#### 9.86.3.4 `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 programatically through `setUserLevel()`. Widgets that are always accessible should be visible at 'User'. Widgets that are only accessible to scientists managing the facility should be visible at 'Scientist'. Widgets that are only accessible to engineers maintaining the facility should be visible at 'Engineer'.

#### 9.86.3.5 `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.

#### 9.86.3.6 `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.

#### 9.86.3.7 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.

#### 9.86.3.8 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 programatically through `setUserLevel()` Widgets that are always visible should be visible at 'User'. Widgets that are only used by scientists managing the facility should be visible at 'Scientist'. Widgets that are only used by engineers maintaining the facility should be visible at 'Engineer'.

#### 9.86.3.9 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.

#### 9.86.3.10 bool QEScript::visible [read, write]

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

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

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

## 9.87 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,**
- Transperency, Rotation, ColourHue, ColourSaturation,**
- ColourValue, ColourIndex, Penwidth }**
- enum [UserLevels](#) { [User](#) = userLevelTypes::USERLEVEL\_USER, [Scientist](#) = userLevelTypes::USERLEVEL\_SCIENTIST, [Engineer](#) = userLevelTypes::USERLEVEL\_ENGINEER }

## Signals

- void [dbValueChanged1](#) (const qlonglong &out)
- void [dbValueChanged2](#) (const qlonglong &out)
- void [dbValueChanged3](#) (const qlonglong &out)
- void [dbValueChanged4](#) (const qlonglong &out)
- void [dbValueChanged5](#) (const qlonglong &out)
- void [dbValueChanged6](#) (const qlonglong &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.*
- [UserLevels](#) [getUserLevelVisibilityProperty](#) ()  
*Access function for [userLevelVisibility](#) property - refer to [userLevelVisibility](#) property for details.*
- void [setUserLevelVisibilityProperty](#) ([UserLevels](#) level)  
*Access function for [userLevelVisibility](#) property - refer to [userLevelVisibility](#) property for details.*
- [UserLevels](#) [getUserLevelEnabledProperty](#) ()  
*Access function for [userLevelEnabled](#) property - refer to [userLevelEnabled](#) property for details.*
- void [setUserLevelEnabledProperty](#) ([UserLevels](#) level)  
*Access function for [userLevelEnabled](#) property - refer to [userLevelEnabled](#) property for details.*

## Properties

- QString [variable1](#)
- QString [variable2](#)
- QString [variable3](#)
- QString [variable4](#)
- QString [variable5](#)
- QString [variable6](#)
- QString [variableSubstitutions](#)
- bool [variableAsToolTip](#)
- bool [allowDrop](#)
- bool [visible](#)
- unsigned int
- QString [userLevelUserStyle](#)
- QString [userLevelScientistStyle](#)
- QString [userLevelEngineerStyle](#)
- [UserLevels](#) [userLevelVisibility](#)
- [UserLevels](#) [userLevelEnabled](#)
- bool [displayAlarmState](#)
- [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](#)

### 9.87.1 Detailed Description

This class is a EPICS aware shape widget based on the Qt widget. One of several shapes can be drawn within the widget, and up to 6 variables can be used to animate various attributes of the shape. For example to represent beam positino and size, an ellipse can be drawn with four variables animating its vertcal and horizontal size and position. It is tightly integrated with the base class QEWidget which provides generic support such as macro substitutions, drag/drop, and standard properties.

### 9.87.2 Member Enumeration Documentation

#### 9.87.2.1 enum QEShape::animationOptions

Options for how a variable will animate the shape.

### 9.87.2.2 enum QEShape::shapeOptions

Options for the type of shape.

### 9.87.2.3 enum QEShape::UserLevels

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

#### Enumerator:

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

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

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

## 9.87.3 Constructor & Destructor Documentation

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

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

### 9.87.3.2 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.

## 9.87.4 Member Function Documentation

### 9.87.4.1 void QEShape::dbValueChanged1 ( const qlonglong & out ) [signal]

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

### 9.87.4.2 void QEShape::dbValueChanged2 ( const qlonglong & out ) [signal]

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

**9.87.4.3** void QEShape::dbValueChanged3 ( const qlonglong & *out* ) [signal]

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

**9.87.4.4** void QEShape::dbValueChanged4 ( const qlonglong & *out* ) [signal]

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

**9.87.4.5** void QEShape::dbValueChanged5 ( const qlonglong & *out* ) [signal]

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

**9.87.4.6** void QEShape::dbValueChanged6 ( const qlonglong & *out* ) [signal]

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

## 9.87.5 Property Documentation

**9.87.5.1** 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.

**9.87.5.2** 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.

**9.87.5.3** 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.

**9.87.5.4 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.

**9.87.5.5 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.

**9.87.5.6 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.

**9.87.5.7 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.

**9.87.5.8 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'.

**9.87.5.9 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'.

**9.87.5.10 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'.

**9.87.5.11 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'.

**9.87.5.12 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'.



**9.87.5.13 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'.

**9.87.5.14 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'.

**9.87.5.15 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'.

**9.87.5.16 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'.

**9.87.5.17 QColor QEShape::color9** [read, write]

Used by the color animation to determine the color based on a data value. The scaled and offset data is used as an index to select color properties 'color1' to 'color10'.

**9.87.5.18 bool QEShape::displayAlarmState** [read, write]

If set (default) widget will indicate the alarm state of any variable data 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.

**9.87.5.19 unsigned QEShape::int** [read, write]

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

The number of points to use when drawing shapes that are defined by a variable number of points, such as polyline, polygon, path, and series of points.

Sets the width of the pen. Used for the following shapes: Line, Points, Polyline, Polygon, Rect, RoundedRect, Ellipse, Arc, Chord, Pie, Path

**9.87.5.20** `double QEShape::offset1` [read, write]

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

**9.87.5.21** `double QEShape::offset2` [read, write]

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

**9.87.5.22** `double QEShape::offset3` [read, write]

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

**9.87.5.23** `double QEShape::offset4` [read, write]

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

**9.87.5.24** `double QEShape::offset5` [read, write]

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

**9.87.5.25** `double QEShape::offset6` [read, write]

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

**9.87.5.26** `QPoint QEShape::point1` [read, write]

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

**9.87.5.27** `QPoint QEShape::point10` [read, write]

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

**9.87.5.28** `QPoint QEShape::point2` [read, write]

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

**9.87.5.41 UserLevels QEShape::userLevelEnabled** [read, write]

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

**9.87.5.42 QString QEShape::userLevelEngineerStyle** [read, write]

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

**9.87.5.43 QString QEShape::userLevelScientistStyle** [read, write]

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

**9.87.5.44** `QString QEShape::userLevelUserStyle` [read, write]

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

**9.87.5.45** `UserLevels QEShape::userLevelVisibility` [read, write]

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

**9.87.5.46** `QString QEShape::variable1` [read, write]

EPICS variable name (CA PV). This variable is read and used to animate an attribute of the shape. The value read is first scaled and offset by properties scale1 and offset1 then the attribute selected for animation is selected by the property animation1.

**9.87.5.47** `QString QEShape::variable2` [read, write]

EPICS variable name (CA PV). This variable is read and used to animate an attribute of the shape. The value read is first scaled and offset by properties scale2 and offset2 then the attribute selected for animation is selected by the property animation2.

**9.87.5.48** `QString QEShape::variable3` [read, write]

EPICS variable name (CA PV). This variable is read and used to animate an attribute of the shape. The value read is first scaled and offset by properties scale3 and offset3 then the attribute selected for animation is selected by the property animation3.

**9.87.5.49** `QString QEShape::variable4` [read, write]

EPICS variable name (CA PV). This variable is read and used to animate an attribute of the shape. The value read is first scaled and offset by properties scale4 and offset4 then the attribute selected for animation is selected by the property animation4.

**9.87.5.50 QString QEShape::variable5** [read, write]

EPICS variable name (CA PV). This variable is read and used to animate an attribute of the shape. The value read is first scaled and offset by properties scale5 and offset5 then the attribute selected for animation is selected by the property animation5.

**9.87.5.51 QString QEShape::variable6** [read, write]

EPICS variable name (CA PV). This variable is read and used to animate an attribute of the shape. The value read is first scaled and offset by properties scale6 and offset6 then the attribute selected for animation is selected by the property animation6.

**9.87.5.52 bool QEShape::variableAsToolTip** [read, write]

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

**9.87.5.53 QString QEShape::variableSubstitutions** [read, write]

Macro substitutions. The default is no substitutions. The format is NAME1=VALUE1[, NAME2=VALUE2... Values may be quoted strings. For example, 'SAMPLE=SAM1, NAME = "Ref foil"' These substitutions are applied to all the variable names.

**9.87.5.54 bool QEShape::visible** [read, write]

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

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

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

## 9.88 QESlider Class Reference

### Public Types

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

### Signals

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

## Public Member Functions

- **QESlider** (QWidget \*parent=0)
- **QESlider** (const QString &variableName, QWidget \*parent=0)
- void **setWriteOnChange** (bool [writeOnChange](#))
- bool **getWriteOnChange** ()
- void **setSubscribe** (bool subscribe)
- bool **getSubscribe** ()
- void **setScale** (double scaleIn)
- double **getScale** ()
- void **setOffset** (double offsetIn)
- double **getOffset** ()
- [UserLevels](#) **getUserLevelVisibilityProperty** ()
 

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

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

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

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

## Protected Member Functions

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

## Protected Attributes

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

## Properties

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

## 9.88.1 Member Enumeration Documentation

### 9.88.1.1 enum QESlider::UserLevels

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

#### Enumerator:

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

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

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

## 9.88.2 Member Function Documentation

### 9.88.2.1 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.

## 9.88.3 Member Data Documentation

### 9.88.3.1 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).



### 9.88.4 Property Documentation

#### 9.88.4.1 `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.

#### 9.88.4.2 `bool QESlider::displayAlarmState` [read, write]

If set (default) widget will indicate the alarm state of any variable data 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.

#### 9.88.4.3 `unsigned QESlider::int` [read, write]

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

#### 9.88.4.4 `bool QESlider::subscribe` [read, write]

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

#### 9.88.4.5 `UserLevels QESlider::userLevelEnabled` [read, write]

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

#### 9.88.4.6 `QString QESlider::userLevelEngineerStyle` [read, write]

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

#### 9.88.4.7 QString QESlider::userLevelScientistStyle [read, write]

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

#### 9.88.4.8 QString QESlider::userLevelUserStyle [read, write]

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

#### 9.88.4.9 UserLevels QESlider::userLevelVisibility [read, write]

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

#### 9.88.4.10 QString QESlider::variable [read, write]

EPICS variable name (CA PV)

#### 9.88.4.11 bool QESlider::variableAsToolTip [read, write]

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

#### 9.88.4.12 QString QESlider::variableSubstitutions [read, write]

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

## 9.88.4.13 bool QESlider::visible [read, write]

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

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

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

## 9.89 QESpinBox Class Reference

### Public Types

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

### Signals

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

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

### Public Member Functions

- **QESpinBox** (QWidget \*parent=0)
- **QESpinBox** (const QString &variableName, QWidget \*parent=0)
- void **setWriteOnChange** (bool writeOnChangeIn)
- bool **getWriteOnChange** ()
- void **setSubscribe** (bool subscribe)
- bool **getSubscribe** ()
- void **setAddUnitsAsSuffix** (bool addUnitsAsSuffixIn)
- bool **getAddUnitsAsSuffix** ()
- void **setUseDbPrecisionForDecimals** (bool useDbPrecisionForDecimalln)
- bool **getUseDbPrecisionForDecimals** ()
- [UserLevels](#) **getUserLevelVisibilityProperty** ()  
*Access function for [userLevelVisibility](#) property - refer to [userLevelVisibility](#) property for details.*
- void **setUserLevelVisibilityProperty** ([UserLevels](#) level)  
*Access function for [userLevelVisibility](#) property - refer to [userLevelVisibility](#) property for details.*
- [UserLevels](#) **getUserLevelEnabledProperty** ()

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

- void [setUserLevelEnabledProperty](#) (UserLevels level)

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

### Protected Member Functions

- void **establishConnection** (unsigned int variableIndex)
- void **dragEnterEvent** (QDragEnterEvent \*event)
- void **dropEvent** (QDropEvent \*event)
- void **setDrop** (QVariant drop)
- QVariant **getDrop** ()
- QString **copyVariable** ()
- QVariant **copyData** ()
- void **paste** (QVariant s)
- QMenu \* **getDefaultContextMenu** ()

### Protected Attributes

- QEFloatingFormatting **floatingFormatting**
- bool **writeOnChange**
- bool **addUnitsAsSuffix**
- bool **useDbPrecisionForDecimal**

### Properties

- QString [variable](#)
- QString [variableSubstitutions](#)
- bool [variableAsToolTip](#)
- bool [allowDrop](#)
- bool [visible](#)
- unsigned [int](#)
- QString [userLevelUserStyle](#)
- QString [userLevelScientistStyle](#)
- QString [userLevelEngineerStyle](#)
- [UserLevels](#) [userLevelVisibility](#)
- [UserLevels](#) [userLevelEnabled](#)
- bool [displayAlarmState](#)
- bool [subscribe](#)
- bool **useDbPrecision**
- bool **addUnits**

## 9.89.1 Member Enumeration Documentation

### 9.89.1.1 enum QESpinBox::UserLevels

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

#### Enumerator:

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

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

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

## 9.89.2 Member Function Documentation

### 9.89.2.1 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.

## 9.89.3 Property Documentation

### 9.89.3.1 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.

### 9.89.3.2 bool QESpinBox::displayAlarmState [read, write]

If set (default) widget will indicate the alarm state of any variable data 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.

### 9.89.3.3 unsigned QESpinBox::int [read, write]

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

#### 9.89.3.4 `bool QESpinBox::subscribe` [read, write]

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

#### 9.89.3.5 `UserLevels QESpinBox::userLevelEnabled` [read, write]

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

#### 9.89.3.6 `QString QESpinBox::userLevelEngineerStyle` [read, write]

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

#### 9.89.3.7 `QString QESpinBox::userLevelScientistStyle` [read, write]

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

#### 9.89.3.8 `QString QESpinBox::userLevelUserStyle` [read, write]

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

#### 9.89.3.9 `UserLevels QESpinBox::userLevelVisibility` [read, write]

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

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

**9.89.3.10** `QString QESpinBox::variable` [read, write]

EPICS variable name (CA PV)

**9.89.3.11** `bool QESpinBox::variableAsToolTip` [read, write]

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

**9.89.3.12** `QString QESpinBox::variableSubstitutions` [read, write]

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

**9.89.3.13** `bool QESpinBox::visible` [read, write]

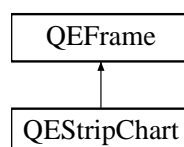
Display the widget. Default is true. Setting this property false is usefull if widget is only used to provide a signal - for example, when supplying data to a [QELink](#) widget. Note, when false the widget will still be visible in Qt Designer.

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

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

## 9.90 QEStriptChart Class Reference

Inheritance diagram for QEStriptChart:



## Public Types

- enum **Constants** { **NUMBER\_OF\_PVS** = 12 }

## Public Member Functions

- **QEStripChart** (QWidget \*parent=0)
- QSize **sizeHint** () const
- QDateTime **getStartDateTime** ()
- QDateTime **getEndDateTime** ()
- void **setEndDateTime** (QDateTime endTimeIn)
- int **getDuration** ()
- void **setDuration** (int durationIn)
- double **getYMinimum** ()
- void **setYMinimum** (const double yMinimumIn)
- double **getYMaximum** ()
- void **setYMaximum** (const double yMaximumIn)
- void **setYRange** (const double yMinimumIn, const double yMaximumIn)

## Protected Member Functions

- void **mousePressEvent** (QMouseEvent \*event)
- void **dragEnterEvent** (QDragEnterEvent \*event)
- void **dropEvent** (QDropEvent \*event)
- QString **copyVariable** ()
- QVariant **copyData** ()
- void **paste** (QVariant s)
- void **setup** ()
- qcaobject::QCaObject \* **createQcaltem** (unsigned int variableIndex)
- void **establishConnection** (unsigned int variableIndex)
- void **saveConfiguration** (PersistenceManager \*pm)
- void **restoreConfiguration** (PersistenceManager \*pm, restorePhases restorePhase)
- void **addToPredefinedList** (const QString &pvName)
- QStringList **getPredefinedPVNameList** ()
- QString **getPredefinedItem** (int i)
- void **setRecalclsRequired** ()
- void **setReplotsRequired** ()
- void **evaluateAllowDrop** ()



## Properties

- int **duration**
- double **yMinimum**
- double **yMaximum**
- QString **variable1**
- QString **variable2**
- QString **variable3**
- QString **variable4**
- QString **variable5**
- QString **variable6**
- QString **variable7**
- QString **variable8**
- QString **variable9**
- QString **variable10**
- QString **variable11**
- QString **variable12**
- 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](#)

### 9.90.1 Property Documentation

#### 9.90.1.1 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

## 9.91 QEStripChartAdjustPVDialo Class Reference

### Public Member Functions

- **QEStripChartAdjustPVDialo** (QWidget \*parent=0)
- void **setValueScaling** (const [ValueScaling](#) &valueScale)
- [ValueScaling](#) **getValueScaling** () const
- void **setSupport** (const double min, const double max, const QEDisplayRanges &loprHop, const QEDisplayRanges &plotted, const QEDisplayRanges &buffered)

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

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

## 9.92 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 QEArchiveInterface::How how)
- void **setLineDrawMode** (const QEStripChartNames::LineDrawModes mode)

### 9.92.1 Constructor & Destructor Documentation

**9.92.1.1** `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

## 9.93 QEStripChartItem Class Reference

### Signals

- void **itemContextMenuRequested** (const unsigned int, const QPoint &)
- void **requestAction** (const QEActionRequests &)

### Public Member Functions

- **QEStripChartItem** ([QEStripChart](#) \*chart, unsigned int slot, QWidget \*parent)
- bool **isInUse** ()
- bool **isCalculation** ()
- void **setPvName** (QString pvName, QString substitutions)
- QString **getPvName** ()
- bool **isScaled** ()
- bool **getUseReceiveTime** ()
- QArchiveInterface::How **getArchiveReadHow** ()
- QEStripChartNames::LineDrawModes **getLineDrawMode** ()
- void **setColour** (const QColor &colour)
- QColor **getColour** ()
- QEDisplayRanges **getLoprHopr** (bool doScale)
- QEDisplayRanges **getDisplayMinMax** (bool doScale)
- QEDisplayRanges **getBufferedMinMax** (bool doScale)
- QCaDataPointList **determinePlotPoints** ()
- void **readArchive** ()
- void **normalise** ()
- void **plotData** ()
- void **saveConfiguration** (PMElement &parentElement)
- void **restoreConfiguration** (PMElement &parentElement)

### Public Attributes

- QCaVariableNamePropertyManager **pvNamePropertyManager**

### Protected Member Functions

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

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

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

## 9.94 QEStripChartNames Class Reference

### Public Types

- enum **ChartTimeModes** { **tmRealTime**, **tmPaused**, **tmHistorical** }
- enum **ChartYRanges** {  
**manual**, **operatingRange**, **plotted**, **buffered**,  
**dynamic**, **normalised** }
- enum **PlayModes** {  
**play**, **pause**, **forward**, **backward**,  
**selectTimes** }
- enum **StateModes** { **previous**, **next** }
- enum **VideoModes** { **normal**, **reverse** }
- enum **YScaleModes** { **linear**, **log** }
- enum **LineDrawModes** { **ldmHide**, **ldmRegular**, **ldmBold** }
- enum **ContextMenuOptions** {  
**SCCM\_NONE** = **contextMenu::CM\_SPECIFIC\_WIDGETS\_START\_HERE**, **SCCM\_READ\_ARCHIVE**, **SCCM\_SCALE\_CHART\_AUTO**, **SCCM\_SCALE\_CHART\_PLOTTED**,  
**SCCM\_SCALE\_CHART\_BUFFERED**, **SCCM\_SCALE\_PV\_RESET**, **SCCM\_SCALE\_PV\_GENERAL**, **SCCM\_SCALE\_PV\_AUTO**,  
**SCCM\_SCALE\_PV\_PLOTTED**, **SCCM\_SCALE\_PV\_BUFFERED**, **SCCM\_SCALE\_PV\_CENTRE**, **SCCM\_PLOT\_RECTANGULAR**,  
**SCCM\_PLOT\_SMOOTH**, **SCCM\_PLOT\_SERVER\_TIME**, **SCCM\_PLOT\_CLIENT\_TIME**, **SCCM\_ARCH\_LINEAR**,  
**SCCM\_ARCH\_PLOTBIN**, **SCCM\_ARCH\_RAW**, **SCCM\_ARCH\_SHEET**, **SCCM\_ARCH\_AVERAGED**,  
**SCCM\_LINE\_HIDE**, **SCCM\_LINE\_REGULAR**, **SCCM\_LINE\_BOLD**, **SCCM\_LINE\_COLOUR**,  
**SCCM\_PV\_EDIT\_NAME**, **SCCM\_ADD\_TO\_PREDEFINED**, **SCCM\_PV\_WRITE\_TRACE**, **SCCM\_PV\_STATS**,  
**SCCM\_PV\_CLEAR**, **SCCM\_PV\_ADD\_NAME**, **SCCM\_PV\_PASTE\_NAME**, **SCCM\_PREDEFINED\_01**,  
**SCCM\_PREDEFINED\_02**, **SCCM\_PREDEFINED\_03**, **SCCM\_PREDEFINED\_04**, **SCCM\_PREDEFINED\_05**,  
**SCCM\_PREDEFINED\_06**, **SCCM\_PREDEFINED\_07**, **SCCM\_PREDEFINED\_08**, **SCCM\_PREDEFINED\_09**,  
**SCCM\_PREDEFINED\_10** }

### Static Public Attributes

- static const ContextMenuOptions **ContextMenuFirst** = **SCCM\_READ\_ARCHIVE**

- static const ContextMenuOptions **ContextMenuItemLast** = SCCM\_PREDEFINED\_10
- static const int **NumberPrefinedItems** = (SCCM\_PREDEFINED\_10 - SCCM\_PREDEFINED\_01 + 1)

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

- /tmp/epicsqt/trunk/framework/widgets/QEStriptChart/QEStriptChartNames.h

## 9.95 QEStriptChartPushButtonSpecifications Struct Reference

### Public Attributes

- int **gap**
- int **width**
- 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/QEStriptChart/QEStriptChartToolBar.cpp

## 9.96 QEStriptChartRangeDialog Class Reference

### Public Member Functions

- **QEStriptChartRangeDialog** (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/QEStriptChart/QEStriptChartRangeDialog.h
- /tmp/epicsqt/trunk/framework/widgets/QEStriptChart/QEStriptChartRangeDialog.cpp

## 9.97 QEStriptChartState 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

## 9.98 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

## 9.99 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

## 9.100 QEStripChartTimeDialog Class Reference

### Public Member Functions

- **QEStripChartTimeDialog** (QWidget \*parent=0)
- void **setMaximumDateTime** (QDateTime datetime)
- void **setStartDateTime** (QDateTime datetime)
- QDateTime **getStartDateTime** ()
- void **setEndDateTime** (QDateTime datetime)
- QDateTime **getEndDateTime** ()

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

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

## 9.101 QEStripChartToolBar Class Reference

This class holds all the StripChart tool bar widgets.

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

### Classes

- class [OwnWidgets](#)

### Signals

- void **stateSelected** (const QEStripChartNames::StateModes mode)
- void **videoModeSelected** (const QEStripChartNames::VideoModes mode)
- void **yScaleModeSelected** (const QEStripChartNames::YScaleModes mode)
- void **yRangeSelected** (const QEStripChartNames::ChartYRanges scale)
- void **durationSelected** (const int seconds)
- void **timeZoneSelected** (const Qt::TimeSpec timeSpec)
- void **playModeSelected** (const QEStripChartNames::PlayModes mode)
- void **readArchiveSelected** ()

### Public Member Functions

- **QEStripChartToolBar** (QWidget \*parent=0)
- void **setYRangeStatus** (const QString &status)
- void **setTimeStatus** (const QString &timeStatus)
- void **setStateSelectionEnabled** (const QEStripChartNames::StateModes mode, const bool enabled)

### Static Public Attributes

- static const int **designHeight** = 44

### Protected Member Functions

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

#### 9.101.1 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

## 9.102 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](#)

#### 9.102.1 Member Data Documentation

9.102.1.1 **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



### 9.102.2 Property Documentation

#### 9.102.2.1 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

## 9.103 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

## 9.104 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

## 9.105 screenSelectDialog Class Reference

### Public Types

- enum **screens** { **PRIMARY\_SCREEN** = -3, **THIS\_SCREEN** = -2, **ALL\_SCREEN** = -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

## 9.106 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

## 9.107 trace Class Reference

### Public Attributes

- QVector< QDateTime > **timeStamps**

- QVector< double > **xdata**
- QVector< double > **ydata**
- QwtPlotCurve \* **curve**
- QColor **color**
- QString **legend**
- bool **waveform**
- QwtPlotCurve::CurveStyle **style**

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

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

## 9.108 userInfoStruct Class Reference

### Public Attributes

- bool **enable**
- double **value1**
- double **value2**
- QString **elementText**

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

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

## 9.109 QEPeriodic::userInfoStructArray Struct Reference

### Public Attributes

- [userInfoStruct](#) **array** [NUM\_ELEMENTS]

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

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

## 9.110 ValueScaling Class Reference

### Public Member Functions

- void **reset** ()
- void **assign** (const [ValueScaling](#) &s)
- void **set** (const double dIn, const double mIn, const double cIn)

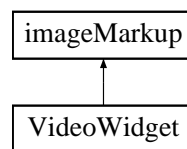
- void **get** (double &dOut, double &mOut, double &cOut) const
- void **map** (const double fromLower, const double fromUpper, const double toLower, const double toUpper)
- bool **isScaled** () const
- double **value** (const double x) const
- QEDisplayRanges **value** (const QEDisplayRanges &x) const
- 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

## 9.111 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** (const 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

## 9.112 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, 27](#)
- [\\_Field, 27](#)
- [\\_Item, 28](#)
- [\\_QDialogItem, 29](#)
- [\\_QPushButtonGroup, 29](#)
- [\\_QTableWidgetFileBrowser, 29](#)
- [\\_QTableWidgetLog, 30](#)
- [\\_QTableWidgetScript, 30](#)
- [addUnits](#)
  - [QEAnalogProgressBar, 65](#)
  - [QECheckBox, 80](#)
  - [QELabel, 160](#)
  - [QELineEdit, 168](#)
  - [QENumericEdit, 179](#)
  - [QEPushButton, 201](#)
  - [QERadioButton, 218](#)
- [alarmSeverityDisplayMode](#)
  - [QEAnalogProgressBar, 65](#)
- [alignment](#)
  - [QECheckBox, 80](#)
  - [QEPushButton, 201](#)
  - [QERadioButton, 218](#)
- [allowDrop](#)
  - [QEAnalogProgressBar, 65](#)
  - [QEBitStatus, 72](#)
  - [QECheckBox, 81](#)
  - [QEComboBox, 92](#)
  - [QEConfiguredLayout, 96](#)
  - [QEFileBrowser, 102](#)
  - [QEFrame, 106](#)
  - [QEGenericEdit, 114](#)
  - [QEGroupBox, 118](#)
  - [QEImage, 140](#)
  - [QELabel, 160](#)
  - [QELog, 174](#)
  - [QEPeriodic, 184](#)
  - [QEPlot, 192](#)
  - [QEPushButton, 201](#)
  - [QERadioButton, 218](#)
  - [QEScript, 234](#)
  - [QEShape, 241](#)
  - [QESlider, 251](#)
  - [QESpinBox, 255](#)
- [altReadbackVariable](#)
  - [QEPushButton, 201](#)
- [animation1](#)
  - [QEShape, 241](#)
- [animation2](#)
  - [QEShape, 241](#)
- [animation3](#)
  - [QEShape, 241](#)
- [animation4](#)
  - [QEShape, 242](#)
- [animation5](#)
  - [QEShape, 242](#)
- [animation6](#)
  - [QEShape, 242](#)
- [animationOptions](#)
  - [QEShape, 239](#)
- [Append](#)
  - [QEAnalogProgressBar, 64](#)
  - [QECheckBox, 77](#)
  - [QELabel, 158](#)
  - [QELineEdit, 166](#)
  - [QEPushButton, 198](#)
  - [QERadioButton, 215](#)
- [areaColor](#)
  - [QEImage, 140](#)
- [areaInfo, 30](#)
- [arguments](#)
  - [QECheckBox, 81](#)
  - [QEPushButton, 201](#)
  - [QERadioButton, 218](#)
- [arguments1](#)
  - [QEImage, 141](#)
- [arguments2](#)
  - [QEImage, 141](#)
- [arrayAction](#)
  - [QEAnalogProgressBar, 65](#)
  - [QECheckBox, 81](#)
  - [QELabel, 160](#)

- QELineEdit, [168](#)
- QEPushButton, [201](#)
- QERadioButton, [218](#)
- ArrayActions
  - QEAnalogProgressBar, [64](#)
  - QECheckBox, [77](#)
  - QELabel, [158](#)
  - QELineEdit, [166](#)
  - QEPushButton, [198](#)
  - QERadioButton, [215](#)
- Ascii
  - QEAnalogProgressBar, [64](#)
  - QECheckBox, [77](#)
  - QELabel, [158](#)
  - QELineEdit, [166](#)
  - QEPushButton, [198](#)
  - QERadioButton, [215](#)
- autoBrightnessContrast
  - QEImage, [141](#)
- Automatic
  - QEAnalogProgressBar, [64](#)
  - QECheckBox, [78](#)
  - QELabel, [158](#)
  - QELineEdit, [167](#)
  - QEPushButton, [199](#)
  - QERadioButton, [216](#)
- autoScale
  - QENumericEdit, [179](#)
- backgroundColour
  - QEAnalogIndicator, [59](#)
- Bar
  - QEAnalogIndicator, [58](#)
- Bayer
  - QEImage, [137](#)
- BayerBG
  - QEImage, [137](#)
- BayerGB
  - QEImage, [137](#)
- BayerGR
  - QEImage, [137](#)
- BayerRG
  - QEImage, [137](#)
- beamColor
  - QEImage, [141](#)
- beamXVariable
  - QEImage, [141](#)
- beamYVariable
  - QEImage, [141](#)
- bitDepthVariable
  - QEImage, [141](#)
- borderColour
  - QEAnalogIndicator, [59](#)
- Bottom\_To\_Top
  - QEAnalogIndicator, [59](#)
- BOUNDING\_RECTANGLE
  - QEImage, [137](#)
- BoundingBoxRectangle
  - QEImage, [137](#)
- briefInfoArea
  - QEImage, [141](#)
- CenterAndSize
  - QEImage, [137](#)
- centreAngle
  - QEAnalogIndicator, [59](#)
- clickCheckedText
  - QECheckBox, [81](#)
  - QEPushButton, [202](#)
  - QERadioButton, [219](#)
- clicked
  - QECheckBox, [80](#)
  - QEPushButton, [200](#)
  - QERadioButton, [217](#)
- clickText
  - QECheckBox, [81](#)
  - QEPushButton, [202](#)
  - QERadioButton, [219](#)
- clippingHighVariable
  - QEImage, [141](#)
- clippingLowVariable
  - QEImage, [142](#)
- clippingOnOffVariable
  - QEImage, [142](#)
- color1
  - QEShape, [242](#)
- color10
  - QEShape, [242](#)
- color2
  - QEShape, [242](#)
- color3
  - QEShape, [242](#)
- color4
  - QEShape, [242](#)
- color5
  - QEShape, [242](#)
- color6
  - QEShape, [243](#)
- color7
  - QEShape, [243](#)

- color8
  - QEShape, [243](#)
- color9
  - QEShape, [243](#)
- confirmAction
  - QECheckBox, [82](#)
  - QEPushButton, [202](#)
  - QERadioButton, [219](#)
- confirmText
  - QECheckBox, [82](#)
  - QEPushButton, [202](#)
  - QERadioButton, [219](#)
- confirmWrite
  - QEGenericEdit, [114](#)
- contrastReversal
  - QEImage, [142](#)
- creationOption
  - QECheckBox, [82](#)
  - QEPushButton, [202](#)
  - QERadioButton, [219](#)
- CreationOptionNames
  - QECheckBox, [77](#)
  - QEPushButton, [198](#)
  - QERadioButton, [215](#)
- customisationName
  - QECheckBox, [82](#)
  - QEPushButton, [203](#)
  - QERadioButton, [220](#)
- dbElementChanged
  - QEPeriodic, [184](#)
- dbValueChanged
  - QEAnalogProgressBar, [65](#)
  - QEBitStatus, [71](#)
  - QECheckBox, [80](#)
  - QEComboBox, [91](#)
  - QEImage, [140](#)
  - QELabel, [160](#)
  - QELineEdit, [167](#)
  - QENumericEdit, [179](#)
  - QEPeriodic, [184](#)
  - QEPlot, [191](#)
  - QEPushButton, [200](#)
  - QERadioButton, [217](#)
  - QESlider, [250](#)
  - QESpinBox, [255](#)
- dbValueChanged1
  - QEShape, [240](#)
- dbValueChanged2
  - QEShape, [240](#)
- dbValueChanged3
  - QEShape, [240](#)
- dbValueChanged4
  - QEShape, [241](#)
- dbValueChanged5
  - QEShape, [241](#)
- dbValueChanged6
  - QEShape, [241](#)
- Default
  - QEAnalogProgressBar, [64](#)
  - QECheckBox, [78](#)
  - QELabel, [158](#)
  - QELineEdit, [167](#)
  - QEPushButton, [198](#)
  - QERadioButton, [216](#)
- dimension1Variable
  - QEImage, [142](#)
- dimension2Variable
  - QEImage, [142](#)
- dimension3Variable
  - QEImage, [142](#)
- dimensionsVariable
  - QEImage, [142](#)
- displayAlarmState
  - QEAnalogProgressBar, [66](#)
  - QEBitStatus, [72](#)
  - QECheckBox, [82](#)
  - QEComboBox, [92](#)
  - QEConfiguredLayout, [96](#)
  - QEFileBrowser, [102](#)
  - QEFrame, [106](#)
  - QEGenericEdit, [114](#)
  - QEGroupBox, [118](#)
  - QEImage, [142](#)
  - QELabel, [160](#)
  - QELog, [174](#)
  - QEPeriodic, [184](#)
  - QEPlot, [192](#)
  - QEPushButton, [203](#)
  - QERadioButton, [220](#)
  - QEScript, [234](#)
  - QEShape, [243](#)
  - QESlider, [251](#)
  - QESpinBox, [255](#)
- displayArea1Selection
  - QEImage, [143](#)
- displayArea2Selection
  - QEImage, [143](#)
- displayArea3Selection
  - QEImage, [143](#)



- displayArea4Selection
  - QEImage, [143](#)
- displayBeamSelection
  - QEImage, [143](#)
- displayButtonBar
  - QEImage, [140](#)
- displayCursorPixelInfo
  - QEImage, [143](#)
- displayEllipse
  - QEImage, [143](#)
- displayHozSliceSelection
  - QEImage, [143](#)
- displayProfileSelection
  - QEImage, [144](#)
- displayTargetSelection
  - QEImage, [144](#)
- displayVertSliceSelection
  - QEImage, [144](#)
- DockBottom
  - QCheckBox, [77](#)
  - QEPushButton, [198](#)
  - QERadioButton, [215](#)
- DockBottomTabbed
  - QCheckBox, [78](#)
  - QEPushButton, [198](#)
  - QERadioButton, [215](#)
- DockFloating
  - QCheckBox, [78](#)
  - QEPushButton, [198](#)
  - QERadioButton, [215](#)
- DockLeft
  - QCheckBox, [77](#)
  - QEPushButton, [198](#)
  - QERadioButton, [215](#)
- DockLeftTabbed
  - QCheckBox, [78](#)
  - QEPushButton, [198](#)
  - QERadioButton, [215](#)
- DockRight
  - QCheckBox, [77](#)
  - QEPushButton, [198](#)
  - QERadioButton, [215](#)
- DockRightTabbed
  - QCheckBox, [78](#)
  - QEPushButton, [198](#)
  - QERadioButton, [215](#)
- DockTop
  - QCheckBox, [77](#)
  - QEPushButton, [198](#)
  - QERadioButton, [215](#)
- DockTopTabbed
  - QCheckBox, [77](#)
  - QEPushButton, [198](#)
  - QERadioButton, [215](#)
- DottedFullCrosshair
  - QEImage, [139](#)
- drawMarkup
  - markupHLine, [44](#)
  - markupVLine, [49](#)
- ellipseColor
  - QEImage, [144](#)
- ellipseHVariable
  - QEImage, [144](#)
- EllipseVariableDefinitions
  - QEImage, [137](#)
- ellipseVariableDefinitions
  - QEImage, [137](#)
- ellipseWVariable
  - QEImage, [144](#)
- ellipseXVariable
  - QEImage, [144](#)
- ellipseYVariable
  - QEImage, [144](#)
- enableArea1Selection
  - QEImage, [144](#)
- enableArea2Selection
  - QEImage, [145](#)
- enableArea3Selection
  - QEImage, [145](#)
- enableArea4Selection
  - QEImage, [145](#)
- enableBeamSelection
  - QEImage, [145](#)
- enableHozSliceSelection
  - QEImage, [145](#)
- enableProfileSelection
  - QEImage, [145](#)
- enableTargetSelection
  - QEImage, [145](#)
- enableVertSliceSelection
  - QEImage, [145](#)
- Engineer
  - QEAnalogProgressBar, [65](#)
  - QEBitStatus, [71](#)
  - QCheckBox, [79](#)
  - QComboBox, [91](#)
  - QEConfiguredLayout, [96](#)
  - QEFileBrowser, [101](#)
  - QEFrame, [106](#)

- QEGenericEdit, [113](#)
- QEGroupBox, [118](#)
- QEImage, [139](#)
- QELabel, [159](#)
- QELog, [174](#)
- QEPeriodic, [184](#)
- QEPlot, [191](#)
- QEPushButton, [200](#)
- QERadioButton, [217](#)
- QEScript, [233](#)
- QEShape, [240](#)
- QESlider, [250](#)
- QESpinBox, [255](#)
- externalControls
  - QEImage, [146](#)
- FFBuffer, [32](#)
- FFThread, [32](#)
- Fit
  - QEImage, [138](#)
- Fixed
  - QEAnalogProgressBar, [64](#)
  - QECheckBox, [78](#)
  - QELabel, [158](#)
  - QELineEdit, [167](#)
  - QEPushButton, [199](#)
  - QERadioButton, [216](#)
- flipRotateMenu, [33](#)
- Floating
  - QEAnalogProgressBar, [64](#)
  - QECheckBox, [78](#)
  - QELabel, [158](#)
  - QELineEdit, [167](#)
  - QEPushButton, [198](#)
  - QERadioButton, [216](#)
- fontColour
  - QEAnalogIndicator, [59](#)
- foregroundColour
  - QEAnalogIndicator, [59](#)
- format
  - QEAnalogProgressBar, [66](#)
  - QECheckBox, [82](#)
  - QELabel, [160](#)
  - QELineEdit, [168](#)
  - QEPushButton, [203](#)
  - QERadioButton, [220](#)
- formatOption
  - QEImage, [146](#)
- FormatOptions
  - QEImage, [137](#)
- Formats
  - QEAnalogProgressBar, [64](#)
  - QECheckBox, [78](#)
  - QELabel, [158](#)
  - QELineEdit, [166](#)
  - QEPushButton, [198](#)
  - QERadioButton, [215](#)
- formatVariable
  - QEImage, [146](#)
- fullScreenWindow, [33](#)
- getConfirmWrite
  - QEGenericEdit, [113](#)
- getSubscribe
  - QEGenericEdit, [113](#)
- getWriteOnEnter
  - QEGenericEdit, [113](#)
- getWriteOnFinish
  - QEGenericEdit, [113](#)
- getWriteOnLoseFocus
  - QEGenericEdit, [113](#)
- guiFile
  - QECheckBox, [83](#)
  - QEPushButton, [203](#)
  - QERadioButton, [220](#)
- heightVariable
  - QEImage, [146](#)
- histogram, [34](#)
- histogramScroll, [34](#)
- historicImage, [34](#)
- horizontalFlip
  - QEImage, [146](#)
- hozSliceColor
  - QEImage, [146](#)
- Icon
  - QECheckBox, [79](#)
  - QEPushButton, [199](#)
  - QERadioButton, [217](#)
- imageContextMenu, [35](#)
- imageDisplayProperties, [36](#)
- imageDisplayProperties::rgbPixel, [267](#)
- imageInfo, [37](#)
- imageMarkup, [38](#)
- imageUpdateIndicator, [40](#)
- imageVariable
  - QEImage, [146](#)
- Index
  - QEAnalogProgressBar, [64](#)

- QERadioButton, 221
- leadingZeros
  - QENumericEdit, 179
- Left\_To\_Right
  - QEAnalogIndicator, 59
- lineProfileArrayVariable
  - QEImage, 147
- lineProfileThicknessVariable
  - QEImage, 147
- lineProfileX1Variable
  - QEImage, 147
- lineProfileX2Variable
  - QEImage, 147
- lineProfileY1Variable
  - QEImage, 147
- lineProfileY2Variable
  - QEImage, 147
- LocalEnumeration
  - QEAnalogProgressBar, 64
  - QECheckBox, 78
  - QELabel, 158
  - QELineEdit, 167
  - QEPushButton, 199
  - QERadioButton, 216
- localEnumeration
  - QEAnalogProgressBar, 66
  - QECheckBox, 83
  - QEComboBox, 92
  - QELabel, 161
  - QELineEdit, 168
  - QEPushButton, 204
  - QERadioButton, 221
- logBrightness
  - QEImage, 147
- loginWidget, 40
- LogOutput
  - QECheckBox, 79
  - QEImage, 138
  - QEPushButton, 199
  - QERadioButton, 216
- logScale
  - QEAnalogIndicator, 59
- logScaleInterval
  - QEAnalogIndicator, 59
- majorInterval
  - QEAnalogIndicator, 59
- markupCrosshair1, 40
- markupCrosshair2, 41
- markupDisplayMenu, 42

- markupEllipse, [42](#)
- markupHLine, [43](#)
  - drawMarkup, [44](#)
- markupItem, [44](#)
- markupLine, [46](#)
- markupRegion, [47](#)
- markupText, [48](#)
- markupVLine, [48](#)
  - drawMarkup, [49](#)
- maximum
  - QEAAnalogIndicator, [60](#)
  - QENumericEdit, [180](#)
- Meter
  - QEAAnalogIndicator, [58](#)
- minimum
  - QEAAnalogIndicator, [60](#)
  - QENumericEdit, [180](#)
- minorInterval
  - QEAAnalogIndicator, [60](#)
- mode
  - QEAAnalogIndicator, [60](#)
- Modes
  - QEAAnalogIndicator, [58](#)
- Mono
  - QEImage, [137](#)
- mpegSource, [49](#)
  - updateImage, [50](#)
- mpegSourceObject, [50](#)
- NewTab
  - QECHECKBOX, [77](#)
  - QEPushButton, [198](#)
  - QERadioButton, [215](#)
- NewWindow
  - QECHECKBOX, [77](#)
  - QEPushButton, [198](#)
  - QERadioButton, [215](#)
- None
  - QECHECKBOX, [78](#)
  - QEImage, [138](#)
  - QEPushButton, [199](#)
  - QERadioButton, [216](#)
- NoRotation
  - QEImage, [138](#)
- notation
  - QEAAnalogProgressBar, [67](#)
  - QECHECKBOX, [84](#)
  - QELabel, [162](#)
  - QELineEdit, [169](#)
  - QEPushButton, [205](#)
  - QERadioButton, [222](#)
- Notations
  - QEAAnalogProgressBar, [64](#)
  - QECHECKBOX, [78](#)
  - QELabel, [158](#)
  - QELineEdit, [167](#)
  - QEPushButton, [199](#)
  - QERadioButton, [216](#)
- offset1
  - QEShape, [243](#)
- offset2
  - QEShape, [244](#)
- offset3
  - QEShape, [244](#)
- offset4
  - QEShape, [244](#)
- offset5
  - QEShape, [244](#)
- offset6
  - QEShape, [244](#)
- Open
  - QECHECKBOX, [77](#)
  - QEPushButton, [198](#)
  - QERadioButton, [215](#)
- orientation
  - QEAAnalogIndicator, [60](#)
- Orientations
  - QEAAnalogIndicator, [58](#)
- password
  - QECHECKBOX, [84](#)
  - QEPushButton, [205](#)
  - QERadioButton, [222](#)
- PeriodicDialog, [51](#)
- PeriodicElementSetupForm, [51](#)
- PeriodicSetupDialog, [52](#)
- Picture
  - QELabel, [159](#)
- pixmap0
  - QECHECKBOX, [84](#)
  - QELabel, [162](#)
  - QEPushButton, [205](#)
  - QERadioButton, [222](#)
- pixmap1
  - QECHECKBOX, [84](#)
  - QELabel, [162](#)
  - QEPushButton, [205](#)
  - QERadioButton, [222](#)
- pixmap2

- QCheckBox, [84](#)
- QELabel, [162](#)
- QEPushButton, [205](#)
- QERadioButton, [222](#)
- pixmap3
  - QCheckBox, [85](#)
  - QELabel, [162](#)
  - QEPushButton, [205](#)
  - QERadioButton, [222](#)
- pixmap4
  - QCheckBox, [85](#)
  - QELabel, [162](#)
  - QEPushButton, [205](#)
  - QERadioButton, [222](#)
- pixmap5
  - QCheckBox, [85](#)
  - QELabel, [162](#)
  - QEPushButton, [205](#)
  - QERadioButton, [222](#)
- pixmap6
  - QCheckBox, [85](#)
  - QELabel, [162](#)
  - QEPushButton, [205](#)
  - QERadioButton, [222](#)
- pixmap7
  - QCheckBox, [85](#)
  - QELabel, [162](#)
  - QEPushButton, [206](#)
  - QERadioButton, [223](#)
- playbackTimer, [52](#)
- point1
  - QEShape, [244](#)
- point10
  - QEShape, [244](#)
- point2
  - QEShape, [244](#)
- point3
  - QEShape, [244](#)
- point4
  - QEShape, [245](#)
- point5
  - QEShape, [245](#)
- point6
  - QEShape, [245](#)
- point7
  - QEShape, [245](#)
- point8
  - QEShape, [245](#)
- point9
  - QEShape, [245](#)
- pointInfo, [52](#)
- precision
  - QEAnalogProgressBar, [67](#)
  - QCheckBox, [85](#)
  - QELabel, [163](#)
  - QELineEdit, [169](#)
  - QENumericEdit, [180](#)
  - QEPushButton, [206](#)
  - QERadioButton, [223](#)
- pressed
  - QCheckBox, [80](#)
  - QEPushButton, [200](#)
  - QERadioButton, [217](#)
- pressText
  - QCheckBox, [85](#)
  - QEPushButton, [206](#)
  - QERadioButton, [223](#)
- prioritySubstitutions
  - QCheckBox, [85](#)
  - QEPushButton, [206](#)
  - QERadioButton, [223](#)
- profileColor
  - QEImage, [148](#)
- profileHozArrayVariable
  - QEImage, [148](#)
- profileHozThicknessVariable
  - QEImage, [148](#)
- profileHozVariable
  - QEImage, [148](#)
- profilePlot, [53](#)
- profileVertArrayVariable
  - QEImage, [148](#)
- profileVertThicknessVariable
  - QEImage, [148](#)
- profileVertVariable
  - QEImage, [148](#)
- program
  - QCheckBox, [86](#)
  - QEPushButton, [206](#)
  - QERadioButton, [223](#)
- program1
  - QEImage, [148](#)
- program2
  - QEImage, [148](#)
- programStartupOption
  - QCheckBox, [86](#)
  - QEPushButton, [206](#)
  - QERadioButton, [223](#)
- programStartupOption1
  - QEImage, [149](#)

- programStartupOption2
  - QEImage, [149](#)
- ProgramStartupOptionNames
  - QCheckBox, [78](#)
  - QEImage, [137](#)
  - QEPushButton, [199](#)
  - QERadioButton, [216](#)
- QBitStatus, [53](#)
- QEAnalogIndicator, [55](#)
  - backgroundColour, [59](#)
  - Bar, [58](#)
  - borderColour, [59](#)
  - Bottom\_To\_Top, [59](#)
  - centreAngle, [59](#)
  - fontColour, [59](#)
  - foregroundColour, [59](#)
  - Left\_To\_Right, [59](#)
  - logScale, [59](#)
  - logScaleInterval, [59](#)
  - majorInterval, [59](#)
  - maximum, [60](#)
  - Meter, [58](#)
  - minimum, [60](#)
  - minorInterval, [60](#)
  - mode, [60](#)
  - Modes, [58](#)
  - orientation, [60](#)
  - Orientations, [58](#)
  - Right\_To\_Left, [59](#)
  - Scale, [58](#)
  - showScale, [60](#)
  - showText, [60](#)
  - spanAngle, [60](#)
  - Top\_To\_Bottom, [59](#)
  - value, [60](#)
- QEAnalogIndicator::Band, [31](#)
- QEAnalogIndicator::BandList, [31](#)
- QEAnalogProgressBar, [61](#)
  - addUnits, [65](#)
  - alarmSeverityDisplayMode, [65](#)
  - allowDrop, [65](#)
  - Append, [64](#)
  - arrayAction, [65](#)
  - ArrayActions, [64](#)
  - Ascii, [64](#)
  - Automatic, [64](#)
  - dbValueChanged, [65](#)
  - Default, [64](#)
  - displayAlarmState, [66](#)
  - Engineer, [65](#)
  - Fixed, [64](#)
  - Floating, [64](#)
  - format, [66](#)
  - Formats, [64](#)
  - Index, [64](#)
  - int, [66](#)
  - Integer, [64](#)
  - leadingZero, [66](#)
  - LocalEnumeration, [64](#)
  - localEnumeration, [66](#)
  - notation, [67](#)
  - Notations, [64](#)
  - precision, [67](#)
  - QEAnalogProgressBar, [65](#)
  - Scientific, [64](#)
  - Scientist, [65](#)
  - Time, [64](#)
  - trailingZeros, [67](#)
  - UnsignedInteger, [64](#)
  - useDbDisplayLimits, [67](#)
  - useDbPrecision, [67](#)
  - User, [65](#)
  - userLevelEnabled, [68](#)
  - userLevelEngineerStyle, [68](#)
  - UserLevels, [64](#)
  - userLevelScientistStyle, [68](#)
  - userLevelUserStyle, [68](#)
  - userLevelVisibility, [68](#)
  - value, [69](#)
  - variable, [69](#)
  - variableAsToolTip, [69](#)
  - variableSubstitutions, [69](#)
  - visible, [69](#)
- QEBitStatus, [69](#)
  - allowDrop, [72](#)
  - dbValueChanged, [71](#)
  - displayAlarmState, [72](#)
  - Engineer, [71](#)
  - int, [72](#)
  - Scientist, [71](#)
  - User, [71](#)
  - userLevelEnabled, [72](#)
  - userLevelEngineerStyle, [72](#)
  - UserLevels, [71](#)
  - userLevelScientistStyle, [72](#)
  - userLevelUserStyle, [73](#)
  - userLevelVisibility, [73](#)
  - variable, [73](#)
  - variableAsToolTip, [73](#)

- variableSubstitutions, [73](#)
- visible, [73](#)
- QCheckBox, [74](#)
  - addUnits, [80](#)
  - alignment, [80](#)
  - allowDrop, [81](#)
  - Append, [77](#)
  - arguments, [81](#)
  - arrayAction, [81](#)
  - ArrayActions, [77](#)
  - Ascii, [77](#)
  - Automatic, [78](#)
  - clickCheckedText, [81](#)
  - clicked, [80](#)
  - clickText, [81](#)
  - confirmAction, [82](#)
  - confirmText, [82](#)
  - creationOption, [82](#)
  - CreationOptionNames, [77](#)
  - customisationName, [82](#)
  - dbValueChanged, [80](#)
  - Default, [78](#)
  - displayAlarmState, [82](#)
  - DockBottom, [77](#)
  - DockBottomTabbed, [78](#)
  - DockFloating, [78](#)
  - DockLeft, [77](#)
  - DockLeftTabbed, [78](#)
  - DockRight, [77](#)
  - DockRightTabbed, [78](#)
  - DockTop, [77](#)
  - DockTopTabbed, [77](#)
  - Engineer, [79](#)
  - Fixed, [78](#)
  - Floating, [78](#)
  - format, [82](#)
  - Formats, [78](#)
  - guiFile, [83](#)
  - Icon, [79](#)
  - Index, [77](#)
  - int, [83](#)
  - Integer, [78](#)
  - labelText, [83](#)
  - leadingZero, [83](#)
  - LocalEnumeration, [78](#)
  - localEnumeration, [83](#)
  - LogOutput, [79](#)
  - NewTab, [77](#)
  - NewWindow, [77](#)
  - None, [78](#)
  - notation, [84](#)
  - Notations, [78](#)
  - Open, [77](#)
  - password, [84](#)
  - pixmap0, [84](#)
  - pixmap1, [84](#)
  - pixmap2, [84](#)
  - pixmap3, [85](#)
  - pixmap4, [85](#)
  - pixmap5, [85](#)
  - pixmap6, [85](#)
  - pixmap7, [85](#)
  - precision, [85](#)
  - pressed, [80](#)
  - pressText, [85](#)
  - prioritySubstitutions, [85](#)
  - program, [86](#)
  - programStartupOption, [86](#)
  - ProgramStartupOptionNames, [78](#)
  - QCheckBox, [79](#)
  - released, [80](#)
  - releaseText, [86](#)
  - requestAction, [80](#)
  - Scientific, [78](#)
  - Scientist, [79](#)
  - State, [79](#)
  - StdOutput, [79](#)
  - subscribe, [86](#)
  - Terminal, [78](#)
  - Text, [79](#)
  - TextAndIcon, [79](#)
  - Time, [78](#)
  - trailingZeros, [86](#)
  - UnsignedInteger, [78](#)
  - updateOption, [86](#)
  - UpdateOptions, [79](#)
  - useDbPrecision, [86](#)
  - User, [79](#)
  - userLevelEnabled, [87](#)
  - userLevelEngineerStyle, [87](#)
  - UserLevels, [79](#)
  - userLevelScientistStyle, [87](#)
  - userLevelUserStyle, [87](#)
  - userLevelVisibility, [87](#)
  - variable, [88](#)
  - variableAsToolTip, [88](#)
  - variableSubstitutions, [88](#)
  - visible, [88](#)
  - writeOnClick, [88](#)
  - writeOnPress, [88](#)

- writeOnRelease, 88
- QECheckBoxManager, 89
- QEComboBox, 89
  - allowDrop, 92
  - dbValueChanged, 91
  - displayAlarmState, 92
  - Engineer, 91
  - int, 92
  - localEnumeration, 92
  - Scientist, 91
  - subscribe, 92
  - useDbEnumerations, 91
  - User, 91
  - userLevelEnabled, 92
  - userLevelEngineerStyle, 92
  - UserLevels, 91
  - userLevelScientistStyle, 93
  - userLevelUserStyle, 93
  - userLevelVisibility, 93
  - variable, 93
  - variableAsToolTip, 93
  - variableSubstitutions, 93
  - visible, 94
  - writeOnChange, 91
- QEConfiguredLayout, 94
  - allowDrop, 96
  - displayAlarmState, 96
  - Engineer, 96
  - int, 96
  - Scientist, 96
  - User, 96
  - userLevelEnabled, 96
  - userLevelEngineerStyle, 97
  - UserLevels, 96
  - userLevelScientistStyle, 97
  - userLevelUserStyle, 97
  - userLevelVisibility, 97
  - variableAsToolTip, 97
  - visible, 98
- QEConfiguredLayoutManager, 98
- QEFileBrowser, 98
  - allowDrop, 102
  - displayAlarmState, 102
  - Engineer, 101
  - int, 102
  - Scientist, 101
  - selected, 101
  - User, 101
  - userLevelEnabled, 102
  - userLevelEngineerStyle, 102
  - UserLevels, 101
  - userLevelScientistStyle, 102
  - userLevelUserStyle, 103
  - userLevelVisibility, 103
  - variable, 103
  - variableAsToolTip, 103
  - variableSubstitutions, 103
  - visible, 103
- QEForm, 104
- QEFrame, 105
  - allowDrop, 106
  - displayAlarmState, 106
  - Engineer, 106
  - int, 107
  - Scientist, 106
  - User, 106
  - userLevelEnabled, 107
  - userLevelEngineerStyle, 107
  - UserLevels, 106
  - userLevelScientistStyle, 107
  - userLevelUserStyle, 107
  - userLevelVisibility, 108
  - variableAsToolTip, 108
  - visible, 108
- QEGenericButton, 108
- QEGenericEdit, 110
  - allowDrop, 114
  - confirmWrite, 114
  - displayAlarmState, 114
  - Engineer, 113
  - getConfirmWrite, 113
  - getSubscribe, 113
  - getWriteOnEnter, 113
  - getWriteOnFinish, 113
  - getWriteOnLoseFocus, 113
  - int, 115
  - QEGenericEdit, 113
  - Scientist, 113
  - setConfirmWrite, 113
  - setSubscribe, 114
  - setWriteOnEnter, 114
  - setWriteOnFinish, 114
  - setWriteOnLoseFocus, 114
  - subscribe, 115
  - User, 113
  - userLevelEnabled, 115
  - userLevelEngineerStyle, 115
  - UserLevels, 112
  - userLevelScientistStyle, 115
  - userLevelUserStyle, 115



- userLevelVisibility, 116
- variable, 116
- variableAsToolTip, 116
- variableSubstitutions, 116
- visible, 116
- writeOnEnter, 116
- writeOnFinish, 116
- writeOnLoseFocus, 117
- QEGroupBox, 117
  - allowDrop, 118
  - displayAlarmState, 118
  - Engineer, 118
  - int, 118
  - Scientist, 118
  - substitutedTitle, 119
  - textSubstitutions, 119
  - User, 118
  - userLevelEnabled, 119
  - userLevelEngineerStyle, 119
  - UserLevels, 118
  - userLevelScientistStyle, 119
  - userLevelUserStyle, 119
  - userLevelVisibility, 120
  - variableAsToolTip, 120
  - visible, 120
- QEImage, 120
  - allowDrop, 140
  - areaColor, 140
  - arguments1, 141
  - arguments2, 141
  - autoBrightnessContrast, 141
  - Bayer, 137
  - BayerBG, 137
  - BayerGB, 137
  - BayerGR, 137
  - BayerRG, 137
  - beamColor, 141
  - beamXVariable, 141
  - beamYVariable, 141
  - bitDepthVariable, 141
  - BOUNDING\_RECTANGLE, 137
  - BoundingRectangle, 137
  - briefInfoArea, 141
  - CenterAndSize, 137
  - clippingHighVariable, 141
  - clippingLowVariable, 142
  - clippingOnOffVariable, 142
  - contrastReversal, 142
  - dbValueChanged, 140
  - dimension1Variable, 142
  - dimension2Variable, 142
  - dimension3Variable, 142
  - dimensionsVariable, 142
  - displayAlarmState, 142
  - displayArea1Selection, 143
  - displayArea2Selection, 143
  - displayArea3Selection, 143
  - displayArea4Selection, 143
  - displayBeamSelection, 143
  - displayButtonBar, 140
  - displayCursorPixelInfo, 143
  - displayEllipse, 143
  - displayHozSliceSelection, 143
  - displayProfileSelection, 144
  - displayTargetSelection, 144
  - displayVertSliceSelection, 144
  - DottedFullCrosshair, 139
  - ellipseColor, 144
  - ellipseHVariable, 144
  - EllipseVariableDefinitions, 137
  - ellipseVariableDefinitions, 137
  - ellipseWVariable, 144
  - ellipseXVariable, 144
  - ellipseYVariable, 144
  - enableArea1Selection, 144
  - enableArea2Selection, 145
  - enableArea3Selection, 145
  - enableArea4Selection, 145
  - enableBeamSelection, 145
  - enableHozSliceSelection, 145
  - enableProfileSelection, 145
  - enableTargetSelection, 145
  - enableVertSliceSelection, 145
  - Engineer, 139
  - externalControls, 146
  - Fit, 138
  - formatOption, 146
  - FormatOptions, 137
  - formatVariable, 146
  - heightVariable, 146
  - horizontalFlip, 146
  - hozSliceColor, 146
  - imageVariable, 146
  - initialHosScrollPos, 146
  - initialVertScrollPos, 140
  - int, 147
  - lineProfileArrayVariable, 147
  - lineProfileThicknessVariable, 147
  - lineProfileX1Variable, 147
  - lineProfileX2Variable, 147

- lineProfileY1Variable, [147](#)
- lineProfileY2Variable, [147](#)
- logBrightness, [147](#)
- LogOutput, [138](#)
- Mono, [137](#)
- None, [138](#)
- NoRotation, [138](#)
- profileColor, [148](#)
- profileHozArrayVariable, [148](#)
- profileHozThicknessVariable, [148](#)
- profileHozVariable, [148](#)
- profileVertArrayVariable, [148](#)
- profileVertThicknessVariable, [148](#)
- profileVertVariable, [148](#)
- program1, [148](#)
- program2, [148](#)
- programStartupOption1, [149](#)
- programStartupOption2, [149](#)
- ProgramStartupOptionNames, [137](#)
- QEImage, [140](#)
- regionOfInterest1HVariable, [149](#)
- regionOfInterest1WVariable, [149](#)
- regionOfInterest1XVariable, [149](#)
- regionOfInterest1YVariable, [149](#)
- regionOfInterest2HVariable, [149](#)
- regionOfInterest2WVariable, [149](#)
- regionOfInterest2XVariable, [150](#)
- regionOfInterest2YVariable, [150](#)
- regionOfInterest3HVariable, [150](#)
- regionOfInterest3WVariable, [150](#)
- regionOfInterest3XVariable, [150](#)
- regionOfInterest3YVariable, [150](#)
- regionOfInterest4HVariable, [150](#)
- regionOfInterest4WVariable, [150](#)
- regionOfInterest4XVariable, [150](#)
- regionOfInterest4YVariable, [151](#)
- RESIZE\_OPTION\_FIT, [138](#)
- RESIZE\_OPTION\_ZOOM, [138](#)
- resizeOption, [151](#)
- ResizeOptions, [138](#)
- resizeOptions, [138](#)
- rgb1, [137](#)
- rgb2, [137](#)
- rgb3, [137](#)
- Rotate180, [138](#)
- Rotate90Left, [138](#)
- Rotate90Right, [138](#)
- rotation, [151](#)
- ROTATION\_0, [138](#)
- ROTATION\_180, [139](#)
- ROTATION\_90\_LEFT, [139](#)
- ROTATION\_90\_RIGHT, [138](#)
- RotationOptions, [138](#)
- rotationOptions, [138](#)
- Scientist, [139](#)
- selectOptions, [139](#)
- showTime, [151](#)
- SO\_AREA4, [139](#)
- SO\_BEAM, [139](#)
- SO\_HSLICE, [139](#)
- SO\_NONE, [139](#)
- SO\_PANNING, [139](#)
- SO\_PROFILE, [139](#)
- SO\_TARGET, [139](#)
- SO\_VSLICE, [139](#)
- SolidSmallCrosshair, [139](#)
- StdOutput, [138](#)
- targetColor, [151](#)
- TargetOptions, [139](#)
- targetTriggerVariable, [151](#)
- targetXVariable, [151](#)
- targetYVariable, [151](#)
- Terminal, [138](#)
- timeColor, [151](#)
- URL, [152](#)
- useFalseColour, [152](#)
- User, [139](#)
- userLevelEnabled, [152](#)
- userLevelEngineerStyle, [152](#)
- UserLevels, [139](#)
- userLevelScientistStyle, [152](#)
- userLevelUserStyle, [152](#)
- userLevelVisibility, [153](#)
- variableAsToolTip, [153](#)
- variableSubstitutions, [153](#)
- verticalFlip, [153](#)
- vertSliceColor, [153](#)
- visible, [153](#)
- widthVariable, [153](#)
- yuv422, [137](#)
- yuv444, [137](#)
- Zoom, [138](#)
- QEImageMarkupThickness, [154](#)
- QEImageOptionsDialog, [154](#)
- QELabel, [155](#)
  - addUnits, [160](#)
  - allowDrop, [160](#)
  - Append, [158](#)
  - arrayAction, [160](#)
  - ArrayActions, [158](#)

- Ascii, [158](#)
- Automatic, [158](#)
- dbValueChanged, [160](#)
- Default, [158](#)
- displayAlarmState, [160](#)
- Engineer, [159](#)
- Fixed, [158](#)
- Floating, [158](#)
- format, [160](#)
- Formats, [158](#)
- Index, [158](#)
- int, [161](#)
- Integer, [158](#)
- leadingZero, [161](#)
- LocalEnumeration, [158](#)
- localEnumeration, [161](#)
- notation, [162](#)
- Notations, [158](#)
- Picture, [159](#)
- pixmap0, [162](#)
- pixmap1, [162](#)
- pixmap2, [162](#)
- pixmap3, [162](#)
- pixmap4, [162](#)
- pixmap5, [162](#)
- pixmap6, [162](#)
- pixmap7, [162](#)
- precision, [163](#)
- QELabel, [159](#)
- Scientific, [158](#)
- Scientist, [159](#)
- Text, [159](#)
- Time, [158](#)
- trailingZeros, [163](#)
- UnsignedInteger, [158](#)
- UPDATE\_PIXMAP, [159](#)
- UPDATE\_TEXT, [159](#)
- updateOption, [163](#)
- UpdateOptions, [158](#)
- updateOptions, [159](#)
- useDbPrecision, [163](#)
- User, [159](#)
- userLevelEnabled, [163](#)
- userLevelEngineerStyle, [163](#)
- UserLevels, [159](#)
- userLevelScientistStyle, [163](#)
- userLevelUserStyle, [164](#)
- userLevelVisibility, [164](#)
- variable, [164](#)
- variableAsToolTip, [164](#)
- variableSubstitutions, [164](#)
- visible, [164](#)
- QELineEdit, [165](#)
  - addUnits, [168](#)
  - Append, [166](#)
  - arrayAction, [168](#)
  - ArrayActions, [166](#)
  - Ascii, [166](#)
  - Automatic, [167](#)
  - dbValueChanged, [167](#)
  - Default, [167](#)
  - Fixed, [167](#)
  - Floating, [167](#)
  - format, [168](#)
  - Formats, [166](#)
  - Index, [166](#)
  - int, [168](#)
  - Integer, [167](#)
  - leadingZero, [168](#)
  - LocalEnumeration, [167](#)
  - localEnumeration, [168](#)
  - notation, [169](#)
  - Notations, [167](#)
  - precision, [169](#)
  - QELineEdit, [167](#)
  - Scientific, [167](#)
  - Time, [167](#)
  - trailingZeros, [169](#)
  - UnsignedInteger, [167](#)
  - useDbPrecision, [169](#)
- QELineEditManager, [170](#)
- QELink, [170](#)
- QELog, [172](#)
  - allowDrop, [174](#)
  - displayAlarmState, [174](#)
  - Engineer, [174](#)
  - int, [174](#)
  - Scientist, [174](#)
  - User, [174](#)
  - userLevelEnabled, [174](#)
  - userLevelEngineerStyle, [175](#)
  - UserLevels, [174](#)
  - userLevelScientistStyle, [175](#)
  - userLevelUserStyle, [175](#)
  - userLevelVisibility, [175](#)
  - variableAsToolTip, [175](#)
  - visible, [176](#)
- QELogin, [176](#)
- QELoginDialog, [177](#)
- QENumericEdit, [177](#)

- addUnits, 179
- autoScale, 179
- dbValueChanged, 179
- leadingZeros, 179
- maximum, 180
- minimum, 180
- precision, 180
- QENumericEdit, 179
- QENumericEditManager, 180
- QEPeriodic, 181
  - allowDrop, 184
  - dbElementChanged, 184
  - dbValueChanged, 184
  - displayAlarmState, 184
  - Engineer, 184
  - int, 184
  - readbackLabelVariable1, 185
  - readbackLabelVariable2, 185
  - Scientist, 184
  - subscribe, 185
  - User, 184
  - userLevelEnabled, 185
  - userLevelEngineerStyle, 185
  - UserLevels, 184
  - userLevelScientistStyle, 185
  - userLevelUserStyle, 186
  - userLevelVisibility, 186
  - variableAsToolTip, 186
  - variableSubstitutions, 186
  - visible, 186
  - writeButtonVariable1, 186
  - writeButtonVariable2, 186
- QEPeriodic::elementInfoStruct, 31
- QEPeriodic::userInfoStructArray, 269
- QEPeriodicComponentData, 187
- QEPeriodicTaskMenu, 187
- QEPeriodicTaskMenuFactory, 187
- QEPlot, 188
  - allowDrop, 192
  - dbValueChanged, 191
  - displayAlarmState, 192
  - Engineer, 191
  - int, 192
  - Scientist, 191
  - User, 191
  - userLevelEnabled, 192
  - userLevelEngineerStyle, 192
  - UserLevels, 191
  - userLevelScientistStyle, 193
  - userLevelUserStyle, 193
  - userLevelVisibility, 193
  - variable1, 193
  - variable2, 193
  - variable3, 193
  - variable4, 193
  - variableAsToolTip, 194
  - variableSubstitutions, 194
  - visible, 194
- QEPushButton, 194
  - addUnits, 201
  - alignment, 201
  - allowDrop, 201
  - altReadbackVariable, 201
  - Append, 198
  - arguments, 201
  - arrayAction, 201
  - ArrayActions, 198
  - Ascii, 198
  - Automatic, 199
  - clickCheckedText, 202
  - clicked, 200
  - clickText, 202
  - confirmAction, 202
  - confirmText, 202
  - creationOption, 202
  - CreationOptionNames, 198
  - customisationName, 203
  - dbValueChanged, 200
  - Default, 198
  - displayAlarmState, 203
  - DockBottom, 198
  - DockBottomTabbed, 198
  - DockFloating, 198
  - DockLeft, 198
  - DockLeftTabbed, 198
  - DockRight, 198
  - DockRightTabbed, 198
  - DockTop, 198
  - DockTopTabbed, 198
  - Engineer, 200
  - Fixed, 199
  - Floating, 198
  - format, 203
  - Formats, 198
  - guiFile, 203
  - Icon, 199
  - Index, 198
  - int, 203
  - Integer, 199
  - labelText, 204

leadingZero, 204  
LocalEnumeration, 199  
localEnumeration, 204  
LogOutput, 199  
NewTab, 198  
NewWindow, 198  
None, 199  
notation, 205  
Notations, 199  
Open, 198  
password, 205  
pixmap0, 205  
pixmap1, 205  
pixmap2, 205  
pixmap3, 205  
pixmap4, 205  
pixmap5, 205  
pixmap6, 205  
pixmap7, 206  
precision, 206  
pressed, 200  
pressText, 206  
prioritySubstitutions, 206  
program, 206  
programStartupOption, 206  
ProgramStartupOptionNames, 199  
QEPushButton, 200  
released, 200  
releaseText, 206  
requestAction, 201  
Scientific, 199  
Scientist, 200  
State, 199  
StdOutput, 199  
subscribe, 207  
Terminal, 199  
Text, 199  
TextAndIcon, 199  
Time, 199  
trailingZeros, 207  
UnsignedInteger, 199  
updateOption, 207  
UpdateOptions, 199  
useDbPrecision, 207  
User, 200  
userLevelEnabled, 207  
userLevelEngineerStyle, 207  
UserLevels, 199  
userLevelScientistStyle, 207  
userLevelUserStyle, 208  
userLevelVisibility, 208  
variable, 208  
variableAsToolTip, 208  
variableSubstitutions, 208  
visible, 208  
writeOnClick, 209  
writeOnPress, 209  
writeOnRelease, 209  
QEPVNameLists, 209  
QEPvProperties, 209  
    variable, 210  
    variableSubstitutions, 210  
QEPvPropertiesManager, 211  
QERadioButton, 211  
    addUnits, 218  
    alignment, 218  
    allowDrop, 218  
    Append, 215  
    arguments, 218  
    arrayAction, 218  
    ArrayActions, 215  
    Ascii, 215  
    Automatic, 216  
    clickCheckedText, 219  
    clicked, 217  
    clickText, 219  
    confirmAction, 219  
    confirmText, 219  
    creationOption, 219  
    CreationOptionNames, 215  
    customisationName, 220  
    dbValueChanged, 217  
    Default, 216  
    displayAlarmState, 220  
    DockBottom, 215  
    DockBottomTabbed, 215  
    DockFloating, 215  
    DockLeft, 215  
    DockLeftTabbed, 215  
    DockRight, 215  
    DockRightTabbed, 215  
    DockTop, 215  
    DockTopTabbed, 215  
    Engineer, 217  
    Fixed, 216  
    Floating, 216  
    format, 220  
    Formats, 215  
    guiFile, 220  
    Icon, 217

- Index, [215](#)
- int, [220](#)
- Integer, [216](#)
- labelText, [221](#)
- leadingZero, [221](#)
- LocalEnumeration, [216](#)
- localEnumeration, [221](#)
- LogOutput, [216](#)
- NewTab, [215](#)
- NewWindow, [215](#)
- None, [216](#)
- notation, [222](#)
- Notations, [216](#)
- Open, [215](#)
- password, [222](#)
- pixmap0, [222](#)
- pixmap1, [222](#)
- pixmap2, [222](#)
- pixmap3, [222](#)
- pixmap4, [222](#)
- pixmap5, [222](#)
- pixmap6, [222](#)
- pixmap7, [223](#)
- precision, [223](#)
- pressed, [217](#)
- pressText, [223](#)
- prioritySubstitutions, [223](#)
- program, [223](#)
- programStartupOption, [223](#)
- ProgramStartupOptionNames, [216](#)
- QERadioButton, [217](#)
- released, [218](#)
- releaseText, [223](#)
- requestAction, [218](#)
- Scientific, [216](#)
- Scientist, [217](#)
- State, [217](#)
- StdOutput, [216](#)
- subscribe, [224](#)
- Terminal, [216](#)
- Text, [217](#)
- TextAndIcon, [217](#)
- Time, [216](#)
- trailingZeros, [224](#)
- UnsignedInteger, [216](#)
- updateOption, [224](#)
- UpdateOptions, [216](#)
- useDbPrecision, [224](#)
- User, [217](#)
- userLevelEnabled, [224](#)
- userLevelEngineerStyle, [224](#)
- UserLevels, [217](#)
- userLevelScientistStyle, [224](#)
- userLevelUserStyle, [225](#)
- userLevelVisibility, [225](#)
- variable, [225](#)
- variableAsToolTip, [225](#)
- variableSubstitutions, [225](#)
- visible, [225](#)
- writeOnClick, [226](#)
- writeOnPress, [226](#)
- writeOnRelease, [226](#)
- QERecipe, [226](#)
- QERecordFieldName, [228](#)
- QERecordSpec, [229](#)
- QERecordSpecList, [229](#)
- QEScript, [229](#)
  - allowDrop, [234](#)
  - displayAlarmState, [234](#)
  - Engineer, [233](#)
  - int, [234](#)
  - Scientist, [233](#)
  - User, [233](#)
  - userLevelEnabled, [234](#)
  - userLevelEngineerStyle, [234](#)
  - UserLevels, [233](#)
  - userLevelScientistStyle, [234](#)
  - userLevelUserStyle, [235](#)
  - userLevelVisibility, [235](#)
  - variableAsToolTip, [235](#)
  - visible, [235](#)
- QEShape, [235](#)
  - allowDrop, [241](#)
  - animation1, [241](#)
  - animation2, [241](#)
  - animation3, [241](#)
  - animation4, [242](#)
  - animation5, [242](#)
  - animation6, [242](#)
  - animationOptions, [239](#)
  - color1, [242](#)
  - color10, [242](#)
  - color2, [242](#)
  - color3, [242](#)
  - color4, [242](#)
  - color5, [242](#)
  - color6, [243](#)
  - color7, [243](#)
  - color8, [243](#)
  - color9, [243](#)

- dbValueChanged1, [240](#)
- dbValueChanged2, [240](#)
- dbValueChanged3, [240](#)
- dbValueChanged4, [241](#)
- dbValueChanged5, [241](#)
- dbValueChanged6, [241](#)
- displayAlarmState, [243](#)
- Engineer, [240](#)
- int, [243](#)
- offset1, [243](#)
- offset2, [244](#)
- offset3, [244](#)
- offset4, [244](#)
- offset5, [244](#)
- offset6, [244](#)
- point1, [244](#)
- point10, [244](#)
- point2, [244](#)
- point3, [244](#)
- point4, [245](#)
- point5, [245](#)
- point6, [245](#)
- point7, [245](#)
- point8, [245](#)
- point9, [245](#)
- QEShape, [240](#)
- scale2, [245](#)
- scale3, [245](#)
- scale4, [245](#)
- scale5, [246](#)
- scale6, [246](#)
- Scientist, [240](#)
- shapeOptions, [239](#)
- User, [240](#)
- userLevelEnabled, [246](#)
- userLevelEngineerStyle, [246](#)
- UserLevels, [240](#)
- userLevelScientistStyle, [246](#)
- userLevelUserStyle, [246](#)
- userLevelVisibility, [247](#)
- variable1, [247](#)
- variable2, [247](#)
- variable3, [247](#)
- variable4, [247](#)
- variable5, [247](#)
- variable6, [248](#)
- variableAsToolTip, [248](#)
- variableSubstitutions, [248](#)
- visible, [248](#)
- QESlider, [248](#)
- allowDrop, [251](#)
- dbValueChanged, [250](#)
- displayAlarmState, [251](#)
- Engineer, [250](#)
- int, [251](#)
- Scientist, [250](#)
- subscribe, [251](#)
- User, [250](#)
- userLevelEnabled, [251](#)
- userLevelEngineerStyle, [251](#)
- UserLevels, [250](#)
- userLevelScientistStyle, [251](#)
- userLevelUserStyle, [252](#)
- userLevelVisibility, [252](#)
- variable, [252](#)
- variableAsToolTip, [252](#)
- variableSubstitutions, [252](#)
- visible, [252](#)
- writeOnChange, [250](#)
- QESpinBox, [253](#)
- allowDrop, [255](#)
- dbValueChanged, [255](#)
- displayAlarmState, [255](#)
- Engineer, [255](#)
- int, [255](#)
- Scientist, [255](#)
- subscribe, [255](#)
- User, [255](#)
- userLevelEnabled, [256](#)
- userLevelEngineerStyle, [256](#)
- UserLevels, [255](#)
- userLevelScientistStyle, [256](#)
- userLevelUserStyle, [256](#)
- userLevelVisibility, [256](#)
- variable, [257](#)
- variableAsToolTip, [257](#)
- variableSubstitutions, [257](#)
- visible, [257](#)
- QEStriptChart, [257](#)
- variableSubstitutions, [259](#)
- QEStriptChartAdjustPVDialo, [260](#)
- QEStriptChartContextMenu, [260](#)
- QEStriptChartContextMenu, [260](#)
- QEStriptChartItem, [261](#)
- QEStriptChartNames, [262](#)
- QEStriptChartPushButtonSpecifications, [263](#)
- QEStriptChartRangeDialog, [263](#)
- QEStriptChartState, [263](#)
- QEStriptChartStateList, [264](#)
- QEStriptChartStatistics, [264](#)

- QEStripChartTimeDialog, 265
- QEStripChartToolBar, 265
- QEStripChartToolBar::OwnWidgets, 51
- QESubstitutedLabel, 266
  - labelText, 266
  - textSubstitutions, 267
- readbackLabelVariable1
  - QEPeiodic, 185
- readbackLabelVariable2
  - QEPeiodic, 185
- recording, 267
- regionOfInterest1HVariable
  - QEImage, 149
- regionOfInterest1WVariable
  - QEImage, 149
- regionOfInterest1XVariable
  - QEImage, 149
- regionOfInterest1YVariable
  - QEImage, 149
- regionOfInterest2HVariable
  - QEImage, 149
- regionOfInterest2WVariable
  - QEImage, 149
- regionOfInterest2XVariable
  - QEImage, 150
- regionOfInterest2YVariable
  - QEImage, 150
- regionOfInterest3HVariable
  - QEImage, 150
- regionOfInterest3WVariable
  - QEImage, 150
- regionOfInterest3XVariable
  - QEImage, 150
- regionOfInterest3YVariable
  - QEImage, 150
- regionOfInterest4HVariable
  - QEImage, 150
- regionOfInterest4WVariable
  - QEImage, 150
- regionOfInterest4XVariable
  - QEImage, 150
- regionOfInterest4YVariable
  - QEImage, 151
- released
  - QECheckBox, 80
  - QEPushButton, 200
  - QERadioButton, 218
- releaseText
  - QECheckBox, 86
- QEPushButton, 206
- QERadioButton, 223
- requestAction
  - QECheckBox, 80
  - QEPushButton, 201
  - QERadioButton, 218
- RESIZE\_OPTION\_FIT
  - QEImage, 138
- RESIZE\_OPTION\_ZOOM
  - QEImage, 138
- resizeOption
  - QEImage, 151
- ResizeOptions
  - QEImage, 138
- resizeOptions
  - QEImage, 138
- rgb1
  - QEImage, 137
- rgb2
  - QEImage, 137
- rgb3
  - QEImage, 137
- Right\_To\_Left
  - QEAnalogIndicator, 59
- Rotate180
  - QEImage, 138
- Rotate90Left
  - QEImage, 138
- Rotate90Right
  - QEImage, 138
- rotation
  - QEImage, 151
- ROTATION\_0
  - QEImage, 138
- ROTATION\_180
  - QEImage, 139
- ROTATION\_90\_LEFT
  - QEImage, 139
- ROTATION\_90\_RIGHT
  - QEImage, 138
- RotationOptions
  - QEImage, 138
- rotationOptions
  - QEImage, 138
- Scale
  - QEAnalogIndicator, 58
- scale2
  - QEShape, 245
- scale3



- QEGenericEdit, 114
- shapeOptions
  - QEShape, 239
- showScale
  - QEAnalogIndicator, 60
- showText
  - QEAnalogIndicator, 60
- showTime
  - QEImage, 151
- SO\_AREA4
  - QEImage, 139
- SO\_BEAM
  - QEImage, 139
- SO\_HSLICE
  - QEImage, 139
- SO\_NONE
  - QEImage, 139
- SO\_PANNING
  - QEImage, 139
- SO\_PROFILE
  - QEImage, 139
- SO\_TARGET
  - QEImage, 139
- SO\_VSLICE
  - QEImage, 139
- SolidSmallCrosshair
  - QEImage, 139
- spanAngle
  - QEAnalogIndicator, 60
- State
  - QECheckBox, 79
  - QEPushButton, 199
  - QERadioButton, 217
- StdOutput
  - QECheckBox, 79
  - QEImage, 138
  - QEPushButton, 199
  - QERadioButton, 216
- subscribe
  - QECheckBox, 86
  - QEComboBox, 92
  - QEGenericEdit, 115
  - QEPeriodic, 185
  - QEPushButton, 207
  - QERadioButton, 224
  - QESlider, 251
  - QESpinBox, 255
- substitutedTitle
  - QEGroupBox, 119

- targetColor
  - QEImage, 151
- TargetOptions
  - QEImage, 139
- targetTriggerVariable
  - QEImage, 151
- targetXVariable
  - QEImage, 151
- targetYVariable
  - QEImage, 151
- Terminal
  - QCheckBox, 78
  - QEImage, 138
  - QEPushButton, 199
  - QERadioButton, 216
- Text
  - QCheckBox, 79
  - QELabel, 159
  - QEPushButton, 199
  - QERadioButton, 217
- TextAndIcon
  - QCheckBox, 79
  - QEPushButton, 199
  - QERadioButton, 217
- textSubstitutions
  - QEGroupBox, 119
  - QESubstitutedLabel, 267
- Time
  - QEAnalogProgressBar, 64
  - QCheckBox, 78
  - QELabel, 158
  - QELineEdit, 167
  - QEPushButton, 199
  - QERadioButton, 216
- timeColor
  - QEImage, 151
- Top\_To\_Bottom
  - QEAnalogIndicator, 59
- trace, 268
- trailingZeros
  - QEAnalogProgressBar, 67
  - QCheckBox, 86
  - QELabel, 163
  - QELineEdit, 169
  - QEPushButton, 207
  - QERadioButton, 224
- UnsignedInteger
  - QEAnalogProgressBar, 64
  - QCheckBox, 78
  - QELabel, 158
  - QELineEdit, 167
  - QEPushButton, 199
  - QERadioButton, 216
- UPDATE\_PIXMAP
  - QELabel, 159
- UPDATE\_TEXT
  - QELabel, 159
- updateImage
  - mpegSource, 50
- updateOption
  - QCheckBox, 86
  - QELabel, 163
  - QEPushButton, 207
  - QERadioButton, 224
- UpdateOptions
  - QCheckBox, 79
  - QELabel, 158
  - QEPushButton, 199
  - QERadioButton, 216
- updateOptions
  - QELabel, 159
- URL
  - QEImage, 152
- useDbDisplayLimits
  - QEAnalogProgressBar, 67
- useDbEnumerations
  - QEComboBox, 91
- useDbPrecision
  - QEAnalogProgressBar, 67
  - QCheckBox, 86
  - QELabel, 163
  - QELineEdit, 169
  - QEPushButton, 207
  - QERadioButton, 224
- useFalseColour
  - QEImage, 152
- User
  - QEAnalogProgressBar, 65
  - QEBitStatus, 71
  - QCheckBox, 79
  - QEComboBox, 91
  - QEConfiguredLayout, 96
  - QEFileBrowser, 101
  - QEFrame, 106
  - QEGenericEdit, 113
  - QEGroupBox, 118
  - QEImage, 139
  - QELabel, 159
  - QELog, 174

- QEPeriodic, [184](#)
- QEPlot, [191](#)
- QEPushButton, [200](#)
- QERadioButton, [217](#)
- QEScript, [233](#)
- QEShape, [240](#)
- QESlider, [250](#)
- QESpinBox, [255](#)
- userInfoStruct, [269](#)
- userLevelEnabled
  - QEAnalogProgressBar, [68](#)
  - QEBitStatus, [72](#)
  - QECheckBox, [87](#)
  - QEComboBox, [92](#)
  - QEConfiguredLayout, [96](#)
  - QEFileBrowser, [102](#)
  - QEFrame, [107](#)
  - QEGenericEdit, [115](#)
  - QEGroupBox, [119](#)
  - QEImage, [152](#)
  - QELabel, [163](#)
  - QELog, [174](#)
  - QEPeriodic, [185](#)
  - QEPlot, [192](#)
  - QEPushButton, [207](#)
  - QERadioButton, [224](#)
  - QEScript, [234](#)
  - QEShape, [246](#)
  - QESlider, [251](#)
  - QESpinBox, [256](#)
- userLevelEngineerStyle
  - QEAnalogProgressBar, [68](#)
  - QEBitStatus, [72](#)
  - QECheckBox, [87](#)
  - QEComboBox, [92](#)
  - QEConfiguredLayout, [97](#)
  - QEFileBrowser, [102](#)
  - QEFrame, [107](#)
  - QEGenericEdit, [115](#)
  - QEGroupBox, [119](#)
  - QEImage, [152](#)
  - QELabel, [163](#)
  - QELog, [175](#)
  - QEPeriodic, [185](#)
  - QEPlot, [192](#)
  - QEPushButton, [207](#)
  - QERadioButton, [224](#)
  - QEScript, [234](#)
  - QEShape, [246](#)
  - QESlider, [251](#)
  - QESpinBox, [256](#)
- QESpinBox, [256](#)
- UserLevels
  - QEAnalogProgressBar, [64](#)
  - QEBitStatus, [71](#)
  - QECheckBox, [79](#)
  - QEComboBox, [91](#)
  - QEConfiguredLayout, [96](#)
  - QEFileBrowser, [101](#)
  - QEFrame, [106](#)
  - QEGenericEdit, [112](#)
  - QEGroupBox, [118](#)
  - QEImage, [139](#)
  - QELabel, [159](#)
  - QELog, [174](#)
  - QEPeriodic, [184](#)
  - QEPlot, [191](#)
  - QEPushButton, [199](#)
  - QERadioButton, [217](#)
  - QEScript, [233](#)
  - QEShape, [240](#)
  - QESlider, [250](#)
  - QESpinBox, [255](#)
- userLevelScientistStyle
  - QEAnalogProgressBar, [68](#)
  - QEBitStatus, [72](#)
  - QECheckBox, [87](#)
  - QEComboBox, [93](#)
  - QEConfiguredLayout, [97](#)
  - QEFileBrowser, [102](#)
  - QEFrame, [107](#)
  - QEGenericEdit, [115](#)
  - QEGroupBox, [119](#)
  - QEImage, [152](#)
  - QELabel, [163](#)
  - QELog, [175](#)
  - QEPeriodic, [185](#)
  - QEPlot, [193](#)
  - QEPushButton, [207](#)
  - QERadioButton, [224](#)
  - QEScript, [234](#)
  - QEShape, [246](#)
  - QESlider, [251](#)
  - QESpinBox, [256](#)
- userLevelUserStyle
  - QEAnalogProgressBar, [68](#)
  - QEBitStatus, [73](#)
  - QECheckBox, [87](#)
  - QEComboBox, [93](#)
  - QEConfiguredLayout, [97](#)
  - QEFileBrowser, [103](#)

- QEFrame, 107
- QEGenericEdit, 115
- QEGroupBox, 119
- QEImage, 152
- QELabel, 164
- QELog, 175
- QEPeriodic, 186
- QEPlot, 193
- QEPushButton, 208
- QERadioButton, 225
- QEScript, 235
- QEShape, 246
- QESlider, 252
- QESpinBox, 256
- userLevelVisibility
  - QEAnalogProgressBar, 68
  - QEBitStatus, 73
  - QECheckBox, 87
  - QEComboBox, 93
  - QEConfiguredLayout, 97
  - QEFileBrowser, 103
  - QEFrame, 108
  - QEGenericEdit, 116
  - QEGroupBox, 120
  - QEImage, 153
  - QELabel, 164
  - QELog, 175
  - QEPeriodic, 186
  - QEPlot, 193
  - QEPushButton, 208
  - QERadioButton, 225
  - QEScript, 235
  - QEShape, 247
  - QESlider, 252
  - QESpinBox, 256
- value
  - QEAnalogIndicator, 60
  - QEAnalogProgressBar, 69
- ValueScaling, 269
- variable
  - QEAnalogProgressBar, 69
  - QEBitStatus, 73
  - QECheckBox, 88
  - QEComboBox, 93
  - QEFileBrowser, 103
  - QEGenericEdit, 116
  - QELabel, 164
  - QEPushButton, 208
  - QEPvProperties, 210
  - QERadioButton, 225
  - QESlider, 252
  - QESpinBox, 257
- variable1
  - QEPlot, 193
  - QEShape, 247
- variable2
  - QEPlot, 193
  - QEShape, 247
- variable3
  - QEPlot, 193
  - QEShape, 247
- variable4
  - QEPlot, 193
  - QEShape, 247
- variable5
  - QEShape, 247
- variable6
  - QEShape, 248
- variableAsToolTip
  - QEAnalogProgressBar, 69
  - QEBitStatus, 73
  - QECheckBox, 88
  - QEComboBox, 93
  - QEConfiguredLayout, 97
  - QEFileBrowser, 103
  - QEFrame, 108
  - QEGenericEdit, 116
  - QEGroupBox, 120
  - QEImage, 153
  - QELabel, 164
  - QELog, 175
  - QEPeriodic, 186
  - QEPlot, 194
  - QEPushButton, 208
  - QERadioButton, 225
  - QEScript, 235
  - QEShape, 248
  - QESlider, 252
  - QESpinBox, 257
- variableSubstitutions
  - QEAnalogProgressBar, 69
  - QEBitStatus, 73
  - QECheckBox, 88
  - QEComboBox, 93
  - QEFileBrowser, 103
  - QEGenericEdit, 116
  - QEImage, 153
  - QELabel, 164
  - QEPeriodic, 186

- QEPlot, [194](#)
- QEPushButton, [208](#)
- QEPvProperties, [210](#)
- QERadioButton, [225](#)
- QEShape, [248](#)
- QESlider, [252](#)
- QESpinBox, [257](#)
- QESTripChart, [259](#)
- verticalFlip
  - QEImage, [153](#)
- vertSliceColor
  - QEImage, [153](#)
- VideoWidget, [270](#)
- visible
  - QEAnalogProgressBar, [69](#)
  - QEBitStatus, [73](#)
  - QECheckBox, [88](#)
  - QEComboBox, [94](#)
  - QEConfiguredLayout, [98](#)
  - QEFileBrowser, [103](#)
  - QEFrame, [108](#)
  - QEGenericEdit, [116](#)
  - QEGroupBox, [120](#)
  - QEImage, [153](#)
  - QELabel, [164](#)
  - QELog, [176](#)
  - QEPeriodic, [186](#)
  - QEPlot, [194](#)
  - QEPushButton, [208](#)
  - QERadioButton, [225](#)
  - QEScript, [235](#)
  - QEShape, [248](#)
  - QESlider, [252](#)
  - QESpinBox, [257](#)
- widthVariable
  - QEImage, [153](#)
- writeButtonVariable1
  - QEPeriodic, [186](#)
- writeButtonVariable2
  - QEPeriodic, [186](#)
- writeOnChange
  - QEComboBox, [91](#)
  - QESlider, [250](#)
- writeOnClick
  - QECheckBox, [88](#)
  - QEPushButton, [209](#)
  - QERadioButton, [226](#)
- writeOnEnter
  - QEGenericEdit, [116](#)
- writeOnFinish
  - QEGenericEdit, [116](#)
- writeOnLoseFocus
  - QEGenericEdit, [117](#)
- writeOnPress
  - QECheckBox, [88](#)
  - QEPushButton, [209](#)
  - QERadioButton, [226](#)
- writeOnRelease
  - QECheckBox, [88](#)
  - QEPushButton, [209](#)
  - QERadioButton, [226](#)
- yuv422
  - QEImage, [137](#)
- yuv444
  - QEImage, [137](#)
- Zoom
  - QEImage, [138](#)
- zoomMenu, [271](#)