

EPICS QT Framework

3.0.1

Generated by Doxygen 1.7.4

Tue Jan 20 2015 13:39:45

Contents

1 QE framework - EPICS aware Qt Widgets and data access classes	1
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
2 GNU General Public License	5
3 ASgui screen shots	7
4 other applications using epicsqt widgets	13
5 Qt Designer	15
6 Qt Creator	17
7 Class Index	19
7.1 Class Hierarchy	19
8 Class Index	23
8.1 Class List	23
9 Class Documentation	27
9.1 _CopyPaste Class Reference	27

9.2	_Field Class Reference	27
9.3	_Item Class Reference	28
9.4	_QDialogItem Class Reference	29
9.5	_QPushButtonGroup Class Reference	29
9.6	_QTableWidgetFileBrowser Class Reference	29
9.7	_QTableWidgetLog Class Reference	30
9.8	_QTableWidgetScript Class Reference	30
9.9	arealInfo Class Reference	30
9.10	QEAnalogIndicator::Band Struct Reference	31
9.11	QEAnalogIndicator::BandList Class Reference	31
9.12	QEPeriodic::elementInfoStruct Struct Reference	31
9.13	FFBuffer Class Reference	32
9.14	FFThread Class Reference	32
9.15	flipRotateMenu Class Reference	33
9.16	fullScreenWindow Class Reference	33
9.17	histogram Class Reference	34
9.18	histogramScroll Class Reference	34
9.19	historicImage Class Reference	34
9.20	imageContextMenu Class Reference	35
9.21	imageDisplayProperties Class Reference	36
9.22	imageInfo Class Reference	37
9.23	imageMarkup Class Reference	38
9.24	imageUpdateIndicator Class Reference	40
9.25	loginWidget Class Reference	40
9.26	markupCrosshair1 Class Reference	40
9.27	markupCrosshair2 Class Reference	41
9.28	markupDisplayMenu Class Reference	42
9.29	markupEllipse Class Reference	42
9.30	markupHLine Class Reference	43
9.30.1	Member Function Documentation	44
9.30.1.1	drawMarkup	44
9.31	markupItem Class Reference	44
9.32	markupLine Class Reference	46
9.33	markupRegion Class Reference	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 updatelImage	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 QEAnalogIndicator 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 DisplayAlarmStateOptions	64
9.47.1.3 Formats	65
9.47.1.4 Notations	65
9.47.1.5 UserLevels	65
9.47.2 Constructor & Destructor Documentation	65
9.47.2.1 QEAnalogProgressBar	65
9.47.2.2 QEAnalogProgressBar	66
9.47.3 Member Function Documentation	66
9.47.3.1 dbValueChanged	66
9.47.4 Property Documentation	66
9.47.4.1 addUnits	66
9.47.4.2 alarmSeverityDisplayMode	66
9.47.4.3 allowDrop	66
9.47.4.4 arrayAction	66
9.47.4.5 displayAlarmState	67
9.47.4.6 displayAlarmStateOption	67
9.47.4.7 format	67
9.47.4.8 int	67
9.47.4.9 leadingZero	67
9.47.4.10 localEnumeration	67
9.47.4.11 notation	68
9.47.4.12 precision	68
9.47.4.13 trailingZeros	68
9.47.4.14 useDbDisplayLimits	68
9.47.4.15 useDbPrecision	69
9.47.4.16 userLevelEnabled	69

9.47.4.17 userLevelEngineerStyle	69
9.47.4.18 userLevelScientistStyle	69
9.47.4.19 userLevelUserStyle	69
9.47.4.20 userLevelVisibility	69
9.47.4.21 value	70
9.47.4.22 variable	70
9.47.4.23 variableAsToolTip	70
9.47.4.24 variableSubstitutions	70
9.47.4.25 visible	70
9.48 QEBitStatus Class Reference	70
9.48.1 Member Enumeration Documentation	73
9.48.1.1 DisplayAlarmStateOptions	73
9.48.1.2 UserLevels	73
9.48.2 Member Function Documentation	73
9.48.2.1 dbValueChanged	73
9.48.3 Property Documentation	73
9.48.3.1 allowDrop	73
9.48.3.2 displayAlarmState	73
9.48.3.3 displayAlarmStateOption	74
9.48.3.4 int	74
9.48.3.5 userLevelEnabled	74
9.48.3.6 userLevelEngineerStyle	74
9.48.3.7 userLevelScientistStyle	74
9.48.3.8 userLevelUserStyle	75
9.48.3.9 userLevelVisibility	75
9.48.3.10 variable	75
9.48.3.11 variableAsToolTip	75
9.48.3.12 variableSubstitutions	75
9.48.3.13 visible	75
9.49 QECheckBox Class Reference	76
9.49.1 Member Enumeration Documentation	80
9.49.1.1 ArrayActions	80
9.49.1.2 CreationOptionNames	80
9.49.1.3 DisplayAlarmStateOptions	80

9.49.1.4	Formats	81
9.49.1.5	Notations	81
9.49.1.6	ProgramStartupOptionNames	81
9.49.1.7	UpdateOptions	81
9.49.1.8	UserLevels	82
9.49.2	Constructor & Destructor Documentation	82
9.49.2.1	QECheckBox	82
9.49.2.2	QECheckBox	82
9.49.3	Member Function Documentation	82
9.49.3.1	clicked	82
9.49.3.2	dbValueChanged	82
9.49.3.3	pressed	83
9.49.3.4	released	83
9.49.3.5	requestAction	83
9.49.4	Property Documentation	83
9.49.4.1	addUnits	83
9.49.4.2	alignment	83
9.49.4.3	allowDrop	83
9.49.4.4	arguments	83
9.49.4.5	arrayAction	84
9.49.4.6	clickCheckedText	84
9.49.4.7	clickText	84
9.49.4.8	confirmAction	84
9.49.4.9	confirmText	84
9.49.4.10	creationOption	85
9.49.4.11	customisationName	85
9.49.4.12	displayAlarmState	85
9.49.4.13	displayAlarmStateOption	85
9.49.4.14	format	85
9.49.4.15	guiFile	86
9.49.4.16	int	86
9.49.4.17	labelText	86
9.49.4.18	leadingZero	86
9.49.4.19	localEnumeration	86

9.49.4.20 notation	87
9.49.4.21 password	87
9.49.4.22 pixmap0	87
9.49.4.23 pixmap1	87
9.49.4.24 pixmap2	87
9.49.4.25 pixmap3	88
9.49.4.26 pixmap4	88
9.49.4.27 pixmap5	88
9.49.4.28 pixmap6	88
9.49.4.29 pixmap7	88
9.49.4.30 precision	88
9.49.4.31 pressText	88
9.49.4.32 prioritySubstitutions	88
9.49.4.33 program	89
9.49.4.34 programStartupOption	89
9.49.4.35 releaseText	89
9.49.4.36 subscribe	89
9.49.4.37 trailingZeros	89
9.49.4.38 updateOption	89
9.49.4.39 useDbPrecision	89
9.49.4.40 userLevelEnabled	89
9.49.4.41 userLevelEngineerStyle	90
9.49.4.42 userLevelScientistStyle	90
9.49.4.43 userLevelUserStyle	90
9.49.4.44 userLevelVisibility	90
9.49.4.45 variable	90
9.49.4.46 variableAsToolTip	91
9.49.4.47 variableSubstitutions	91
9.49.4.48 visible	91
9.49.4.49 writeOnClick	91
9.49.4.50 writeOnPress	91
9.49.4.51 writeOnRelease	91
9.50 QECheckBoxManager Class Reference	91
9.51 QEComboBox Class Reference	92

9.51.1 Member Enumeration Documentation	94
9.51.1.1 DisplayAlarmStateOptions	94
9.51.1.2 UserLevels	95
9.51.2 Member Function Documentation	95
9.51.2.1 dbValueChanged	95
9.51.3 Member Data Documentation	95
9.51.3.1 useDbEnumerations	95
9.51.3.2 writeOnChange	95
9.51.4 Property Documentation	95
9.51.4.1 allowDrop	95
9.51.4.2 displayAlarmState	95
9.51.4.3 displayAlarmStateOption	96
9.51.4.4 int	96
9.51.4.5 localEnumeration	96
9.51.4.6 subscribe	96
9.51.4.7 userLevelEnabled	96
9.51.4.8 userLevelEngineerStyle	96
9.51.4.9 userLevelScientistStyle	97
9.51.4.10 userLevelUserStyle	97
9.51.4.11 userLevelVisibility	97
9.51.4.12 variable	97
9.51.4.13 variableAsToolTip	97
9.51.4.14 variableSubstitutions	97
9.51.4.15 visible	98
9.52 QEConfiguredLayout Class Reference	98
9.52.1 Member Enumeration Documentation	100
9.52.1.1 DisplayAlarmStateOptions	100
9.52.1.2 UserLevels	100
9.52.2 Property Documentation	101
9.52.2.1 allowDrop	101
9.52.2.2 displayAlarmState	101
9.52.2.3 displayAlarmStateOption	101
9.52.2.4 int	101
9.52.2.5 userLevelEnabled	101

9.52.2.6 userLevelEngineerStyle	102
9.52.2.7 userLevelScientistStyle	102
9.52.2.8 userLevelUserStyle	102
9.52.2.9 userLevelVisibility	102
9.52.2.10 variableAsToolTip	102
9.52.2.11 visible	102
9.53 QEConfiguredLayoutManager Class Reference	103
9.54 QEFileBrowser Class Reference	103
9.54.1 Detailed Description	106
9.54.2 Member Enumeration Documentation	106
9.54.2.1 DisplayAlarmStateOptions	106
9.54.2.2 UserLevels	106
9.54.3 Member Function Documentation	107
9.54.3.1 selected	107
9.54.4 Property Documentation	107
9.54.4.1 allowDrop	107
9.54.4.2 displayAlarmState	107
9.54.4.3 displayAlarmStateOption	107
9.54.4.4 int	107
9.54.4.5 userLevelEnabled	108
9.54.4.6 userLevelEngineerStyle	108
9.54.4.7 userLevelScientistStyle	108
9.54.4.8 userLevelUserStyle	108
9.54.4.9 userLevelVisibility	108
9.54.4.10 variable	109
9.54.4.11 variableAsToolTip	109
9.54.4.12 variableSubstitutions	109
9.54.4.13 visible	109
9.55 QEForm Class Reference	109
9.56 QEFrame Class Reference	111
9.56.1 Member Enumeration Documentation	112
9.56.1.1 DisplayAlarmStateOptions	112
9.56.1.2 UserLevels	112
9.56.2 Property Documentation	112

9.56.2.1	allowDrop	112
9.56.2.2	displayAlarmState	113
9.56.2.3	displayAlarmStateOption	113
9.56.2.4	int	113
9.56.2.5	userLevelEnabled	113
9.56.2.6	userLevelEngineerStyle	113
9.56.2.7	userLevelScientistStyle	114
9.56.2.8	userLevelUserStyle	114
9.56.2.9	userLevelVisibility	114
9.56.2.10	variableAsToolTip	114
9.56.2.11	visible	114
9.57	QEGenericButton Class Reference	114
9.58	QEGenericEdit Class Reference	116
9.58.1	Member Enumeration Documentation	119
9.58.1.1	DisplayAlarmStateOptions	119
9.58.1.2	UserLevels	119
9.58.2	Constructor & Destructor Documentation	120
9.58.2.1	QEGenericEdit	120
9.58.2.2	QEGenericEdit	120
9.58.3	Member Function Documentation	120
9.58.3.1	getConfirmWrite	120
9.58.3.2	getSubscribe	120
9.58.3.3	getWriteOnEnter	120
9.58.3.4	getWriteOnFinish	120
9.58.3.5	getWriteOnLoseFocus	120
9.58.3.6	setConfirmWrite	120
9.58.3.7	setSubscribe	121
9.58.3.8	setWriteOnEnter	121
9.58.3.9	setWriteOnFinish	121
9.58.3.10	setWriteOnLoseFocus	121
9.58.4	Property Documentation	121
9.58.4.1	allowDrop	121
9.58.4.2	confirmWrite	121
9.58.4.3	displayAlarmState	121

9.58.4.4	displayAlarmStateOption	122
9.58.4.5	int	122
9.58.4.6	subscribe	122
9.58.4.7	userLevelEnabled	122
9.58.4.8	userLevelEngineerStyle	122
9.58.4.9	userLevelScientistStyle	123
9.58.4.10	userLevelUserStyle	123
9.58.4.11	userLevelVisibility	123
9.58.4.12	variable	123
9.58.4.13	variableAsToolTip	123
9.58.4.14	variableSubstitutions	123
9.58.4.15	visible	124
9.58.4.16	writeOnEnter	124
9.58.4.17	writeOnFinish	124
9.58.4.18	writeOnLoseFocus	124
9.59	QEGroupBox Class Reference	124
9.59.1	Member Enumeration Documentation	125
9.59.1.1	DisplayAlarmStateOptions	125
9.59.1.2	UserLevels	126
9.59.2	Property Documentation	126
9.59.2.1	allowDrop	126
9.59.2.2	displayAlarmState	126
9.59.2.3	displayAlarmStateOption	126
9.59.2.4	int	127
9.59.2.5	substitutedTitle	127
9.59.2.6	textSubstitutions	127
9.59.2.7	userLevelEnabled	127
9.59.2.8	userLevelEngineerStyle	127
9.59.2.9	userLevelScientistStyle	127
9.59.2.10	userLevelUserStyle	128
9.59.2.11	userLevelVisibility	128
9.59.2.12	variableAsToolTip	128
9.59.2.13	visible	128
9.60	QEImage Class Reference	128

9.60.1	Member Enumeration Documentation	145
9.60.1.1	DisplayAlarmStateOptions	145
9.60.1.2	ellipseVariableDefinitions	146
9.60.1.3	EllipseVariableDefinitions	146
9.60.1.4	FormatOptions	146
9.60.1.5	ProgramStartupOptionNames	146
9.60.1.6	ResizeOptions	147
9.60.1.7	resizeOptions	147
9.60.1.8	RotationOptions	147
9.60.1.9	rotationOptions	147
9.60.1.10	selectOptions	148
9.60.1.11	TargetOptions	148
9.60.1.12	UserLevels	148
9.60.2	Constructor & Destructor Documentation	148
9.60.2.1	QEImage	148
9.60.2.2	QEImage	149
9.60.3	Member Function Documentation	149
9.60.3.1	dbValueChanged	149
9.60.4	Member Data Documentation	149
9.60.4.1	displayButtonBar	149
9.60.4.2	initialVertScrollPos	149
9.60.5	Property Documentation	149
9.60.5.1	allowDrop	149
9.60.5.2	areaColor	149
9.60.5.3	arguments1	150
9.60.5.4	arguments2	150
9.60.5.5	autoBrightnessContrast	150
9.60.5.6	beamColor	150
9.60.5.7	beamXVariable	150
9.60.5.8	beamYVariable	150
9.60.5.9	bitDepthVariable	150
9.60.5.10	briefInfoArea	150
9.60.5.11	clippingHighVariable	150
9.60.5.12	clippingLowVariable	151

9.60.5.13 clippingOnOffVariable	151
9.60.5.14 contrastReversal	151
9.60.5.15 dimension1Variable	151
9.60.5.16 dimension2Variable	151
9.60.5.17 dimension3Variable	151
9.60.5.18 dimensionsVariable	151
9.60.5.19 displayAlarmState	151
9.60.5.20 displayAlarmStateOption	152
9.60.5.21 displayArea1Selection	152
9.60.5.22 displayArea2Selection	152
9.60.5.23 displayArea3Selection	152
9.60.5.24 displayArea4Selection	152
9.60.5.25 displayBeamSelection	152
9.60.5.26 displayCursorPixelInfo	152
9.60.5.27 displayEllipse	153
9.60.5.28 displayHozSliceSelection	153
9.60.5.29 displayProfileSelection	153
9.60.5.30 displayTargetSelection	153
9.60.5.31 displayVertSliceSelection	153
9.60.5.32 ellipseColor	153
9.60.5.33 ellipseHVariable	153
9.60.5.34 ellipseWVariable	153
9.60.5.35 ellipseXVariable	153
9.60.5.36 ellipseYVariable	154
9.60.5.37 enableArea1Selection	154
9.60.5.38 enableArea2Selection	154
9.60.5.39 enableArea3Selection	154
9.60.5.40 enableArea4Selection	154
9.60.5.41 enableBeamSelection	154
9.60.5.42 enableHozSliceSelection	154
9.60.5.43 enableProfileSelection	154
9.60.5.44 enableTargetSelection	155
9.60.5.45 enableVertSliceSelection	155
9.60.5.46 externalControls	155

9.60.5.47	formatOption	155
9.60.5.48	formatVariable	155
9.60.5.49	heightVariable	155
9.60.5.50	horizontalFlip	155
9.60.5.51	hozSliceColor	155
9.60.5.52	imageVariable	156
9.60.5.53	initialHosScrollPos	156
9.60.5.54	int	156
9.60.5.55	lineProfileArrayVariable	156
9.60.5.56	lineProfileThicknessVariable	156
9.60.5.57	lineProfileX1Variable	156
9.60.5.58	lineProfileX2Variable	156
9.60.5.59	lineProfileY1Variable	156
9.60.5.60	lineProfileY2Variable	157
9.60.5.61	logBrightness	157
9.60.5.62	profileColor	157
9.60.5.63	profileHozArrayVariable	157
9.60.5.64	profileHozThicknessVariable	157
9.60.5.65	profileHozVariable	157
9.60.5.66	profileVertArrayVariable	157
9.60.5.67	profileVertThicknessVariable	157
9.60.5.68	profileVertVariable	157
9.60.5.69	program1	158
9.60.5.70	program2	158
9.60.5.71	programStartupOption1	158
9.60.5.72	programStartupOption2	158
9.60.5.73	regionOfInterest1HVariable	158
9.60.5.74	regionOfInterest1WVariable	158
9.60.5.75	regionOfInterest1XVariable	158
9.60.5.76	regionOfInterest1YVariable	159
9.60.5.77	regionOfInterest2HVariable	159
9.60.5.78	regionOfInterest2WVariable	159
9.60.5.79	regionOfInterest2XVariable	159
9.60.5.80	regionOfInterest2YVariable	159

9.60.5.81 regionOfInterest3HVariable	159
9.60.5.82 regionOfInterest3WVariable	159
9.60.5.83 regionOfInterest3XVariable	159
9.60.5.84 regionOfInterest3YVariable	159
9.60.5.85 regionOfInterest4HVariable	160
9.60.5.86 regionOfInterest4WVariable	160
9.60.5.87 regionOfInterest4XVariable	160
9.60.5.88 regionOfInterest4YVariable	160
9.60.5.89 resizeOption	160
9.60.5.90 rotation	160
9.60.5.91 showTime	160
9.60.5.92 targetColor	160
9.60.5.93 targetTriggerVariable	160
9.60.5.94 targetXVariable	161
9.60.5.95 targetYVariable	161
9.60.5.96 timeColor	161
9.60.5.97 URL	161
9.60.5.98 useFalseColour	161
9.60.5.99 userLevelEnabled	161
9.60.5.100 userLevelEngineerStyle	161
9.60.5.101 userLevelScientistStyle	162
9.60.5.102 userLevelUserStyle	162
9.60.5.103 userLevelVisibility	162
9.60.5.104 variableAsToolTip	162
9.60.5.105 variableSubstitutions	162
9.60.5.106 verticalFlip	162
9.60.5.107 vertSliceColor	162
9.60.5.108 visible	163
9.60.5.109 widthVariable	163
9.61 QEImageMarkupThickness Class Reference	163
9.62 QEImageOptionsDialog Class Reference	163
9.63 QELabel Class Reference	164
9.63.1 Detailed Description	167
9.63.2 Member Enumeration Documentation	168

9.63.2.1	ArrayActions	168
9.63.2.2	DisplayAlarmStateOptions	168
9.63.2.3	Formats	168
9.63.2.4	Notations	168
9.63.2.5	updateOptions	169
9.63.2.6	UpdateOptions	169
9.63.2.7	UserLevels	169
9.63.3	Constructor & Destructor Documentation	169
9.63.3.1	QELabel	169
9.63.3.2	QELabel	169
9.63.4	Member Function Documentation	170
9.63.4.1	dbValueChanged	170
9.63.5	Property Documentation	170
9.63.5.1	addUnits	170
9.63.5.2	allowDrop	170
9.63.5.3	arrayAction	170
9.63.5.4	displayAlarmState	170
9.63.5.5	displayAlarmStateOption	171
9.63.5.6	format	171
9.63.5.7	int	171
9.63.5.8	leadingZero	171
9.63.5.9	localEnumeration	171
9.63.5.10	notation	172
9.63.5.11	pixmap0	172
9.63.5.12	pixmap1	172
9.63.5.13	pixmap2	172
9.63.5.14	pixmap3	172
9.63.5.15	pixmap4	172
9.63.5.16	pixmap5	173
9.63.5.17	pixmap6	173
9.63.5.18	pixmap7	173
9.63.5.19	precision	173
9.63.5.20	trailingZeros	173
9.63.5.21	updateOption	173

9.63.5.22 useDbPrecision	173
9.63.5.23 userLevelEnabled	173
9.63.5.24 userLevelEngineerStyle	174
9.63.5.25 userLevelScientistStyle	174
9.63.5.26 userLevelUserStyle	174
9.63.5.27 userLevelVisibility	174
9.63.5.28 variable	174
9.63.5.29 variableAsToolTip	175
9.63.5.30 variableSubstitutions	175
9.63.5.31 visible	175
9.64 QELineEdit Class Reference	175
9.64.1 Member Enumeration Documentation	177
9.64.1.1 ArrayActions	177
9.64.1.2 Formats	177
9.64.1.3 Notations	177
9.64.2 Constructor & Destructor Documentation	177
9.64.2.1 QELineEdit	177
9.64.2.2 QELineEdit	178
9.64.3 Member Function Documentation	178
9.64.3.1 dbValueChanged	178
9.64.4 Property Documentation	178
9.64.4.1 addUnits	178
9.64.4.2 arrayAction	178
9.64.4.3 format	178
9.64.4.4 int	178
9.64.4.5 leadingZero	179
9.64.4.6 localEnumeration	179
9.64.4.7 notation	179
9.64.4.8 precision	180
9.64.4.9 trailingZeros	180
9.64.4.10 useDbPrecision	180
9.65 QELineEditManager Class Reference	180
9.66 QELink Class Reference	180
9.67 QELog Class Reference	182

9.67.1 Member Enumeration Documentation	185
9.67.1.1 DisplayAlarmStateOptions	185
9.67.1.2 UserLevels	185
9.67.2 Property Documentation	185
9.67.2.1 allowDrop	185
9.67.2.2 displayAlarmState	185
9.67.2.3 displayAlarmStateOption	185
9.67.2.4 int	186
9.67.2.5 userLevelEnabled	186
9.67.2.6 userLevelEngineerStyle	186
9.67.2.7 userLevelScientistStyle	186
9.67.2.8 userLevelUserStyle	186
9.67.2.9 userLevelVisibility	187
9.67.2.10 variableAsToolTip	187
9.67.2.11 visible	187
9.68 QELogin Class Reference	187
9.69 QELoginDialog Class Reference	188
9.70 QENumericEdit Class Reference	188
9.70.1 Detailed Description	190
9.70.2 Constructor & Destructor Documentation	190
9.70.2.1 QENumericEdit	190
9.70.2.2 QENumericEdit	190
9.70.3 Member Function Documentation	191
9.70.3.1 dbValueChanged	191
9.70.4 Property Documentation	191
9.70.4.1 addUnits	191
9.70.4.2 autoScale	191
9.70.4.3 leadingZeros	191
9.70.4.4 maximum	191
9.70.4.5 minimum	191
9.70.4.6 precision	191
9.71 QENumericEditManager Class Reference	192
9.72 QEPeriodic Class Reference	192
9.72.1 Member Enumeration Documentation	195

9.72.1.1	DisplayAlarmStateOptions	195
9.72.1.2	UserLevels	196
9.72.2	Member Function Documentation	196
9.72.2.1	dbElementChanged	196
9.72.2.2	dbValueChanged	196
9.72.3	Member Data Documentation	196
9.72.3.1	allowDrop	196
9.72.4	Property Documentation	196
9.72.4.1	displayAlarmState	196
9.72.4.2	displayAlarmStateOption	197
9.72.4.3	int	197
9.72.4.4	readbackLabelVariable1	197
9.72.4.5	readbackLabelVariable2	197
9.72.4.6	subscribe	197
9.72.4.7	userLevelEnabled	197
9.72.4.8	userLevelEngineerStyle	198
9.72.4.9	userLevelScientistStyle	198
9.72.4.10	userLevelUserStyle	198
9.72.4.11	userLevelVisibility	198
9.72.4.12	variableAsToolTip	198
9.72.4.13	variableSubstitutions	198
9.72.4.14	visible	199
9.72.4.15	writeButtonVariable1	199
9.72.4.16	writeButtonVariable2	199
9.73	QEPeriodicComponentData Class Reference	199
9.74	QEPeriodicTaskMenu Class Reference	199
9.75	QEPeriodicTaskMenuFactory Class Reference	200
9.76	QEPlot Class Reference	200
9.76.1	Member Enumeration Documentation	204
9.76.1.1	DisplayAlarmStateOptions	204
9.76.1.2	UserLevels	204
9.76.2	Member Function Documentation	204
9.76.2.1	dbValueChanged	204
9.76.2.2	dbValueChanged	204

9.76.3 Member Data Documentation	205
9.76.3.1 allowDrop	205
9.76.4 Property Documentation	205
9.76.4.1 displayAlarmState	205
9.76.4.2 displayAlarmStateOption	205
9.76.4.3 int	205
9.76.4.4 userLevelEnabled	205
9.76.4.5 userLevelEngineerStyle	206
9.76.4.6 userLevelScientistStyle	206
9.76.4.7 userLevelUserStyle	206
9.76.4.8 userLevelVisibility	206
9.76.4.9 variable1	206
9.76.4.10 variable2	206
9.76.4.11 variable3	207
9.76.4.12 variable4	207
9.76.4.13 variableAsToolTip	207
9.76.4.14 variableSubstitutions	207
9.76.4.15 visible	207
9.77 QEPushButton Class Reference	207
9.77.1 Member Enumeration Documentation	211
9.77.1.1 ArrayActions	211
9.77.1.2 CreationOptionNames	211
9.77.1.3 DisplayAlarmStateOptions	212
9.77.1.4 Formats	212
9.77.1.5 Notations	212
9.77.1.6 ProgramStartupOptionNames	213
9.77.1.7 UpdateOptions	213
9.77.1.8 UserLevels	213
9.77.2 Constructor & Destructor Documentation	214
9.77.2.1 QEPushButton	214
9.77.2.2 QEPushButton	214
9.77.3 Member Function Documentation	214
9.77.3.1 clicked	214
9.77.3.2 dbValueChanged	214

9.77.3.3	pressed	214
9.77.3.4	released	214
9.77.3.5	requestAction	214
9.77.4	Property Documentation	215
9.77.4.1	addUnits	215
9.77.4.2	alignment	215
9.77.4.3	allowDrop	215
9.77.4.4	altReadbackVariable	215
9.77.4.5	arguments	215
9.77.4.6	arrayAction	215
9.77.4.7	clickCheckedText	216
9.77.4.8	clickText	216
9.77.4.9	confirmAction	216
9.77.4.10	confirmText	216
9.77.4.11	creationOption	216
9.77.4.12	customisationName	216
9.77.4.13	displayAlarmState	217
9.77.4.14	displayAlarmStateOption	217
9.77.4.15	format	217
9.77.4.16	guiFile	217
9.77.4.17	int	217
9.77.4.18	labelText	218
9.77.4.19	leadingZero	218
9.77.4.20	localEnumeration	218
9.77.4.21	notation	219
9.77.4.22	password	219
9.77.4.23	pixmap0	219
9.77.4.24	pixmap1	219
9.77.4.25	pixmap2	219
9.77.4.26	pixmap3	219
9.77.4.27	pixmap4	219
9.77.4.28	pixmap5	219
9.77.4.29	pixmap6	220
9.77.4.30	pixmap7	220

9.77.4.31 precision	220
9.77.4.32 pressText	220
9.77.4.33 prioritySubstitutions	220
9.77.4.34 program	220
9.77.4.35 programStartupOption	220
9.77.4.36 releaseText	221
9.77.4.37 subscribe	221
9.77.4.38 trailingZeros	221
9.77.4.39 updateOption	221
9.77.4.40 useDbPrecision	221
9.77.4.41 userLevelEnabled	221
9.77.4.42 userLevelEngineerStyle	221
9.77.4.43 userLevelScientistStyle	222
9.77.4.44 userLevelUserStyle	222
9.77.4.45 userLevelVisibility	222
9.77.4.46 variable	222
9.77.4.47 variableAsToolTip	222
9.77.4.48 variableSubstitutions	222
9.77.4.49 visible	223
9.77.4.50 writeOnClick	223
9.77.4.51 writeOnPress	223
9.77.4.52 writeOnRelease	223
9.78 QEPVNameLists Class Reference	223
9.79 QEPvProperties Class Reference	223
9.79.1 Property Documentation	225
9.79.1.1 variable	225
9.79.1.2 variableSubstitutions	225
9.80 QEPvPropertiesManager Class Reference	225
9.81 QERadioButton Class Reference	226
9.81.1 Member Enumeration Documentation	229
9.81.1.1 ArrayActions	229
9.81.1.2 CreationOptionNames	230
9.81.1.3 DisplayAlarmStateOptions	230
9.81.1.4 Formats	231

9.81.1.5	Notations	231
9.81.1.6	ProgramStartupOptionNames	231
9.81.1.7	UpdateOptions	231
9.81.1.8	UserLevels	232
9.81.2	Constructor & Destructor Documentation	232
9.81.2.1	QERadioButton	232
9.81.2.2	QERadioButton	232
9.81.3	Member Function Documentation	232
9.81.3.1	clicked	232
9.81.3.2	dbValueChanged	232
9.81.3.3	pressed	233
9.81.3.4	released	233
9.81.3.5	requestAction	233
9.81.4	Property Documentation	233
9.81.4.1	addUnits	233
9.81.4.2	alignment	233
9.81.4.3	allowDrop	233
9.81.4.4	arguments	233
9.81.4.5	arrayAction	234
9.81.4.6	clickCheckedText	234
9.81.4.7	clickText	234
9.81.4.8	confirmAction	234
9.81.4.9	confirmText	234
9.81.4.10	creationOption	235
9.81.4.11	customisationName	235
9.81.4.12	displayAlarmState	235
9.81.4.13	displayAlarmStateOption	235
9.81.4.14	format	235
9.81.4.15	guiFile	236
9.81.4.16	int	236
9.81.4.17	labelText	236
9.81.4.18	leadingZero	236
9.81.4.19	localEnumeration	236
9.81.4.20	notation	237

9.81.4.21 password	237
9.81.4.22 pixmap0	237
9.81.4.23 pixmap1	237
9.81.4.24 pixmap2	237
9.81.4.25 pixmap3	238
9.81.4.26 pixmap4	238
9.81.4.27 pixmap5	238
9.81.4.28 pixmap6	238
9.81.4.29 pixmap7	238
9.81.4.30 precision	238
9.81.4.31 pressText	238
9.81.4.32 prioritySubstitutions	238
9.81.4.33 program	239
9.81.4.34 programStartupOption	239
9.81.4.35 releaseText	239
9.81.4.36 subscribe	239
9.81.4.37 trailingZeros	239
9.81.4.38 updateOption	239
9.81.4.39 useDbPrecision	239
9.81.4.40 userLevelEnabled	239
9.81.4.41 userLevelEngineerStyle	240
9.81.4.42 userLevelScientistStyle	240
9.81.4.43 userLevelUserStyle	240
9.81.4.44 userLevelVisibility	240
9.81.4.45 variable	240
9.81.4.46 variableAsToolTip	241
9.81.4.47 variableSubstitutions	241
9.81.4.48 visible	241
9.81.4.49 writeOnClick	241
9.81.4.50 writeOnPress	241
9.81.4.51 writeOnRelease	241
9.82 QRRecipe Class Reference	241
9.83 QRRecordFieldName Class Reference	243
9.84 QRRecordSpec Class Reference	244

9.85 QERecordSpecList Class Reference	244
9.86 QEScript Class Reference	244
9.86.1 Detailed Description	249
9.86.2 Member Enumeration Documentation	249
9.86.2.1 DisplayAlarmStateOptions	249
9.86.2.2 UserLevels	249
9.86.3 Property Documentation	249
9.86.3.1 allowDrop	249
9.86.3.2 displayAlarmState	249
9.86.3.3 displayAlarmStateOption	250
9.86.3.4 int	250
9.86.3.5 userLevelEnabled	250
9.86.3.6 userLevelEngineerStyle	250
9.86.3.7 userLevelScientistStyle	250
9.86.3.8 userLevelUserStyle	251
9.86.3.9 userLevelVisibility	251
9.86.3.10 variableAsToolTip	251
9.86.3.11 visible	251
9.87 QEShape Class Reference	251
9.87.1 Detailed Description	256
9.87.2 Member Enumeration Documentation	256
9.87.2.1 animationOptions	256
9.87.2.2 DisplayAlarmStateOptions	256
9.87.2.3 shapeOptions	256
9.87.2.4 UserLevels	256
9.87.3 Constructor & Destructor Documentation	257
9.87.3.1 QEShape	257
9.87.3.2 QEShape	257
9.87.4 Member Function Documentation	257
9.87.4.1 dbValueChanged1	257
9.87.4.2 dbValueChanged2	257
9.87.4.3 dbValueChanged3	257
9.87.4.4 dbValueChanged4	257
9.87.4.5 dbValueChanged5	257

9.87.4.6 dbValueChanged6	258
9.87.5 Property Documentation	258
9.87.5.1 allowDrop	258
9.87.5.2 animation1	258
9.87.5.3 animation2	258
9.87.5.4 animation3	258
9.87.5.5 animation4	258
9.87.5.6 animation5	258
9.87.5.7 animation6	258
9.87.5.8 color1	259
9.87.5.9 color10	259
9.87.5.10 color2	259
9.87.5.11 color3	259
9.87.5.12 color4	259
9.87.5.13 color5	259
9.87.5.14 color6	259
9.87.5.15 color7	259
9.87.5.16 color8	259
9.87.5.17 color9	260
9.87.5.18 displayAlarmState	260
9.87.5.19 displayAlarmStateOption	260
9.87.5.20 int	260
9.87.5.21 offset1	260
9.87.5.22 offset2	260
9.87.5.23 offset3	261
9.87.5.24 offset4	261
9.87.5.25 offset5	261
9.87.5.26 offset6	261
9.87.5.27 point1	261
9.87.5.28 point10	261
9.87.5.29 point2	261
9.87.5.30 point3	261
9.87.5.31 point4	261
9.87.5.32 point5	262

9.87.5.33 point6	262
9.87.5.34 point7	262
9.87.5.35 point8	262
9.87.5.36 point9	262
9.87.5.37 scale2	262
9.87.5.38 scale3	262
9.87.5.39 scale4	262
9.87.5.40 scale5	262
9.87.5.41 scale6	263
9.87.5.42 userLevelEnabled	263
9.87.5.43 userLevelEngineerStyle	263
9.87.5.44 userLevelScientistStyle	263
9.87.5.45 userLevelUserStyle	263
9.87.5.46 userLevelVisibility	263
9.87.5.47 variable1	264
9.87.5.48 variable2	264
9.87.5.49 variable3	264
9.87.5.50 variable4	264
9.87.5.51 variable5	264
9.87.5.52 variable6	264
9.87.5.53 variableAsToolTip	265
9.87.5.54 variableSubstitutions	265
9.87.5.55 visible	265
9.88 QESlider Class Reference	265
9.88.1 Member Enumeration Documentation	267
9.88.1.1 DisplayAlarmStateOptions	267
9.88.1.2 UserLevels	268
9.88.2 Member Function Documentation	268
9.88.2.1 dbValueChanged	268
9.88.3 Member Data Documentation	268
9.88.3.1 writeOnChange	268
9.88.4 Property Documentation	268
9.88.4.1 allowDrop	268
9.88.4.2 displayAlarmState	268

9.88.4.3	displayAlarmStateOption	269
9.88.4.4	int	269
9.88.4.5	subscribe	269
9.88.4.6	userLevelEnabled	269
9.88.4.7	userLevelEngineerStyle	269
9.88.4.8	userLevelScientistStyle	269
9.88.4.9	userLevelUserStyle	270
9.88.4.10	userLevelVisibility	270
9.88.4.11	variable	270
9.88.4.12	variableAsToolTip	270
9.88.4.13	variableSubstitutions	270
9.88.4.14	visible	270
9.89	QESpinBox Class Reference	271
9.89.1	Member Enumeration Documentation	273
9.89.1.1	DisplayAlarmStateOptions	273
9.89.1.2	UserLevels	273
9.89.2	Member Function Documentation	274
9.89.2.1	dbValueChanged	274
9.89.3	Property Documentation	274
9.89.3.1	allowDrop	274
9.89.3.2	displayAlarmState	274
9.89.3.3	displayAlarmStateOption	274
9.89.3.4	int	274
9.89.3.5	subscribe	275
9.89.3.6	userLevelEnabled	275
9.89.3.7	userLevelEngineerStyle	275
9.89.3.8	userLevelScientistStyle	275
9.89.3.9	userLevelUserStyle	275
9.89.3.10	userLevelVisibility	275
9.89.3.11	variable	276
9.89.3.12	variableAsToolTip	276
9.89.3.13	variableSubstitutions	276
9.89.3.14	visible	276
9.90	QEStripChart Class Reference	276

9.90.1 Property Documentation	278
9.90.1.1 variableSubstitutions	278
9.91 QEStripChartAdjustPVDialog Class Reference	279
9.92 QEStripChartContextMenu Class Reference	279
9.92.1 Constructor & Destructor Documentation	279
9.92.1.1 QEStripChartContextMenu	279
9.93 QEStripChartDurationDialog Class Reference	280
9.94 QEStripChartItem Class Reference	280
9.95 QEStripChartNames Class Reference	281
9.96 QEStripChartPushButtonSpecifications Struct Reference	282
9.97 QEStripChartRangeDialog Class Reference	282
9.98 QEStripChartState Class Reference	283
9.99 QEStripChartStateList Class Reference	283
9.100QEStripChartStatistics Class Reference	284
9.101QEStripChartTimeDialog Class Reference	284
9.102QEStripChartToolBar Class Reference	284
9.102.1 Detailed Description	285
9.103QESubstitutedLabel Class Reference	285
9.103.1 Member Data Documentation	286
9.103.1.1 labelText	286
9.103.2 Property Documentation	286
9.103.2.1 textSubstitutions	286
9.104recording Class Reference	286
9.105imageDisplayProperties::rgbPixel Struct Reference	287
9.106screenSelectDialog Class Reference	287
9.107selectMenu Class Reference	288
9.108trace Class Reference	288
9.109userInfoStruct Class Reference	288
9.110QEPeriodic::userInfoStructArray Struct Reference	289
9.111ValueScaling Class Reference	289
9.112VideoWidget Class Reference	289
9.113zoomMenu Class Reference	291

Chapter 1

QE framework - EPICS aware Qt Widgets and data access classes

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

1.1 Documentation

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

1.2 License

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

1.3 Platforms

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

1.4 Screenshots

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

Screenshots are only available in the HTML docs.

1.5 Downloads

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

For getting a development snapshot from the SVN repository:

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

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

1.6 Installation

Read [QE_GettingStarted.pdf](#) in the documentation for setting up an enviroment for building or using the epicsqt framework.

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

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

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

1.7 Support

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

1.8 Related Projects

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

1.9 Credits:

Authors:

Andrew Rhyder, Anthony Owen, Glenn Jackson

Project admin:

Andrew Rhyder <andrew.rhyder@synchrotron.org.au>

Chapter 2

GNU General Public License

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

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

See the GNU General Public License for more details.

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

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

Chapter 3

ASgui screen shots

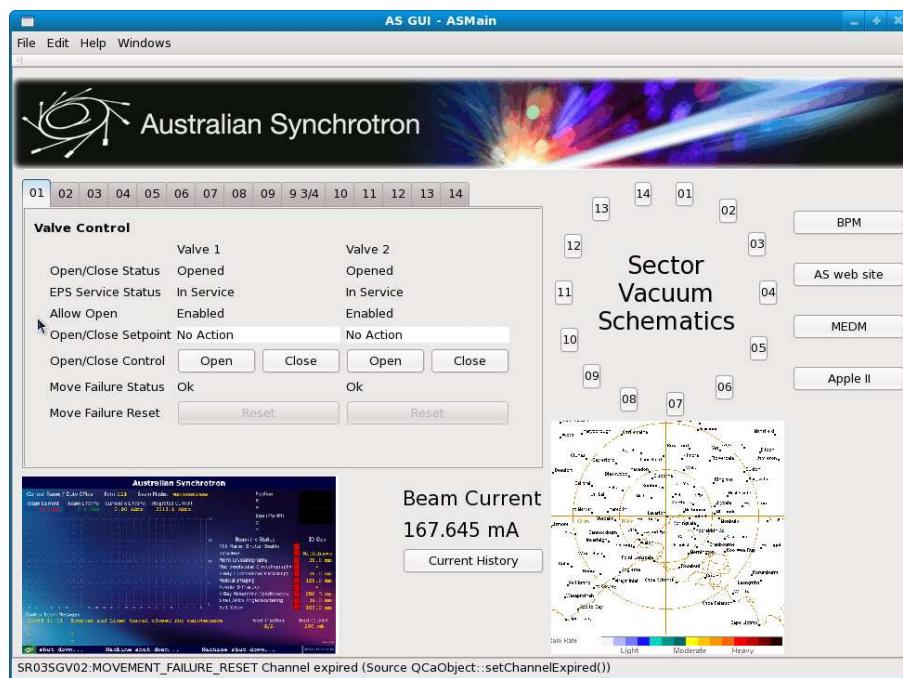


Figure 3.1: Australian Synchrotron mock up

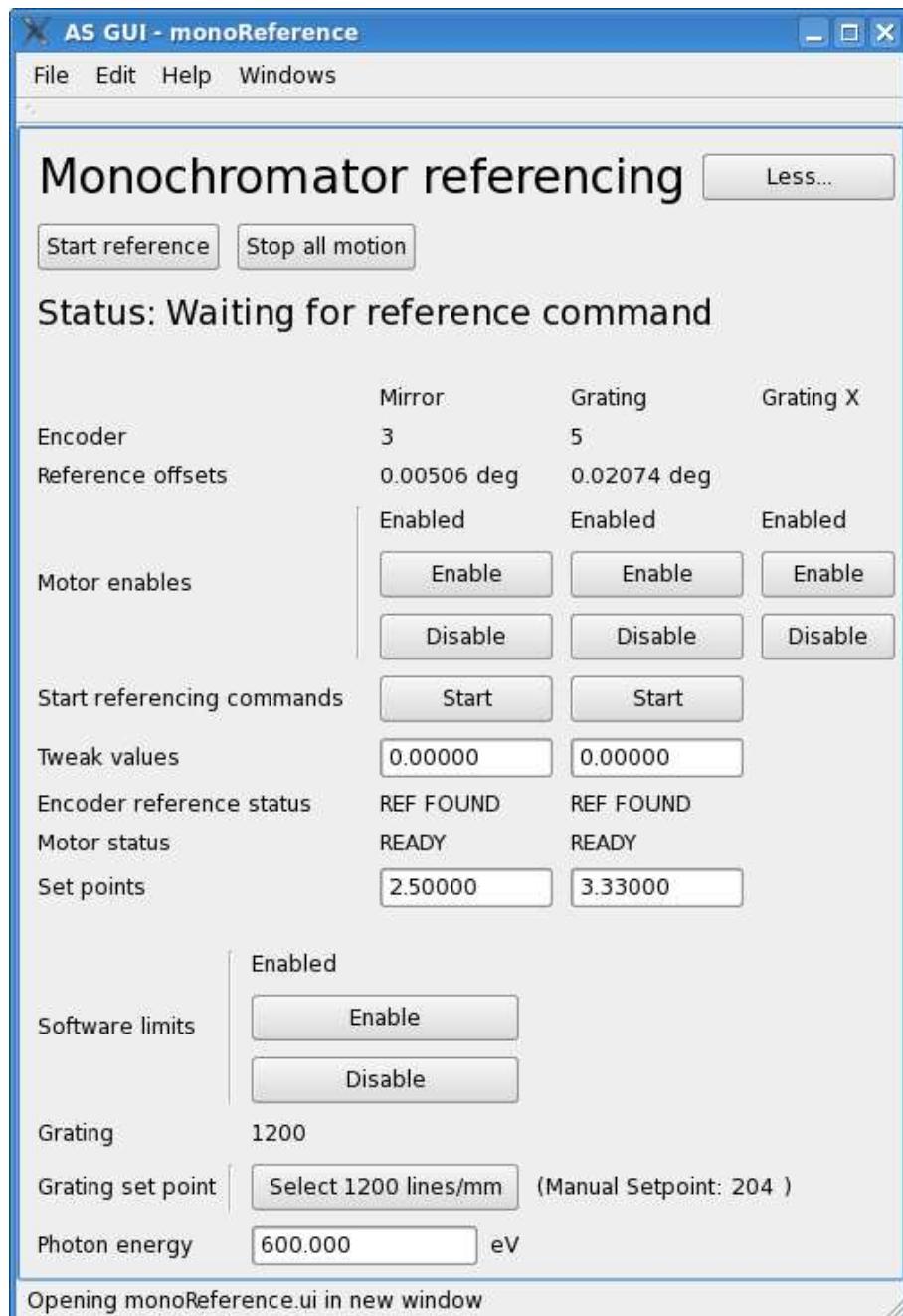


Figure 3.2: Monochromator referencing

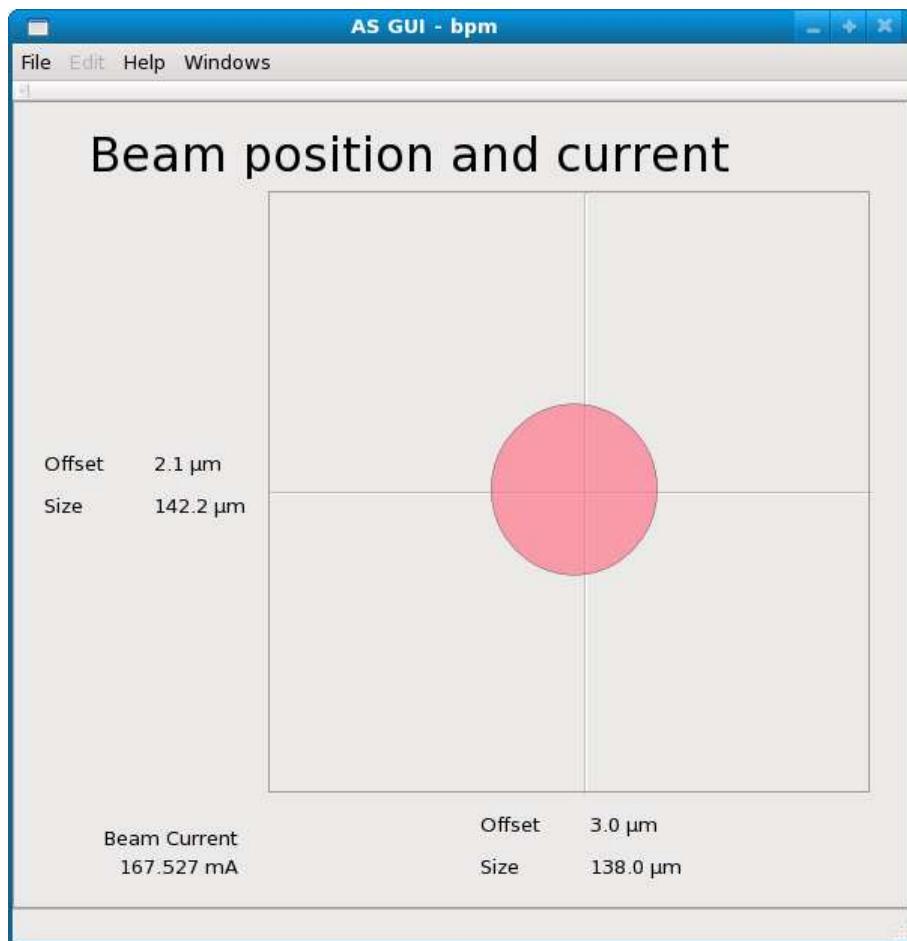


Figure 3.3: Beam position monitor

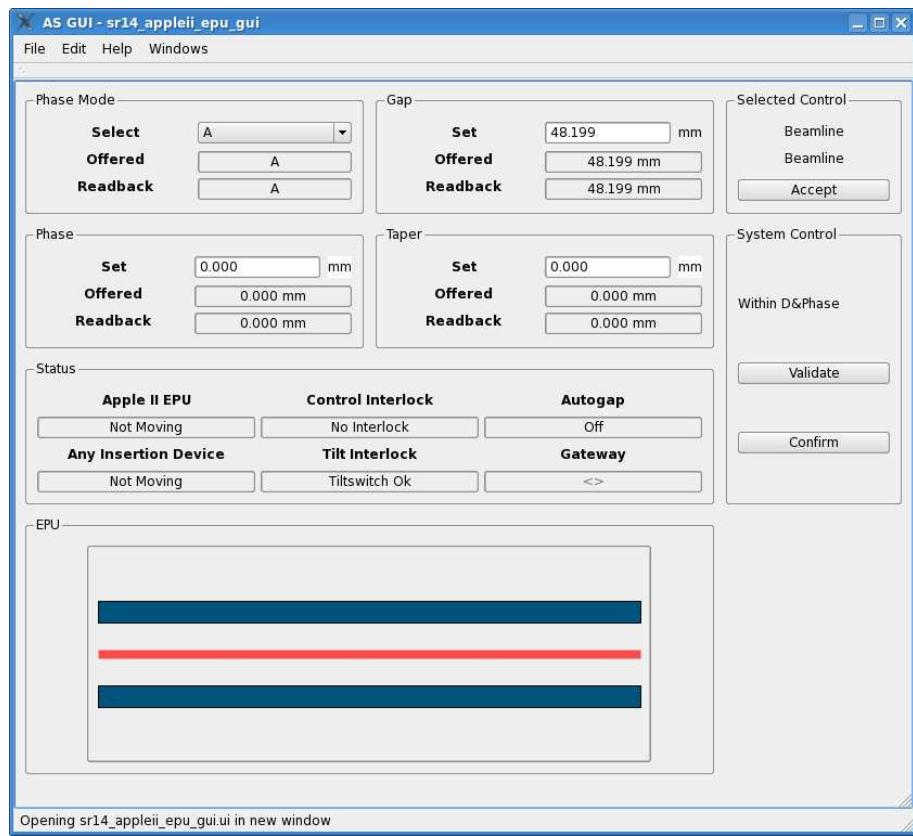


Figure 3.4: Insertion device



Figure 3.5: Injection efficiency monitor

Chapter 4

other applications using epicsqt widgets

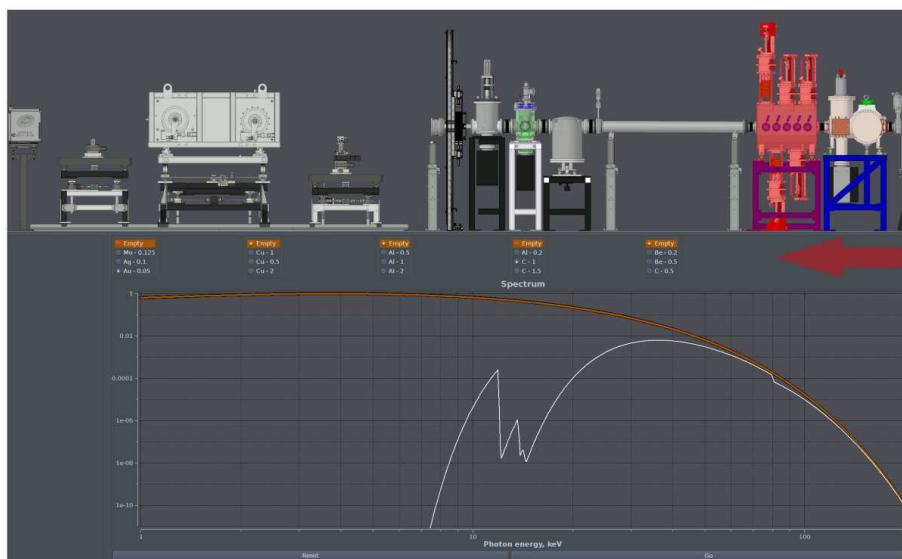


Figure 4.1: Medical Imaging beamline

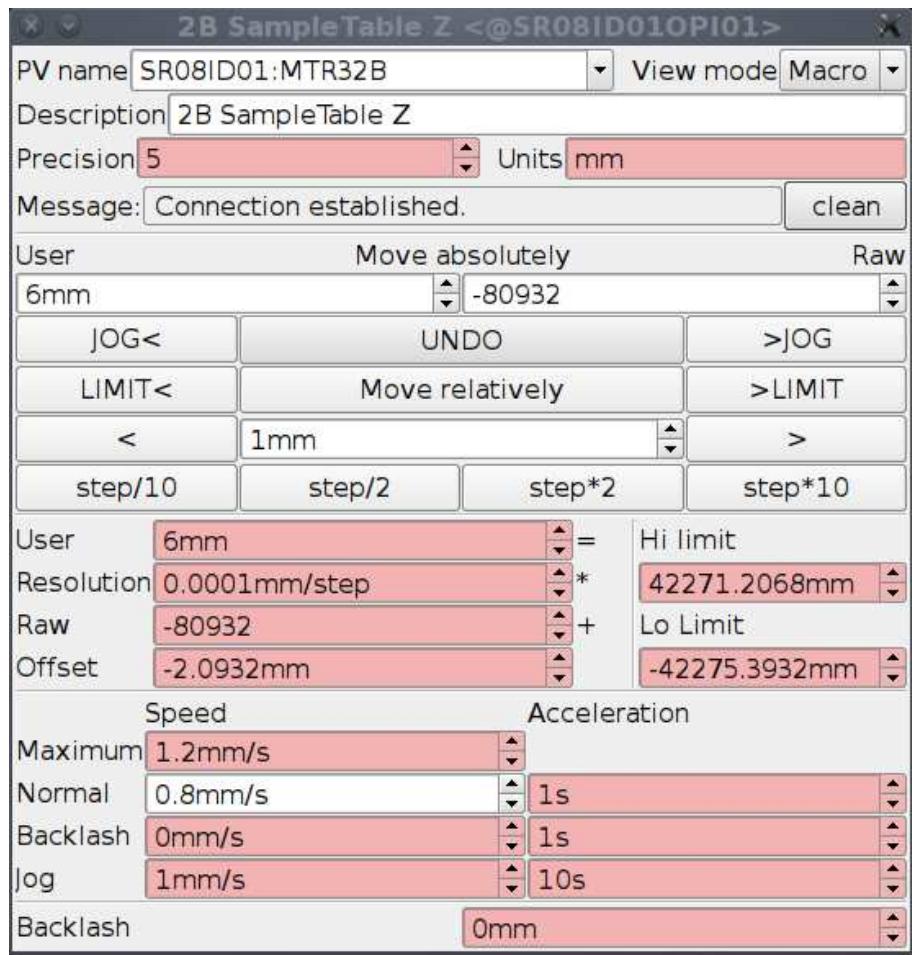


Figure 4.2: Motor controller

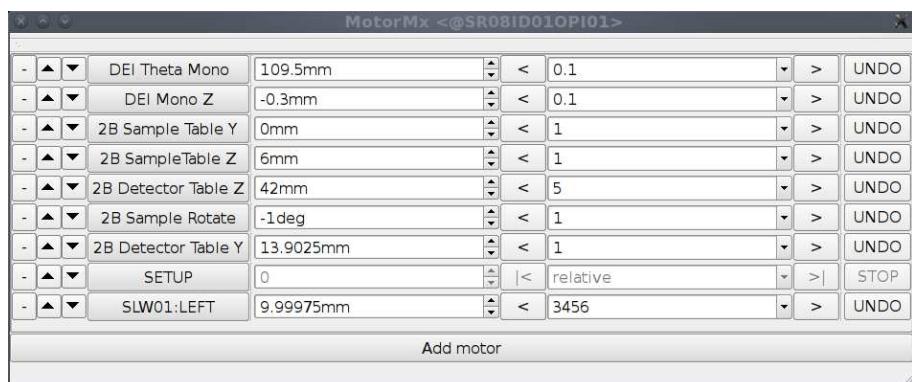


Figure 4.3: Motor controller

Chapter 5

Qt Designer

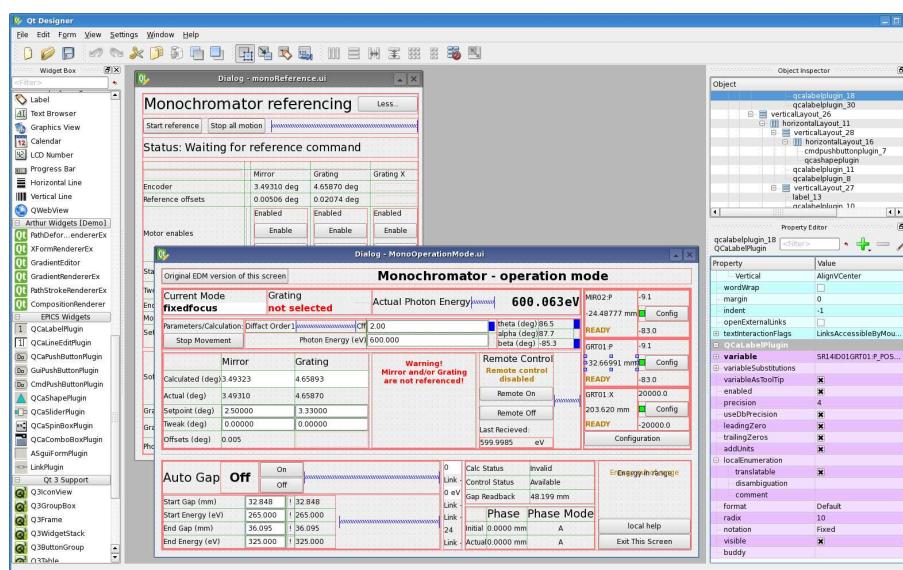


Figure 5.1: Editing multiple GUIs

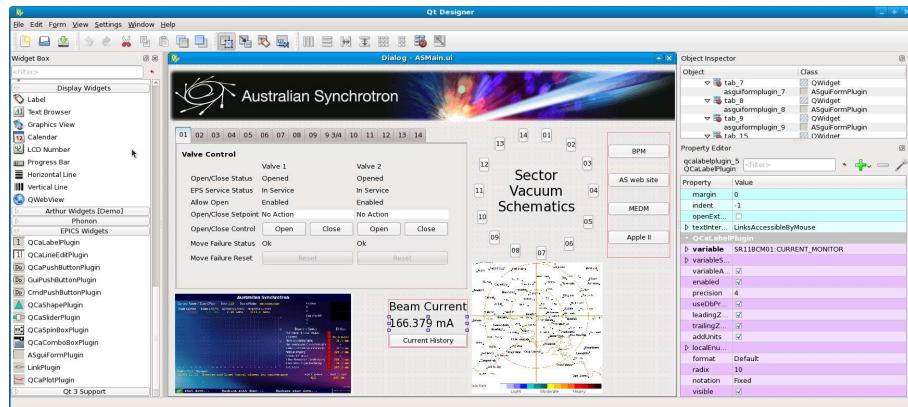


Figure 5.2: Editing a GUI

Chapter 6

Qt Creator

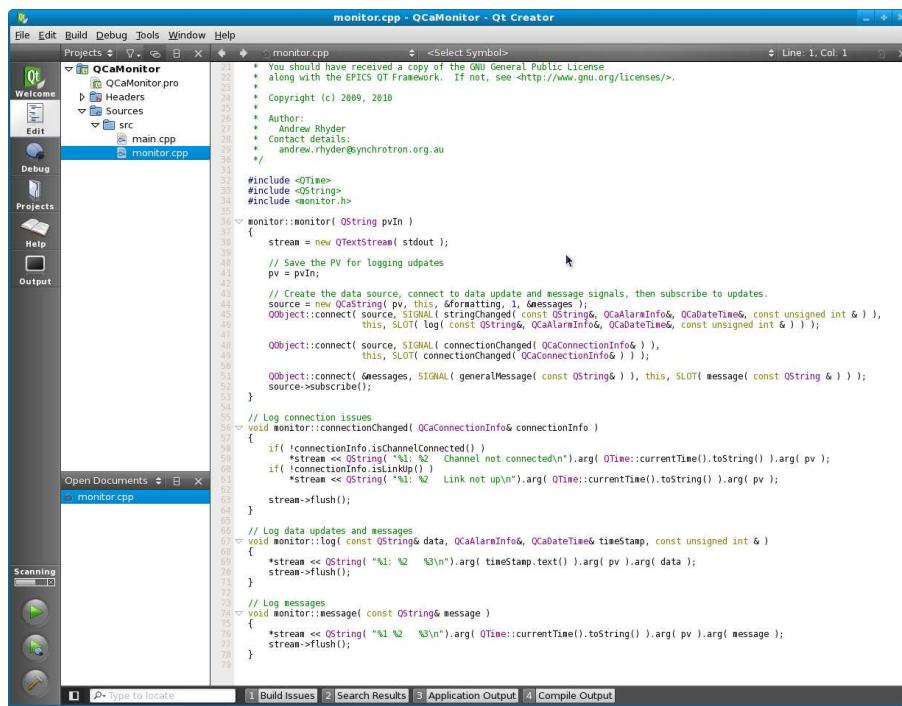


Figure 6.1: Application using epicsqt data source classes

Chapter 7

Class Index

7.1 Class Hierarchy

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

_CopyPaste	27
_Field	27
_Item	28
_QDialogItem	29
_QPushButtonGroup	29
_QTableWidgetFileBrowser	29
_QTableWidgetLog	30
_QTableWidgetScript	30
areaInfo	30
QEAnalogIndicator::Band	31
QEAnalogIndicator::BandList	31
QEPeriodic::elementInfoStruct	31
FFBuffer	32
FFTThread	32
flipRotateMenu	33
fullScreenWindow	33
histogram	34
histogramScroll	34
historicImage	34
imageContextMenu	35
imageDisplayProperties	36
imageInfo	37
QEImage	128
imageMarkup	38
VideoWidget	289
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
QEImage	128
mpegSourceObject	50
QEStripChartToolBar::OwnWidgets	51
PeriodicDialog	51
PeriodicElementSetupForm	51
PeriodicSetupDialog	52
playbackTimer	52
pointInfo	52
profilePlot	53
QBitStatus	53
QEBitStatus	70
QEAnalogIndicator	55
QEAnalogProgressBar	61
QECheckBoxManager	91
QEComboBox	92
QEConfiguredLayout	98
QEConfiguredLayoutManager	103
QEFileBrowser	103
QEForm	109
QEFrame	111
QE_pvProperties	223
QEStripChart	276
QEGenericButton	114
QECheckBox	76
QEPushButton	207
QERadioButton	226
QEGenericEdit	116
QELineEdit	175
QENumericEdit	188
QEGroupBox	124
QEImageMarkupThickness	163
QEImageOptionsDialog	163
QELabel	164
QELineEditManager	180
QELink	180
QELog	182
QELogin	187
QELoginDialog	188
QE_NUMERIC_EDIT_MANAGER	192

QEPeriodic	192
QEPeriodicComponentData	199
QEPeriodicTaskMenu	199
QEPeriodicTaskMenuFactory	200
QEPlot	200
QEPVNameLists	223
QEPvPropertiesManager	225
QERecipe	241
QERecordFieldName	243
QERecordSpec	244
QERecordSpecList	244
QEScript	244
QEShape	251
QESlider	265
QESpinBox	271
QEStripChartAdjustPVDialog	279
QEStripChartContextMenu	279
QEStripChartDurationDialog	280
QEStripChartItem	280
QEStripChartNames	281
QEStripChartPushButtonSpecifications	282
QEStripChartRangeDialog	282
QEStripChartState	283
QEStripChartStateList	283
QEStripChartStatistics	284
QEStripChartTimeDialog	284
QEStripChartToolBar	284
QESubstitutedLabel	285
recording	286
imageDisplayProperties::rgbPixel	287
screenSelectDialog	287
selectMenu	288
trace	288
userInfoStruct	288
QEPeriodic::userInfoStructArray	289
ValueScaling	289
zoomMenu	291

Chapter 8

Class Index

8.1 Class List

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

_CopyPaste	27
_Field	27
_Item	28
_QDialogItem	29
_QPushButtonGroup	29
_QTableWidgetFileBrowser	29
_QTableWidgetLog	30
_QTableWidgetScript	30
areaInfo	30
QEAnalogIndicator::Band	31
QEAnalogIndicator::BandList	31
QEPeriodic::elementInfoStruct	31
FFBuffer	32
FFTthread	32
flipRotateMenu	33
fullScreenWindow	33
histogram	34
histogramScroll	34
historicImage	34
imageContextMenu	35
imageDisplayProperties	36
imageInfo	37
imageMarkup	38
imageUpdateIndicator	40
loginWidget	40
markupCrosshair1	40
markupCrosshair2	41
markupDisplayMenu	42
markupEllipse	42

markupHLine	43
markupItem	44
markupLine	46
markupRegion	47
markupText	48
markupVLine	48
mpegSource	49
mpegSourceObject	50
QEStripChartToolBar::OwnWidgets	51
PeriodicDialog	51
PeriodicElementSetupForm	51
PeriodicSetupDialog	52
playbackTimer	52
pointInfo	52
profilePlot	53
QBitStatus	53
QEAnalogIndicator	55
QEAnalogProgressBar	61
QEBitStatus	70
QECheckBox	76
QECheckBoxManager	91
QEComboBox	92
QEConfiguredLayout	98
QEConfiguredLayoutManager	103
QEFileBrowser	103
QEForm	109
QEFrame	111
QEGenericButton	114
QEGenericEdit	116
QEGroupBox	124
QEImage	128
QEImageMarkupThickness	163
QEImageOptionsDialog	163
QELabel	164
QELineEdit	175
QELineEditManager	180
QELink	180
QELog	182
QELogin	187
QELoginDialog	188
QENumericEdit (The QENumericEdit class This class is similar to QELineEdit (both of which are derived from QLineEdit). However this class is tailored specifically for editing numerical values)	188
QENumericEditManager	192
QEPeriodic	192
QEPeriodicComponentData	199
QEPeriodicTaskMenu	199
QEPeriodicTaskMenuFactory	200
QEPlot	200
QEPushButton	207

QEPVNameLists	223
QEPvProperties	223
QEPvPropertiesManager	225
QERadioButton	226
QERecipe	241
QERecordFieldName	243
QERecordSpec	244
QERecordSpecList	244
QEScript	244
QEShape	251
QESlider	265
QESpinBox	271
QEStripChart	276
QEStripChartAdjustPVDDialog	279
QEStripChartContextMenu	279
QEStripChartDurationDialog	280
QEStripChartItem	280
QEStripChartNames	281
QEStripChartPushButtonSpecifications	282
QEStripChartRangeDialog	282
QEStripChartState	283
QEStripChartStateList	283
QEStripChartStatistics	284
QEStripChartTimeDialog	284
QEStripChartToolBar (This class holds all the StripChart tool bar widgets)	284
QESubstitutedLabel	285
recording	286
imageDisplayProperties::rgbPixel	287
screenSelectDialog	287
selectMenu	288
trace	288
userInfoStruct	288
QEPeriodic::userInfoStructArray	289
ValueScaling	289
VideoWidget	289
zoomMenu	291

Chapter 9

Class Documentation

9.1 _CopyPaste Class Reference

Public Member Functions

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

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

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

9.2 _Field Class Reference

Public Member Functions

- QEWidget * **getWidget** ()

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

Public Attributes

- QEWidget * **qeWidget**

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

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

9.3 [_Item Class Reference](#)

Public Member Functions

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

Public Attributes

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

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

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

9.4 _QDialogItem Class Reference

Public Member Functions

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

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

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

9.5 _QPushButtonGroup Class Reference

Public Slots

- void **buttonGroupClicked** ()

Public Member Functions

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

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

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

9.6 _QTableWidgetFileBrowser Class Reference

Public Member Functions

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

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

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

9.7 _QTableWidgetLog Class Reference

Public Member Functions

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

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

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

9.8 _QTableWidgetScript Class Reference

Public Member Functions

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

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

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

9.9 arealInfo Class Reference

Public Member Functions

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

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

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

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

9.10 QEAnalogIndicator::Band Struct Reference

Public Attributes

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

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

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

9.11 QEAnalogIndicator::BandList Class Reference

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

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

9.12 QEPeriodic::elementInfoStruct Struct Reference

Public Attributes

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

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

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

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

9.13 FFBuffer Class Reference

Public Member Functions

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

Public Attributes

- QMutex * **mutex**
- unsigned char * **mem**
- AVFrame * **pFrame**
- PixelFormat **pix_fmt**
- int **width**
- int **height**
- int **refs**

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

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

9.14 FFThread Class Reference

Public Slots

- void **stopGracefully** ()

Signals

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

Public Member Functions

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

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

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

9.15 flipRotateMenu Class Reference

Public Member Functions

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

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

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

9.16 fullScreenWindow Class Reference

Signals

- void **fullScreenResize** ()

Public Member Functions

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

Protected Member Functions

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

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

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

9.17 histogram Class Reference

Public Member Functions

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

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

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

9.18 histogramScroll Class Reference

Public Member Functions

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

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

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

9.19 historicImage Class Reference

Public Member Functions

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

Public Attributes

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

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

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

9.20 imageContextMenu Class Reference

Public Types

- enum **imageContextMenuOptions** {
 ICM_NONE = contextMenu::CM_SPECIFIC_WIDGETS_START_HERE, **ICM_SAVE**,
 ICM_PAUSE, **ICM_ENABLE_TIME**,
 ICM_ENABLE_CURSOR_PIXEL, **ICM_ABOUT_IMAGE**, **ICM_ENABLE_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 **getContextMenuItem** (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/QElImage/imageContextMenu.h
- /tmp/epicsqt/trunk/framework/widgets/QElImage/imageContextMenu.cpp

9.21 imageDisplayProperties Class Reference

Classes

- struct [rgbPixel](#)

Signals

- void **brightnessContrastAutolmage** ()
- void **imageDisplayPropertiesChange** ()

Public Member Functions

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

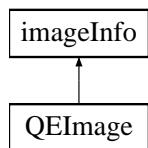
- `rgbPixel * pixelLookup`
- `QLabel * histXLabel`

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)`

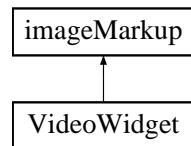
- bool **getBriefInfoArea** ()
- void **freshImage** (QDateTime &time)

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

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

9.23 imageMarkup Class Reference

Inheritance diagram for imageMarkup:



Public Types

- enum **markupIds** {

MARKUP_ID_REGION1, MARKUP_ID_REGION2, MARKUP_ID_REGION3, MARKUP_ID_REGION4,

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

9.25 loginWidget Class Reference

Public Member Functions

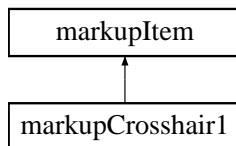
- **loginWidget** (QELogin *ownerIn)
- userLevelTypes::userLevels **getUserType** ()
- QString **getPassword** ()
- void **clearPassword** ()

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

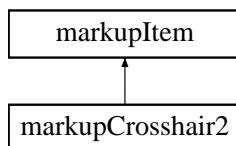
- **markupCrosshair1** (*imageMarkup* *ownerIn, const bool interactiveIn, const bool reportOnMoveIn, const QString legendIn)
- void **startDrawing** (const QPoint pos)
- void **setArea** ()
- void **drawMarkup** (QPainter &p)
- void **moveTo** (const QPoint pos)
- bool **isOver** (const QPoint point,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

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

- /tmp/epicsqt/trunk/framework/widgets/QEImage/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** ()
- QCursors **cursorForHandle** (const markupItem::markupHandles handle)
- QPoint **getPoint1** ()
- QPoint **getPoint2** ()
- QCursors **defaultCursor** ()
- void **nonInteractiveUpdate** (QPoint p1, QPoint p2)

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

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

9.28 markupDisplayMenu Class Reference

Public Member Functions

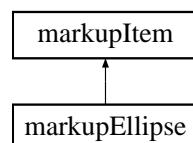
- **markupDisplayMenu** (QWidget *parent=0)
- void **setDisplayed** (imageContextMenu::imageContextMenuOptions option, bool state)
- void **setItemText** (imageContextMenu::imageContextMenuOptions option, QString title)
- bool **isDisplayed** (imageContextMenu::imageContextMenuOptions option)
- void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

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

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

9.29 markupEllipse Class Reference

Inheritance diagram for markupEllipse:



Public Member Functions

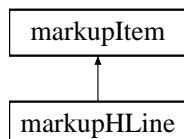
- **markupEllipse** ([imageMarkup](#) *ownerIn, const bool interactiveIn, const bool reportOnMoveIn, const QString legendIn)
- void **startDrawing** (const QPoint pos)
- void **setArea** ()
- void **drawMarkup** (QPainter &p)
- void **moveTo** (const QPoint pos)
- bool **isOver** (const QPoint point,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

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

- /tmp/epicsqt/trunk/framework/widgets/QEImage/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, QCursors *cursor)
- QPoint **origin** ()
- QCursors **cursorForHandle** (const markupItem::markupHandles handle)
- QPoint **getPoint1** ()
- QPoint **getPoint2** ()
- QCursors **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

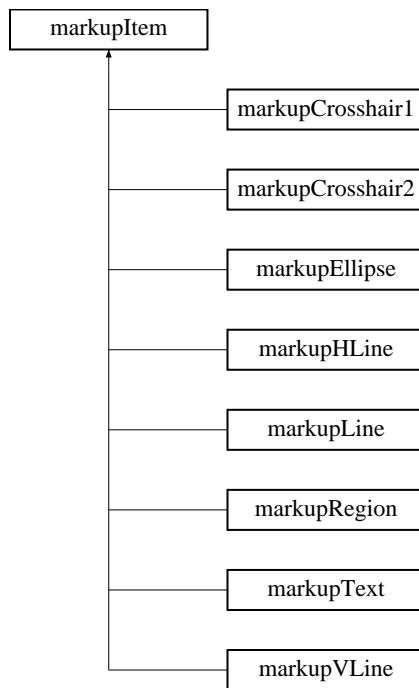
Implements [markupItem](#).

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

- /tmp/epicsqt/trunk/framework/widgets/QElImage/markupHLine.h
- /tmp/epicsqt/trunk/framework/widgets/QElImage/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 QCursors **cursorForHandle** (const markupItem::markupHandles handle)=0

- virtual QPoint **getPoint1** ()=0
- virtual QPoint **getPoint2** ()=0
- virtual QCursors **defaultCursor** ()=0
- virtual void **nonInteractiveUpdate** (QPoint, QPoint)
- void **setThickness** (const unsigned int thicknessIn)
- unsigned int **getThickness** ()
- void **setLegend** (const QString legendIn)
- const QString **getLegend** ()
- void **setColor** (QColor colorIn)
- QColor **getColor** ()

Public Attributes

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

Protected Types

- enum **isOverOptions** { OVER_LINE, OVER_BORDER, OVER_AREA }
- enum **legendJustification** { ABOVE_RIGHT, BELOW_LEFT, BELOW_RIGHT }

Protected Member Functions

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

Protected Attributes

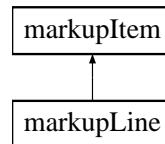
- markupHandles **activeHandle**
- *imageMarkup* * **owner**
- unsigned int **thickness**
- unsigned int **maxThickness**

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

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

9.32 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** ()

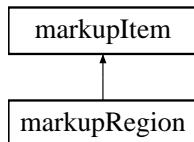
- void **drawMarkup** (QPainter &p)
- void **moveTo** (const QPoint pos)
- bool **isOver** (const QPoint point, QCursor *cursor)
- QPoint **origin** ()
- QCursors **cursorForHandle** (const markupItem::markupHandles handle)
- QPoint **getPoint1** ()
- QPoint **getPoint2** ()
- QCursors **defaultCursor**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

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

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

9.33 markupRegion Class Reference

Inheritance diagram for markupRegion:



Public Member Functions

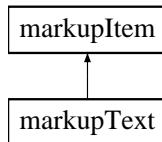
- **markupRegion** (imageMarkup *ownerIn, const bool interactiveIn, const bool reportOnMoveIn, const QString legendIn)
- void **startDrawing** (const QPoint pos)
- void **setArea** ()
- void **drawMarkup** (QPainter &p)
- void **moveTo** (const QPoint pos)
- bool **isOver** (const QPoint point, QCursors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

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

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

9.34 markupText Class Reference

Inheritance diagram for markupText:



Public Member Functions

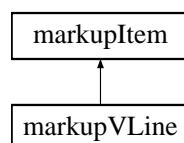
- **markupText** (*imageMarkup* *ownerIn, const bool interactiveIn, const bool reportOnMoveIn, const QString legendIn)
- void **setText** (QString textIn)
- void **startDrawing** (const QPoint pos)
- void **setArea** ()
- void **drawMarkup** (QPainter &p)
- void **moveTo** (const QPoint pos)
- bool **isOver**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

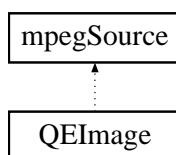
Implements [markupItem](#).

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

- QPushButtons * **pushButtons** [NUMBER_OF_BUTTONS]
- QLabel * **yScaleStatus**
- QLabel * **timeStatus**
- QLabel * **durationStatus**

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

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

9.39 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 pln)
- void **clearX** ()
- void **clearY** ()
- bool **getStatus** ()
- QPoint **getPoint** ()

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

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

9.44 profilePlot Class Reference

Public Types

- enum **plotDirections** { **PROFILEPLOT_LR**, **PROFILEPLOT_RL**, **PROFILEPLOT_TB**, **PROFILEPLOT_BT** }

Public Member Functions

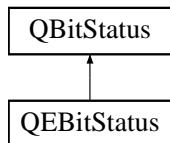
- **profilePlot** (plotDirections plotDirectionIn)
- void **setProfile** (QVector< QPointF > *profile, double minX, double maxX, double minY, double maxY, QString title, QPoint start, QPoint end, unsigned int thicknessIn)
- void **clearProfile** ()

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**

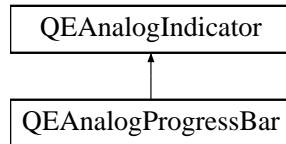
The documentation for this class was generated from the following files:

- /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.

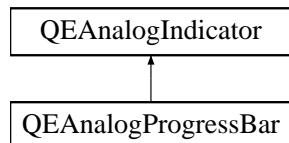
Reimplemented in [QEAnalogProgressBar](#).

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 **DisplayAlarmStateOptions** { **Never** = standardProperties::DISPLAY_ALARM_STATE_NEVER, **Always** = standardProperties::DISPLAY_ALARM_STATE_ALWAYS, **WhenInAlarm** = standardProperties::DISPLAY_ALARM_STATE_WHEN_IN_ALARM }
- enum **AlarmSeverityDisplayModes** { **foreground**, **background** }
- enum **Formats** {
 Default = QQStringFormatting::FORMAT_DEFAULT, **Floating** = QQStringFormatting::FORMAT_FLOATING, **Integer** = QQStringFormatting::FORMAT_INTEGER, **UnsignedInteger** = QQStringFormatting::FORMAT_UNSIGNEDINTEGER,
 Time = QQStringFormatting::FORMAT_TIME, **LocalEnumeration** = QQStringFormatting::FORMAT_LOCAL_ENUMERATE }
- enum **Notations** { **Fixed** = QQStringFormatting::NOTATION_FIXED, **Scientific** = QQStringFormatting::NOTATION_SCIENTIFIC, **Automatic** = QQStringFormatting::NOTATION_AUTOMATIC }
- enum **ArrayActions** { **Append** = QQStringFormatting::APPEND, **Ascii** = QQStringFormatting::ASCII, **Index** = QQStringFormatting::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

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

Property access function for `variableSubstitutions` property. This has special behaviour to work well within designer.
- QString `getVariableNameSubstitutionsProperty` ()
Property access function for `variableSubstitutions` property. This has special behaviour to work well within designer.
- UserLevels `getUserLevelVisibilityProperty` ()
Access function for `userLevelVisibility` property - refer to `userLevelVisibility` property for details.
- void `setUserLevelVisibilityProperty` (UserLevels level)
Access function for `userLevelVisibility` property - refer to `userLevelVisibility` property for details.
- UserLevels `getUserLevelEnabledProperty` ()
Access function for `userLevelEnabled` property - refer to `userLevelEnabled` property for details.
- void `setUserLevelEnabledProperty` (UserLevels level)
Access function for `userLevelEnabled` property - refer to `userLevelEnabled` property for details.
- DisplayAlarmStateOptions `getDisplayAlarmStateOptionProperty` ()
Access function for `displayAlarmStateOption` property - refer to `displayAlarmStateOption` property for details.
- void `setDisplayAlarmStateOptionProperty` (DisplayAlarmStateOptions option)
Access function for `displayAlarmStateOption` property - refer to `displayAlarmStateOption` property for details.
- void `setFormatProperty` (Formats format)
Access function for `format` property - refer to `format` property for details.
- Formats `getFormatProperty` ()
Access function for `format` property - refer to `format` property for details.
- void `setNotationProperty` (Notations notation)
Access function for `notation` property - refer to `notation` property for details.
- Notations `getNotationProperty` ()
Access function for `notation` property - refer to `notation` property for details.
- void `setArrayActionProperty` (ArrayActions arrayAction)
Access function for `arrayAction` property - refer to `arrayAction` property for details.
- ArrayActions `getArrayActionProperty` ()
Access function for `arrayAction` property - refer to `arrayAction` property for details.
- QEAnalogProgressBar (`QWidget *parent=0`)

- `QEAnalogProgressBar` (const `QString &variableName, QWidget *parent=0`)
- virtual ~`QEAnalogProgressBar` ()
Destruction.
- void `setUseDbDisplayLimits` (bool `useDbDisplayLimitsIn`)
Access function for `useDbDisplayLimits` property - refer to `useDbDisplayLimits` property for details.
- bool `getUseDbDisplayLimits` ()
Access function for `useDbDisplayLimits` property - refer to `useDbDisplayLimits` property for details.
- void `setAlarmSeverityDisplayStyle` (`AlarmSeverityDisplayModes` value)
Access function for `#AlarmSeverityDisplayModes` property - refer to `#AlarmSeverityDisplayModes` property for details.
- `AlarmSeverityDisplayModes` `getAlarmSeverityDisplayStyle` ()
Access function for `#AlarmSeverityDisplayModes` property - refer to `#AlarmSeverityDisplayModes` property for details.

Protected Member Functions

- `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`
- `DisplayAlarmStateOptions displayAlarmStateOption`
- `AlarmSeverityDisplayModes alarmSeverityDisplayStyle`
- `bool useDbDisplayLimits`
- `int value`
- `bool isActive`

Alternative to isEnabled. Default is true.
- `int precision`
- `bool useDbPrecision`
- `bool leadingZero`
- `bool trailingZeros`
- `bool addUnits`
- `QString localEnumeration`
- `Formats format`
- `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 `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.

9.47.1.2 enum QEAnalogProgressBar::DisplayAlarmStateOptions

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

Enumerator:

Never Refer to `DISPLAY_ALARM_STATE_NEVER` for details.

Always Refer to `DISPLAY_ALARM_STATE_ALWAYS` for details.

WhenInAlarm Refer to `DISPLAY_ALARM_STATE_WHEN_IN_ALARM` for details.

9.47.1.3 enum QEAnalogProgressBar::Formats

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

Enumerator:

Default Format as best appropriate for the data type.

Floating Format as a floating point number.

Integer Format as an integer.

UnsignedInteger Format as an unsigned integer.

Time Format as a time.

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

9.47.1.4 enum QEAnalogProgressBar::Notations

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

Enumerator:

Fixed Refer to QEStringFormatting::NOTATION_FIXED for details.

Scientific Refer to QEStringFormatting::NOTATION_SCIENTIFIC for details.

Automatic Refer to QEStringFormatting::NOTATION_AUTOMATIC for details.

9.47.1.5 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::alarmSeverityDisplayStyle [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 QEAnalogProgressBar::arrayAction [read, write]

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

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

9.47.4.5 bool QEAnalogProgressBar::displayAlarmState [read, write]

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

9.47.4.6 DisplayAlarmStateOptions QEAnalogProgressBar::displayAlarmStateOption [read, write]

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

9.47.4.7 Formats QEAnalogProgressBar::format [read, write]

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

9.47.4.8 unsigned QEAnalogProgressBar::int [read, write]

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

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

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

9.47.4.9 bool QEAnalogProgressBar::leadingZero [read, write]

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

9.47.4.10 QString QEAnalogProgressBar::localEnumeration [read, write]

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

Format is:

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

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

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

may be included in a string to indicate a line break

Examples are:

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

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

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

9.47.4.11 Notations QEAnalogProgressBar::notation [read, write]

Notation used for numerical formatting. Default is fixed.

9.47.4.12 int QEAnalogProgressBar::precision [read, write]

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

9.47.4.13 bool QEAnalogProgressBar::trailingZeros [read, write]

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

9.47.4.14 bool QEAnalogProgressBar::useDbDisplayLimits [read, write]

Use the EPICS database display limits

9.47.4.15 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.

9.47.4.16 UserLevels QEAnalogProgressBar::userLevelEnabled [read, write]

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

9.47.4.17 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.

9.47.4.18 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.

9.47.4.19 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.

9.47.4.20 UserLevels QEAnalogProgressBar::userLevelVisibility [read, write]

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

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

9.47.4.21 int QEAnalogProgressBar::value [read, write]

Current indicated value.

Reimplemented from [QEAnalogIndicator](#).

9.47.4.22 QString QEAnalogProgressBar::variable [read, write]

EPICS variable name (CA PV)

9.47.4.23 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.

9.47.4.24 QString QEAnalogProgressBar::variableSubstitutions [read, write]

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

9.47.4.25 bool QEAnalogProgressBar::visible [read, write]

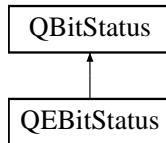
Display the widget. Default is true. Setting this property false is useful if widget is only used to provide a signal - for example, when supplying data to a [QELink](#) widget. Note, when false the widget will still be visible in Qt Designer.

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

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

9.48 QEBitStatus Class Reference

Inheritance diagram for QEBitStatus:



Public Types

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

Signals

- void `dbValueChanged` (const long &out)

Public Member Functions

- void `setVariableNameProperty` (QString variableName)

Property access function for `variable` property. This has special behaviour to work well within designer.
- QString `getVariableNameProperty` ()

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

Property access function for `variableSubstitutions` property. This has special behaviour to work well within designer.
- QString `getVariableNameSubstitutionsProperty` ()

Property access function for `variableSubstitutions` property. This has special behaviour to work well within designer.
- UserLevels `getUserLevelVisibilityProperty` ()

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

Access function for `userLevelVisibility` property - refer to `userLevelVisibility` property for details.
- UserLevels `getUserLevelEnabledProperty` ()

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

- Access function for `userLevelEnabled` property - refer to `userLevelEnabled` property for details.*
- `DisplayAlarmStateOptions getDisplayAlarmStateOptionProperty ()`
Access function for `displayAlarmStateOption` property - refer to `displayAlarmStateOption` property for details.
 - `void setDisplayAlarmStateOptionProperty (DisplayAlarmStateOptions option)`
Access function for `displayAlarmStateOption` property - refer to `displayAlarmStateOption` property for details.
 - `QEBitStatus (QWidget *parent=0)`
 - `QEBitStatus (const QString &variableName, QWidget *parent=0)`

Protected Member Functions

- `void establishConnection (unsigned int variableIndex)`
- `void dragEnterEvent (QDragEnterEvent *event)`
- `void dropEvent (QDropEvent *event)`
- `void mousePressEvent (QMouseEvent *event)`
- `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`
- `DisplayAlarmStateOptions displayAlarmStateOption`
- `double value`
- `bool isActive`
- `bool isValid`

9.48.1 Member Enumeration Documentation

9.48.1.1 enum QEBitStatus::DisplayAlarmStateOptions

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

Enumerator:

Never Refer to DISPLAY_ALARM_STATE_NEVER for details.

Always Refer to DISPLAY_ALARM_STATE_ALWAYS for details.

WhenInAlarm Refer to DISPLAY_ALARM_STATE_WHEN_IN_ALARM for details.

9.48.1.2 enum QEBitStatus::UserLevels

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

Enumerator:

User Refer to USERLEVEL_USER for details.

Scientist Refer to USERLEVEL_SCIENTIST for details.

Engineer Refer to USERLEVEL_ENGINEER for details.

9.48.2 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]

DEPRECATED. USE [displayAlarmStateOption](#) INSTEAD. If set (default) widget will indicate the alarm state of any variable data it is displaying. If clear widget will never indicate the alarm state of any variable data it is displaying. Typically the background

colour is set to indicate the alarm state. Note, this property is included in the set of standard properties as it applies to most widgets. It will do nothing for widgets that don't display data.

9.48.3.3 `DisplayAlarmStateOptions` `QEBitStatus::displayAlarmStateOption` [read, write]

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

9.48.3.4 `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.

9.48.3.5 `UserLevels` `QEBitStatus::userLevelEnabled` [read, write]

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

9.48.3.6 `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.

9.48.3.7 `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.

9.48.3.8 `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.

9.48.3.9 `UserLevels QEBitStatus::userLevelVisibility [read, write]`

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

9.48.3.10 `QString QEBitStatus::variable [read, write]`

EPICS variable name (CA PV)

9.48.3.11 `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.

9.48.3.12 `QString QEBitStatus::variableSubstitutions [read, write]`

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

9.48.3.13 `bool QEBitStatus::visible [read, write]`

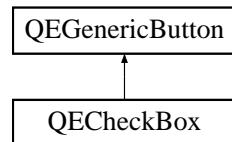
Display the widget. Default is true. Setting this property false is useful if widget is only used to provide a signal - for example, when supplying data to a [QELink](#) widget. Note, when false the widget will still be visible in Qt Designer.

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

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

9.49 QECheckBox Class Reference

Inheritance diagram for QECheckBox:



Public Types

- enum `UserLevels` { `User` = userLevelTypes::USERLEVEL_USER, `Scientist` = userLevelTypes::USERLEVEL_SCIENTIST, `Engineer` = userLevelTypes::USERLEVEL_ENGINEER }
 - enum `DisplayAlarmStateOptions` { `Never` = standardProperties::DISPLAY_ALARM_STATE_NEVER, `Always` = standardProperties::DISPLAY_ALARM_STATE_ALWAYS, `WhenInAlarm` = standardProperties::DISPLAY_ALARM_STATE_WHEN_IN_ALARM }
 - enum `Formats` {
 `Default` = QEStringFormatting::FORMAT_DEFAULT, `Floating` = QEStringFormatting::FORMAT_FLOATING, `Integer` = QEStringFormatting::FORMAT_INTEGER, `UnsignedInteger` = QEStringFormatting::FORMAT_UNSIGNEDINTEGER,
 `Time` = QEStringFormatting::FORMAT_TIME, `LocalEnumeration` = QEStringFormatting::FORMAT_LOCAL_ENUMERATE }
 - enum `Notations` { `Fixed` = QEStringFormatting::NOTATION_FIXED, `Scientific` = QEStringFormatting::NOTATION_SCIENTIFIC, `Automatic` = QEStringFormatting::NOTATION_AUTOMATIC }
 - enum `ArrayActions` { `Append` = QEStringFormatting::APPEND, `Ascii` = QEStringFormatting::ASCII, `Index` = QEStringFormatting::INDEX }
 - enum `UpdateOptions` { `Text` = QEGenericButton::UPDATE_TEXT, `Icon` = QEGenericButton::UPDATE_ICON, `TextAndIcon` = QEGenericButton::UPDATE_TEXT_AND_ICON, `State` = QEGenericButton::UPDATE_STATE }
- User friendly enumerations for updateOption property - refer to QEGenericButton::updateOptions for details.*
- enum `ProgramStartupOptionNames` { `None` = applicationLauncher::PSO_NONE, `Terminal` = applicationLauncher::PSO_TERMINAL, `LogOutput` = applicationLauncher::PSO_LOGOUTPUT, `StdOutput` = applicationLauncher::PSO_STDOUPUT }
 - enum `CreationOptionNames` {
 `Open` = QEActionRequests::OptionOpen, `NewTab` = QEActionRequests::OptionNewTab,
 `NewWindow` = QEActionRequests::OptionNewWindow, `DockTop` = QEActionRequests::OptionTopDockWindow,

```

DockBottom = QEActionRequests::OptionBottomDockWindow, DockLeft = QE-
ActionRequests::OptionLeftDockWindow, DockRight = QEActionRequests::OptionRightDockWindow,
DockTopTabbed = QEActionRequests::OptionTopDockWindowTabbed,
DockBottomTabbed = QEActionRequests::OptionBottomDockWindowTabbed, Dock-
LeftTabbed = QEActionRequests::OptionLeftDockWindowTabbed, DockRightTabbed
= QEActionRequests::OptionRightDockWindowTabbed, DockFloating = QEAction-
Requests::OptionFloatingDockWindow }
```

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

Public Slots

- void **requestAction** (const QEActionRequests &request)
- void **setDefaultStyle** (const QString &style)

Update the default style applied to this widget.

Signals

- void **dbValueChanged** (const QString &out)
- void **requestResend** ()
Internal use only. Used when changing a property value to force a re-display to reflect the new property value.
- void **newGui** (const QEActionRequests &request)
Internal use only. Request a new GUI is created. Typically, this is caught by the QEGui application.
- void **pressed** (int value)
- void **released** (int value)
- void **clicked** (int value)
- void **programCompleted** ()
Program started by button has completed.

Public Member Functions

- **QECheckBox** (QWidget *parent=0)
- **QECheckBox** (const QString &variableName, QWidget *parent=0)
- void **setVariableNameProperty** (QString variableName)
*Property access function for **variable** property. This has special behaviour to work well within designer.*
- QString **getVariableNameProperty** ()
*Property access function for **variable** property. This has special behaviour to work well within designer.*
- void **setVariableNameSubstitutionsProperty** (QString variableNameSubstitutions)
*Property access function for **variableSubstitutions** property. This has special behaviour to work well within designer.*

- `QString getVariableNameSubstitutionsProperty ()`
Property access function for `variableSubstitutions` property. This has special behaviour to work well within designer.
- `UserLevels getUserLevelVisibilityProperty ()`
Access function for `userLevelVisibility` property - refer to `userLevelVisibility` property for details.
- `void setUserLevelVisibilityProperty (UserLevels level)`
Access function for `userLevelVisibility` property - refer to `userLevelVisibility` property for details.
- `UserLevels getUserLevelEnabledProperty ()`
Access function for `userLevelEnabled` property - refer to `userLevelEnabled` property for details.
- `void setUserLevelEnabledProperty (UserLevels level)`
Access function for `userLevelEnabled` property - refer to `userLevelEnabled` property for details.
- `DisplayAlarmStateOptions getDisplayAlarmStateOptionProperty ()`
Access function for `displayAlarmStateOption` property - refer to `displayAlarmStateOption` property for details.
- `void setDisplayAlarmStateOptionProperty (DisplayAlarmStateOptions option)`
Access function for `displayAlarmStateOption` property - refer to `displayAlarmStateOption` property for details.
- `void setFormatProperty (Formats format)`
Access function for `format` property - refer to `format` property for details.
- `Formats getFormatProperty ()`
Access function for `format` property - refer to `format` property for details.
- `void 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`
- `DisplayAlarmStateOptions displayAlarmStateOption`
- `int precision`
- `bool useDbPrecision`
- `bool leadingZero`
- `bool trailingZeros`
- `bool addUnits`
- `QString localEnumeration`
- `Formats format`
- `Notations notation`
- `ArrayActions arrayAction`
- `Qt::Alignment alignment`
- `UpdateOptions updateOption`
- `QPixmap pixmap0`
- `QPixmap pixmap1`
- `QPixmap pixmap2`
- `QPixmap pixmap3`
- `QPixmap pixmap4`
- `QPixmap pixmap5`
- `QPixmap pixmap6`
- `QPixmap pixmap7`
- `QString password`
- `bool confirmAction`
- `QString confirmText`
- `bool writeOnPress`
- `bool writeOnRelease`
- `bool writeOnClick`
- `QString pressText`
- `QString releaseText`
- `QString clickText`
- `QString clickCheckedText`
- `QString labelText`
- `QString program`
- `QStringList arguments`
- `ProgramStartupOptionNames programStartupOption`
- `QString guiFile`
- `CreationOptionNames creationOption`
- `QString prioritySubstitutions`
- `QString customisationName`

9.49.1 Member Enumeration Documentation

9.49.1.1 enum QECheckBox::ArrayActions

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

Enumerator:

Append Refer to QEStringFormatting::APPEND for details.

Ascii Refer to QEStringFormatting::ASCII for details.

Index Refer to QEStringFormatting::INDEX for details.

9.49.1.2 enum QECheckBox::CreationOptionNames

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

Enumerator:

Open Replace the current GUI with the new GUI.

NewTab Open new GUI in a new tab.

NewWindow Open new GUI in a new window.

DockTop Open new GUI in a top dock window.

DockBottom Open new GUI in a bottom dock window.

DockLeft Open new GUI in a left dock window.

DockRight Open new GUI in a right dock window.

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

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

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

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

DockFloating Open new GUI in a floating dock window.

9.49.1.3 enum QECheckBox::DisplayAlarmStateOptions

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

Enumerator:

Never Refer to DISPLAY_ALARM_STATE_NEVER for details.

Always Refer to DISPLAY_ALARM_STATE_ALWAYS for details.

WhenInAlarm Refer to DISPLAY_ALARM_STATE_WHEN_IN_ALARM for details.

9.49.1.4 enum QECheckBox::Formats

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

Enumerator:

Default Format as best appropriate for the data type.

Floating Format as a floating point number.

Integer Format as an integer.

UnsignedInteger Format as an unsigned integer.

Time Format as a time.

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

9.49.1.5 enum QECheckBox::Notations

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

Enumerator:

Fixed Refer to QEStringFormatting::NOTATION_FIXED for details.

Scientific Refer to QEStringFormatting::NOTATION_SCIENTIFIC for details.

Automatic Refer to QEStringFormatting::NOTATION_AUTOMATIC for details.

9.49.1.6 enum QECheckBox::ProgramStartupOptionNames

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

Enumerator:

None Just run the program.

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

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

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

9.49.1.7 enum QECheckBox::UpdateOptions

User friendly enumerations for updateOption property - refer to QEGenericButton::updateOptions for details.

Enumerator:

Text Data updates will update the button text.

Icon Data updates will update the button icon.

TextAndIcon Data updates will update the button text and icon.

State Data updates will update the button state (checked or unchecked)

9.49.1.8 enum QECheckBox::UserLevels

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

Enumerator:

User Refer to USERLEVEL_USER for details.

Scientist Refer to USERLEVEL_SCIENTIST for details.

Engineer Refer to USERLEVEL_ENGINEER for details.

9.49.2 Constructor & Destructor Documentation

9.49.2.1 QECheckBox::QECheckBox (QWidget * *parent* = 0)

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

9.49.2.2 QECheckBox::QECheckBox (const QString & *variableName*, QWidget * *parent* = 0)

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

9.49.3 Member Function Documentation

9.49.3.1 void QECheckBox::clicked (int *value*) [signal]

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

9.49.3.2 void QECheckBox::dbValueChanged (const QString & *out*) [signal]

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

9.49.3.3 void QECheckBox::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 QECheckBox::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 QECheckBox::requestAction (const QEActionRequests & request) [inline, slot]

Default slot used to create a new GUI if there is no slot indicated in the ContainerProfile class. This slot is typically used when the button is pressed within the Designer preview window to allow the operation of the button to be tested. If an application does not specify a slot to use for creating new windows (through the ContainerProfile class) a window will still be created through this slot, but it will not respect the window creation options or any other window related application constraints. For example, the QEGui application does provide a slot for creating new GUIs in the ContainerProfile class which respects the creation options, knows how to add tabs in the application, and extend the application's window menu in the menu bar.

9.49.4 Property Documentation

9.49.4.1 bool QECheckBox::addUnits [read, write]

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

9.49.4.2 Qt::Alignment QECheckBox::alignment [read, write]

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

9.49.4.3 bool QECheckBox::allowDrop [read, write]

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

9.49.4.4 QStringList QECheckBox::arguments [read, write]

Arguments for program specified in the 'program' property.

9.49.4.5 ArrayActions QECheckBox::arrayAction [read, write]

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

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

9.49.4.6 QString QECheckBox::clickCheckedText [read, write]

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

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

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

Reimplemented from [QEGenericButton](#).

9.49.4.7 QString QECheckBox::clickText [read, write]

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

Reimplemented from [QEGenericButton](#).

9.49.4.8 bool QECheckBox::confirmAction [read, write]

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

9.49.4.9 QString QECheckBox::confirmText [read, write]

Text used to confirm action if confirmation dialog is presented

Reimplemented from [QEGenericButton](#).

9.49.4.10 CreationOptionNames QECheckBox::creationOption [read, write]

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

Reimplemented from [QEGenericButton](#).

9.49.4.11 QString QECheckBox::customisationName [read, write]

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

Reimplemented from [QEGenericButton](#).

9.49.4.12 bool QECheckBox::displayAlarmState [read, write]

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

9.49.4.13 DisplayAlarmStateOptions QECheckBox::displayAlarmStateOption [read, write]

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

9.49.4.14 Formats QECheckBox::format [read, write]

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

9.49.4.15 `QString QECheckBox::guiFile` [read, write]

File name of GUI to be presented on button click. File name can be absolute, relative to the path of the QEform in which the `QEPushButton` is located, relative to the any path in the path list published in the ContainerProfile class, or relative to the current path. See `QEWidget::openQEFfile()` in `QEWidget.cpp` for details.

9.49.4.16 `unsigned QECheckBox::int` [read, write]

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

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.

9.49.4.17 `QString QECheckBox::labelText` [read, write]

Button label text (prior to substitution). Macro substitutions will be applied to this text and the result will be set as the button text. Used when data updates are not being represented in the button text. IF NOT LEFT EMPTY, THIS TEXT WILL TAKE PRIORITY OVER THE PUSH BUTTON 'text' PROPERTY! For example, a button in a sub form may have a 'labelText' property of 'Turn Pump On'. When the sub form is used twice in a main form with substitutions PUMPNUM=1 and PUMPNUM=2 respectively, the two identical buttons in the sub forms will have the labels 'Turn Pump 1 On' and 'Turn Pump 2 On' respectively.

Reimplemented from `QEGenericButton`.

9.49.4.18 `bool QECheckBox::leadingZero` [read, write]

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

9.49.4.19 `QString QECheckBox::localEnumeration` [read, write]

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

Format is:

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

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

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

may be included in a string to indicate a line break

Examples are:

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

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

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

9.49.4.20 Notations QECheckBox::notation [read, write]

Notation used for numerical formatting. Default is fixed.

9.49.4.21 QString QECheckBox::password [read, write]

Password user will need to enter before any action is taken

Reimplemented from [QEGenericButton](#).

9.49.4.22 QPixmap QECheckBox:: pixmap0 [read, write]

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

9.49.4.23 QPixmap QECheckBox:: pixmap1 [read, write]

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

9.49.4.24 QPixmap QECheckBox:: pixmap2 [read, write]

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

9.49.4.25 QPixmap QECheckBox:: pixmap3 [read, write]

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

9.49.4.26 QPixmap QECheckBox:: pixmap4 [read, write]

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

9.49.4.27 QPixmap QECheckBox:: pixmap5 [read, write]

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

9.49.4.28 QPixmap QECheckBox:: pixmap6 [read, write]

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

9.49.4.29 QPixmap QECheckBox:: pixmap7 [read, write]

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

9.49.4.30 int QECheckBox:: precision [read, write]

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

9.49.4.31 QString QECheckBox:: pressText [read, write]

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

Reimplemented from [QEGenericButton](#).

9.49.4.32 QString QECheckBox:: prioritySubstitutions [read, write]

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

Reimplemented from [QEGenericButton](#).

9.49.4.33 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

9.49.4.34 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.

9.49.4.35 QString QECheckBox::releaseText [read, write]

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

Reimplemented from [QEGenericButton](#).

9.49.4.36 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)

9.49.4.37 bool QECheckBox::trailingZeros [read, write]

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

9.49.4.38 UpdateOptions QECheckBox::updateOption [read, write]

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

Reimplemented from [QEGenericButton](#).

9.49.4.39 bool QECheckBox::useDbPrecision [read, write]

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

9.49.4.40 UserLevels QECheckBox::userLevelEnabled [read, write]

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

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

9.49.4.41 `QString QECheckBox::userLevelEngineerStyle` [read, write]

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

9.49.4.42 `QString QECheckBox::userLevelScientistStyle` [read, write]

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

9.49.4.43 `QString QECheckBox::userLevelUserStyle` [read, write]

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

9.49.4.44 `UserLevels QECheckBox::userLevelVisibility` [read, write]

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

9.49.4.45 `QString QECheckBox::variable` [read, write]

EPICS variable name (CA PV)

9.49.4.46 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.

9.49.4.47 QString QECheckBox::variableSubstitutions [read, write]

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

9.49.4.48 bool QECheckBox::visible [read, write]

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

9.49.4.49 bool QECheckBox::writeOnClick [read, write]

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

Reimplemented from [QEGenericButton](#).

9.49.4.50 bool QECheckBox::writeOnPress [read, write]

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

Reimplemented from [QEGenericButton](#).

9.49.4.51 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 }
- enum **DisplayAlarmStateOptions** { **Never** = standardProperties::DISPLAY_ALARM_STATE_NEVER, **Always** = standardProperties::DISPLAY_ALARM_STATE_ALWAYS, **WhenInAlarm** = standardProperties::DISPLAY_ALARM_STATE_WHEN_IN_ALARM }

Public Slots

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

Signals

- void **dbValueChanged** (const qulonglong &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 writeOnChangeEvent)
- bool **getWriteOnChange** ()
- void **setSubscribe** (bool subscribe)
- bool **getSubscribe** ()
- void **setUseDbEnumerations** (bool useDbEnumerations)
- bool **getUseDbEnumerations** ()
- void **setLocalEnumerations** (const QString &localEnumerations)
- QString **getLocalEnumerations** ()
- void **setVariableNameProperty** (QString variableName)
Property access function for `variable` property. This has special behaviour to work well within designer.
- QString **getVariableNameProperty** ()
Property access function for `variable` property. This has special behaviour to work well within designer.
- void **setVariableNameSubstitutionsProperty** (QString variableNameSubstitutions)
Property access function for `variableSubstitutions` property. This has special behaviour to work well within designer.
- QString **getVariableNameSubstitutionsProperty** ()
Property access function for `variableSubstitutions` property. This has special behaviour to work well within designer.
- UserLevels **getUserLevelVisibilityProperty** ()
Access function for `userLevelVisibility` property - refer to `userLevelVisibility` property for details.
- void **setUserLevelVisibilityProperty** (UserLevels level)
Access function for `userLevelVisibility` property - refer to `userLevelVisibility` property for details.
- UserLevels **getUserLevelEnabledProperty** ()
Access function for `userLevelEnabled` property - refer to `userLevelEnabled` property for details.
- void **setUserLevelEnabledProperty** (UserLevels level)
Access function for `userLevelEnabled` property - refer to `userLevelEnabled` property for details.
- DisplayAlarmStateOptions **getDisplayAlarmStateOptionProperty** ()
Access function for `displayAlarmStateOption` property - refer to `displayAlarmStateOption` property for details.
- void **setDisplayAlarmStateOptionProperty** (DisplayAlarmStateOptions option)
Access function for `displayAlarmStateOption` property - refer to `displayAlarmStateOption` property for details.

Protected Member Functions

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

Protected Attributes

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

Properties

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

9.51.1 Member Enumeration Documentation

9.51.1.1 enum QEComboBox::DisplayAlarmStateOptions

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

Enumerator:

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

9.51.1.2 enum QEComboBox::UserLevels

User friendly enumerations for `userLevelVisibility` and `userLevelEnabled` properties - refer to `userLevelVisibility` and `userLevelEnabled` properties and `userLevel` enumeration for details.

Enumerator:

User Refer to `USERLEVEL_USER` for details.

Scientist Refer to `USERLEVEL_SCIENTIST` for details.

Engineer Refer to `USERLEVEL_ENGINEER` for details.

9.51.2 Member Function Documentation

9.51.2.1 void QEComboBox::dbValueChanged (const qulonglong & out) [signal]

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

9.51.3 Member Data Documentation

9.51.3.1 bool QEComboBox::useDbEnumerations [read, write, protected]

Use database enumerations - defaults to true

9.51.3.2 bool QEComboBox::writeOnChange [read, write, protected]

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

9.51.4 Property Documentation

9.51.4.1 bool QEComboBox::allowDrop [read, write]

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

9.51.4.2 bool QEComboBox::displayAlarmState [read, write]

DEPRECATED. USE `displayAlarmStateOption` INSTEAD. If set (default) widget will indicate the alarm state of any variable data it is displaying. If clear widget will never indicate the alarm state of any variable data it is displaying. Typically the background

colour is set to indicate the alarm state. Note, this property is included in the set of standard properties as it applies to most widgets. It will do nothing for widgets that don't display data.

9.51.4.3 `DisplayAlarmStateOptions QEComboBox::displayAlarmStateOption` [read, write]

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

9.51.4.4 `unsigned QEComboBox::int` [read, write]

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

9.51.4.5 `QString QEComboBox::localEnumeration` [read, write]

Enumerations values used when `useDbEnumerations` is false.

9.51.4.6 `bool QEComboBox::subscribe` [read, write]

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

9.51.4.7 `UserLevels QEComboBox::userLevelEnabled` [read, write]

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

9.51.4.8 `QString QEComboBox::userLevelEngineerStyle` [read, write]

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

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

9.51.4.9 **QString QEComboBox::userLevelScientistStyle** [read, write]

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

9.51.4.10 **QString QEComboBox::userLevelUserStyle** [read, write]

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

9.51.4.11 **UserLevels QEComboBox::userLevelVisibility** [read, write]

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

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

EPICS variable name (CA PV)

9.51.4.13 **bool QEComboBox::variableAsToolTip** [read, write]

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

9.51.4.14 **QString QEComboBox::variableSubstitutions** [read, write]

Macro substitutions. The default is no substitutions. The format is NAME1=VALUE1[,] NAME2=VALUE2... Values may be quoted strings. For example, 'PUMP=PMP3, NAME

= "My Pump" These substitutions are applied to variable names for all QE widgets. In some widgets are also used for other purposes.

9.51.4.15 bool QEComboBox::visible [read, write]

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

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

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

9.52 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 }
- enum **DisplayAlarmStateOptions** { **Never** = standardProperties::DISPLAY_ALARM_STATE_NEVER, **Always** = standardProperties::DISPLAY_ALARM_STATE_ALWAYS, **WhenInAlarm** = standardProperties::DISPLAY_ALARM_STATE_WHEN_IN_ALARM }

Public Member Functions

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

- void **refreshFields** ()
- void **userLevelChanged** (userLevelTypes::userLevels pValue)
- void **setConfigurationTypeProperty** (configurationTypesProperty pConfigurationType)

- configurationTypesProperty **getConfigurationTypeProperty** ()
- void **setOptionsLayoutProperty** (optionsLayoutProperty pOptionsLayout)
- optionsLayoutProperty **getOptionsLayoutProperty** ()
- **UserLevels getUserLevelVisibilityProperty** ()
Access function for userLevelVisibility property - refer to userLevelVisibility property for details.
- void **setUserLevelVisibilityProperty** (**UserLevels** level)
Access function for userLevelVisibility property - refer to userLevelVisibility property for details.
- **UserLevels getUserLevelEnabledProperty** ()
Access function for userLevelEnabled property - refer to userLevelEnabled property for details.
- void **setUserLevelEnabledProperty** (**UserLevels** level)
Access function for userLevelEnabled property - refer to userLevelEnabled property for details.
- **DisplayAlarmStateOptions getDisplayAlarmStateOptionProperty** ()
Access function for displayAlarmStateOption property - refer to displayAlarmStateOption property for details.
- void **setDisplayAlarmStateOptionProperty** (**DisplayAlarmStateOptions** option)
Access function for displayAlarmStateOption property - refer to displayAlarmStateOption property for details.

Public Attributes

- QList< **_Item** * > **itemList**
- QList< **_Field** * > **currentFieldList**

Protected Attributes

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

Properties

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

Change the order of the widgets. Valid orders are: TOP, BOTTOM, LEFT and RIG.
- bool **variableAsToolTip**
- bool **allowDrop**
- bool **visible**
- unsigned **int**
- QString **userLevelUserStyle**
- QString **userLevelScientistStyle**
- QString **userLevelEngineerStyle**
- **UserLevels userLevelVisibility**
- **UserLevels userLevelEnabled**
- bool **displayAlarmState**
- **DisplayAlarmStateOptions displayAlarmStateOption**

9.52.1 Member Enumeration Documentation

9.52.1.1 enum QEConfiguredLayout::DisplayAlarmStateOptions

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

Enumerator:

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

9.52.1.2 enum QEConfiguredLayout::UserLevels

User friendly enumerations for **userLevelVisibility** and **userLevelEnabled** properties - refer to **userLevelVisibility** and **userLevelEnabled** properties and **userLevel** enumeration for details.

Enumerator:

- User** Refer to USERLEVEL_USER for details.
- Scientist** Refer to USERLEVEL_SCIENTIST for details.
- Engineer** Refer to USERLEVEL_ENGINEER for details.

9.52.2 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]

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

9.52.2.3 DisplayAlarmStateOptions QEConfiguredLayout::displayAlarmStateOption [read, write]

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

9.52.2.4 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.

9.52.2.5 UserLevels QEConfiguredLayout::userLevelEnabled [read, write]

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

9.52.2.6 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.

9.52.2.7 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.

9.52.2.8 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.

9.52.2.9 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'.

9.52.2.10 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.

9.52.2.11 bool QEConfiguredLayout::visible [read, write]

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

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

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

9.53 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 }
- enum **DisplayAlarmStateOptions** { **Never** = standardProperties::DISPLAY_ALARM_STATE_NEVER, **Always** = standardProperties::DISPLAY_ALARM_STATE_ALWAYS, **WhenInAlarm** = standardProperties::DISPLAY_ALARM_STATE_WHEN_IN_ALARM }

Signals

- void **selected** (QString pFilename)

Public Member Functions

- **QEFileBrowser** (QWidget *pParent=0)
- void **setVariableName** (QString pValue)
- QString **getVariableName** ()
- void **setVariableNameSubstitutions** (QString pValue)
- QString **getVariableNameSubstitutions** ()
- void **setDirectoryPath** (QString pValue)
- QString **getDirectoryPath** ()
- void **setShowDirectoryPath** (bool pValue)
- bool **getShowDirectoryPath** ()
- void **setShowDirectoryBrowser** (bool pValue)
- bool **getShowDirectoryBrowser** ()
- void **setShowRefresh** (bool pValue)
- bool **getShowRefresh** ()
- void **setShowTable** (bool pValue)
- bool **getShowTable** ()
- void **setShowColumnTime** (bool pValue)
- bool **getShowColumnTime** ()
- void **setShowColumnSize** (bool pValue)
- bool **getShowColumnSize** ()
- void **setShowColumnFilename** (bool pValue)
- bool **getShowColumnFilename** ()
- void **setShowFileExtension** (bool pValue)
- bool **getShowFileExtension** ()
- void **setFileFilter** (QString pValue)
- QString **getFileFilter** ()
- void **setFileDialogDirectoriesOnly** (bool pValue)
- bool **getFileDialogDirectoriesOnly** ()
- void **setOptionsLayout** (int pValue)
- int **getOptionsLayout** ()
- void **updateTable** ()
- void **setOptionsLayoutProperty** (optionsLayoutProperty pOptionsLayout)
- optionsLayoutProperty **getOptionsLayoutProperty** ()
- UserLevels **getUserLevelVisibilityProperty** ()
Access function for `userLevelVisibility` property - refer to `userLevelVisibility` property for details.
- void **setUserLevelVisibilityProperty** (UserLevels level)
Access function for `userLevelVisibility` property - refer to `userLevelVisibility` property for details.
- UserLevels **getUserLevelEnabledProperty** ()
Access function for `userLevelEnabled` property - refer to `userLevelEnabled` property for details.
- void **setUserLevelEnabledProperty** (UserLevels level)
Access function for `userLevelEnabled` property - refer to `userLevelEnabled` property for details.

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

Protected Attributes

- **QLineEdit * qlineEditDirectoryPath**
- **QPushButton * qpushButtonDirectoryBrowser**
- **QPushButton * qpushButtonRefresh**
- **_QTableWidgetFileBrowser * qtableWidgetFileBrowser**
- **QString fileFilter**
*Specify which files to browse. To specify more than one filter, please separate them with a ;. Example: *.py;*.ui (this will only display files with an extension .py or .ui).*
- **bool showFileExtension**
Show/hide the extension of files.
- **bool fileDialogDirectoriesOnly**
Enable/disable the browsing of directories-only when opening the dialog window.
- **int optionsLayout**

Properties

- **QString variable**
- **QString variableSubstitutions**
- **QString directoryPath**
Default directory where to browse files when `QEFileBrowser` is launched for the first time.
- **bool showDirectoryPath**
Show/hide directory path line edit where the user can specify the directory to browse files.
- **bool showDirectoryBrowser**
Show/hide button to open the dialog window to browse for directories and files.
- **bool showRefresh**
Show/hide button to refresh the table containing the list of files being browsed.
- **bool showTable**
Show/hide table containing the list of files being browsed.
- **bool showColumnTime**
Show/hide column containing the time of creation of files.
- **bool showColumnSize**
Show/hide column containing the size (in bytes) of files.
- **bool showColumnFilename**
Show/hide column containing the name of files.

- optionsLayoutProperty [optionsLayout](#)
Change the order of the widgets. Valid orders are: TOP, BOTTOM, LEFT and RIGHT.
- bool [variableAsToolTip](#)
- bool [allowDrop](#)
- bool [visible](#)
- unsigned [int](#)
- QString [userLevelUserStyle](#)
- QString [userLevelScientistStyle](#)
- QString [userLevelEngineerStyle](#)
- [UserLevels userLevelVisibility](#)
- [UserLevels userLevelEnabled](#)
- bool [displayAlarmState](#)
- [DisplayAlarmStateOptions displayAlarmStateOption](#)

9.54.1 Detailed Description

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

9.54.2 Member Enumeration Documentation

9.54.2.1 enum QEFileBrowser::DisplayAlarmStateOptions

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

Enumerator:

Never Refer to DISPLAY_ALARM_STATE_NEVER for details.

Always Refer to DISPLAY_ALARM_STATE_ALWAYS for details.

WhenInAlarm Refer to DISPLAY_ALARM_STATE_WHEN_IN_ALARM for details.

9.54.2.2 enum QEFileBrowser::UserLevels

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

Enumerator:

User Refer to USERLEVEL_USER for details.

Scientist Refer to USERLEVEL_SCIENTIST for details.

Engineer Refer to USERLEVEL_ENGINEER for details.

9.54.3 Member Function Documentation

9.54.3.1 void QEFileBrowser::selected (QString *pFilename*) [signal]

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

9.54.4 Property Documentation

9.54.4.1 bool QEFileBrowser::allowDrop [read, write]

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

9.54.4.2 bool QEFileBrowser::displayAlarmState [read, write]

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

9.54.4.3 DisplayAlarmStateOptions QEFileBrowser::displayAlarmStateOption [read, write]

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

9.54.4.4 unsigned QEFileBrowser::int [read, write]

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

9.54.4.5 UserLevels QEFfileBrowser::userLevelEnabled [read, write]

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

9.54.4.6 QString QEFfileBrowser::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.

9.54.4.7 QString QEFfileBrowser::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.

9.54.4.8 QString QEFfileBrowser::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.

9.54.4.9 UserLevels QEFfileBrowser::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'.

9.54.4.10 QString QEFileBrowser::variable [read, write]

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

9.54.4.11 bool QEFileBrowser::variableAsToolTip [read, write]

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

9.54.4.12 QString QEFileBrowser::variableSubstitutions [read, write]

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

9.54.4.13 bool QEFileBrowser::visible [read, write]

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

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

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

9.55 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 &uifileNameIn, QWidget *parent=0)
- void **commonInit** (const bool alertIfUINotFoundIn, const bool loadManuallyIn)

- void **setQEguiTitle** (const QString titleIn)
- QString **getQEguiTitle** ()
- QString **getFullFileName** ()
- QString **getUiFileName** ()
- void **setFileMonitoringIsEnabled** (bool fileMonitoringIsEnabled)
- bool **getFileMonitoringIsEnabled** ()
- void **setHandleGuiLaunchRequests** (bool handleGuiLaunchRequests)
- bool **getHandleGuiLaunchRequests** ()
- void **setResizeContents** (bool resizeContentsIn)
- 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

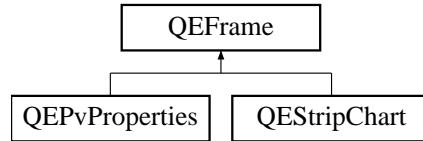
- QString **uiFile**
- QString **variableSubstitutions**
- unsigned **int**
- MessageFilterOptions **messageFormFilter**
- MessageFilterOptions **messageSourceFilter**

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 }
- enum `DisplayAlarmStateOptions` { `Never` = standardProperties::DISPLAY_ALARM_STATE_NEVER, `Always` = standardProperties::DISPLAY_ALARM_STATE_ALWAYS, `WhenInAlarm` = standardProperties::DISPLAY_ALARM_STATE_WHEN_IN_ALARM }

Public Member Functions

- `UserLevels getUserLevelVisibilityProperty ()`
Access function for `userLevelVisibility` property - refer to `userLevelVisibility` property for details.
- `void setUserLevelVisibilityProperty (UserLevels level)`
Access function for `userLevelVisibility` property - refer to `userLevelVisibility` property for details.
- `UserLevels getUserLevelEnabledProperty ()`
Access function for `userLevelEnabled` property - refer to `userLevelEnabled` property for details.
- `void setUserLevelEnabledProperty (UserLevels level)`
Access function for `userLevelEnabled` property - refer to `userLevelEnabled` property for details.
- `DisplayAlarmStateOptions getDisplayAlarmStateOptionProperty ()`
Access function for `displayAlarmStateOption` property - refer to `displayAlarmStateOption` property for details.
- `void setDisplayAlarmStateOptionProperty (DisplayAlarmStateOptions option)`
Access function for `displayAlarmStateOption` property - refer to `displayAlarmStateOption` property for details.
- `QEFrame (QWidget *parent=0)`
- `QSize sizeHint () const`

Properties

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

9.56.1 Member Enumeration Documentation

9.56.1.1 enum QEFrame::DisplayAlarmStateOptions

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

Enumerator:

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

9.56.1.2 enum QEFrame::UserLevels

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

Enumerator:

- User** Refer to USERLEVEL_USER for details.
Scientist Refer to USERLEVEL_SCIENTIST for details.
Engineer Refer to USERLEVEL_ENGINEER for details.

9.56.2 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]

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

9.56.2.3 DisplayAlarmStateOptions QEFrame::displayAlarmStateOption [read, write]

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

9.56.2.4 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.

9.56.2.5 UserLevels QEFrame::userLevelEnabled [read, write]

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

9.56.2.6 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.

9.56.2.7 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.

9.56.2.8 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.

9.56.2.9 UserLevels QEFrame::userLevelVisibility [read, write]

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

9.56.2.10 bool QEFrame::variableAsToolTip [read, write]

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

9.56.2.11 bool QEFrame::visible [read, write]

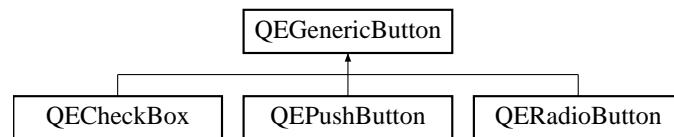
Display the widget. Default is true. Setting this property false is useful if widget is only used to provide a signal - for example, when supplying data to a [QELink](#) widget. Note, when false the widget will still be visible in Qt Designer.

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

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

9.57 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 **getDefaultValueOption** ()=0
- void **startGui** (const QEActionRequests &request)
- void **setup** ()
- void **establishConnection** (unsigned int variableIndex)

Protected Attributes

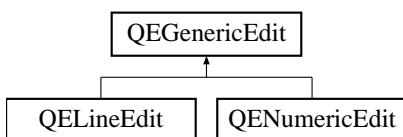
- applicationLauncher **programLauncher**

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 }
- enum `DisplayAlarmStateOptions` { `Never` = standardProperties::DISPLAY_ALARM_STATE_NEVER, `Always` = standardProperties::DISPLAY_ALARM_STATE_ALWAYS, `WhenInAlarm` = standardProperties::DISPLAY_ALARM_STATE_WHEN_IN_ALARM }

Public Slots

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

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.
- `DisplayAlarmStateOptions getDisplayAlarmStateOptionProperty` ()
 - Access function for `displayAlarmStateOption` property - refer to `displayAlarmStateOption` property for details.
- void `setDisplayAlarmStateOptionProperty` (DisplayAlarmStateOptions option)
 - Access function for `displayAlarmStateOption` property - refer to `displayAlarmStateOption` property for details.
- `QEGenericEdit` (QWidget *parent=0)
- `QEGenericEdit` (const QString &variableName, QWidget *parent=0)
- void `setWriteOnLoseFocus` (bool writeOnLoseFocus)
- bool `getWriteOnLoseFocus` ()
- void `setWriteOnEnter` (bool writeOnEnter)
- bool `getWriteOnEnter` ()
- void `setWriteOnFinish` (bool writeOnFinish)
- bool `getWriteOnFinish` ()
- void `setConfirmWrite` (bool confirmWrite)
- bool `getConfirmWrite` ()
- void `setSubscribe` (bool subscribe)
- bool `getSubscribe` ()
- void `writeValue` (qcaobject::QCaObject *qca, QVariant newValue)
- void `writeNow` ()

Protected Member Functions

- void `setDataIfNoFocus` (const QVariant &value, QCaAlarmInfo &alarmInfo, QCaDateTime &dateTime)
- bool `getIsConnected` ()
- bool `getIsFirstUpdate` ()
- 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`
- `DisplayAlarmStateOptions displayAlarmStateOption`

9.58.1 Member Enumeration Documentation

9.58.1.1 enum QEGenericEdit::DisplayAlarmStateOptions

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

Enumerator:

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

9.58.1.2 enum QEGenericEdit::UserLevels

User friendly enumerations for `userLevelVisibility` and `userLevelEnabled` properties - refer to `userLevelVisibility` and `userLevelEnabled` properties and `userLevel` enumeration for details.

Enumerator:

- User** Refer to `USERLEVEL_USER` for details.
- Scientist** Refer to `USERLEVEL_SCIENTIST` for details.
- Engineer** Refer to `USERLEVEL_ENGINEER` for details.

9.58.2 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]

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

9.58.4.4 DisplayAlarmStateOptions `QEGenericEdit::displayAlarmStateOption` [read, write]

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

9.58.4.5 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 [QLineEdit](#).

9.58.4.6 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)

9.58.4.7 UserLevels `QEGenericEdit::userLevelEnabled` [read, write]

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

9.58.4.8 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.

9.58.4.9 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.

9.58.4.10 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.

9.58.4.11 UserLevels QEGenericEdit::userLevelVisibility [read, write]

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

9.58.4.12 QString QEGenericEdit::variable [read, write]

EPICS variable name (CA PV)

9.58.4.13 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.

9.58.4.14 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.

9.58.4.15 bool QEGenericEdit::visible [read, write]

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

9.58.4.16 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').

9.58.4.17 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).

9.58.4.18 bool QEGenericEdit::writeOnLoseFocus [read, write]

Sets if this widget automatically writes any changes when it loses focus. Default is 'false' (does not write any changes when it loses focus).

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

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

9.59 QEGroupBox Class Reference

Public Types

- enum [UserLevels](#) { [User](#) = userLevelTypes::USERLEVEL_USER, [Scientist](#) = userLevelTypes::USERLEVEL_SCIENTIST, [Engineer](#) = userLevelTypes::USERLEVEL_ENGINEER }
- enum [DisplayAlarmStateOptions](#) { [Never](#) = standardProperties::DISPLAY_ALARM_STATE_NEVER, [Always](#) = standardProperties::DISPLAY_ALARM_STATE_ALWAYS, [WhenInAlarm](#) = standardProperties::DISPLAY_ALARM_STATE_WHEN_IN_ALARM }

Public Member Functions

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

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

Protected Member Functions

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

Properties

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

9.59.1 Member Enumeration Documentation

9.59.1.1 enum QEGroupBox::DisplayAlarmStateOptions

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

Enumerator:

Never Refer to DISPLAY_ALARM_STATE_NEVER for details.

Always Refer to DISPLAY_ALARM_STATE_ALWAYS for details.

WhenInAlarm Refer to DISPLAY_ALARM_STATE_WHEN_IN_ALARM for details.

9.59.1.2 enum QEGroupBox::UserLevels

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

Enumerator:

User Refer to USERLEVEL_USER for details.

Scientist Refer to USERLEVEL_SCIENTIST for details.

Engineer Refer to USERLEVEL_ENGINEER for details.

9.59.2 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]

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

9.59.2.3 DisplayAlarmStateOptions QEGroupBox::displayAlarmStateOption [read, write]

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

9.59.2.4 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.

9.59.2.5 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

9.59.2.6 QString QEGroupBox::textSubstitutions [read, write]

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

9.59.2.7 UserLevels QEGroupBox::userLevelEnabled [read, write]

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

9.59.2.8 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.

9.59.2.9 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.

9.59.2.10 `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.

9.59.2.11 `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'.

9.59.2.12 `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.

9.59.2.13 `bool QEGroupBox::visible` [read, write]

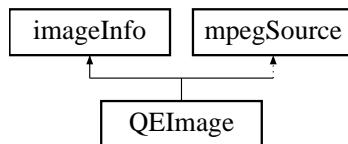
Display the widget. Default is true. Setting this property false is useful if widget is only used to provide a signal - for example, when supplying data to a [QELink](#) widget. Note, when false the widget will still be visible in Qt Designer.

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

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

9.60 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 `DisplayAlarmStateOptions` { `Never = standardProperties::DISPLAY_ALARM_-`

 `STATE_NEVER, Always = standardProperties::DISPLAY_ALARM_STATE_ALWAYS,`

 `WhenInAlarm = standardProperties::DISPLAY_ALARM_STATE_WHEN_IN_ALARM` }
- enum `FormatOptions` {

 `Mono = imageDataFormats::MONO, Bayer = imageDataFormats::BAYERRG, Bay-`

 `erGB = imageDataFormats::BAYERGB, BayerBG = imageDataFormats::BAYERBG,`

 `BayerGR = imageDataFormats::BAYERGR, BayerRG = imageDataFormats::BAYERRG,`

 `rgb1 = imageDataFormats::RGB1, rgb2 = imageDataFormats::RGB2,`

 `rgb3 = imageDataFormats::RGB3, yuv444 = imageDataFormats::YUV444, yuv422`

 `= imageDataFormats::YUV422, yuv421 = imageDataFormats::YUV421` }
- enum `EllipseVariableDefinitions` { `BoundingRectangle = BOUNDING_RECTANGLE,`

 `CenterAndSize = CENTRE_AND_SIZE` }
- enum `TargetOptions` { `DottedFullCrosshair = VideoWidget::CROSSHAIR1, SolidS-`

 `mallCrosshair = VideoWidget::CROSSHAIR2` }
- enum `ResizeOptions` { `Zoom = QEImage::RESIZE_OPTION_ZOOM, Fit = QEImage::RESIZE_-`

 `OPTION_FIT` }
- enum `RotationOptions` { `NoRotation = 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_STDOUPUT` }

Public Slots

- void `setImageFile` (QString name)
- void `setSelectPanMode` ()

Framework use only. Slot to allow external setting of selection menu options.
- void `setSelectVSliceMode` ()

Framework use only. Slot to allow external setting of selection menu options.

- void **setSelectHSliceMode ()**
Framework use only. Slot to allow external setting of selection menu options.
- void **setSelectArea1Mode ()**
Framework use only. Slot to allow external setting of selection menu options.
- void **setSelectArea2Mode ()**
Framework use only. Slot to allow external setting of selection menu options.
- void **setSelectArea3Mode ()**
Framework use only. Slot to allow external setting of selection menu options.
- void **setSelectArea4Mode ()**
Framework use only. Slot to allow external setting of selection menu options.
- void **setSelectProfileMode ()**
Framework use only. Slot to allow external setting of selection menu options.
- void **setSelectTargetMode ()**
Framework use only. Slot to allow external setting of selection menu options.
- void **setSelectBeamMode ()**
Framework use only. Slot to allow external setting of selection menu options.
- void **pauseClicked ()**
Framework use only. Slot to allow external setting of selection menu options.
- void **saveClicked ()**
Framework use only. Slot to allow external setting of selection menu options.
- void **targetClicked ()**
Framework use only. Slot to allow external setting of selection menu options.
- void **imageDisplayPropsDestroyed (QObject *)**
Framework use only. Slot to catch deletion of components (such as profile plots) that have been passed to the application for presentation.
- void **vSliceDisplayDestroyed (QObject *)**
Framework use only. Slot to catch deletion of components (such as profile plots) that have been passed to the application for presentation.
- void **hSliceDisplayDestroyed (QObject *)**
Framework use only. Slot to catch deletion of components (such as profile plots) that have been passed to the application for presentation.
- void **profileDisplayDestroyed (QObject *)**
Framework use only. Slot to catch deletion of components (such as profile plots) that have been passed to the application for presentation.
- void **recorderDestroyed (QObject *)**
Framework use only. Slot to catch deletion of components (such as profile plots) that have been passed to the application for presentation.
- void **showProfile ()**
Show the arbitrary line (profile) markup. --refer to refer to - refer to [enableProfileSelection](#) property and #displayMarkups property for details.
- void **hideProfile ()**
Hide the arbitrary line (profile) markup but note that if its PV changes it will reshew unless DisplayMarkups has been set to off - refer to [enableProfileSelection](#) property and #displayMarkups property for details.
- void **showArea1 ()**

Show the area1 markup - refer to [enableArea1Selection](#) property and #displayMarkups property for details.

- void [hideArea1](#) ()

Hide the area1 markup but note that if its PV changes it will reshown unless Display-Markups has been set to off - refer to [enableArea1Selection](#) property and #display-Markups property for details.

- void [setDisplayMarkupsOn](#) ()

Set markup display to on to show all markups that change either due to user or PV activity, even if their setDisplay????Selection is off - refer to #displayMarkups property for details.

- void [setDisplayMarkupsOff](#) ()

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

Signals

- void [dbValueChanged](#) (const QString &out)
- void [requestResend](#) ()

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

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

Public Member Functions

- [QEImage](#) (QWidget *parent=0)

- [QEImage](#) (const QString &variableName, QWidget *parent=0)

- [~QEImage](#) ()

Destructor.

- [selectOptions](#) [getSelectionOption](#) ()

- void [setBitDepth](#) (unsigned int bitDepthIn)

Access function for #bitDepth property - refer to #bitDepth property for details.

- unsigned int [getBitDepth](#) ()

Access function for #bitDepth property - refer to #bitDepth property for details.

- void [setFormatOption](#) (imageDataFormats::formatOptions formatOption)

Access function for formatOption property - refer to formatOption property for details.

- imageDataFormats::formatOptions [getFormatOption](#) ()

Access function for formatOption property - refer to formatOption property for details.

- void [setResizeOption](#) (resizeOptions resizeModeIn)

Access function for #resizeOption property - refer to #resizeOption property for details.

- [resizeOptions](#) [getResizeOption](#) ()

Access function for #resizeOption property - refer to #resizeOption property for details.

- void [setZoom](#) (int zoomIn)

Access function for zoom property - refer to zoom property for details.

- int [getZoom](#) ()

- **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` ()

- 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 setName (QString nameIn)`
Access function for `name` property - refer to `#name` property for details.
 - `QString getName ()`
Access function for `name` property - refer to `#name` property for details.
 - `void setProgram1 (QString program)`
Access function for `program1` property - refer to `program1` property for details.
 - `QString getProgram1 ()`
Access function for `program1` property - refer to `program1` property for details.
 - `void setProgram2 (QString program)`
Access function for `program2` property - refer to `program2` property for details.
 - `QString getProgram2 ()`
Access function for `program2` property - refer to `program2` property for details.

- void `setArguments1` (QStringList arguments)
Access function for `arguments1` property - refer to `arguments1` property for details.
- QStringList `getArguments1` ()
Access function for `arguments1` property - refer to `arguments1` property for details.
- void `setArguments2` (QStringList arguments)
Access function for `arguments2` property - refer to `arguments2` property for details.
- QStringList `getArguments2` ()
Access function for `arguments2` property - refer to `arguments2` property for details.
- void `setProgramStartupOption1` (applicationLauncher::programStartupOptions programStartupOption)
Access function for `programStartupOption1` property - refer to `programStartupOption1` property for details.
- applicationLauncher::programStartupOptions `getProgramStartupOption1` ()
Access function for `programStartupOption1` property - refer to `programStartupOption1` property for details.
- void `setProgramStartupOption2` (applicationLauncher::programStartupOptions programStartupOption)
Access function for `programStartupOption2` property - refer to `programStartupOption2` property for details.
- applicationLauncher::programStartupOptions `getProgramStartupOption2` ()
Access function for `programStartupOption2` property - refer to `programStartupOption2` property for details.
- QString `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 data.
 - QString `getSubstitutedUrl` ()
Access function for `URL` property - refer to `URL` property for data.
 - void `setVariableNameSubstitutionsProperty` (QString variableNameSubstitutions)

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

- QString `getVariableNameSubstitutionsProperty` ()
Property access function for `variableSubstitutions` property. This has special behaviour to work well within designer.
- UserLevels `getUserLevelVisibilityProperty` ()

- Access function for `userLevelVisibility` property - refer to `userLevelVisibility` property for details.
- void `setUserLevelVisibilityProperty (UserLevels level)`
Access function for `userLevelVisibility` property - refer to `userLevelVisibility` property for details.
- `UserLevels getUserLevelEnabledProperty ()`
Access function for `userLevelEnabled` property - refer to `userLevelEnabled` property for details.
- void `setUserLevelEnabledProperty (UserLevels level)`
Access function for `userLevelEnabled` property - refer to `userLevelEnabled` property for details.
- `DisplayAlarmStateOptions getDisplayAlarmStateOptionProperty ()`
Access function for `displayAlarmStateOption` property - refer to `displayAlarmStateOption` property for details.
- void `setDisplayAlarmStateOptionProperty (DisplayAlarmStateOptions option)`
Access function for `displayAlarmStateOption` property - refer to `displayAlarmStateOption` property for details.
- void `setFormatOptionProperty (FormatOptions formatOption)`
Access function for `formatOption` property - refer to `formatOption` property for details.
- `FormatOptions getFormatOptionProperty ()`
Access function for `formatOption` property - refer to `formatOption` property for details.
- void `setBitDepthProperty (unsigned int bitDepth)`
Access function for `#bitDepth` property - refer to `#bitDepth` property for details.
- unsigned int `getBitDepthProperty ()`
Access function for `#bitDepth` property - refer to `#bitDepth` property for details.
- `EllipseVariableDefinitions getEllipseVariableDefinitionProperty ()`
Access function for `#EllipseVariableDefinition` property - refer to `#EllipseVariableDefinition` property for details.
- void `setEllipseVariableDefinitionProperty (EllipseVariableDefinitions variableUsage)`
Access function for `EllipseVariableDefinitions` property - refer to `EllipseVariableDefinitions` property for details.
- `TargetOptions getTargetOptionProperty ()`
Access function for `targetOption` property - refer to `targetOption` property for details.
- void `setTargetOptionProperty (TargetOptions option)`
Access function for `targetOption` property - refer to `targetOption` property for details.
- `TargetOptions getBeamOptionProperty ()`
Access function for `beamOption` property - refer to `beamOption` property for details.
- void `setBeamOptionProperty (TargetOptions option)`
Access function for `beamOption` property - refer to `beamOption` property for details.
- void `setResizeOptionProperty (ResizeOptions resizeOption)`
Access function for `#resizeOption` property - refer to `#resizeOption` property for details.
- `ResizeOptions getResizeOptionProperty ()`
Access function for `#resizeOption` property - refer to `#resizeOption` property for details.
- void `setRotationOptionProperty (RotationOptions rotation)`

- *Access function for #rotation property - refer to #rotation property for details.*
- [RotationOptions getRotationProperty \(\)](#)
 - *Access function for #rotation property - refer to #rotation property for details.*
- [void setProgramStartupOptionProperty1 \(ProgramStartupOptionNames programStartupOption\)](#)
 - *Access function for #ProgramStartupOptionNames1 property - refer to #ProgramStartupOptionNames1 property for details.*
- [ProgramStartupOptionNames getProgramStartupOptionProperty1 \(\)](#)
 - *Access function for #ProgramStartupOptionNames1 property - refer to #ProgramStartupOptionNames1 property for details.*
- [void setProgramStartupOptionProperty2 \(ProgramStartupOptionNames programStartupOption\)](#)
 - *Access function for #ProgramStartupOptionNames2 property - refer to #ProgramStartupOptionNames2 property for details.*
- [ProgramStartupOptionNames getProgramStartupOptionProperty2 \(\)](#)
 - *Access function for #ProgramStartupOptionNames2 property - refer to #ProgramStartupOptionNames2 property for details.*

Protected Types

- enum **variableIndexes** {
 - IMAGE_VARIABLE, FORMAT_VARIABLE, BIT_DEPTH_VARIABLE, WIDTH_VARIABLE,**
 - HEIGHT_VARIABLE, NUM_DIMENSIONS_VARIABLE, DIMENSION_0_VARIABLE, DIMENSION_1_VARIABLE,**
 - DIMENSION_2_VARIABLE, ROI1_X_VARIABLE, ROI1_Y_VARIABLE, ROI1_W_VARIABLE,**
 - ROI1_H_VARIABLE, ROI2_X_VARIABLE, ROI2_Y_VARIABLE, ROI2_W_VARIABLE,**
 - ROI2_H_VARIABLE, ROI3_X_VARIABLE, ROI3_Y_VARIABLE, ROI3_W_VARIABLE,**
 - ROI3_H_VARIABLE, ROI4_X_VARIABLE, ROI4_Y_VARIABLE, ROI4_W_VARIABLE,**
 - ROI4_H_VARIABLE, TARGET_X_VARIABLE, TARGET_Y_VARIABLE, BEAM_X_VARIABLE,**
 - BEAM_Y_VARIABLE, TARGET_TRIGGER_VARIABLE, CLIPPING_ONOFF_VARIABLE, CLIPPING_LOW_VARIABLE,**
 - CLIPPING_HIGH_VARIABLE, PROFILE_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

- QEStringFormatting **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`

- `DisplayAlarmStateOptions displayAlarmStateOption`
- `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 QEImage::DisplayAlarmStateOptions

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

Enumerator:

Never Refer to `DISPLAY_ALARM_STATE_NEVER` for details.

Always Refer to `DISPLAY_ALARM_STATE_ALWAYS` for details.

WhenInAlarm Refer to `DISPLAY_ALARM_STATE_WHEN_IN_ALARM` for details.

9.60.1.2 enum QEImage::ellipseVariableDefinitions

Options for the use of ellipse markup variables.

Enumerator:

BOUNDING_RECTANGLE Variables define bounding rectangle of ellipse.

9.60.1.3 enum QEImage::EllipseVariableDefinitions

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

Enumerator:

BoundingRectangle Refer to BOUNDING_RECTANGLE for details.

CenterAndSize Refer to CENTRE_AND_SIZE for details.

9.60.1.4 enum QEImage::FormatOptions

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

Enumerator:

Mono Grey scale.

Bayer Colour (Bayer Red Green)

BayerGB Colour (Bayer Green Blue)

BayerBG Colour (Bayer Blue Green)

BayerGR Colour (Bayer Green Red)

BayerRG Colour (Bayer Red Green)

rgb1 Colour (24 bit RGB)

rgb2 Colour (??? bit RGB)

rgb3 Colour (??? bit RGB)

yuv444 Colour (???)

yuv422 Colour (???)

9.60.1.5 enum QEImage::ProgramStartupOptionNames

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

Enumerator:

None Just run the program.

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

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

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

9.60.1.6 enum QEImage::ResizeOptions

User friendly enumerations for #resizeOption property

Enumerator:

Zoom Zoom to selected percentage.

Fit Zoom to fit the current window size.

9.60.1.7 enum QEImage::resizeOptions

Image resize options

Enumerator:

RESIZE_OPTION_ZOOM Zoom to selected percentage.

RESIZE_OPTION_FIT Zoom to fit the current window size.

9.60.1.8 enum QEImage::RotationOptions

User friendly enumerations for #rotation property

Enumerator:

NoRotation No image rotation.

Rotate90Right Rotate image 90 degrees clockwise.

Rotate90Left Rotate image 90 degrees anticlockwise.

Rotate180 Rotate image 180 degrees.

9.60.1.9 enum QEImage::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.

9.60.1.10 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.

9.60.1.11 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.

9.60.1.12 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 QEImage::QEImage (QWidget * *parent* = 0)

Create without a variable. Use `setVariableName'n'Property()` - where 'n' is a number from 0 to 40 - and `setSubstitutionsProperty()` to define variables and, optionally, macro

substitutions later. Note, each variable property is named by function (such as `imageVariable` and `widthVariable`) but given a numeric get and set property access function such as `setVariableName22Property()`. Refer to the property definitions to determine what 'set' and 'get' function is used for each variable, or use Qt library functions to set or get the variable names by name.

9.60.2.2 `QEImage::QEImage (const QString & variableName, QWidget * parent = 0)`

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

9.60.3 Member Function Documentation

9.60.3.1 `void QEImage::dbValueChanged (const QString & out) [signal]`

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

9.60.4 Member Data Documentation

9.60.4.1 `bool QEImage::displayButtonBar [read, write, protected]`

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

9.60.4.2 `int QEImage::initialVertScrollPos [read, write, protected]`

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

9.60.5 Property Documentation

9.60.5.1 `bool QEImage::allowDrop [read, write]`

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

9.60.5.2 `QColor QEImage::areaColor [read, write]`

Used to select the color of the area selection markups.

9.60.5.3 QStringList QEImage::arguments1 [read, write]

Arguments for program specified in the 'program1' property.

9.60.5.4 QStringList QEImage::arguments2 [read, write]

Arguments for program specified in the 'program2' property.

9.60.5.5 bool QEImage::autoBrightnessContrast [read, write]

If true, auto set local brightness and contrast when any area is selected. The brightness and contrast is set to use the full range of pixels in the selected area.

9.60.5.6 QColor QEImage::beamColor [read, write]

Used to select the color of the beam marker.

9.60.5.7 QString QEImage::beamXVariable [read, write]

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

9.60.5.8 QString QEImage::beamYVariable [read, write]

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

9.60.5.9 QString QEImage::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 QEImage::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 QEImage::displayAlarmState [read, write]

DEPRECATED. USE `displayAlarmStateOption` INSTEAD. If set (default) widget will indicate the alarm state of any variable data it is displaying. If clear widget will never indicate the alarm state of any variable data it is displaying. Typically the background colour is set to indicate the alarm state. Note, this property is included in the set of

standard properties as it applies to most widgets. It will do nothing for widgets that don't display data.

9.60.5.20 `DisplayAlarmStateOptions QEImage::displayAlarmStateOption` [read, write]

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

9.60.5.21 `bool QEImage::displayArea1Selection` [read, write]

If true, selected area 1 will be displayed on the image. Note, this property is ignored unless the [enableArea1Selection](#) property is true.

9.60.5.22 `bool QEImage::displayArea2Selection` [read, write]

If true, selected area 2 will be displayed on the image. Note, this property is ignored unless the [enableArea2Selection](#) property is true.

9.60.5.23 `bool QEImage::displayArea3Selection` [read, write]

If true, selected area 3 will be displayed on the image. Note, this property is ignored unless the [enableArea3Selection](#) property is true.

9.60.5.24 `bool QEImage::displayArea4Selection` [read, write]

If true, selected area 4 will be displayed on the image. Note, this property is ignored unless the [enableArea4Selection](#) property is true.

9.60.5.25 `bool QEImage::displayBeamSelection` [read, write]

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

9.60.5.26 `bool QEImage::displayCursorPixelInfo` [read, write]

If true, an area will be presented under the image with textual information about the pixel under the cursor, and for other selections such as selected areas.

9.60.5.27 `bool QEImage::displayEllipse [read, write]`

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

9.60.5.28 `bool QEImage::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.

9.60.5.29 `bool QEImage::displayProfileSelection [read, write]`

If true, the selected arbitrary line will be displayed on the image. Note, this property is ignored unless the [enableProfileSelection](#) property is true.

9.60.5.30 `bool QEImage::displayTargetSelection [read, write]`

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

9.60.5.31 `bool QEImage::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.

9.60.5.32 `QColor QEImage::ellipseColor [read, write]`

Used to select the color of the ellipse marker.

9.60.5.33 `QString QEImage::ellipseHVariable [read, write]`

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

9.60.5.34 `QString QEImage::ellipseWVariable [read, write]`

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

9.60.5.35 `QString QEImage::ellipseXVariable [read, write]`

EPICS variable name (CA PV). This variable is used to read an ellipse X (center or top left corner of bounding rectangle depending on property `ellipseDefinition`).

9.60.5.36 QString QEImage::ellipseYVariable [read, write]

EPICS variable name (CA PV). This variable is used to read an ellipse Y (center or top left corner of bounding rectangle depending on property `ellipseDefinition`).

9.60.5.37 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

9.60.5.38 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

9.60.5.39 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

9.60.5.40 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

9.60.5.41 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.

9.60.5.42 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` is true.

9.60.5.43 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.

9.60.5.44 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.

9.60.5.45 bool QEImage::enableVertSliceSelection [read, write]

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

9.60.5.46 bool QEImage::externalControls [read, write]

If true, image controls and views such as brightness controls and profile plots are hosted by the application as dock windows, toolbars, etc. Refer to the #ContainerProfile class and the #windowCustomisation class to see how this class asks an application to act as a host.

9.60.5.47 FormatOptions QEImage::formatOption [read, write]

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

9.60.5.48 QString QEImage::formatVariable [read, write]

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

9.60.5.49 QString QEImage::heightVariable [read, write]

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

9.60.5.50 bool QEImage::horizontalFlip [read, write]

If true, flip image horizontally.

9.60.5.51 QColor QEImage::hozSliceColor [read, write]

Used to select the color of the horizontal slice markup.

9.60.5.52 QString QEImage::imageVariable [read, write]

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

9.60.5.53 int QEImage::initialHosScrollPos [read, write]

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

9.60.5.54 unsigned QEImage::int [read, write]

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

Bit depth. Note, EPICS data type size will typically be adequate for the number of bits required (one byte for up to 8 bits, 2 bytes for up to 16 bits, etc), but can be larger (for example, 4 bytes for 24 bits) and may be larger than nessesary (4 bytes for 8 bits).

9.60.5.55 QString QEImage::lineProfileArrayVariable [read, write]

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

9.60.5.56 QString QEImage::lineProfileThicknessVariable [read, write]

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

9.60.5.57 QString QEImage::lineProfileX1Variable [read, write]

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

9.60.5.58 QString QEImage::lineProfileX2Variable [read, write]

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

9.60.5.59 QString QEImage::lineProfileY1Variable [read, write]

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

9.60.5.60 QString QEImage::lineProfileY2Variable [read, write]

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

9.60.5.61 bool QEImage::logBrightness [read, write]

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

9.60.5.62 QColor QEImage::profileColor [read, write]

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

9.60.5.63 QString QEImage::profileHozArrayVariable [read, write]

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

9.60.5.64 QString QEImage::profileHozThicknessVariable [read, write]

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

9.60.5.65 QString QEImage::profileHozVariable [read, write]

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

9.60.5.66 QString QEImage::profileVertArrayVariable [read, write]

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

9.60.5.67 QString QEImage::profileVertThicknessVariable [read, write]

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

9.60.5.68 QString QEImage::profileVertVariable [read, write]

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

9.60.5.69 QString QEImage::program1 [read, write]

Program to run when a request is made to pass on the current image to the first external application. No attempt to run a program is made if this property is empty. Example: paint.exe

9.60.5.70 QString QEImage::program2 [read, write]

Program to run when a request is made to pass on the current image to the second external application. No attempt to run a program is made if this property is empty. Example: paint.exe

9.60.5.71 ProgramStartupOptionNames QEImage::programStartupOption1 [read, write]

Startup options for the program specified in the 'program1' property. Just run the command, run the command within a terminal, or display the output in QE message system.

9.60.5.72 ProgramStartupOptionNames QEImage::programStartupOption2 [read, write]

Startup options for the program specified in the 'program2' property. Just run the command, run the command within a terminal, or display the output in QE message system.

9.60.5.73 QString QEImage::regionOfInterest1HVariable [read, write]

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

9.60.5.74 QString QEImage::regionOfInterest1WVariable [read, write]

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

9.60.5.75 QString QEImage::regionOfInterest1XVariable [read, write]

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

9.60.5.76 QString QEImage::regionOfInterest1YVariable [read, write]

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

9.60.5.77 QString QEImage::regionOfInterest2HVariable [read, write]

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

9.60.5.78 QString QEImage::regionOfInterest2WVariable [read, write]

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

9.60.5.79 QString QEImage::regionOfInterest2XVariable [read, write]

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

9.60.5.80 QString QEImage::regionOfInterest2YVariable [read, write]

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

9.60.5.81 QString QEImage::regionOfInterest3HVariable [read, write]

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

9.60.5.82 QString QEImage::regionOfInterest3WVariable [read, write]

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

9.60.5.83 QString QEImage::regionOfInterest3XVariable [read, write]

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

9.60.5.84 QString QEImage::regionOfInterest3YVariable [read, write]

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

9.60.5.85 QString QEImage::regionOfInterest4HVariable [read, write]

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

9.60.5.86 QString QEImage::regionOfInterest4WVariable [read, write]

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

9.60.5.87 QString QEImage::regionOfInterest4XVariable [read, write]

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

9.60.5.88 QString QEImage::regionOfInterest4YVariable [read, write]

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

9.60.5.89 ResizeOptions QEImage::resizeOption [read, write]

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

9.60.5.90 RotationOptions QEImage::rotation [read, write]

Image rotation option.

9.60.5.91 bool QEImage::showTime [read, write]

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

9.60.5.92 QColor QEImage::targetColor [read, write]

Used to select the color of the target marker.

9.60.5.93 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.

9.60.5.94 QString QEImage::targetXVariable [read, write]

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

9.60.5.95 QString QEImage::targetYVariable [read, write]

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

9.60.5.96 QColor QEImage::timeColor [read, write]

Used to select the color of the timestamp.

9.60.5.97 QString QEImage::URL [read, write]

MPEG stream URL. If this is specified, this will be used as the source of the image in preference to variables (variables defining the image data, width, and height will be ignored)

9.60.5.98 bool QEImage::useFalseColour [read, write]

If true, the apply false colour to the image.

9.60.5.99 UserLevels QEImage::userLevelEnabled [read, write]

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

9.60.5.100 QString QEImage::userLevelEngineerStyle [read, write]

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

9.60.5.101 QString QEImage::userLevelScientistStyle [read, write]

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

9.60.5.102 QString QEImage::userLevelUserStyle [read, write]

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

9.60.5.103 UserLevels QEImage::userLevelVisibility [read, write]

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

9.60.5.104 bool QEImage::variableAsToolTip [read, write]

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

9.60.5.105 QString QEImage::variableSubstitutions [read, write]

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

9.60.5.106 bool QEImage::verticalFlip [read, write]

If true, flip image vertically.

9.60.5.107 QColor QEImage::vertSliceColor [read, write]

Used to select the color of the vertical slice markup.

9.60.5.108 bool QEImage::visible [read, write]

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

9.60.5.109 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 `DisplayAlarmStateOptions` { `Never` = standardProperties::DISPLAY_ALARM_STATE_NEVER, `Always` = standardProperties::DISPLAY_ALARM_STATE_ALWAYS, `WhenInAlarm` = standardProperties::DISPLAY_ALARM_STATE_WHEN_IN_ALARM }
- enum `Formats` {
 `Default` = QEStringFormatting::FORMAT_DEFAULT, `Floating` = QEStringFormatting::FORMAT_FLOATING, `Integer` = QEStringFormatting::FORMAT_INTEGER, `UnsignedInteger` = QEStringFormatting::FORMAT_UNSIGNEDINTEGER,
 `Time` = QEStringFormatting::FORMAT_TIME, `LocalEnumeration` = QEStringFormatting::FORMAT_LOCAL_ENUMERATE }
- enum `Notations` { `Fixed` = QEStringFormatting::NOTATION_FIXED, `Scientific` = QEStringFormatting::NOTATION_SCIENTIFIC, `Automatic` = QEStringFormatting::NOTATION_AUTOMATIC }
- enum `ArrayActions` { `Append` = QEStringFormatting::APPEND, `Ascii` = QEStringFormatting::ASCII, `Index` = QEStringFormatting::INDEX }
- enum `UpdateOptions` { `Text` = QELabel::UPDATE_TEXT, `Picture` = QELabel::UPDATE_PIXMAP }

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

Public Slots

- void `setDefaultStyle` (const QString &style)

Update the default style applied to this widget.

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)`
• `void setVariableNameProperty (QString variableName)`
Property access function for `variable` property. This has special behaviour to work well within designer.
- `QString getVariableNameProperty ()`
Property access function for `variable` property. This has special behaviour to work well within designer.
- `void setVariableNameSubstitutionsProperty (QString variableNameSubstitutions)`
Property access function for `variableSubstitutions` property. This has special behaviour to work well within designer.
- `QString getVariableNameSubstitutionsProperty ()`
Property access function for `variableSubstitutions` property. This has special behaviour to work well within designer.
- `UserLevels getUserLevelVisibilityProperty ()`
Access function for `userLevelVisibility` property - refer to `userLevelVisibility` property for details.
- `void setUserLevelVisibilityProperty (UserLevels level)`
Access function for `userLevelVisibility` property - refer to `userLevelVisibility` property for details.
- `UserLevels getUserLevelEnabledProperty ()`
Access function for `userLevelEnabled` property - refer to `userLevelEnabled` property for details.
- `void setUserLevelEnabledProperty (UserLevels level)`
Access function for `userLevelEnabled` property - refer to `userLevelEnabled` property for details.
- `DisplayAlarmStateOptions getDisplayAlarmStateOptionProperty ()`
Access function for `displayAlarmStateOption` property - refer to `displayAlarmStateOption` property for details.
- `void setDisplayAlarmStateOptionProperty (DisplayAlarmStateOptions option)`
Access function for `displayAlarmStateOption` property - refer to `displayAlarmStateOption` property for details.
- `void setFormatProperty (Formats format)`
Access function for `format` property - refer to `format` property for details.
- `Formats getFormatProperty ()`
Access function for `format` property - refer to `format` property for details.
- `void 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](#)
- [DisplayAlarmStateOptions displayAlarmStateOption](#)
- 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 `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.

9.63.2.2 enum QELabel::DisplayAlarmStateOptions

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

Enumerator:

Never Refer to `DISPLAY_ALARM_STATE_NEVER` for details.

Always Refer to `DISPLAY_ALARM_STATE_ALWAYS` for details.

WhenInAlarm Refer to `DISPLAY_ALARM_STATE_WHEN_IN_ALARM` for details.

9.63.2.3 enum QELabel::Formats

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

Enumerator:

Default Format as best appropriate for the data type.

Floating Format as a floating point number.

Integer Format as an integer.

UnsignedInteger Format as an unsigned integer.

Time Format as a time.

LocalEnumeration Format as a selection from the `localEnumeration` property.

9.63.2.4 enum QELabel::Notations

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

Enumerator:

Fixed Refer to `QQStringFormatting::NOTATION_FIXED` for details.

Scientific Refer to `QQStringFormatting::NOTATION_SCIENTIFIC` for details.

Automatic Refer to `QQStringFormatting::NOTATION_AUTOMATIC` for details.

9.63.2.5 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::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.

9.63.2.7 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]

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

9.63.5.5 DisplayAlarmStateOptions QELabel::displayAlarmStateOption [read, write]

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

9.63.5.6 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).

9.63.5.7 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.

9.63.5.8 bool QELabel::leadingZero [read, write]

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

9.63.5.9 QString QELabel::localEnumeration [read, write]

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

Format is:

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

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

Values may be numeric or textual Values do not have to be in any order, but first match wins Values may be quoted Strings may be quoted Consecutive values do not have to

be present. Operator is assumed to be equality if not present. White space is ignored except within quoted strings.

may be included in a string to indicate a line break

Examples are:

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

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

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

9.63.5.10 Notations QELabel::notation [read, write]

Notation used for numerical formatting. Default is fixed.

9.63.5.11 QPixmap QELabel:: pixmap0 [read, write]

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

9.63.5.12 QPixmap QELabel:: pixmap1 [read, write]

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

9.63.5.13 QPixmap QELabel:: pixmap2 [read, write]

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

9.63.5.14 QPixmap QELabel:: pixmap3 [read, write]

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

9.63.5.15 QPixmap QELabel:: pixmap4 [read, write]

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

9.63.5.16 QPixmap QELabel::pixmap5 [read, write]

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

9.63.5.17 QPixmap QELabel::pixmap6 [read, write]

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

9.63.5.18 QPixmap QELabel::pixmap7 [read, write]

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

9.63.5.19 int QELabel::precision [read, write]

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

9.63.5.20 bool QELabel::trailingZeros [read, write]

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

9.63.5.21 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.

9.63.5.22 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.

9.63.5.23 UserLevels QELabel::userLevelEnabled [read, write]

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

Widgets that are only accessible to engineers maintaining the facility should be visible at 'Engineer'.

9.63.5.24 `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.

9.63.5.25 `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.

9.63.5.26 `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.

9.63.5.27 `UserLevels QELabel::userLevelVisibility` [read, write]

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

9.63.5.28 `QString QELabel::variable` [read, write]

EPICS variable name (CA PV)

9.63.5.29 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.

9.63.5.30 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.

9.63.5.31 bool QELabel::visible [read, write]

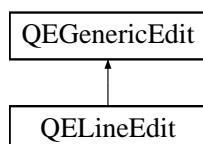
Display the widget. Default is true. Setting this property false is useful if widget is only used to provide a signal - for example, when supplying data to a [QELink](#) widget. Note, when false the widget will still be visible in Qt Designer.

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](#) = [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` = 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.
- `QELLineEdit` (QWidget *parent=0)
- `QELLineEdit` (const QString &variableName, QWidget *parent=0)

Properties

- int `precision`
- bool `useDbPrecision`
- bool `leadingZero`
- bool `trailingZeros`
- bool `addUnits`
- QString `localEnumeration`
- `Formats format`
- unsigned `int`
- `Notations notation`
- `ArrayActions arrayAction`

9.64.1 Member Enumeration Documentation

9.64.1.1 enum QELlineEdit::ArrayActions

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

Enumerator:

Append Refer to QEStringFormatting::APPEND for details.

Ascii Refer to QEStringFormatting::ASCII for details.

Index Refer to QEStringFormatting::INDEX for details.

9.64.1.2 enum QELlineEdit::Formats

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

Enumerator:

Default Format as best appropriate for the data type.

Floating Format as a floating point number.

Integer Format as an integer.

UnsignedInteger Format as an unsigned integer.

Time Format as a time.

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

9.64.1.3 enum QELlineEdit::Notations

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

Enumerator:

Fixed Refer to QEStringFormatting::NOTATION_FIXED for details.

Scientific Refer to QEStringFormatting::NOTATION_SCIENTIFIC for details.

Automatic Refer to QEStringFormatting::NOTATION_AUTOMATIC for details.

9.64.2 Constructor & Destructor Documentation

9.64.2.1 QELlineEdit::QELlineEdit (QWidget * parent = 0)

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

9.64.2.2 QELineEdit::QELineEdit (const QString & *variableName*, QWidget * *parent* = 0)

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

9.64.3 Member Function Documentation**9.64.3.1 void QELineEdit::dbValueChanged (const QString & *out*) [signal]**

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

9.64.4 Property Documentation**9.64.4.1 bool QELineEdit::addUnits [read, write]**

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

9.64.4.2 ArrayActions QELineEdit::arrayAction [read, write]

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

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

9.64.4.3 Formats QELineEdit::format [read, write]

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

9.64.4.4 unsigned QELineEdit::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 QELineEdit::leadingZero [read, write]

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

9.64.4.6 QString QELineEdit::localEnumeration [read, write]

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

Format is:

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

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

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

may be included in a string to indicate a line break

Examples are:

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

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

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

9.64.4.7 Notations QELineEdit::notation [read, write]

Notation used for numerical formatting. Default is fixed.

9.64.4.8 int QELineEdit::precision [read, write]

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

9.64.4.9 bool QELineEdit::trailingZeros [read, write]

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

9.64.4.10 bool QELineEdit::useDbPrecision [read, write]

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

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

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

9.65 QELineEditManager Class Reference

Public Member Functions

- **QELineEditManager** (QObject *parent=0)
- bool **isContainer** () const
- bool **isInitialized** () const
- QIcon **icon** () const
- QString **group** () const
- QString **includeFile** () const
- QString **name** () const
- QString **toolTip** () const
- QString **whatsThis** () const
- QWidget * **createWidget** (QWidget *parent)
- void **initialize** (QDesignerFormEditorInterface *core)

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

- /tmp/epicsqt/trunk/framework/widgets/QELineEdit/QELineEditManager.h

9.66 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 }
- enum **DisplayAlarmStateOptions** { **Never** = standardProperties::DISPLAY_ALARM_STATE_NEVER, **Always** = standardProperties::DISPLAY_ALARM_STATE_ALWAYS, **WhenInAlarm** = standardProperties::DISPLAY_ALARM_STATE_WHEN_IN_ALARM }

Public Member Functions

- **QELog** (QWidget *pParent=0)
- void **setShowColumnType** (bool pValue)
- bool **getShowColumnType** ()
- void **setShowColumnTime** (bool pValue)
- bool **getShowColumnTime** ()

- void **setShowColumnMessage** (bool pValue)
- bool **getShowColumnMessage** ()
- void **setShowMessageFilter** (bool pValue)
- bool **getShowMessageFilter** ()
- void **setShowClear** (bool pValue)
- bool **getShowClear** ()
- void **setShowSave** (bool pValue)
- bool **getShowSave** ()
- void **setOptionsLayout** (int pValue)
- int **getOptionsLayout** ()
- void **setScrollToBottom** (bool pValue)
- bool **getScrollToBottom** ()
- void **setInfoColor** (QColor pValue)
- QColor **getInfoColor** ()
- void **setWarningColor** (QColor pValue)
- QColor **getWarningColor** ()
- void **setErrorColor** (QColor pValue)
- QColor **getErrorColor** ()
- void **clearLog** ()
- void **addLog** (int pType, QString pMessage)
- void **refreshLog** ()
- void **setOptionsLayoutProperty** (optionsLayoutProperty pOptionsLayout)
- optionsLayoutProperty **getOptionsLayoutProperty** ()
- MessageFilterOptions **getMessageFormFilter** ()
- void **setMessageFormFilter** (MessageFilterOptions messageFormFilter)
- MessageFilterOptions **getMessageSourceFilter** ()
- void **setMessageSourceFilter** (MessageFilterOptions messageSourceFilter)
- **UserLevels getUserLevelVisibilityProperty** ()
Access function for userLevelVisibility property - refer to [userLevelVisibility](#) property for details.
- void **setUserLevelVisibilityProperty** (**UserLevels** level)
Access function for userLevelVisibility property - refer to [userLevelVisibility](#) property for details.
- **UserLevels getUserLevelEnabledProperty** ()
Access function for userLevelEnabled property - refer to [userLevelEnabled](#) property for details.
- void **setUserLevelEnabledProperty** (**UserLevels** level)
Access function for userLevelEnabled property - refer to [userLevelEnabled](#) property for details.
- **DisplayAlarmStateOptions getDisplayAlarmStateOptionProperty** ()
Access function for displayAlarmStateOption property - refer to [displayAlarmStateOption](#) property for details.
- void **setDisplayAlarmStateOptionProperty** (**DisplayAlarmStateOptions** option)
Access function for displayAlarmStateOption property - refer to [displayAlarmStateOption](#) property for details.

Protected Attributes

- `_QTableWidgetLog * qTableWidgetLog`
- `QCheckBox * qCheckBoxInfoMessage`
- `QCheckBox * qCheckBoxWarningMessage`
- `QCheckBox * qCheckBoxErrorMessage`
- `QPushButton * qPushButtonClear`
- `QPushButton * qPushButtonSave`
- `QColor qColorInfo`
- `QColor qColorWarning`
- `QColor qColorError`
- `bool scrollToBottom`
- `int optionsLayout`

Properties

- `bool showColumnType`
- `bool showColumnTime`
- `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`
- `DisplayAlarmStateOptions displayAlarmStateOption`

9.67.1 Member Enumeration Documentation

9.67.1.1 enum QELog::DisplayAlarmStateOptions

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

Enumerator:

Never Refer to DISPLAY_ALARM_STATE_NEVER for details.

Always Refer to DISPLAY_ALARM_STATE_ALWAYS for details.

WhenInAlarm Refer to DISPLAY_ALARM_STATE_WHEN_IN_ALARM for details.

9.67.1.2 enum QELog::UserLevels

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

Enumerator:

User Refer to USERLEVEL_USER for details.

Scientist Refer to USERLEVEL_SCIENTIST for details.

Engineer Refer to USERLEVEL_ENGINEER for details.

9.67.2 Property Documentation

9.67.2.1 bool QELog::allowDrop [read, write]

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

9.67.2.2 bool QELog::displayAlarmState [read, write]

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

9.67.2.3 DisplayAlarmStateOptions QELog::displayAlarmStateOption [read, write]

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

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

9.67.2.4 `unsigned QELog::int` [read, write]

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

9.67.2.5 `UserLevels QELog::userLevelEnabled` [read, write]

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

9.67.2.6 `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.

9.67.2.7 `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.

9.67.2.8 `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.

9.67.2.9 **UserLevels** QELog::userLevelVisibility [read, write]

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

9.67.2.10 **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.

9.67.2.11 **bool** QELog::visible [read, write]

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

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

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

9.68 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:

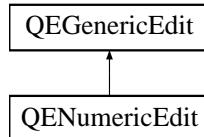
- `/tmp/epicsqt/trunk/framework/widgets/QELogin/QELogin.h`
- `/tmp/epicsqt/trunk/framework/widgets/QELogin/QELogin.cpp`

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 specifically 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 QEFixedPointRadix::Radicies value)
- QEFixedPointRadix::Radicies `getRadix` ()
- void `setSeparator` (const QEFixedPointRadix::Separators value)
- QEFixedPointRadix::Separators `getSeparator` ()

Protected Member Functions

- void `keyPressEvent` (QKeyEvent *event)
- void `focusInEvent` (QFocusEvent *event)
- void `mouseReleaseEvent` (QMouseEvent *event)
- void `establishConnection` (unsigned int variableIndex)
- qcaobject::QCaObject * `createQcalItem` (unsigned int variableIndex)
- int `getPrecision` ()
- int `getLeadingZeros` ()
- double `getMinimum` ()
- double `getMaximum` ()
- int `maximumSignificance` ()
- int `getRadixValue` ()
- void `setValue` (const QVariant &value)
Sets the underlying QLineEdit widget to the given value.
- QVariant `getValue` ()
Gets the underlying value.
- bool `writeData` (const QVariant &value, QString &message)
Write the data to the channel.

Protected Attributes

- QEFloatingFormatting **floatingFormatting**

Properties

- bool **autoScale**
- QEFixedPointRadix::Radicies **radix**
Specify radix, default is Decimal.
- QEFixedPointRadix::Separators **separator**
Specify digit 'thousands' separator character, default is none.
- int **precision**
- int **leadingZeros**
- double **minimum**
- double **maximum**
- bool **addUnits**

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 specifically 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 QENumericEdit::leadingZeros [read, write]**

Specifies the number of leading zeros. This is only used if autoScale is false. Strictly speaking, this should be an unsigned int, but designer properties editor much 'nicer' with integers.

9.70.4.4 **double QENumericEdit::maximum [read, write]**

Specifies the maximum allowed value. This is only used if autoScale is false.

9.70.4.5 **double QENumericEdit::minimum [read, write]**

Specifies the mimimum allowed value. This is only used if autoScale is false.

9.70.4.6 **int QENumericEdit::precision [read, write]**

Precision used for the display and editing of numbers. The default is 4. This is only used if autoScale is false. Strictly speaking, this should be an unsigned int, but designer properties editor much 'nicer' with integers.

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

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

9.71 QENumericEditManager Class Reference

Public Member Functions

- **QENumericEditManager** (QObject *parent=0)
- bool **isContainer** () const
- bool **isInitialized** () const
- QIcon **icon** () const
- QString **group** () const
- QString **includeFile** () const
- QString **name** () const
- QString **toolTip** () const
- QString **whatsThis** () const
- QWidget * **createWidget** (QWidget *parent)
- void **initialize** (QDesignerFormEditorInterface *core)

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

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

9.72 QEPeriodic Class Reference

Classes

- struct [elementInfoStruct](#)
- struct [userInfoStructArray](#)

Public Types

- enum **variableTypes** {

 VARIABLE_TYPE_NUMBER, VARIABLE_TYPE_ATOMIC_WEIGHT, VARIABLE_TYPE_MELTING_POINT, VARIABLE_TYPE_BOILING_POINT,

 VARIABLE_TYPE_DENSITY, VARIABLE_TYPE_GROUP, VARIABLE_TYPE_IONIZATION_ENERGY, VARIABLE_TYPE_USER_VALUE_1,

 VARIABLE_TYPE_USER_VALUE_2 }
- enum **presentationOptions** { PRESENTATION_BUTTON_AND_LABEL, PRESENTATION_BUTTON_ONLY, PRESENTATION_LABEL_ONLY }
- enum **UserLevels** { User = userLevelTypes::USERLEVEL_USER, Scientist = userLevelTypes::USERLEVEL_SCIENTIST, Engineer = userLevelTypes::USERLEVEL_ENGINEER }
- enum **DisplayAlarmStateOptions** { Never = standardProperties::DISPLAY_ALARM_STATE_NEVER, Always = standardProperties::DISPLAY_ALARM_STATE_ALWAYS, WhenInAlarm = standardProperties::DISPLAY_ALARM_STATE_WHEN_IN_ALARM }

- enum **PresentationOptions** { **buttonAndLabel** = QEPeriodic::PRESENTATION_BUTTON_AND_LABEL, **buttonOnly** = QEPeriodic::PRESENTATION_BUTTON_ONLY, **labelOnly** = QEPeriodic::PRESENTATION_LABEL_ONLY }
- enum **VariableTypes** {
 Number = QEPeriodic::VARIABLE_TYPE_NUMBER, **atomicWeight** = QEPeriodic::VARIABLE_TYPE_ATOMIC_WEIGHT, **meltingPoint** = QEPeriodic::VARIABLE_TYPE_MELTING_POINT, **boilingPoint** = QEPeriodic::VARIABLE_TYPE_BOILING_POINT,
 density = QEPeriodic::VARIABLE_TYPE_DENSITY, **group** = QEPeriodic::VARIABLE_TYPE_GROUP, **ionizationEnergy** = QEPeriodic::VARIABLE_TYPE_IONIZATION_ENERGY, **userValue1** = QEPeriodic::VARIABLE_TYPE_USER_VALUE_1,
 userValue2 = QEPeriodic::VARIABLE_TYPE_USER_VALUE_2 }

Signals

- void **dbValueChanged** (const double &out)
- void **dbElementChanged** (const QString &out)
- void **requestResend** ()

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

Public Member Functions

- **QEPeriodic** (QWidget *parent=0)
- **QEPeriodic** (const QString &variableName, QWidget *parent=0)
- void **setSubscribe** (bool subscribe)
- bool **getSubscribe** ()
- void **setPresentationOption** (presentationOptions presentationOptionIn)
- presentationOptions **getPresentationOption** ()
- void **setVariableType1** (variableTypes variableType1In)
- variableTypes **getVariableType1** ()
- void **setVariableType2** (variableTypes variableType2In)
- variableTypes **getVariableType2** ()
- void **setVariableTolerance1** (double variableTolerance1In)
- double **getVariableTolerance1** ()
- void **setVariableTolerance2** (double variableTolerance2In)
- double **getVariableTolerance2** ()
- void **setUserInfo** (QString userInfo)
- QString **getUserInfo** ()
- void **setVariableNameSubstitutionsProperty** (QString variableNameSubstitutions)

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

- QString **getVariableNameSubstitutionsProperty** ()

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

- **UserLevels getUserLevelVisibilityProperty ()**
Access function for `userLevelVisibility` property - refer to `userLevelVisibility` property for details.
- **void setUserLevelVisibilityProperty (UserLevels level)**
Access function for `userLevelVisibility` property - refer to `userLevelVisibility` property for details.
- **UserLevels getUserLevelEnabledProperty ()**
Access function for `userLevelEnabled` property - refer to `userLevelEnabled` property for details.
- **void setUserLevelEnabledProperty (UserLevels level)**
Access function for `userLevelEnabled` property - refer to `userLevelEnabled` property for details.
- **DisplayAlarmStateOptions getDisplayAlarmStateOptionProperty ()**
Access function for `displayAlarmStateOption` property - refer to `displayAlarmStateOption` property for details.
- **void setDisplayAlarmStateOptionProperty (DisplayAlarmStateOptions option)**
Access function for `displayAlarmStateOption` property - refer to `displayAlarmStateOption` property for details.
- **void setPresentationOptionProperty (PresentationOptions presentationOption)**
 - **PresentationOptions getPresentationOptionProperty ()**
 - **void setVariableType1Property (VariableTypes variableType)**
 - **void setVariableType2Property (VariableTypes variableType)**
 - **VariableTypes getVariableType1Property ()**
 - **VariableTypes getVariableType2Property ()**

Public Attributes

- **userInfoStruct userInfo [NUM_ELEMENTS]**

Static Public Attributes

- static **elementInfoStruct elementInfo [NUM_ELEMENTS]**

Protected Member Functions

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

Protected Attributes

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

Properties

- QString **writeButtonVariable1**
- QString **writeButtonVariable2**
- QString **readbackLabelVariable1**
- QString **readbackLabelVariable2**
- QString **variableSubstitutions**
- bool **subscribe**
- bool **variableAsToolTip**
- bool **visible**
- unsigned **int**
- QString **userLevelUserStyle**
- QString **userLevelScientistStyle**
- QString **userLevelEngineerStyle**
- **UserLevels userLevelVisibility**
- **UserLevels userLevelEnabled**
- bool **displayAlarmState**
- **DisplayAlarmStateOptions displayAlarmStateOption**
- **PresentationOptions presentationOption**
- **VariableTypes variableType1**
- **VariableTypes variableType2**
- QString **userInfo**

9.72.1 Member Enumeration Documentation

9.72.1.1 enum QEPeriodic::DisplayAlarmStateOptions

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

Enumerator:

Never Refer to **DISPLAY_ALARM_STATE_NEVER** for details.

Always Refer to **DISPLAY_ALARM_STATE_ALWAYS** for details.

WhenInAlarm Refer to **DISPLAY_ALARM_STATE_WHEN_IN_ALARM** for details.

9.72.1.2 enum QEPeriodic::UserLevels

User friendly enumerations for `userLevelVisibility` and `userLevelEnabled` properties - refer to `userLevelVisibility` and `userLevelEnabled` properties and `userLevel` enumeration for details.

Enumerator:

User Refer to `USERLEVEL_USER` for details.

Scientist Refer to `USERLEVEL_SCIENTIST` for details.

Engineer Refer to `USERLEVEL_ENGINEER` for details.

9.72.2 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]

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

9.72.4.2 DisplayAlarmStateOptions QEPeriodic::displayAlarmStateOption [read, write]

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

9.72.4.3 unsigned QEPeriodic::int [read, write]

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

9.72.4.4 QString QEPeriodic::readbackLabelVariable1 [read, write]

EPICS variable name (CA PV). This variable is used to read the value to the first of two positioners to determine which (if any) element is currently selected.

9.72.4.5 QString QEPeriodic::readbackLabelVariable2 [read, write]

EPICS variable name (CA PV). This variable is used to read the value to the second of two positioners to determine which (if any) element is currently selected.

9.72.4.6 bool QEPeriodic::subscribe [read, write]

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

9.72.4.7 UserLevels QEPeriodic::userLevelEnabled [read, write]

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

9.72.4.8 QString QEPeriodic::userLevelEngineerStyle [read, write]

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

9.72.4.9 QString QEPeriodic::userLevelScientistStyle [read, write]

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

9.72.4.10 QString QEPeriodic::userLevelUserStyle [read, write]

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

9.72.4.11 UserLevels QEPeriodic::userLevelVisibility [read, write]

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

9.72.4.12 bool QEPeriodic::variableAsToolTip [read, write]

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

9.72.4.13 QString QEPeriodic::variableSubstitutions [read, write]

Macro substitutions. The default is no substitutions. The format is NAME1=VALUE1[,] NAME2=VALUE2... Values may be quoted strings. For example, 'SAMPLE=SAM1, NAME = "Ref foil"' These substitutions are applied to all the variable names.

9.72.4.14 bool QEPeriodic::visible [read, write]

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

9.72.4.15 QString QEPeriodic::writeButtonVariable1 [read, write]

EPICS variable name (CA PV). This variable is used to write a value to the first of two positioners that will position the select element.

9.72.4.16 QString QEPeriodic::writeButtonVariable2 [read, write]

EPICS variable name (CA PV). This variable is used to write a value to the second of two positioners that will position the select element.

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

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

9.73 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 **DisplayAlarmStateOptions** { **Never** = standardProperties::DISPLAY_ALARM_STATE_NEVER, **Always** = standardProperties::DISPLAY_ALARM_STATE_ALWAYS, **WhenInAlarm** = standardProperties::DISPLAY_ALARM_STATE_WHEN_IN_ALARM }
- enum **TraceStyles** { **Lines** = QwtPlotCurve::Lines, **Sticks** = QwtPlotCurve::Sticks, **Steps** = QwtPlotCurve::Steps, **Dots** = QwtPlotCurve::Dots }

Signals

- void **dbValueChanged** (const double &out)
- void **dbValueChanged** (const QVector< double > &out)

Public Member Functions

- **QEPlot** (QWidget *parent=0)
- **QEPlot** (const QString &variableName, QWidget *parent=0)
- QSize **sizeHint** () const
- void **setYMin** (double yMin)
- double **getYMin** ()
- void **setYMax** (double yMax)
- double **getYMax** ()
- void **setAutoScale** (bool autoScale)
- bool **getAutoScale** ()
- void **setAxisEnableX** (bool axisEnableXIn)
- bool **getAxisEnableX** ()
- void **setAxisEnableY** (bool axisEnableYIn)
- bool **getAxisEnableY** ()
- QString **getTitle** ()
- void **setBackgroundColor** (QColor backgroundColor)
- QColor **getBackgroundColor** ()
- void **setTraceStyle** (QwtPlotCurve::CurveStyle traceStyle, const unsigned int variableIndex)
- QwtPlotCurve::CurveStyle **getTraceStyle** (const unsigned int variableIndex)
- void **setTraceColor** (QColor traceColor, const unsigned int variableIndex)
- void **setTraceColor1** (QColor traceColor)
- void **setTraceColor2** (QColor traceColor)
- void **setTraceColor3** (QColor traceColor)
- void **setTraceColor4** (QColor traceColor)
- QColor **getTraceColor** (const unsigned int variableIndex)
- QColor **getTraceColor1** ()
- QColor **getTraceColor2** ()
- QColor **getTraceColor3** ()
- QColor **getTraceColor4** ()
- void **setTraceLegend1** (QString traceLegend)
- void **setTraceLegend2** (QString traceLegend)
- void **setTraceLegend3** (QString traceLegend)
- void **setTraceLegend4** (QString traceLegend)
- QString **getTraceLegend1** ()
- QString **getTraceLegend2** ()
- QString **getTraceLegend3** ()
- QString **getTraceLegend4** ()
- void **setXUnit** (QString xUnit)
- QString **getXUnit** ()
- void **setYUnit** (QString yUnit)
- QString **getYUnit** ()
- void **setGridEnableMajorX** (bool gridEnableMajorXIn)
- void **setGridEnableMajorY** (bool gridEnableMajorYIn)
- void **setGridEnableMinorX** (bool gridEnableMinorXIn)
- void **setGridEnableMinorY** (bool gridEnableMinorYIn)

- bool **getGridEnableMajorX** ()
- bool **getGridEnableMajorY** ()
- bool **getGridEnableMinorX** ()
- bool **getGridEnableMinorY** ()
- void **setGridMajorColor** (QColor gridMajorColorIn)
- void **setGridMinorColor** (QColor gridMinorColorIn)
- QColor **getGridMajorColor** ()
- QColor **getGridMinorColor** ()
- void **setXStart** (double xStart)
- double **getXStart** ()
- void **setXIncrement** (double xIncrement)
- double **getXIncrement** ()
- void **setTimeSpan** (unsigned int timeSpan)
- unsigned int **getTimeSpan** ()
- void **setTickRate** (unsigned int tickRate)
- unsigned int **getTickRate** ()
- void **setVariableNameSubstitutionsProperty** (QString variableNameSubstitutions)

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

- QString **getVariableNameSubstitutionsProperty** ()

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

- UserLevels **getUserLevelVisibilityProperty** ()

Access function for `userLevelVisibility` property - refer to `userLevelVisibility` property for details.

- void **setUserLevelVisibilityProperty** (UserLevels level)

Access function for `userLevelVisibility` property - refer to `userLevelVisibility` property for details.

- UserLevels **getUserLevelEnabledProperty** ()

Access function for `userLevelEnabled` property - refer to `userLevelEnabled` property for details.

- void **setUserLevelEnabledProperty** (UserLevels level)

Access function for `userLevelEnabled` property - refer to `userLevelEnabled` property for details.

- DisplayAlarmStateOptions **getDisplayAlarmStateOptionProperty** ()

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

- void **setDisplayAlarmStateOptionProperty** (DisplayAlarmStateOptions option)

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

- void **setTraceStyle1** (TraceStyles traceStyle)

- void **setTraceStyle2** (TraceStyles traceStyle)

- void **setTraceStyle3** (TraceStyles traceStyle)

- void **setTraceStyle4** (TraceStyles traceStyle)

- TraceStyles **getTraceStyle1** ()

- TraceStyles **getTraceStyle2** ()

- TraceStyles **getTraceStyle3** ()

- TraceStyles **getTraceStyle4** ()

Protected Member Functions

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

Protected Attributes

- QEFloatingFormatting **floatingFormatting**
- bool **localEnabled**
- bool **allowDrop**

Properties

- QString **variable1**
- QString **variable2**
- QString **variable3**
- QString **variable4**
- QString **variableSubstitutions**
- bool **variableAsToolTip**
- bool **visible**
- unsigned **int**
- QString **userLevelUserStyle**
- QString **userLevelScientistStyle**
- QString **userLevelEngineerStyle**
- **UserLevels userLevelVisibility**
- **UserLevels userLevelEnabled**
- bool **displayAlarmState**
- **DisplayAlarmStateOptions displayAlarmStateOption**
- QColor **traceColor1**
- QColor **traceColor2**
- QColor **traceColor3**
- QColor **traceColor4**
- TraceStyles **traceStyle1**
- TraceStyles **traceStyle2**
- TraceStyles **traceStyle3**
- TraceStyles **traceStyle4**
- QString **traceLegend1**
- QString **traceLegend2**
- QString **traceLegend3**

- `QString traceLegend4`
- `QString title`
- `QColor backgroundColor`
- `QString xUnit`
- `QString yUnit`

9.76.1 Member Enumeration Documentation

9.76.1.1 enum QEPlot::DisplayAlarmStateOptions

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

Enumerator:

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

9.76.1.2 enum QEPlot::UserLevels

User friendly enumerations for `userLevelVisibility` and `userLevelEnabled` properties - refer to `userLevelVisibility` and `userLevelEnabled` properties and `userLevel` enumeration for details.

Enumerator:

- User** Refer to `USERLEVEL_USER` for details.
Scientist Refer to `USERLEVEL_SCIENTIST` for details.
Engineer Refer to `USERLEVEL_ENGINEER` for details.

9.76.2 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]

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

9.76.4.2 DisplayAlarmStateOptions QEPlot::displayAlarmStateOption [read, write]

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

9.76.4.3 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.

9.76.4.4 UserLevels QEPlot::userLevelEnabled [read, write]

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

9.76.4.5 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.

9.76.4.6 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.

9.76.4.7 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.

9.76.4.8 UserLevels QEPlot::userLevelVisibility [read, write]

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

9.76.4.9 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.

9.76.4.10 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.

9.76.4.11 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.

9.76.4.12 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.

9.76.4.13 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.

9.76.4.14 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.

9.76.4.15 bool QEPlot::visible [read, write]

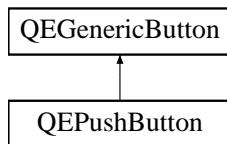
Display the widget. Default is true. Setting this property false is useful if widget is only used to provide a signal - for example, when supplying data to a [QELink](#) widget. Note, when false the widget will still be visible in Qt Designer.

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

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

9.77 QEPushButton Class Reference

Inheritance diagram for QEPushButton:



Public Types

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

`Default` = QEStringFormatting::FORMAT_DEFAULT, `Floating` = QEStringFormatting::FORMAT_FLOATING, `Integer` = QEStringFormatting::FORMAT_INTEGER, `UnsignedInteger` = QEStringFormatting::FORMAT_UNSIGNEDINTEGER,

`Time` = QEStringFormatting::FORMAT_TIME, `LocalEnumeration` = QEStringFormatting::FORMAT_LOCAL_ENUMERATE }
- enum `Notations` { `Fixed` = QEStringFormatting::NOTATION_FIXED, `Scientific` = QEStringFormatting::NOTATION_SCIENTIFIC, `Automatic` = QEStringFormatting::NOTATION_AUTOMATIC }
- enum `ArrayActions` { `Append` = QEStringFormatting::APPEND, `Ascii` = QEStringFormatting::ASCII, `Index` = QEStringFormatting::INDEX }
- enum `UpdateOptions` { `Text` = QEGenericButton::UPDATE_TEXT, `Icon` = QEGenericButton::UPDATE_ICON, `TextAndIcon` = QEGenericButton::UPDATE_TEXT_AND_ICON, `State` = QEGenericButton::UPDATE_STATE }

User friendly enumerations for updateOption property - refer to QEGenericButton::updateOptions for details.
- enum `ProgramStartupOptionNames` { `None` = applicationLauncher::PSO_NONE, `Terminal` = applicationLauncher::PSO_TERMINAL, `LogOutput` = applicationLauncher::PSO_LOGOUTPUT, `StdOutput` = applicationLauncher::PSO_STDOUPUT }
- enum `CreationOptionNames` {

`Open` = QEActionRequests::OptionOpen, `NewTab` = QEActionRequests::OptionNewTab, `NewWindow` = QEActionRequests::OptionNewWindow, `DockTop` = QEActionRequests::OptionTopDockWindow,

`DockBottom` = QEActionRequests::OptionBottomDockWindow, `DockLeft` = QEActionRequests::OptionLeftDockWindow, `DockRight` = QEActionRequests::OptionRightDockWindow, `DockTopTabbed` = QEActionRequests::OptionTopDockWindowTabbed, `DockBottomTabbed` = QEActionRequests::OptionBottomDockWindowTabbed, `DockLeftTabbed` = QEActionRequests::OptionLeftDockWindowTabbed, `DockRightTabbed` = QEActionRequests::OptionRightDockWindowTabbed, `DockFloating` = QEActionRequests::OptionFloatingDockWindow }

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

Public Slots

- void `requestAction` (const QEActionRequests &request)
- void `setDefaultStyle` (const QString &style)

Update the default style applied to this widget.

Signals

- void `dbValueChanged` (const QString &out)
- void `requestResend` ()

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

Internal use only. Request a new GUI is created. Typically, this is caught by the QEGui application.
- void `pressed` (int value)
- void `released` (int value)
- void `clicked` (int value)
- void `programCompleted` ()

Program started by button has completed.

Public Member Functions

- `QEPushButton` (QWidget *parent=0)
- `QEPushButton` (const QString &variableName, QWidget *parent=0)
- void `setVariableNameSubstitutionsProperty` (QString variableNameSubstitutions)

Property access function for `variableSubstitutions` property. This has special behaviour to work well within designer.
- QString `getVariableNameSubstitutionsProperty` ()

Property access function for `variableSubstitutions` property. This has special behaviour to work well within designer.
- UserLevels `getUserLevelVisibilityProperty` ()

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

Access function for `userLevelVisibility` property - refer to `userLevelVisibility` property for details.
- UserLevels `getUserLevelEnabledProperty` ()

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

Access function for `userLevelEnabled` property - refer to `userLevelEnabled` property for details.
- DisplayAlarmStateOptions `getDisplayAlarmStateOptionProperty` ()

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

Access function for `displayAlarmStateOption` property - refer to `displayAlarmStateOption` property for details.
- void `setFormatProperty` (Formats format)

Access function for `format` property - refer to `format` property for details.

- [Formats getFormatProperty \(\)](#)
Access function for `format` property - refer to `format` property for details.
- [void setNotationProperty \(Notations notation\)](#)
Access function for `notation` property - refer to `notation` property for details.
- [Notations getNotationProperty \(\)](#)
Access function for `notation` property - refer to `notation` property for details.
- [void setArrayActionProperty \(ArrayActions arrayAction\)](#)
Access function for `arrayAction` property - refer to `arrayAction` property for details.
- [ArrayActions getArrayActionProperty \(\)](#)
Access function for `arrayAction` property - refer to `arrayAction` property for details.

Properties

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

- QPixmap [pixmap6](#)
- QPixmap [pixmap7](#)
- QString [password](#)
- bool [confirmAction](#)
- QString [confirmText](#)
- bool [writeOnPress](#)
- bool [writeOnRelease](#)
- bool [writeOnClick](#)
- QString [pressText](#)
- QString [releaseText](#)
- QString [clickText](#)
- QString [clickCheckedText](#)
- QString [labelText](#)
- QString [program](#)
- QStringList [arguments](#)
- ProgramStartupOptionNames [programStartupOption](#)
- QString [guiFile](#)
- CreationOptionNames [creationOption](#)
- QString [prioritySubstitutions](#)
- QString [customisationName](#)

9.77.1 Member Enumeration Documentation

9.77.1.1 enum QEPushButton::ArrayActions

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

Enumerator:

Append Refer to QEStringFormatting::APPEND for details.

Ascii Refer to QEStringFormatting::ASCII for details.

Index Refer to QEStringFormatting::INDEX for details.

9.77.1.2 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::DisplayAlarmStateOptions

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

Enumerator:

Never Refer to DISPLAY_ALARM_STATE_NEVER for details.

Always Refer to DISPLAY_ALARM_STATE_ALWAYS for details.

WhenInAlarm Refer to DISPLAY_ALARM_STATE_WHEN_IN_ALARM for details.

9.77.1.4 enum QEPushButton::Formats

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

Enumerator:

Default Format as best appropriate for the data type.

Floating Format as a floating point number.

Integer Format as an integer.

UnsignedInteger Format as an unsigned integer.

Time Format as a time.

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

9.77.1.5 enum QEPushButton::Notations

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

Enumerator:

Fixed Refer to QEStringFormatting::NOTATION_FIXED for details.

Scientific Refer to QEStringFormatting::NOTATION_SCIENTIFIC for details.

Automatic Refer to QEStringFormatting::NOTATION_AUTOMATIC for details.

9.77.1.6 enum QEPushButton::ProgramStartupOptionNames

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

Enumerator:

None Just run the program.

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

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

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

9.77.1.7 enum QEPushButton::UpdateOptions

User friendly enumerations for updateOption property - refer to QEGenericButton::updateOptions for details.

Enumerator:

Text Data updates will update the button text.

Icon Data updates will update the button icon.

TextAndIcon Data updates will update the button text and icon.

State Data updates will update the button state (checked or unchecked)

9.77.1.8 enum QEPushButton::UserLevels

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

Enumerator:

User Refer to USERLEVEL_USER for details.

Scientist Refer to USERLEVEL_SCIENTIST for details.

Engineer Refer to USERLEVEL_ENGINEER for details.

9.77.2 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 `pressText` property

9.77.3.4 `void QEPushButton::released (int value) [signal]`

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

9.77.3.5 `void QEPushButton::requestAction (const QEActionRequests & request) [inline, slot]`

Default slot used to create a new GUI if there is no slot indicated in the ContainerProfile class. This slot is typically used when the button is pressed within the Designer preview window to allow the operation of the button to be tested. If an application does not specify a slot to use for creating new windows (through the ContainerProfile class) a

window will still be created through this slot, but it will not respect the window creation options or any other window related application constraints. For example, the QEGui application does provide a slot for creating new GUIs in the ContainerProfile class which respects the creation options, knows how to add tabs in the application, and extend the application's window menu in the menu bar.

9.77.4 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]

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

9.77.4.14 DisplayAlarmStateOptions QEPushButton::displayAlarmStateOption [read, write]

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

9.77.4.15 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).

9.77.4.16 QString QEPushButton::guiFile [read, write]

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

9.77.4.17 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.

9.77.4.18 `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](#).

9.77.4.19 `bool QEPushButton::leadingZero` [read, write]

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

9.77.4.20 `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"

9.77.4.21 Notations QEPushButton::notation [read, write]

Notation used for numerical formatting. Default is fixed.

9.77.4.22 QString QEPushButton::password [read, write]

Password user will need to enter before any action is taken

Reimplemented from [QEGenericButton](#).

9.77.4.23 QPixmap QEPushButton:: pixmap0 [read, write]

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

9.77.4.24 QPixmap QEPushButton:: pixmap1 [read, write]

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

9.77.4.25 QPixmap QEPushButton:: pixmap2 [read, write]

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

9.77.4.26 QPixmap QEPushButton:: pixmap3 [read, write]

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

9.77.4.27 QPixmap QEPushButton:: pixmap4 [read, write]

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

9.77.4.28 QPixmap QEPushButton:: pixmap5 [read, write]

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

9.77.4.29 QPixmap QEPushButton:: pixmap6 [read, write]

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

9.77.4.30 QPixmap QEPushButton:: pixmap7 [read, write]

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

9.77.4.31 int QEPushButton:: precision [read, write]

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

9.77.4.32 QString QEPushButton:: pressText [read, write]

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

Reimplemented from [QEGenericButton](#).

9.77.4.33 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](#).

9.77.4.34 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

9.77.4.35 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.

9.77.4.36 QString QEPushButton::releaseText [read, write]

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

Reimplemented from [QEGenericButton](#).

9.77.4.37 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)

9.77.4.38 bool QEPushButton::trailingZeros [read, write]

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

9.77.4.39 UpdateOptions QEPushButton::updateOption [read, write]

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

Reimplemented from [QEGenericButton](#).

9.77.4.40 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.

9.77.4.41 UserLevels QEPushButton::userLevelEnabled [read, write]

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

9.77.4.42 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.

9.77.4.43 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.

9.77.4.44 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.

9.77.4.45 UserLevels QEPushButton::userLevelVisibility [read, write]

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

9.77.4.46 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.

9.77.4.47 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.

9.77.4.48 QString QEPushButton::variableSubstitutions [read, write]

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

9.77.4.49 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.

9.77.4.50 bool QEPushButton::writeOnClick [read, write]

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

Reimplemented from [QEGenericButton](#).

9.77.4.51 bool QEPushButton::writeOnPress [read, write]

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

Reimplemented from [QEGenericButton](#).

9.77.4.52 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

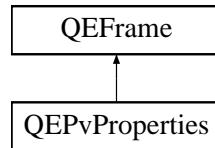
- void **prependOrMoveToFirst** (const QString &item)
- void **saveConfiguration** (PMElement &parentElement)
- void **restoreConfiguration** (PMElement &parentElement)

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

- void **setVariableNameProperty** (QString variableName)

Property access function for `variable` property. This has special behaviour to work well within designer.
- QString **getVariableNameProperty** ()

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

Property access function for `variableSubstitutions` property. This has special behaviour to work well within designer.
- QString **getVariableNameSubstitutionsProperty** ()

Property access function for `variableSubstitutions` property. This has special behaviour to work well within designer.
- **QEPvProperties** (QWidget *parent=0)
- **QEPvProperties** (const QString &variableName, QWidget *parent=0)
- QSize **sizeHint** () const

Protected Member Functions

- void **resizeEvent** (QResizeEvent *event)
- void **establishConnection** (unsigned int variableIndex)
- qcaobject::QCaObject * **createQcalItem** (unsigned int variableIndex)
- void **mousePressEvent** (QMouseEvent *event)
- void **dragEnterEvent** (QDragEnterEvent *event)
- void **dropEvent** (QDropEvent *event)
- void **saveConfiguration** (PersistanceManager *pm)
- void **restoreConfiguration** (PersistanceManager *pm, restorePhases restorePhase)
- QString **copyVariable** ()
- QVariant **copyData** ()
- void **paste** (QVariant s)

Properties

- `QString variable`
- `QString variableSubstitutions`

9.79.1 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

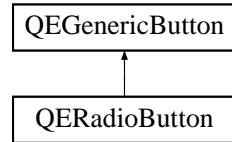
- **`QEPvPropertiesManager`** (`QObject *parent=0`)
- `bool isContainer () const`
- `bool isInitialized () const`
- `QIcon icon () const`
- `QString group () const`
- `QString includeFile () const`
- `QString name () const`
- `QString toolTip () const`
- `QString whatsThis () const`
- `QWidget * createWidget (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 `DisplayAlarmStateOptions` { `Never` = standardProperties::DISPLAY_ALARM_STATE_NEVER, `Always` = standardProperties::DISPLAY_ALARM_STATE_ALWAYS, `WhenInAlarm` = standardProperties::DISPLAY_ALARM_STATE_WHEN_IN_ALARM }
- enum `Formats` {
 `Default` = QEStringFormatting::FORMAT_DEFAULT, `Floating` = QEStringFormatting::FORMAT_FLOATING, `Integer` = QEStringFormatting::FORMAT_INTEGER, `UnsignedInteger` = QEStringFormatting::FORMAT_UNSIGNEDINTEGER,
 `Time` = QEStringFormatting::FORMAT_TIME, `LocalEnumeration` = QEStringFormatting::FORMAT_LOCAL_ENUMERATE }
- enum `Notations` { `Fixed` = QEStringFormatting::NOTATION_FIXED, `Scientific` = QEStringFormatting::NOTATION_SCIENTIFIC, `Automatic` = QEStringFormatting::NOTATION_AUTOMATIC }
- enum `ArrayActions` { `Append` = QEStringFormatting::APPEND, `Ascii` = QEStringFormatting::ASCII, `Index` = QEStringFormatting::INDEX }
- enum `UpdateOptions` { `Text` = QEGenericButton::UPDATE_TEXT, `Icon` = QEGenericButton::UPDATE_ICON, `TextAndIcon` = QEGenericButton::UPDATE_TEXT_AND_ICON, `State` = QEGenericButton::UPDATE_STATE }

User friendly enumerations for updateOption property - refer to QEGenericButton::updateOptions for details.
- enum `ProgramStartupOptionNames` { `None` = applicationLauncher::PSO_NONE, `Terminal` = applicationLauncher::PSO_TERMINAL, `LogOutput` = applicationLauncher::PSO_LOGOUTPUT, `StdOutput` = applicationLauncher::PSO_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
 }

```
= QEActionRequests::OptionRightDockWindowTabbed, DockFloating = QEAction-
Requests::OptionFloatingDockWindow }
```

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

Public Slots

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

Signals

- void `dbValueChanged` (const QString &out)
 - void `requestResend` ()
- Internal use only. Used when changing a property value to force a re-display to reflect the new property value.*
- void `newGui` (const QEActionRequests &request)
- Internal use only. Request a new GUI is created. Typically, this is caught by the QEGui application.*
- void `pressed` (int value)
 - void `released` (int value)
 - void `clicked` (int value)
 - void `programCompleted` ()
- Program started by button has completed.*

Public Member Functions

- `QERadioButton` (QWidget *parent=0)
 - `QERadioButton` (const QString &variableName, QWidget *parent=0)
 - void `setVariableNameProperty` (QString variableName)
- Property access function for `variable` property. This has special behaviour to work well within designer.*
- QString `getVariableNameProperty` ()
- Property access function for `variable` property. This has special behaviour to work well within designer.*
- void `setVariableNameSubstitutionsProperty` (QString variableNameSubstitutions)
- Property access function for `variableSubstitutions` property. This has special behaviour to work well within designer.*
- QString `getVariableNameSubstitutionsProperty` ()
- Property access function for `variableSubstitutions` property. This has special behaviour to work well within designer.*
- UserLevels `getUserLevelVisibilityProperty` ()

- Access function for `userLevelVisibility` property - refer to `userLevelVisibility` property for details.
- void `setUserLevelVisibilityProperty (UserLevels level)`
Access function for `userLevelVisibility` property - refer to `userLevelVisibility` property for details.
- `UserLevels getUserLevelEnabledProperty ()`
Access function for `userLevelEnabled` property - refer to `userLevelEnabled` property for details.
- void `setUserLevelEnabledProperty (UserLevels level)`
Access function for `userLevelEnabled` property - refer to `userLevelEnabled` property for details.
- `DisplayAlarmStateOptions getDisplayAlarmStateOptionProperty ()`
Access function for `displayAlarmStateOption` property - refer to `displayAlarmStateOption` property for details.
- void `setDisplayAlarmStateOptionProperty (DisplayAlarmStateOptions option)`
Access function for `displayAlarmStateOption` property - refer to `displayAlarmStateOption` property for details.
- void `setFormatProperty (Formats format)`
Access function for `format` property - refer to `format` property for details.
- `Formats getFormatProperty ()`
Access function for `format` property - refer to `format` property for details.
- void `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`

- `DisplayAlarmStateOptions displayAlarmStateOption`
- `int precision`
- `bool useDbPrecision`
- `bool leadingZero`
- `bool trailingZeros`
- `bool addUnits`
- `QString localEnumeration`
- `Formats format`
- `Notations notation`
- `ArrayActions arrayAction`
- `Qt::Alignment alignment`
- `UpdateOptions updateOption`
- `QPixmap pixmap0`
- `QPixmap pixmap1`
- `QPixmap pixmap2`
- `QPixmap pixmap3`
- `QPixmap pixmap4`
- `QPixmap pixmap5`
- `QPixmap pixmap6`
- `QPixmap pixmap7`
- `QString password`
- `bool confirmAction`
- `QString confirmText`
- `bool writeOnPress`
- `bool writeOnRelease`
- `bool writeOnClick`
- `QString pressText`
- `QString releaseText`
- `QString clickText`
- `QString clickCheckedText`
- `QString labelText`
- `QString program`
- `QStringList arguments`
- `ProgramStartupOptionNames programStartupOption`
- `QString guiFile`
- `CreationOptionNames creationOption`
- `QString prioritySubstitutions`
- `QString customisationName`

9.81.1 Member Enumeration Documentation

9.81.1.1 enum QERadioButton::ArrayActions

User friendly enumerations for arrayAction property - refer to QEStringFormatting::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.

9.81.1.2 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::DisplayAlarmStateOptions

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

Enumerator:

Never Refer to `DISPLAY_ALARM_STATE_NEVER` for details.

Always Refer to `DISPLAY_ALARM_STATE_ALWAYS` for details.

WhenInAlarm Refer to `DISPLAY_ALARM_STATE_WHEN_IN_ALARM` for details.

9.81.1.4 enum QERadioButton::Formats

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

Enumerator:

Default Format as best appropriate for the data type.

Floating Format as a floating point number.

Integer Format as an integer.

UnsignedInteger Format as an unsigned integer.

Time Format as a time.

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

9.81.1.5 enum QERadioButton::Notations

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

Enumerator:

Fixed Refer to QEStringFormatting::NOTATION_FIXED for details.

Scientific Refer to QEStringFormatting::NOTATION_SCIENTIFIC for details.

Automatic Refer to QEStringFormatting::NOTATION_AUTOMATIC for details.

9.81.1.6 enum QERadioButton::ProgramStartupOptionNames

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

Enumerator:

None Just run the program.

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

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

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

9.81.1.7 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)

9.81.1.8 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]

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

9.81.4.13 DisplayAlarmStateOptions QERadioButton::displayAlarmStateOption [read, write]

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

9.81.4.14 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).

9.81.4.15 `QString QERadioButton::guiFile [read, write]`

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

9.81.4.16 `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.

9.81.4.17 `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](#).

9.81.4.18 `bool QERadioButton::leadingZero [read, write]`

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

9.81.4.19 `QString QERadioButton::localEnumeration [read, write]`

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

Format is:

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

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

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

may be included in a string to indicate a line break

Examples are:

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

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

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

9.81.4.20 Notations QERadioButton::notation [read, write]

Notation used for numerical formatting. Default is fixed.

9.81.4.21 QString QERadioButton::password [read, write]

Password user will need to enter before any action is taken

Reimplemented from [QEGenericButton](#).

9.81.4.22 QPixmap QERadioButton:: pixmap0 [read, write]

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

9.81.4.23 QPixmap QERadioButton:: pixmap1 [read, write]

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

9.81.4.24 QPixmap QERadioButton:: pixmap2 [read, write]

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

9.81.4.25 QPixmap QERadioButton:: pixmap3 [read, write]

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

9.81.4.26 QPixmap QERadioButton:: pixmap4 [read, write]

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

9.81.4.27 QPixmap QERadioButton:: pixmap5 [read, write]

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

9.81.4.28 QPixmap QERadioButton:: pixmap6 [read, write]

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

9.81.4.29 QPixmap QERadioButton:: pixmap7 [read, write]

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

9.81.4.30 int QERadioButton:: precision [read, write]

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

9.81.4.31 QString QERadioButton:: pressText [read, write]

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

Reimplemented from [QEGenericButton](#).

9.81.4.32 QString QERadioButton:: prioritySubstitutions [read, write]

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

Reimplemented from [QEGenericButton](#).

9.81.4.33 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

9.81.4.34 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.

9.81.4.35 QString QERadioButton::releaseText [read, write]

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

Reimplemented from [QEGenericButton](#).

9.81.4.36 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)

9.81.4.37 bool QERadioButton::trailingZeros [read, write]

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

9.81.4.38 UpdateOptions QERadioButton::updateOption [read, write]

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

Reimplemented from [QEGenericButton](#).

9.81.4.39 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.

9.81.4.40 UserLevels QERadioButton::userLevelEnabled [read, write]

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

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

9.81.4.41 `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.

9.81.4.42 `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.

9.81.4.43 `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.

9.81.4.44 `UserLevels QERadioButton::userLevelVisibility` [read, write]

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

9.81.4.45 `QString QERadioButton::variable` [read, write]

EPICS variable name (CA PV)

9.81.4.46 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.

9.81.4.47 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.

9.81.4.48 bool QERadioButton::visible [read, write]

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

9.81.4.49 bool QERadioButton::writeOnClick [read, write]

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

Reimplemented from [QEGenericButton](#).

9.81.4.50 bool QERadioButton::writeOnPress [read, write]

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

Reimplemented from [QEGenericButton](#).

9.81.4.51 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/QEReipe/QEReipe.h
- /tmp/epicsqt/trunk/framework/widgets/QEReipe/QEReipe.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 **pvNameIsValid** (const QString &pvName)
- static bool **extractPvName** (const QString &item, QString &pvName)

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

- /tmp/epicsqt/trunk/framework/widgets/QEPvProperties/QEPvPropertiesUtilities.h
- /tmp/epicsqt/trunk/framework/widgets/QEPvProperties/QEPvPropertiesUtilities.cpp

9.84 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 }
- enum **DisplayAlarmStateOptions** { **Never** = standardProperties::DISPLAY_ALARM_STATE_NEVER, **Always** = standardProperties::DISPLAY_ALARM_STATE_ALWAYS, **WhenInAlarm** = standardProperties::DISPLAY_ALARM_STATE_WHEN_IN_ALARM }

Signals

- void **selected** (QString pFilename)

Public Member Functions

- **QEScript** (QWidget *pParent=0)
- void **setShowScriptList** (bool pValue)
- bool **getShowScriptList** ()
- void **setShowNew** (bool pValue)
- bool **getShowNew** ()
- void **setShowSave** (bool pValue)
- bool **getShowSave** ()
- void **setShowDelete** (bool pValue)
- bool **getShowDelete** ()
- void **setShowExecute** (bool pValue)
- bool **getShowExecute** ()
- void **setShowAbort** (bool pValue)
- bool **getShowAbort** ()
- void **setEditableTable** (bool pValue)
- bool **getEditableTable** ()
- void **setShowTable** (bool pValue)
- bool **getShowTable** ()
- void **setShowTableControl** (bool pValue)
- bool **getShowTableControl** ()
- void **setShowColumnNumber** (bool pValue)
- bool **getShowColumnNumber** ()
- void **setShowColumnEnable** (bool pValue)
- bool **getShowColumnEnable** ()
- void **setShowColumnProgram** (bool pValue)
- bool **getShowColumnProgram** ()
- void **setShowColumnParameters** (bool pValue)
- bool **getShowColumnParameters** ()
- void **setShowColumnWorkingDirectory** (bool pValue)

- bool **getShowColumnWorkingDirectory** ()
- void **setShowColumnTimeout** (bool pValue)
- bool **getShowColumnTimeout** ()
- void **setShowColumnStop** (bool pValue)
- bool **getShowColumnStop** ()
- void **setShowColumnLog** (bool pValue)
- bool **getShowColumnLog** ()
- void **setScriptType** (int pValue)
- int **getScriptType** ()
- void **setScriptFile** (QString pValue)
- QString **getScriptFile** ()
- void **setScriptText** (QString pValue)
- QString **getScriptText** ()
- void **setScriptDefault** (QString pValue)
- QString **getScriptDefault** ()
- void **setExecuteText** (QString pValue)
- QString **getExecuteText** ()
- void **setOptionsLayout** (int pValue)
- int **getOptionsLayout** ()
- void **insertRow** (bool pEnable, QString pProgram, QString pParameter, QString pWorkingDirectory, int pTimeOut, bool pStop, bool pLog)
- bool **saveScriptList** ()
- void **refreshScriptList** ()
- void **refreshWidgets** ()
- void **setScriptTypeProperty** (scriptTypesProperty pScriptType)
- scriptTypesProperty **getScriptTypeProperty** ()
- void **setOptionsLayoutProperty** (optionsLayoutProperty pOptionsLayout)
- optionsLayoutProperty **getOptionsLayoutProperty** ()
- UserLevels **getUserLevelVisibilityProperty** ()
 - Access function for *userLevelVisibility* property - refer to *userLevelVisibility* property for details.
- void **setUserLevelVisibilityProperty** (UserLevels level)
 - Access function for *userLevelVisibility* property - refer to *userLevelVisibility* property for details.
- UserLevels **getUserLevelEnabledProperty** ()
 - Access function for *userLevelEnabled* property - refer to *userLevelEnabled* property for details.
- void **setUserLevelEnabledProperty** (UserLevels level)
 - Access function for *userLevelEnabled* property - refer to *userLevelEnabled* property for details.
- DisplayAlarmStateOptions **getDisplayAlarmStateOptionProperty** ()
 - Access function for *displayAlarmStateOption* property - refer to *displayAlarmStateOption* property for details.
- void **setDisplayAlarmStateOptionProperty** (DisplayAlarmStateOptions option)
 - Access function for *displayAlarmStateOption* property - refer to *displayAlarmStateOption* property for details.

Protected Attributes

- QComboBox * **qComboBoxScriptList**
- QPushButton * **qPushButtonNew**
- QPushButton * **qPushButtonSave**
- QPushButton * **qPushButtonDelete**
- QPushButton * **qPushButtonExecute**
- QPushButton * **qPushButtonAbort**
- QPushButton * **qPushButtonAdd**
- QPushButton * **qPushButtonRemove**
- QPushButton * **qPushButtonUp**
- QPushButton * **qPushButtonDown**
- QPushButton * **qPushButtonCopy**
- QPushButton * **qPushButtonPaste**
- **_QTableWidgetScript** * **qTableWidgetScript**
- QString **scriptFile**
Define the file where to save the scripts (if not defined then the scripts will be saved in a file named "QEScript.xml")
- QString **scriptText**
Define the XML text that contains the scripts.
- QString **scriptDefault**
Define the script (previously saved by the user) that will be load as the default script when the widget starts.
 - int **scriptType**
 - int **optionsLayout**
 - QDomDocument **document**
 - QString **filename**
 - QList< **_CopyPaste** * > **copyPasteList**
 - bool **editableTable**
Enable/disable table edition.
 - bool **isExecuting**

Properties

- bool **showScriptList**
Show/hide combobox that contains the list of existing scripts created by the user.
- bool **showNew**
Show/hide button to reset (initialize) the table that contains the sequence of programs to be executed.
- bool **showSave**
Show/hide button to save/overwrite a new/existing script.
- bool **showDelete**
Show/hide button to delete an existing script.
- bool **showExecute**
Show/hide button to execute a sequence of programs.

- bool `showAbort`
Show/hide button to abort the execution of a sequence of programs.
- bool `showTable`
Show/hide table that contains a sequence of programs to be executed.
- bool `showTableControl`
Show/hide the controls of the table that contains a sequence of programs to be executed.
- bool `showColumnNumber`
Show/hide the column '#' that displays the sequential number of programs.
- bool `showColumnEnable`
Show/hide the column 'Enable' that enables the execution of programs.
- bool `showColumnProgram`
Show/hide the column 'Program' that contains the external programs to be executed.
- bool `showColumnParameters`
Show/hide the column 'Parameters' that contains the parameters that are passed to external programs to be executed.
- bool `showColumnWorkingDirectory`
Show/hide the column 'Directory' that defines the working directory to be used when external programs are executed.
- bool `showColumnTimeout`
Show/hide the column 'Timeout' that defines a time out period in seconds (if equal to 0 then the program runs until it finishes; otherwise if greater than 0 then the program will only run during this amount of seconds and will be aborted beyond this time)
- bool `showColumnStop`
Show/hide the column 'Stop' that enables stopping the execution of subsequent programs when the current one exited with an error code different from 0.
- bool `showColumnLog`
Show/hide the column 'Log' that enables the generation of log messages (these messages may be displayed using the `QELog` widget)
- scriptTypesProperty `scriptType`
Select if the scripts are to be loaded/saved from an XML file or from an XML text.
- QString `executeText`
Define the caption of the button responsible for starting the execution of external programs (if not defined then the caption will be "Execute")
- optionsLayoutProperty `optionsLayout`
Change the order of the widgets. Valid orders are: TOP, BOTTOM, LEFT and RIG.
- bool `variableAsToolTip`
- bool `allowDrop`
- bool `visible`
- unsigned int
- QString `userLevelUserStyle`
- QString `userLevelScientistStyle`
- QString `userLevelEngineerStyle`
- UserLevels `userLevelVisibility`
- UserLevels `userLevelEnabled`
- bool `displayAlarmState`
- DisplayAlarmStateOptions `displayAlarmStateOption`

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::DisplayAlarmStateOptions

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

Enumerator:

Never Refer to DISPLAY_ALARM_STATE_NEVER for details.

Always Refer to DISPLAY_ALARM_STATE_ALWAYS for details.

WhenInAlarm Refer to DISPLAY_ALARM_STATE_WHEN_IN_ALARM for details.

9.86.2.2 enum QEScript::UserLevels

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

Enumerator:

User Refer to USERLEVEL_USER for details.

Scientist Refer to USERLEVEL_SCIENTIST for details.

Engineer Refer to USERLEVEL_ENGINEER for details.

9.86.3 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]

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

9.86.3.3 DisplayAlarmStateOptions `QEScript::displayAlarmStateOption` [read, write]

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

9.86.3.4 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.

9.86.3.5 UserLevels `QEScript::userLevelEnabled` [read, write]

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

9.86.3.6 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.

9.86.3.7 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.

9.86.3.8 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.

9.86.3.9 UserLevels QEScript::userLevelVisibility [read, write]

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

9.86.3.10 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.

9.86.3.11 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 }
- enum `DisplayAlarmStateOptions` { `Never` = standardProperties::DISPLAY_ALARM_STATE_NEVER, `Always` = standardProperties::DISPLAY_ALARM_STATE_ALWAYS, `WhenInAlarm` = standardProperties::DISPLAY_ALARM_STATE_WHEN_IN_ALARM }

Signals

- void `dbValueChanged1` (const qulonglong &out)
- void `dbValueChanged2` (const qulonglong &out)
- void `dbValueChanged3` (const qulonglong &out)
- void `dbValueChanged4` (const qulonglong &out)
- void `dbValueChanged5` (const qulonglong &out)
- void `dbValueChanged6` (const qulonglong &out)

Public Member Functions

- `QEShape` (QWidget *parent=0)
- `QEShape` (const QString &variableName, QWidget *parent=0)
- void `scaleBy` (const int m, const int d)

Scale the widgets my m/d.
- void `setAnimation` (`animationOptions` animation, const int index)

Access function for #animation' properties - refer to animation' properties for details.
- `animationOptions getAnimation` (const int index)

Access function for #animation' properties - refer to animation' properties for details.
- void `setScale` (const double scale, const int index)

Access function for #scale' properties - refer to scale' properties for details.
- double `getScale` (const int index)

Access function for #scale' properties - refer to scale' properties for details.
- void `setOffset` (const double offset, const int index)

Access function for #offset' properties - refer to offset' properties for details.
- double `getOffset` (const int index)

Access function for #offset' properties - refer to offset' properties for details.
- void `setBorder` (const bool border)

Access function for #border' properties - refer to border' properties for details.
- bool `getBorder` ()

Access function for #border' properties - refer to border' properties for details.
- void `setFill` (const bool fill)

- `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.
- `bool getDrawBorder ()`
Access function for #draw border' properties - refer to draw border' properties for details.
- `void setLineWidth (const unsigned int lineWidth)`
Access function for #line width' properties - refer to line width' properties for details.
- `unsigned int getLineWidth ()`
Access function for #line width' properties - refer to line width' properties for details.
- `void setStartAngle (const double startAngle)`
Access function for #start angle' properties - refer to start angle' properties for details.
- `double getStartAngle ()`
Access function for #start angle' properties - refer to start angle' properties for details.
- `void setRotation (const double rotation)`
Access function for #rotation' properties - refer to rotation' properties for details.
- `double getRotation ()`

Access function for #rotation' properties - refer to rotation' properties for details.

- void [setArcLength](#) (const double arcLength)

Access function for #arc length' properties - refer to arc length' properties for details.

- double [getArcLength](#) ()

Access function for #arc length' properties - refer to arc length' properties for details.

- void [setVariableNameSubstitutionsProperty](#) (QString variableNameSubstitutions)

Property access function for [variableSubstitutions](#) property. This has special behaviour to work well within designer.

- QString [getVariableNameSubstitutionsProperty](#) ()

Property access function for [variableSubstitutions](#) property. This has special behaviour to work well within designer.

- [UserLevels getUserLevelVisibilityProperty](#) ()

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

- void [setUserLevelVisibilityProperty](#) ([UserLevels](#) level)

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

- [UserLevels getUserLevelEnabledProperty](#) ()

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

- void [setUserLevelEnabledProperty](#) ([UserLevels](#) level)

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

- [DisplayAlarmStateOptions getDisplayAlarmStateOptionProperty](#) ()

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

- void [setDisplayAlarmStateOptionProperty](#) ([DisplayAlarmStateOptions](#) option)

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

Properties

- QString [variable1](#)
- QString [variable2](#)
- QString [variable3](#)
- QString [variable4](#)
- QString [variable5](#)
- QString [variable6](#)
- QString [variableSubstitutions](#)
- bool [variableAsToolTip](#)
- bool [allowDrop](#)
- bool [visible](#)
- unsigned [int](#)
- QString [userLevelUserStyle](#)

- `QString userLevelScientistStyle`
- `QString userLevelEngineerStyle`
- `UserLevels userLevelVisibility`
- `UserLevels userLevelEnabled`
- `bool displayAlarmState`
- `DisplayAlarmStateOptions displayAlarmStateOption`
- `animationOptions animation1`
- `animationOptions animation2`
- `animationOptions animation3`
- `animationOptions animation4`
- `animationOptions animation5`
- `animationOptions animation6`
- `double scale1`

Scale factor applied to data from the 1st variable before it is used to animate the shape.

- `double scale2`
- `double scale3`
- `double scale4`
- `double scale5`
- `double scale6`
- `double offset1`
- `double offset2`
- `double offset3`
- `double offset4`
- `double offset5`
- `double offset6`
- `QPoint point1`
- `QPoint point2`
- `QPoint point3`
- `QPoint point4`
- `QPoint point5`
- `QPoint point6`
- `QPoint point7`
- `QPoint point8`
- `QPoint point9`
- `QPoint point10`
- `QColor color1`
- `QColor color2`
- `QColor color3`
- `QColor color4`
- `QColor color5`
- `QColor color6`
- `QColor color7`
- `QColor color8`
- `QColor color9`
- `QColor color10`

9.87.1 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 vertical and horizontal size and position. It is tighly integrated with the base class QEWidget which provides generic support such as macro substitutions, drag/drop, and standard properties.

9.87.2 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::DisplayAlarmStateOptions

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

Enumerator:

Never Refer to DISPLAY_ALARM_STATE_NEVER for details.

Always Refer to DISPLAY_ALARM_STATE_ALWAYS for details.

WhenInAlarm Refer to DISPLAY_ALARM_STATE_WHEN_IN_ALARM for details.

9.87.2.3 enum QEShape::shapeOptions

Options for the type of shape.

9.87.2.4 enum QEShape::UserLevels

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

Enumerator:

User Refer to USERLEVEL_USER for details.

Scientist Refer to USERLEVEL_SCIENTIST for details.

Engineer Refer to USERLEVEL_ENGINEER for details.

9.87.3 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 qulonglong & out) [signal]`

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

9.87.4.2 `void QEShape::dbValueChanged2 (const qulonglong & out) [signal]`

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

9.87.4.3 `void QEShape::dbValueChanged3 (const qulonglong & out) [signal]`

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

9.87.4.4 `void QEShape::dbValueChanged4 (const qulonglong & out) [signal]`

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

9.87.4.5 `void QEShape::dbValueChanged5 (const qulonglong & out) [signal]`

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

9.87.4.6 void QEShape::dbValueChanged6 (const qulonglong & out) [signal]

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

9.87.5 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]

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

9.87.5.19 DisplayAlarmStateOptions QEShape::displayAlarmStateOption [read, write]

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

9.87.5.20 unsigned QEShape::int [read, write]

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

The number of points to use when drawing shapes that are defined by a variable number of points, such as polyline, polygon, path, and series of points.

Sets the width of the pen. Used for the following shapes: Line, Points, Polyline, Polygon, Rect, RoundedRectangle, Ellipse, Arc, Chord, Pie, Path

9.87.5.21 double QEShape::offset1 [read, write]

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

9.87.5.22 double QEShape::offset2 [read, write]

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

9.87.5.23 double QEShape::offset3 [read, write]

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

9.87.5.24 double QEShape::offset4 [read, write]

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

9.87.5.25 double QEShape::offset5 [read, write]

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

9.87.5.26 double QEShape::offset6 [read, write]

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

9.87.5.27 QPoint QEShape::point1 [read, write]

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

9.87.5.28 QPoint QEShape::point10 [read, write]

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

9.87.5.29 QPoint QEShape::point2 [read, write]

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

9.87.5.30 QPoint QEShape::point3 [read, write]

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

9.87.5.31 QPoint QEShape::point4 [read, write]

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

9.87.5.32 QPoint QEShape::point5 [read, write]

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

9.87.5.33 QPoint QEShape::point6 [read, write]

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

9.87.5.34 QPoint QEShape::point7 [read, write]

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

9.87.5.35 QPoint QEShape::point8 [read, write]

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

9.87.5.36 QPoint QEShape::point9 [read, write]

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

9.87.5.37 double QEShape::scale2 [read, write]

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

9.87.5.38 double QEShape::scale3 [read, write]

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

9.87.5.39 double QEShape::scale4 [read, write]

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

9.87.5.40 double QEShape::scale5 [read, write]

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

9.87.5.41 double QEShape::scale6 [read, write]

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

9.87.5.42 UserLevels QEShape::userLevelEnabled [read, write]

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

9.87.5.43 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.

9.87.5.44 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.

9.87.5.45 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.

9.87.5.46 UserLevels QEShape::userLevelVisibility [read, write]

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

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

9.87.5.47 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.

9.87.5.48 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.

9.87.5.49 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.

9.87.5.50 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.

9.87.5.51 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.

9.87.5.52 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.

9.87.5.53 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.

9.87.5.54 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.

9.87.5.55 bool QEShape::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/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 }
- enum [DisplayAlarmStateOptions](#) { [Never](#) = standardProperties::DISPLAY_ALARM_STATE_NEVER, [Always](#) = standardProperties::DISPLAY_ALARM_STATE_ALWAYS, [WhenInAlarm](#) = standardProperties::DISPLAY_ALARM_STATE_WHEN_IN_ALARM }

Public Slots

- void [setDefaultStyle](#) (const QString &style)
Update the default style applied to this widget.

Signals

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

Public Member Functions

- **QESlider** (QWidget *parent=0)
- **QESlider** (const QString &variableName, QWidget *parent=0)
- void **setWriteOnChange** (bool writeOnChange)
- bool **getWriteOnChange** ()
- void **setSubscribe** (bool subscribe)
- bool **getSubscribe** ()
- void **setScale** (double scaleIn)
- double **getScale** ()
- void **setOffset** (double offsetIn)
- double **getOffset** ()
- void **setVariableNameProperty** (QString variableName)
Property access function for `variable` property. This has special behaviour to work well within designer.
- QString **getVariableNameProperty** ()
Property access function for `variable` property. This has special behaviour to work well within designer.
- void **setVariableNameSubstitutionsProperty** (QString variableNameSubstitutions)
Property access function for `variableSubstitutions` property. This has special behaviour to work well within designer.
- QString **getVariableNameSubstitutionsProperty** ()
Property access function for `variableSubstitutions` property. This has special behaviour to work well within designer.
- UserLevels **getUserLevelVisibilityProperty** ()
Access function for `userLevelVisibility` property - refer to `userLevelVisibility` property for details.
- void **setUserLevelVisibilityProperty** (UserLevels level)
Access function for `userLevelVisibility` property - refer to `userLevelVisibility` property for details.
- UserLevels **getUserLevelEnabledProperty** ()
Access function for `userLevelEnabled` property - refer to `userLevelEnabled` property for details.
- void **setUserLevelEnabledProperty** (UserLevels level)
Access function for `userLevelEnabled` property - refer to `userLevelEnabled` property for details.
- DisplayAlarmStateOptions **getDisplayAlarmStateOptionProperty** ()
Access function for `displayAlarmStateOption` property - refer to `displayAlarmStateOption` property for details.
- void **setDisplayAlarmStateOptionProperty** (DisplayAlarmStateOptions option)
Access function for `displayAlarmStateOption` property - refer to `displayAlarmStateOption` property for details.

Protected Member Functions

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

Protected Attributes

- QEFloatingFormatting **floatingFormatting**
- bool **writeOnChange**

Properties

- QString **variable**
- QString **variableSubstitutions**
- bool **subscribe**
- bool **variableAsToolTip**
- bool **allowDrop**
- bool **visible**
- unsigned **int**
- QString **userLevelUserStyle**
- QString **userLevelScientistStyle**
- QString **userLevelEngineerStyle**
- UserLevels **userLevelVisibility**
- UserLevels **userLevelEnabled**
- bool **displayAlarmState**
- DisplayAlarmStateOptions **displayAlarmStateOption**
- double **value**
- int **sliderPosition**

9.88.1 Member Enumeration Documentation

9.88.1.1 enum QESlider::DisplayAlarmStateOptions

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

Enumerator:

Never Refer to DISPLAY_ALARM_STATE_NEVER for details.

Always Refer to DISPLAY_ALARM_STATE_ALWAYS for details.

WhenInAlarm Refer to DISPLAY_ALARM_STATE_WHEN_IN_ALARM for details.

9.88.1.2 enum QESlider::UserLevels

User friendly enumerations for `userLevelVisibility` and `userLevelEnabled` properties - refer to `userLevelVisibility` and `userLevelEnabled` properties and `userLevel` enumeration for details.

Enumerator:

User Refer to `USERLEVEL_USER` for details.

Scientist Refer to `USERLEVEL_SCIENTIST` for details.

Engineer Refer to `USERLEVEL_ENGINEER` for details.

9.88.2 Member Function Documentation

9.88.2.1 void QESlider::dbValueChanged (const qulonglong & out) [signal]

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

9.88.3 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]

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

9.88.4.3 DisplayAlarmStateOptions `QESlider::displayAlarmStateOption` [read, write]

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

9.88.4.4 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.

9.88.4.5 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)

9.88.4.6 UserLevels `QESlider::userLevelEnabled` [read, write]

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

9.88.4.7 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.

9.88.4.8 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.

9.88.4.9 **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.

9.88.4.10 **UserLevels QESlider::userLevelVisibility** [read, write]

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

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

EPICS variable name (CA PV)

9.88.4.12 **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.

9.88.4.13 **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.

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

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

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

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

9.89 QESpinBox Class Reference

Public Types

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

Public Slots

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

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

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

- void `setVariableNameSubstitutionsProperty` (QString `variableNameSubstitutions`)

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

- QString `getVariableNameSubstitutionsProperty` ()

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

- UserLevels `getUserLevelVisibilityProperty` ()

Access function for `userLevelVisibility` property - refer to `userLevelVisibility` property for details.

- void `setUserLevelVisibilityProperty` (UserLevels `level`)

Access function for `userLevelVisibility` property - refer to `userLevelVisibility` property for details.

- UserLevels `getUserLevelEnabledProperty` ()

Access function for `userLevelEnabled` property - refer to `userLevelEnabled` property for details.

- void `setUserLevelEnabledProperty` (UserLevels `level`)

Access function for `userLevelEnabled` property - refer to `userLevelEnabled` property for details.

- DisplayAlarmStateOptions `getDisplayAlarmStateOptionProperty` ()

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

- void `setDisplayAlarmStateOptionProperty` (DisplayAlarmStateOptions `option`)

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

Protected Member Functions

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

Protected Attributes

- QEFloatingFormatting `floatingFormatting`
- bool `writeOnChange`
- bool `addUnitsAsSuffix`
- bool `useDbPrecisionForDecimal`

Properties

- QString [variable](#)
- QString [variableSubstitutions](#)
- bool [variableAsToolTip](#)
- bool [allowDrop](#)
- bool [visible](#)
- unsigned [int](#)
- QString [userLevelUserStyle](#)
- QString [userLevelScientistStyle](#)
- QString [userLevelEngineerStyle](#)
- [UserLevels userLevelVisibility](#)
- [UserLevels userLevelEnabled](#)
- bool [displayAlarmState](#)
- [DisplayAlarmStateOptions displayAlarmStateOption](#)
- bool [subscribe](#)
- bool [useDbPrecision](#)
- bool [addUnits](#)
- double [value](#)

9.89.1 Member Enumeration Documentation

9.89.1.1 enum QESpinBox::DisplayAlarmStateOptions

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

Enumerator:

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

9.89.1.2 enum QESpinBox::UserLevels

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

Enumerator:

- User** Refer to USERLEVEL_USER for details.
Scientist Refer to USERLEVEL_SCIENTIST for details.
Engineer Refer to USERLEVEL_ENGINEER for details.

9.89.2 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]`

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

9.89.3.3 `DisplayAlarmStateOptions QESpinBox::displayAlarmStateOption [read, write]`

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

9.89.3.4 `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.

9.89.3.5 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)

9.89.3.6 UserLevels QESpinBox::userLevelEnabled [read, write]

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

9.89.3.7 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.

9.89.3.8 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.

9.89.3.9 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.

9.89.3.10 UserLevels QESpinBox::userLevelVisibility [read, write]

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

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

9.89.3.11 `QString QESpinBox::variable [read, write]`

EPICS variable name (CA PV)

9.89.3.12 `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.

9.89.3.13 `QString QESpinBox::variableSubstitutions [read, write]`

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

9.89.3.14 `bool QESpinBox::visible [read, write]`

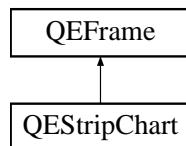
Display the widget. Default is true. Setting this property false is useful if widget is only used to provide a signal - for example, when supplying data to a [QELink](#) widget. Note, when false the widget will still be visible in Qt Designer.

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

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

9.90 QEStripChart Class Reference

Inheritance diagram for QEStripChart:



Public Types

- enum **Constants** { **NUMBER_OF_PVS** = 12 }

Public Member Functions

- **QEStripChart** (QWidget *parent=0)
- QSize **sizeHint** () const
- QDateTime **getStartTime** () const
- QDateTime **getEndTime** () const
- void **setEndTime** (QDateTime endTimeIn)
- int **getDuration** () const
- void **setDuration** (int durationIn)
- double **getYMinimum** () const
- void **setYMinimum** (const double yMinimumIn)
- double **getYMaximum** () const
- void **setYMaximum** (const double yMaximumIn)
- void **setYRange** (const double yMinimumIn, const double yMaximumIn)
- void **setPvName** (unsigned int slot, const QString &pvName)
- QString **getPvName** (unsigned int slot) const
- int **addPvName** (const QString &pvName)

Protected Member Functions

- void **mousePressEvent** (QMouseEvent *event)
- void **dragEnterEvent** (QDragEnterEvent *event)
- void **dropEvent** (QDropEvent *event)
- QString **copyVariable** ()
- QVariant **copyData** ()
- void **paste** (QVariant s)
- qcaobject::QCaObject * **createQcalItem** (unsigned int variableIndex)
- void **establishConnection** (unsigned int variableIndex)
- void **saveConfiguration** (PersistanceManager *pm)
- void **restoreConfiguration** (PersistanceManager *pm, restorePhases restorePhase)

- void **addToPredefinedList** (const QString &pvName)
- QStringList **getPredefinedPVNameList** () const
- QString **getPredefinedItem** (int i) const
- void **setRecalcIsRequired** ()
- void **setReplotIsRequired** ()
- void **evaluateAllowDrop** ()

Properties

- int **duration**
- double **yMinimum**
- double **yMaximum**
- 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 &loprHopr, 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 QEStripChartDurationDialog Class Reference

Public Member Functions

- **QEStripChartDurationDialog** (QWidget *parent=0)
- void **setDuration** (int secs)
- int **getDuration** () const

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

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

9.94 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** ()
- QEArchiveInterface::How **getArchiveReadHow** ()
- QEStripChartNames::LineDrawModes **getLineDrawMode** ()
- void **setColour** (const QColor &colour)
- QColor **getColour** ()
- QEDisplayRanges **getLoprHopr** (bool doScale)
- QEDisplayRanges **getDisplayedMinMax** (bool doScale)
- QEDisplayRanges **getBufferedMinMax** (bool doScale)
- QCaDataPointList **determinePlotPoints** ()
- void **readArchive** ()
- void **normalise** ()
- void **plotData** ()
- void **saveConfiguration** (PMElement &parentElement)
- void **restoreConfiguration** (PMElement &parentElement)

Public Attributes

- QCaVariableNamePropertyManager **pvNamePropertyManager**

Protected Member Functions

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

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

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

9.95 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 **ContextMenuLast** = SCCM_PREDEFINED_-
10
- static const int **NumberPrefdefinedItems** = (SCCM_PREDEFINED_10 - SCCM_-
PREDEFINED_01 + 1)

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

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

9.96 QEStripChartPushButtonSpecifications Struct Reference

Public Attributes

- int **gap**
- int **width**
- int **value**
- bool **isIcon**
- const QString **captionOrIcon**
- const QString **toolTip**
- const char * **member**

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

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

9.97 QEStripChartRangeDialog Class Reference

Public Member Functions

- **QEStripChartRangeDialog** (QWidget *parent=0)
- void **setRange** (const double min, const double max)

- double **getMinimum** ()
- double **getMaximum** ()

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

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

9.98 QEStripChartState Class Reference

Public Member Functions

- void **saveConfiguration** (PMElement &parentElement)
- void **restoreConfiguration** (PMElement &parentElement)

Public Attributes

- bool **isNormalVideo**
- QEStripChartNames::ChartTimeModes **chartTimeMode**
- QEStripChartNames::YScaleModes **yScaleMode**
- QEStripChartNames::ChartYRanges **chartYScale**
- double **yMinimum**
- double **yMaximum**
- int **duration**
- Qt::TimeSpec **timeZoneSpec**
- QDateTime **endDateTime**

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

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

9.99 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

9.100 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

9.101 QEStripChartTimeDialog Class Reference

Public Member Functions

- **QEStripChartTimeDialog** (QWidget *parent=0)
- void **setMaximumDateTime** (QDateTime datetime)
- void **setStartTime** (QDateTime datetime)
- QDateTime **getStartTime** ()
- void **setEndTime** (QDateTime datetime)
- QDateTime **getEndTime** ()

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

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

9.102 QEStripChartToolBar Class Reference

This class holds all the StripChart tool bar widgets.

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

Classes

- class [OwnWidgets](#)

Signals

- void **stateSelected** (const QEStripChartNames::StateModes mode)
- void **videoModeSelected** (const QEStripChartNames::VideoModes mode)
- void **yScaleModeSelected** (const QEStripChartNames::YScaleModes mode)
- void **yRangeSelected** (const QEStripChartNames::ChartYRanges scale)
- void **durationSelected** (const int seconds)
- void **selectDuration** ()
- void **timeZoneSelected** (const Qt::TimeSpec timeSpec)
- void **playModeSelected** (const QEStripChartNames::PlayModes mode)
- void **readArchiveSelected** ()

Public Member Functions

- **QEStripChartToolBar** (QWidget *parent=0)
- void **setYRangeStatus** (const QString &status)
- void **setTimeStatus** (const QString &timeStatus)
- void **setDurationStatus** (const QString &durationStatus)
- void **setStateSelectionEnabled** (const QEStripChartNames::StateModes mode, const bool enabled)

Static Public Attributes

- static const int **designHeight** = 44

Protected Member Functions

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

9.102.1 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

9.103 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`

9.103.1 Member Data Documentation

9.103.1.1 `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

9.103.2 Property Documentation

9.103.2.1 `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`

9.104 recording Class Reference

Signals

- `void byteArrayChanged (const QByteArray &value, unsigned long dataSize, QCaAlarmInfo &alarmInfo, QCaDateTime &timeStamp, const unsigned int &variableIndex)`
- `void playingBack (bool playing)`

Public Member Functions

- **recording** (QWidget *parent=0)
- bool **isRecording** ()
- void **recordImage** (QByteArray image, unsigned long dataSize, QCaAlarmInfo &alarmInfo, QCaDateTime &time)
- void **nextFrameDue** ()

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

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

9.105 imageDisplayProperties::rgbPixel Struct Reference

Public Attributes

- unsigned char **p** [4]

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

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

9.106 screenSelectDialog Class Reference

Public Types

- enum **screens** { **PRIMARY_SCREEN** = -3, **THIS_SCREEN** = -2, **ALL_SCREENS** = -1 }

Public Member Functions

- **screenSelectDialog** (int numScreens, QWidget *parent=0)
- int **getScreenNum** ()

Static Public Member Functions

- static bool **getFullscreenGeometry** (QWidget *target, QRect &geom)

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

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

9.107 selectMenu Class Reference

Public Member Functions

- **selectMenu** (QWidget *parent=0)
- imageContextMenu::imageContextMenuOptions **getSelectOption** (const QPoint &pos)
- void **enable** (imageContextMenu::imageContextMenuOptions option, bool state)
- bool **isEnabled** (imageContextMenu::imageContextMenuOptions option)
- void **setChecked** (const int mode)
- void **setItemText** (imageContextMenu::imageContextMenuOptions option, QString title)

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

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

9.108 trace Class Reference

Public Attributes

- QVector< QCaDateTime > **timeStamps**
- QVector< double > **xdata**
- QVector< double > **ydata**
- QwtPlotCurve * **curve**
- QColor **color**
- QString **legend**
- bool **waveform**
- QwtPlotCurve::CurveStyle **style**

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

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

9.109 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

9.110 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`

9.111 ValueScaling Class Reference

Public Member Functions

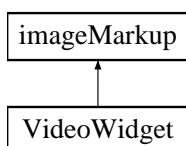
- `void reset ()`
- `void assign (const ValueScaling &s)`
- `void set (const double dIn, const double mIn, const double cIn)`
- `void get (double &dOut, double &mOut, double &cOut) const`
- `void map (const double fromLower, const double fromUpper, const double toLower, const double toUpper)`
- `bool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

9.112 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/QElImage/videowidget.h
- /tmp/epicsqt/trunk/framework/widgets/QElImage/videowidget.cpp

9.113 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, 66
 QECheckBox, 83
 QELabel, 170
 QELineEdit, 178
 QENumericEdit, 191
 QEPushButton, 215
 QERadioButton, 233
alarmSeverityDisplayMode
 QEAnalogProgressBar, 66
alignment
 QECheckBox, 83
 QEPushButton, 215
 QERadioButton, 233
allowDrop
 QEAnalogProgressBar, 66
 QEBitStatus, 73
 QECheckBox, 83
 QEComboBox, 95
 QEConfiguredLayout, 101
 QEFileBrowser, 107
 QEFrame, 112
 QEGenericEdit, 121
 QEGroupBox, 126
 QEImage, 149
 QELabel, 170
 QELog, 185
 QEPeriodic, 196
 QEPlot, 205
 QEPushButton, 215
 QERadioButton, 233
 QEScript, 249

QEShape, 258
QESlider, 268
QESpinBox, 274
altReadbackVariable
 QEPushButton, 215
Always
 QEAnalogProgressBar, 64
 QEBitStatus, 73
 QECheckBox, 80
 QEComboBox, 94
 QEConfiguredLayout, 100
 QEFileBrowser, 106
 QEFrame, 112
 QEGenericEdit, 119
 QEGroupBox, 126
 QEImage, 145
 QELabel, 168
 QELog, 185
 QEPeriodic, 195
 QEPlot, 204
 QEPushButton, 212
 QERadioButton, 230
 QEScript, 249
 QEShape, 256
 QESlider, 267
 QESpinBox, 273
animation1
 QEShape, 258
animation2
 QEShape, 258
animation3
 QEShape, 258
animation4
 QEShape, 258
animation5
 QEShape, 258
animation6
 QEShape, 258
animationOptions
 QEShape, 256
Append

QEAnalogProgressBar, 64
QECheckBox, 80
QELabel, 168
QELineEdit, 177
QEPushButton, 211
QERadioButton, 230
areaColor
 QEImage, 149
arealInfo, 30
arguments
 QECheckBox, 83
 QEPushButton, 215
 QERadioButton, 233
arguments1
 QEImage, 149
arguments2
 QEImage, 150
arrayAction
 QEAnalogProgressBar, 66
 QECheckBox, 83
 QELabel, 170
 QELineEdit, 178
 QEPushButton, 215
 QERadioButton, 233
ArrayActions
 QEAnalogProgressBar, 64
 QECheckBox, 80
 QELabel, 168
 QELineEdit, 177
 QEPushButton, 211
 QERadioButton, 229
Ascii
 QEAnalogProgressBar, 64
 QECheckBox, 80
 QELabel, 168
 QELineEdit, 177
 QEPushButton, 211
 QERadioButton, 230
autoBrightnessContrast
 QEImage, 150
Automatic
 QEAnalogProgressBar, 65
 QECheckBox, 81
 QELabel, 168
 QELineEdit, 177
 QEPushButton, 213
 QERadioButton, 231
autoScale
 QENumericEdit, 191
backgroundColour
 QEAnalogIndicator, 59
Bar
 QEAnalogIndicator, 58
Bayer
 QEImage, 146
BayerBG
 QEImage, 146
BayerGB
 QEImage, 146
BayerGR
 QEImage, 146
BayerRG
 QEImage, 146
beamColor
 QEImage, 150
beamXVariable
 QEImage, 150
beamYVariable
 QEImage, 150
bitDepthVariable
 QEImage, 150
borderColour
 QEAnalogIndicator, 59
Bottom_To_Top
 QEAnalogIndicator, 59
BOUNDING_RECTANGLE
 QEImage, 146
BoundingRectangle
 QEImage, 146
briefInfoArea
 QEImage, 150
CenterAndSize
 QEImage, 146
centreAngle
 QEAnalogIndicator, 59
clickCheckedText
 QECheckBox, 84
 QEPushButton, 215
 QERadioButton, 234
clicked
 QECheckBox, 82
 QEPushButton, 214
 QERadioButton, 232
clickText
 QECheckBox, 84
 QEPushButton, 216
 QERadioButton, 234
clippingHighVariable

QEImage, 150
 clippingLowVariable
 QEImage, 150
 clippingOnOffVariable
 QEImage, 151
 color1
 QEShape, 258
 color10
 QEShape, 259
 color2
 QEShape, 259
 color3
 QEShape, 259
 color4
 QEShape, 259
 color5
 QEShape, 259
 color6
 QEShape, 259
 color7
 QEShape, 259
 color8
 QEShape, 259
 color9
 QEShape, 259
 confirmAction
 QECheckBox, 84
 QEPushButton, 216
 QERadioButton, 234
 confirmText
 QECheckBox, 84
 QEPushButton, 216
 QERadioButton, 234
 confirmWrite
 QEGenericEdit, 121
 contrastReversal
 QEImage, 151
 creationOption
 QECheckBox, 84
 QEPushButton, 216
 QERadioButton, 234
 CreationOptionNames
 QECheckBox, 80
 QEPushButton, 211
 QERadioButton, 230
 customisationName
 QECheckBox, 85
 QEPushButton, 216
 QERadioButton, 235
 dbElementChanged
 QEPeriodic, 196
 dbValueChanged
 QEAnalogProgressBar, 66
 QEBitStatus, 73
 QECheckBox, 82
 QEComboBox, 95
 QEImage, 149
 QELabel, 170
 QELineEdit, 178
 QNumericEdit, 191
 QEPeriodic, 196
 QEPlot, 204
 QEPushButton, 214
 QERadioButton, 232
 QESlider, 268
 QESpinBox, 274
 dbValueChanged1
 QEShape, 257
 dbValueChanged2
 QEShape, 257
 dbValueChanged3
 QEShape, 257
 dbValueChanged4
 QEShape, 257
 dbValueChanged5
 QEShape, 257
 dbValueChanged6
 QEShape, 257
 Default
 QEAnalogProgressBar, 65
 QECheckBox, 81
 QELabel, 168
 QELineEdit, 177
 QEPushButton, 212
 QERadioButton, 231
 dimension1Variable
 QEImage, 151
 dimension2Variable
 QEImage, 151
 dimension3Variable
 QEImage, 151
 dimensionsVariable
 QEImage, 151
 displayAlarmState
 QEAnalogProgressBar, 66
 QEBitStatus, 73
 QECheckBox, 85
 QEComboBox, 95
 QEConfiguredLayout, 101

QEFileBrowser, 107
QEFrame, 112
QEGenericEdit, 121
QEGroupBox, 126
QEImage, 151
QELabel, 170
QELog, 185
QEPeriodic, 196
QEPlot, 205
QEPushButton, 217
QERadioButton, 235
QEScript, 249
QEShape, 260
QESlider, 268
QESpinBox, 274
displayAlarmStateOption
QEAnalogProgressBar, 67
QEBitStatus, 74
QECheckBox, 85
QEComboBox, 96
QEConfiguredLayout, 101
QEFileBrowser, 107
QEFrame, 113
QEGenericEdit, 121
QEGroupBox, 126
QEImage, 152
QELabel, 170
QELog, 185
QEPeriodic, 196
QEPlot, 205
QEPushButton, 217
QERadioButton, 235
QEScript, 249
QEShape, 260
QESlider, 268
QESpinBox, 274
DisplayAlarmStateOptions
QEAnalogProgressBar, 64
QEBitStatus, 73
QECheckBox, 80
QEComboBox, 94
QEConfiguredLayout, 100
QEFileBrowser, 106
QEFrame, 112
QEGenericEdit, 119
QEGroupBox, 125
QEImage, 145
QELabel, 168
QELog, 185
QEPeriodic, 195
QEPlot, 204
QEPushButton, 212
QERadioButton, 230
QEScript, 249
QEShape, 256
QESlider, 267
QESpinBox, 273
displayArea1Selection
QEImage, 152
displayArea2Selection
QEImage, 152
displayArea3Selection
QEImage, 152
displayArea4Selection
QEImage, 152
displayBeamSelection
QEImage, 152
displayButtonBar
QEImage, 149
displayCursorPixelInfo
QEImage, 152
displayEllipse
QEImage, 152
displayHozSliceSelection
QEImage, 153
displayProfileSelection
QEImage, 153
displayTargetSelection
QEImage, 153
displayVertSliceSelection
QEImage, 153
DockBottom
QECheckBox, 80
QEPushButton, 211
QERadioButton, 230
DockBottomTabbed
QECheckBox, 80
QEPushButton, 212
QERadioButton, 230
DockFloating
QECheckBox, 80
QEPushButton, 212
QERadioButton, 230
DockLeft
QECheckBox, 80
QEPushButton, 212
QERadioButton, 230
DockLeftTabbed
QECheckBox, 80
QEPushButton, 212

QERadioButton, 230
 DockRight
 QECheckBox, 80
 QEPushButton, 212
 QERadioButton, 230
 DockRightTabbed
 QECheckBox, 80
 QEPushButton, 212
 QERadioButton, 230
 DockTop
 QECheckBox, 80
 QEPushButton, 211
 QERadioButton, 230
 DockTopTabbed
 QECheckBox, 80
 QEPushButton, 212
 QERadioButton, 230
 DottedFullCrosshair
 QEImage, 148
 drawMarkup
 markupHLine, 44
 markupVLine, 49
 ellipseColor
 QEImage, 153
 ellipseHVariable
 QEImage, 153
 EllipseVariableDefinitions
 QEImage, 146
 ellipseVariableDefinitions
 QEImage, 145
 ellipseWVariable
 QEImage, 153
 ellipseXVariable
 QEImage, 153
 ellipseYVariable
 QEImage, 153
 enableArea1Selection
 QEImage, 154
 enableArea2Selection
 QEImage, 154
 enableArea3Selection
 QEImage, 154
 enableArea4Selection
 QEImage, 154
 enableBeamSelection
 QEImage, 154
 enableHozSliceSelection
 QEImage, 154
 enableProfileSelection
 QEImage, 154
 QEImage, 154
 enableTargetSelection
 QEImage, 154
 enableVertSliceSelection
 QEImage, 155
 Engineer
 QEAnalogProgressBar, 65
 QEBitStatus, 73
 QECheckBox, 82
 QEComboBox, 95
 QEConfiguredLayout, 100
 QEFileBrowser, 106
 QEFrame, 112
 QEGenericEdit, 119
 QEGroupBox, 126
 QEImage, 148
 QELabel, 169
 QELog, 185
 QEPeriodic, 196
 QEPlot, 204
 QEPushButton, 213
 QERadioButton, 232
 QEScript, 249
 QEShape, 256
 QESlider, 268
 QESpinBox, 273
 externalControls
 QEImage, 155
 FFBuffer, 32
 FFThread, 32
 Fit
 QEImage, 147
 Fixed
 QEAnalogProgressBar, 65
 QECheckBox, 81
 QELabel, 168
 QELineEdit, 177
 QEPushButton, 213
 QERadioButton, 231
 flipRotateMenu, 33
 Floating
 QEAnalogProgressBar, 65
 QECheckBox, 81
 QELabel, 168
 QELineEdit, 177
 QEPushButton, 212
 QERadioButton, 231
 fontColour
 QEAnalogIndicator, 59

foregroundColour
QEAnalogIndicator, 59

format
QEAnalogProgressBar, 67
QECheckBox, 85
QELabel, 171
QELineEdit, 178
QEPushButton, 217
QERadioButton, 235

formatOption
QEImage, 155

FormatOptions
QEImage, 146

Formats
QEAnalogProgressBar, 64
QECheckBox, 80
QELabel, 168
QELineEdit, 177
QEPushButton, 212
QERadioButton, 230

formatVariable
QEImage, 155

fullScreenWindow, 33

getConfirmWrite
QEGenericEdit, 120

getSubscribe
QEGenericEdit, 120

getWriteOnEnter
QEGenericEdit, 120

getWriteOnFinish
QEGenericEdit, 120

getWriteOnLoseFocus
QEGenericEdit, 120

guiFile
QECheckBox, 85
QEPushButton, 217
QERadioButton, 235

heightVariable
QEImage, 155

histogram, 34

histogramScroll, 34

historicImage, 34

horizontalFlip
QEImage, 155

hozSliceColor
QEImage, 155

Icon

QECheckBox, 82
QEPushButton, 213
QERadioButton, 232

imageContextMenu, 35

imageDisplayProperties, 36

imageDisplayProperties::rgbPixel, 287

imageInfo, 37

imageMarkup, 38

imageUpdateIndicator, 40

imageVariable
QEImage, 155

Index
QEAnalogProgressBar, 64
QECheckBox, 80
QELabel, 168
QELineEdit, 177
QEPushButton, 211
QERadioButton, 230

initialHosScrollPos
QEImage, 156

initialVertScrollPos
QEImage, 149

int
QEAnalogProgressBar, 67
QEBitStatus, 74
QECheckBox, 86
QEComboBox, 96
QEConfiguredLayout, 101
QEFileBrowser, 107
QEFrame, 113
QEGenericEdit, 122
QEGroupBox, 126
QEImage, 156
QELabel, 171
QELineEdit, 178
QELog, 186
QEPeriodic, 197
QEPlot, 205
QEPushButton, 217
QERadioButton, 236
QEScript, 250
QEShape, 260
QESlider, 269
QESpinBox, 274

Integer
QEAnalogProgressBar, 65
QECheckBox, 81
QELabel, 168
QELineEdit, 177
QEPushButton, 212

QERadioButton, 231
 labelText
 QECheckBox, 86
 QEPushButton, 218
 QERadioButton, 236
 QESubstitutedLabel, 286
 leadingZero
 QEAnalogProgressBar, 67
 QECheckBox, 86
 QELabel, 171
 QELineEdit, 179
 QEPushButton, 218
 QERadioButton, 236
 leadingZeros
 QNumericUpDown, 191
 Left_To_Right
 QEAnalogIndicator, 59
 lineProfileArrayVariable
 QEImage, 156
 lineProfileThicknessVariable
 QEImage, 156
 lineProfileX1Variable
 QEImage, 156
 lineProfileX2Variable
 QEImage, 156
 lineProfileY1Variable
 QEImage, 156
 lineProfileY2Variable
 QEImage, 156
 LocalEnumeration
 QEAnalogProgressBar, 65
 QECheckBox, 81
 QELabel, 168
 QELineEdit, 177
 QEPushButton, 212
 QERadioButton, 231
 localEnumeration
 QEAnalogProgressBar, 67
 QECheckBox, 86
 QEComboBox, 96
 QELabel, 171
 QELineEdit, 179
 QEPushButton, 218
 QERadioButton, 236
 logBrightness
 QEImage, 157
 loginWidget, 40
 LogOutput
 QECheckBox, 81
 QEImage, 147
 QEPushButton, 213
 QERadioButton, 231
 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
 QEAnalogIndicator, 60
 QNumericUpDown, 191
 Meter
 QEAnalogIndicator, 58
 minimum
 QEAnalogIndicator, 60
 QNumericUpDown, 191
 minorInterval
 QEAnalogIndicator, 60
 mode
 QEAnalogIndicator, 60
 Modes
 QEAnalogIndicator, 58
 Mono
 QEImage, 146
 mpegSource, 49
 updateImage, 50
 mpegSourceObject, 50
 Never
 QEAnalogProgressBar, 64
 QEBitStatus, 73
 QECheckBox, 80
 QEComboBox, 94
 QEConfiguredLayout, 100
 QEFileBrowser, 106
 QEFrame, 112

QEGenericEdit, 119
QEGroupBox, 126
QEImage, 145
QELabel, 168
QELog, 185
QEPeriodic, 195
QEPlot, 204
QEPushButton, 212
QERadioButton, 230
QEScript, 249
QEShape, 256
QESlider, 267
QESpinBox, 273
NewTab
 QECheckBox, 80
 QEPushButton, 211
 QERadioButton, 230
NewWindow
 QECheckBox, 80
 QEPushButton, 211
 QERadioButton, 230
None
 QECheckBox, 81
 QEImage, 146
 QEPushButton, 213
 QERadioButton, 231
NoRotation
 QEImage, 147
notation
 QEAnalogProgressBar, 68
 QECheckBox, 87
 QELabel, 172
 QELineEdit, 179
 QEPushButton, 219
 QERadioButton, 237
Notations
 QEAnalogProgressBar, 65
 QECheckBox, 81
 QELabel, 168
 QELineEdit, 177
 QEPushButton, 212
 QERadioButton, 231
offset1
 QEShape, 260
offset2
 QEShape, 260
offset3
 QEShape, 260
offset4

 QEShape, 261
 offset5
 QEShape, 261
 offset6
 QEShape, 261
 Open
 QECheckBox, 80
 QEPushButton, 211
 QERadioButton, 230
 orientation
 QEAnalogIndicator, 60
 Orientations
 QEAnalogIndicator, 58

 password
 QECheckBox, 87
 QEPushButton, 219
 QERadioButton, 237
 PeriodicDialog, 51
 PeriodicElementSetupForm, 51
 PeriodicSetupDialog, 52
 Picture
 QELabel, 169
 pixmap0
 QECheckBox, 87
 QELabel, 172
 QEPushButton, 219
 QERadioButton, 237
 pixmap1
 QECheckBox, 87
 QELabel, 172
 QEPushButton, 219
 QERadioButton, 237
 pixmap2
 QECheckBox, 87
 QELabel, 172
 QEPushButton, 219
 QERadioButton, 237
 pixmap3
 QECheckBox, 87
 QELabel, 172
 QEPushButton, 219
 QERadioButton, 237
 pixmap4
 QECheckBox, 88
 QELabel, 172
 QEPushButton, 219
 QERadioButton, 238
 pixmap5
 QECheckBox, 88

QELabel, 172
 QEPushButton, 219
 QERadioButton, 238
 pixmap6
 QECheckBox, 88
 QELabel, 173
 QEPushButton, 219
 QERadioButton, 238
 pixmap7
 QECheckBox, 88
 QELabel, 173
 QEPushButton, 220
 QERadioButton, 238
 playbackTimer, 52
 point1
 QEShape, 261
 point10
 QEShape, 261
 point2
 QEShape, 261
 point3
 QEShape, 261
 point4
 QEShape, 261
 point5
 QEShape, 261
 point6
 QEShape, 262
 point7
 QEShape, 262
 point8
 QEShape, 262
 point9
 QEShape, 262
 pointInfo, 52
 precision
 QEAnalogProgressBar, 68
 QECheckBox, 88
 QELabel, 173
 QELineEdit, 179
 QENumericEdit, 191
 QEPushButton, 220
 QERadioButton, 238
 pressed
 QECheckBox, 82
 QEPushButton, 214
 QERadioButton, 232
 pressText
 QECheckBox, 88
 QEPushButton, 220
 QERadioButton, 238
 prioritySubstitutions
 QECheckBox, 88
 QEPushButton, 220
 QERadioButton, 238
 profileColor
 QEImage, 157
 profileHozArrayVariable
 QEImage, 157
 profileHozThicknessVariable
 QEImage, 157
 profileHozVariable
 QEImage, 157
 profilePlot, 53
 profileVertArrayVariable
 QEImage, 157
 profileVertThicknessVariable
 QEImage, 157
 profileVertVariable
 QEImage, 157
 program
 QECheckBox, 89
 QEPushButton, 220
 QERadioButton, 239
 program1
 QEImage, 157
 program2
 QEImage, 158
 programStartupOption
 QECheckBox, 89
 QEPushButton, 220
 QERadioButton, 239
 programStartupOption1
 QEImage, 158
 programStartupOption2
 QEImage, 158
 ProgramStartupOptionNames
 QECheckBox, 81
 QEImage, 146
 QEPushButton, 213
 QERadioButton, 231
 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, 66
 alarmSeverityDisplayMode, 66
 allowDrop, 66
 Always, 64
 Append, 64
 arrayAction, 66
 ArrayActions, 64
 Ascii, 64
 Automatic, 65
 dbValueChanged, 66
 Default, 65
 displayAlarmState, 66
 displayAlarmStateOption, 67
 DisplayAlarmStateOptions, 64
 Engineer, 65
 Fixed, 65
 Floating, 65
 format, 67
 Formats, 64
 Index, 64
 int, 67
 Integer, 65
 leadingZero, 67
 LocalEnumeration, 65
 localEnumeration, 67
 Never, 64
 notation, 68
Notations, 65
precision, 68
QEAnalogProgressBar, 65
Scientific, 65
Scientist, 65
Time, 65
trailingZeros, 68
UnsignedInteger, 65
useDbDisplayLimits, 68
useDbPrecision, 68
User, 65
userLevelEnabled, 69
userLevelEngineerStyle, 69
UserLevels, 65
userLevelScientistStyle, 69
userLevelUserStyle, 69
userLevelVisibility, 69
value, 70
variable, 70
variableAsToolTip, 70
variableSubstitutions, 70
visible, 70
WhenInAlarm, 64
QEBitStatus, 70
allowDrop, 73
Always, 73
dbValueChanged, 73
displayAlarmState, 73
displayAlarmStateOption, 74
DisplayAlarmStateOptions, 73
Engineer, 73
int, 74
Never, 73
Scientist, 73
User, 73
userLevelEnabled, 74
userLevelEngineerStyle, 74
UserLevels, 73
userLevelScientistStyle, 74
userLevelUserStyle, 75
userLevelVisibility, 75
variable, 75
variableAsToolTip, 75
variableSubstitutions, 75
visible, 75
WhenInAlarm, 73
QECheckBox, 76
 addUnits, 83
 alignment, 83
 allowDrop, 83

Always, 80
Append, 80
arguments, 83
arrayAction, 83
ArrayActions, 80
Ascii, 80
Automatic, 81
clickCheckedText, 84
clicked, 82
clickText, 84
confirmAction, 84
confirmText, 84
creationOption, 84
CreationOptionNames, 80
customisationName, 85
dbValueChanged, 82
Default, 81
displayAlarmState, 85
displayAlarmStateOption, 85
DisplayAlarmStateOptions, 80
DockBottom, 80
DockBottomTabbed, 80
DockFloating, 80
DockLeft, 80
DockLeftTabbed, 80
DockRight, 80
DockRightTabbed, 80
DockTop, 80
DockTopTabbed, 80
Engineer, 82
Fixed, 81
Floating, 81
format, 85
Formats, 80
guiFile, 85
Icon, 82
Index, 80
int, 86
Integer, 81
labelText, 86
leadingZero, 86
LocalEnumeration, 81
localEnumeration, 86
LogOutput, 81
Never, 80
NewTab, 80
NewWindow, 80
None, 81
notation, 87
Notations, 81
Open, 80
password, 87
 pixmap0, 87
 pixmap1, 87
 pixmap2, 87
 pixmap3, 87
 pixmap4, 88
 pixmap5, 88
 pixmap6, 88
 pixmap7, 88
precision, 88
pressed, 82
pressText, 88
prioritySubstitutions, 88
program, 89
programStartupOption, 89
ProgramStartupOptionNames, 81
QECheckBox, 82
released, 83
releaseText, 89
requestAction, 83
Scientific, 81
Scientist, 82
State, 82
StdOutput, 81
subscribe, 89
Terminal, 81
Text, 82
TextAndIcon, 82
Time, 81
trailingZeros, 89
UnsignedInteger, 81
updateOption, 89
UpdateOptions, 81
useDbPrecision, 89
User, 82
userLevelEnabled, 89
userLevelEngineerStyle, 90
UserLevels, 82
userLevelScientistStyle, 90
userLevelUserStyle, 90
userLevelVisibility, 90
variable, 90
variableAsToolTip, 90
variableSubstitutions, 91
visible, 91
WhenInAlarm, 80
writeOnClick, 91
writeOnPress, 91
writeOnRelease, 91

QECheckBoxManager, 91
QEComboBox, 92
 allowDrop, 95
 Always, 94
 dbValueChanged, 95
 displayAlarmState, 95
 displayAlarmStateOption, 96
 DisplayAlarmStateOptions, 94
 Engineer, 95
 int, 96
 localEnumeration, 96
 Never, 94
 Scientist, 95
 subscribe, 96
 useDbEnumerations, 95
 User, 95
 userLevelEnabled, 96
 userLevelEngineerStyle, 96
 UserLevels, 94
 userLevelScientistStyle, 97
 userLevelUserStyle, 97
 userLevelVisibility, 97
 variable, 97
 variableAsToolTip, 97
 variableSubstitutions, 97
 visible, 98
 WhenInAlarm, 94
 writeOnChange, 95
QEConfiguredLayout, 98
 allowDrop, 101
 Always, 100
 displayAlarmState, 101
 displayAlarmStateOption, 101
 DisplayAlarmStateOptions, 100
 Engineer, 100
 int, 101
 Never, 100
 Scientist, 100
 User, 100
 userLevelEnabled, 101
 userLevelEngineerStyle, 101
 UserLevels, 100
 userLevelScientistStyle, 102
 userLevelUserStyle, 102
 userLevelVisibility, 102
 variableAsToolTip, 102
 visible, 102
 WhenInAlarm, 100
QEConfiguredLayoutManager, 103
QEFileBrowser, 103
 allowDrop, 107
 Always, 106
 displayAlarmState, 107
 displayAlarmStateOption, 107
 DisplayAlarmStateOptions, 106
 Engineer, 106
 int, 107
 Never, 106
 Scientist, 106
 selected, 107
 User, 106
 userLevelEnabled, 107
 userLevelEngineerStyle, 108
 UserLevels, 106
 userLevelScientistStyle, 108
 userLevelUserStyle, 108
 userLevelVisibility, 108
 variable, 108
 variableAsToolTip, 109
 variableSubstitutions, 109
 visible, 109
 WhenInAlarm, 106
QEForm, 109
QEFrame, 111
 allowDrop, 112
 Always, 112
 displayAlarmState, 112
 displayAlarmStateOption, 113
 DisplayAlarmStateOptions, 112
 Engineer, 112
 int, 113
 Never, 112
 Scientist, 112
 User, 112
 userLevelEnabled, 113
 userLevelEngineerStyle, 113
 UserLevels, 112
 userLevelScientistStyle, 113
 userLevelUserStyle, 114
 userLevelVisibility, 114
 variableAsToolTip, 114
 visible, 114
 WhenInAlarm, 112
QEGenericButton, 114
QEGenericEdit, 116
 allowDrop, 121
 Always, 119
 confirmWrite, 121
 displayAlarmState, 121
 displayAlarmStateOption, 121

DisplayAlarmStateOptions, 119
Engineer, 119
getConfirmWrite, 120
getSubscribe, 120
getWriteOnEnter, 120
getWriteOnFinish, 120
getWriteOnLoseFocus, 120
int, 122
Never, 119
QEGenericEdit, 120
Scientist, 119
setConfirmWrite, 120
setSubscribe, 120
setWriteOnEnter, 121
setWriteOnFinish, 121
setWriteOnLoseFocus, 121
subscribe, 122
User, 119
userLevelEnabled, 122
userLevelEngineerStyle, 122
UserLevels, 119
userLevelScientistStyle, 122
userLevelUserStyle, 123
userLevelVisibility, 123
variable, 123
variableAsToolTip, 123
variableSubstitutions, 123
visible, 123
WhenInAlarm, 119
writeOnEnter, 124
writeOnFinish, 124
writeOnLoseFocus, 124
QEGroupBox, 124
allowDrop, 126
Always, 126
displayAlarmState, 126
displayAlarmStateOption, 126
DisplayAlarmStateOptions, 125
Engineer, 126
int, 126
Never, 126
Scientist, 126
substitutedTitle, 127
textSubstitutions, 127
User, 126
userLevelEnabled, 127
userLevelEngineerStyle, 127
UserLevels, 126
userLevelScientistStyle, 127
userLevelUserStyle, 127
userLevelVisibility, 128
variableAsToolTip, 128
visible, 128
WhenInAlarm, 126
QEImage, 128
allowDrop, 149
Always, 145
areaColor, 149
arguments1, 149
arguments2, 150
autoBrightnessContrast, 150
Bayer, 146
BayerBG, 146
BayerGB, 146
BayerGR, 146
BayerRG, 146
beamColor, 150
beamXVariable, 150
beamYVariable, 150
bitDepthVariable, 150
BOUNDING_RECTANGLE, 146
BoundingRectangle, 146
briefInfoArea, 150
CenterAndSize, 146
clippingHighVariable, 150
clippingLowVariable, 150
clippingOnOffVariable, 151
contrastReversal, 151
dbValueChanged, 149
dimension1Variable, 151
dimension2Variable, 151
dimension3Variable, 151
dimensionsVariable, 151
displayAlarmState, 151
displayAlarmStateOption, 152
DisplayAlarmStateOptions, 145
displayArea1Selection, 152
displayArea2Selection, 152
displayArea3Selection, 152
displayArea4Selection, 152
displayBeamSelection, 152
displayButtonBar, 149
displayCursorPixelInfo, 152
displayEllipse, 152
displayHozSliceSelection, 153
displayProfileSelection, 153
displayTargetSelection, 153
displayVertSliceSelection, 153
DottedFullCrosshair, 148
ellipseColor, 153

ellipseHVariable, 153
EllipseVariableDefinitions, 146
ellipseVariableDefinitions, 145
ellipseWVariable, 153
ellipseXVariable, 153
ellipseYVariable, 153
enableArea1Selection, 154
enableArea2Selection, 154
enableArea3Selection, 154
enableArea4Selection, 154
enableBeamSelection, 154
enableHozSliceSelection, 154
enableProfileSelection, 154
enableTargetSelection, 154
enableVertSliceSelection, 155
Engineer, 148
externalControls, 155
Fit, 147
formatOption, 155
FormatOptions, 146
formatVariable, 155
heightVariable, 155
horizontalFlip, 155
hozSliceColor, 155
imageVariable, 155
initialHosScrollPos, 156
initialVertScrollPos, 149
int, 156
lineProfileArrayVariable, 156
lineProfileThicknessVariable, 156
lineProfileX1Variable, 156
lineProfileX2Variable, 156
lineProfileY1Variable, 156
lineProfileY2Variable, 156
logBrightness, 157
LogOutput, 147
Mono, 146
Never, 145
None, 146
NoRotation, 147
profileColor, 157
profileHozArrayVariable, 157
profileHozThicknessVariable, 157
profileHozVariable, 157
profileVertArrayVariable, 157
profileVertThicknessVariable, 157
profileVertVariable, 157
program1, 157
program2, 158
programStartupOption1, 158
programStartupOption2, 158
ProgramStartupOptionNames, 146
QEImage, 148, 149
regionOfInterest1HVariable, 158
regionOfInterest1WVariable, 158
regionOfInterest1XVariable, 158
regionOfInterest1YVariable, 158
regionOfInterest2HVariable, 159
regionOfInterest2WVariable, 159
regionOfInterest2XVariable, 159
regionOfInterest2YVariable, 159
regionOfInterest3HVariable, 159
regionOfInterest3WVariable, 159
regionOfInterest3XVariable, 159
regionOfInterest3YVariable, 159
regionOfInterest4HVariable, 159
regionOfInterest4WVariable, 160
regionOfInterest4XVariable, 160
regionOfInterest4YVariable, 160
RESIZE_OPTION_FIT, 147
RESIZE_OPTION_ZOOM, 147
resizeOption, 160
ResizeOptions, 147
resizeOptions, 147
rgb1, 146
rgb2, 146
rgb3, 146
Rotate180, 147
Rotate90Left, 147
Rotate90Right, 147
rotation, 160
ROTATION_0, 147
ROTATION_180, 147
ROTATION_90_LEFT, 147
ROTATION_90_RIGHT, 147
RotationOptions, 147
rotationOptions, 147
Scientist, 148
selectOptions, 147
showTime, 160
SO_AREA4, 148
SO_BEAM, 148
SO_HSLICE, 148
SO_NONE, 148
SO_PANNING, 148
SO_PROFILE, 148
SO_TARGET, 148
SO_VSLICE, 148
SolidSmallCrosshair, 148
StdOutput, 147

targetColor, 160
TargetOptions, 148
targetTriggerVariable, 160
targetXVariable, 160
targetYVariable, 161
Terminal, 146
timeColor, 161
URL, 161
useFalseColour, 161
User, 148
userLevelEnabled, 161
userLevelEngineerStyle, 161
UserLevels, 148
userLevelScientistStyle, 161
userLevelUserStyle, 162
userLevelVisibility, 162
variableAsToolTip, 162
variableSubstitutions, 162
verticalFlip, 162
vertSliceColor, 162
visible, 162
WhenInAlarm, 145
widthVariable, 163
yuv422, 146
yuv444, 146
Zoom, 147
QEImageMarkupThickness, 163
QEImageOptionsDialog, 163
QELabel, 164
 addUnits, 170
 allowDrop, 170
 Always, 168
 Append, 168
 arrayAction, 170
 ArrayActions, 168
 Ascii, 168
 Automatic, 168
 dbValueChanged, 170
 Default, 168
 displayAlarmState, 170
 displayAlarmStateOption, 170
 DisplayAlarmStateOptions, 168
 Engineer, 169
 Fixed, 168
 Floating, 168
 format, 171
 Formats, 168
 Index, 168
 int, 171
 Integer, 168
leadingZero, 171
LocalEnumeration, 168
localEnumeration, 171
Never, 168
notation, 172
Notations, 168
Picture, 169
 pixmap0, 172
 pixmap1, 172
 pixmap2, 172
 pixmap3, 172
 pixmap4, 172
 pixmap5, 172
 pixmap6, 173
 pixmap7, 173
precision, 173
QELabel, 169
Scientific, 168
Scientist, 169
Text, 169
Time, 168
trailingZeros, 173
UnsignedInteger, 168
UPDATE_PIXMAP, 169
UPDATE_TEXT, 169
updateOption, 173
UpdateOptions, 169
updateOptions, 168
useDbPrecision, 173
User, 169
userLevelEnabled, 173
userLevelEngineerStyle, 174
UserLevels, 169
userLevelScientistStyle, 174
userLevelUserStyle, 174
userLevelVisibility, 174
variable, 174
variableAsToolTip, 174
variableSubstitutions, 175
visible, 175
WhenInAlarm, 168
QELineEdit, 175
 addUnits, 178
 Append, 177
 arrayAction, 178
 ArrayActions, 177
 Ascii, 177
 Automatic, 177
 dbValueChanged, 178
 Default, 177

Fixed, 177
Floating, 177
format, 178
Formats, 177
Index, 177
int, 178
Integer, 177
leadingZero, 179
LocalEnumeration, 177
localEnumeration, 179
notation, 179
Notations, 177
precision, 179
QELineEdit, 177
Scientific, 177
Time, 177
trailingZeros, 180
UnsignedInteger, 177
useDbPrecision, 180
QELineEditManager, 180
QELink, 180
QELog, 182
 allowDrop, 185
 Always, 185
 displayAlarmState, 185
 displayAlarmStateOption, 185
 DisplayAlarmStateOptions, 185
 Engineer, 185
 int, 186
 Never, 185
 Scientist, 185
 User, 185
 userLevelEnabled, 186
 userLevelEngineerStyle, 186
 UserLevels, 185
 userLevelScientistStyle, 186
 userLevelUserStyle, 186
 userLevelVisibility, 187
 variableAsToolTip, 187
 visible, 187
 WhenInAlarm, 185
QELogin, 187
QELoginDialog, 188
QENumericEdit, 188
 addUnits, 191
 autoScale, 191
 dbValueChanged, 191
 leadingZeros, 191
 maximum, 191
 minimum, 191
 precision, 191
 QENumericEdit, 190
 QENumericEditManager, 192
 QEPeriodic, 192
 allowDrop, 196
 Always, 195
 dbElementChanged, 196
 dbValueChanged, 196
 displayAlarmState, 196
 displayAlarmStateOption, 196
 DisplayAlarmStateOptions, 195
 Engineer, 196
 int, 197
 Never, 195
 readbackLabelVariable1, 197
 readbackLabelVariable2, 197
 Scientist, 196
 subscribe, 197
 User, 196
 userLevelEnabled, 197
 userLevelEngineerStyle, 197
 UserLevels, 195
 userLevelScientistStyle, 198
 userLevelUserStyle, 198
 userLevelVisibility, 198
 variableAsToolTip, 198
 variableSubstitutions, 198
 visible, 198
 WhenInAlarm, 195
 writeButtonVariable1, 199
 writeButtonVariable2, 199
QEPeriodic::elementInfoStruct, 31
QEPeriodic::userInfoStructArray, 289
QEPeriodicComponentData, 199
QEPeriodicTaskMenu, 199
QEPeriodicTaskMenuFactory, 200
QEPlot, 200
 allowDrop, 205
 Always, 204
 dbValueChanged, 204
 displayAlarmState, 205
 displayAlarmStateOption, 205
 DisplayAlarmStateOptions, 204
 Engineer, 204
 int, 205
 Never, 204
 Scientist, 204
 User, 204
 userLevelEnabled, 205
 userLevelEngineerStyle, 205

UserLevels, 204
userLevelScientistStyle, 206
userLevelUserStyle, 206
userLevelVisibility, 206
variable1, 206
variable2, 206
variable3, 206
variable4, 207
variableAsToolTip, 207
variableSubstitutions, 207
visible, 207
WhenInAlarm, 204
QEPushButton, 207
 addUnits, 215
 alignment, 215
 allowDrop, 215
 altReadbackVariable, 215
 Always, 212
 Append, 211
 arguments, 215
 arrayAction, 215
 ArrayActions, 211
 Ascii, 211
 Automatic, 213
 clickCheckedText, 215
 clicked, 214
 clickText, 216
 confirmAction, 216
 confirmText, 216
 creationOption, 216
 CreationOptionNames, 211
 customisationName, 216
 dbValueChanged, 214
 Default, 212
 displayAlarmState, 217
 displayAlarmStateOption, 217
 DisplayAlarmStateOptions, 212
 DockBottom, 211
 DockBottomTabbed, 212
 DockFloating, 212
 DockLeft, 212
 DockLeftTabbed, 212
 DockRight, 212
 DockRightTabbed, 212
 DockTop, 211
 DockTopTabbed, 212
 Engineer, 213
 Fixed, 213
 Floating, 212
 format, 217
Formats, 212
guiFile, 217
Icon, 213
Index, 211
int, 217
Integer, 212
labelText, 218
leadingZero, 218
LocalEnumeration, 212
localEnumeration, 218
LogOutput, 213
Never, 212
NewTab, 211
NewWindow, 211
None, 213
notation, 219
Notations, 212
Open, 211
password, 219
 pixmap0, 219
 pixmap1, 219
 pixmap2, 219
 pixmap3, 219
 pixmap4, 219
 pixmap5, 219
 pixmap6, 219
 pixmap7, 220
precision, 220
pressed, 214
pressText, 220
prioritySubstitutions, 220
program, 220
programStartupOption, 220
ProgramStartupOptionNames, 213
QEPushButton, 214
released, 214
releaseText, 220
requestAction, 214
Scientific, 213
Scientist, 213
State, 213
StdOutput, 213
subscribe, 221
Terminal, 213
Text, 213
TextAndIcon, 213
Time, 212
trailingZeros, 221
UnsignedInteger, 212
updateOption, 221

UpdateOptions, 213
useDbPrecision, 221
User, 213
userLevelEnabled, 221
userLevelEngineerStyle, 221
UserLevels, 213
userLevelScientistStyle, 221
userLevelUserStyle, 222
userLevelVisibility, 222
variable, 222
variableAsToolTip, 222
variableSubstitutions, 222
visible, 222
WhenInAlarm, 212
writeOnClick, 223
writeOnPress, 223
writeOnRelease, 223
QEPVNameLists, 223
QEPvProperties, 223
 variable, 225
 variableSubstitutions, 225
QEPvPropertiesManager, 225
QERadioButton, 226
 addUnits, 233
 alignment, 233
 allowDrop, 233
 Always, 230
 Append, 230
 arguments, 233
 arrayAction, 233
 ArrayActions, 229
 Ascii, 230
 Automatic, 231
 clickCheckedText, 234
 clicked, 232
 clickText, 234
 confirmAction, 234
 confirmText, 234
 creationOption, 234
 CreationOptionNames, 230
 customisationName, 235
 dbValueChanged, 232
 Default, 231
 displayAlarmState, 235
 displayAlarmStateOption, 235
 DisplayAlarmStateOptions, 230
 DockBottom, 230
 DockBottomTabbed, 230
 DockFloating, 230
 DockLeft, 230
 DockLeftTabbed, 230
 DockRight, 230
 DockRightTabbed, 230
 DockTop, 230
 DockTopTabbed, 230
 Engineer, 232
 Fixed, 231
 Floating, 231
 format, 235
 Formats, 230
 guiFile, 235
 Icon, 232
 Index, 230
 int, 236
 Integer, 231
 labelText, 236
 leadingZero, 236
 LocalEnumeration, 231
 localEnumeration, 236
 LogOutput, 231
 Never, 230
 NewTab, 230
 NewWindow, 230
 None, 231
 notation, 237
 Notations, 231
 Open, 230
 password, 237
 pixmap0, 237
 pixmap1, 237
 pixmap2, 237
 pixmap3, 237
 pixmap4, 238
 pixmap5, 238
 pixmap6, 238
 pixmap7, 238
 precision, 238
 pressed, 232
 pressText, 238
 prioritySubstitutions, 238
 program, 239
 programStartupOption, 239
 ProgramStartupOptionNames, 231
 QERadioButton, 232
 released, 233
 releaseText, 239
 requestAction, 233
 Scientific, 231
 Scientist, 232
 State, 232

StdOutput, 231
subscribe, 239
Terminal, 231
Text, 232
TextAndIcon, 232
Time, 231
trailingZeros, 239
UnsignedInteger, 231
updateOption, 239
UpdateOptions, 231
useDbPrecision, 239
User, 232
userLevelEnabled, 239
userLevelEngineerStyle, 240
UserLevels, 232
userLevelScientistStyle, 240
userLevelUserStyle, 240
userLevelVisibility, 240
variable, 240
variableAsToolTip, 240
variableSubstitutions, 241
visible, 241
WhenInAlarm, 230
writeOnClick, 241
writeOnPress, 241
writeOnRelease, 241
QERecipe, 241
QERecordFieldName, 243
QERecordSpec, 244
QERecordSpecList, 244
QEScript, 244
 allowDrop, 249
 Always, 249
 displayAlarmState, 249
 displayAlarmStateOption, 249
 DisplayAlarmStateOptions, 249
 Engineer, 249
 int, 250
 Never, 249
 Scientist, 249
 User, 249
 userLevelEnabled, 250
 userLevelEngineerStyle, 250
 UserLevels, 249
 userLevelScientistStyle, 250
 userLevelUserStyle, 250
 userLevelVisibility, 251
 variableAsToolTip, 251
 visible, 251
 WhenInAlarm, 249
QEShape, 251
 allowDrop, 258
 Always, 256
 animation1, 258
 animation2, 258
 animation3, 258
 animation4, 258
 animation5, 258
 animation6, 258
 animationOptions, 256
 color1, 258
 color10, 259
 color2, 259
 color3, 259
 color4, 259
 color5, 259
 color6, 259
 color7, 259
 color8, 259
 color9, 259
 dbValueChanged1, 257
 dbValueChanged2, 257
 dbValueChanged3, 257
 dbValueChanged4, 257
 dbValueChanged5, 257
 dbValueChanged6, 257
 displayAlarmState, 260
 displayAlarmStateOption, 260
 DisplayAlarmStateOptions, 256
 Engineer, 256
 int, 260
 Never, 256
 offset1, 260
 offset2, 260
 offset3, 260
 offset4, 261
 offset5, 261
 offset6, 261
 point1, 261
 point10, 261
 point2, 261
 point3, 261
 point4, 261
 point5, 261
 point6, 262
 point7, 262
 point8, 262
 point9, 262
 QEShape, 257
 scale2, 262

scale3, 262
scale4, 262
scale5, 262
scale6, 262
Scientist, 256
shapeOptions, 256
User, 256
userLevelEnabled, 263
userLevelEngineerStyle, 263
UserLevels, 256
userLevelScientistStyle, 263
userLevelUserStyle, 263
userLevelVisibility, 263
variable1, 264
variable2, 264
variable3, 264
variable4, 264
variable5, 264
variable6, 264
variableAsToolTip, 264
variableSubstitutions, 265
visible, 265
WhenInAlarm, 256
QESlider, 265
allowDrop, 268
Always, 267
dbValueChanged, 268
displayAlarmState, 268
displayAlarmStateOption, 268
DisplayAlarmStateOptions, 267
Engineer, 268
int, 269
Never, 267
Scientist, 268
subscribe, 269
User, 268
userLevelEnabled, 269
userLevelEngineerStyle, 269
UserLevels, 267
userLevelScientistStyle, 269
userLevelUserStyle, 270
userLevelVisibility, 270
variable, 270
variableAsToolTip, 270
variableSubstitutions, 270
visible, 270
WhenInAlarm, 267
writeOnChange, 268
QESpinBox, 271
allowDrop, 274
Always, 273
dbValueChanged, 274
displayAlarmState, 274
displayAlarmStateOption, 274
DisplayAlarmStateOptions, 273
Engineer, 273
int, 274
Never, 273
Scientist, 273
subscribe, 274
User, 273
userLevelEnabled, 275
userLevelEngineerStyle, 275
UserLevels, 273
userLevelScientistStyle, 275
userLevelUserStyle, 275
userLevelVisibility, 275
variable, 276
variableAsToolTip, 276
variableSubstitutions, 276
visible, 276
WhenInAlarm, 273
QEStripChart, 276
variableSubstitutions, 278
QEStripChartAdjustPVDialog, 279
QEStripChartContextMenu, 279
QEStripChartContextMenu, 279
QEStripChartDurationDialog, 280
QEStripChartItem, 280
QEStripChartNames, 281
QEStripChartPushButtonSpecifications, 282
QEStripChartRangeDialog, 282
QEStripChartState, 283
QEStripChartStateList, 283
QEStripChartStatistics, 284
QEStripChartTimeDialog, 284
QEStripChartToolBar, 284
QEStripChartToolBar::OwnWidgets, 51
QESubstitutedLabel, 285
labelText, 286
textSubstitutions, 286
readbackLabelVariable1
QEPeriodic, 197
readbackLabelVariable2
QEPeriodic, 197
recording, 286
regionOfInterest1HVariable
QEImage, 158
regionOfInterest1WVariable

QEImage, 158
 regionOfInterest1XVariable
 QEImage, 158
 regionOfInterest1YVariable
 QEImage, 158
 regionOfInterest2HVariable
 QEImage, 159
 regionOfInterest2WVariable
 QEImage, 159
 regionOfInterest2XVariable
 QEImage, 159
 regionOfInterest2YVariable
 QEImage, 159
 regionOfInterest3HVariable
 QEImage, 159
 regionOfInterest3WVariable
 QEImage, 159
 regionOfInterest3XVariable
 QEImage, 159
 regionOfInterest3YVariable
 QEImage, 159
 regionOfInterest4HVariable
 QEImage, 159
 regionOfInterest4WVariable
 QEImage, 160
 regionOfInterest4XVariable
 QEImage, 160
 regionOfInterest4YVariable
 QEImage, 160
 released
 QECheckBox, 83
 QEPushButton, 214
 QERadioButton, 233
 releaseText
 QECheckBox, 89
 QEPushButton, 220
 QERadioButton, 239
 requestAction
 QECheckBox, 83
 QEPushButton, 214
 QERadioButton, 233
 RESIZE_OPTION_FIT
 QEImage, 147
 RESIZE_OPTION_ZOOM
 QEImage, 147
 resizeOption
 QEImage, 160
 ResizeOptions
 QEImage, 147
 resizeOptions

QEImage, 147
 rgb1
 QEImage, 146
 rgb2
 QEImage, 146
 rgb3
 QEImage, 146
 Right_To_Left
 QEAnalogIndicator, 59
 Rotate180
 QEImage, 147
 Rotate90Left
 QEImage, 147
 Rotate90Right
 QEImage, 147
 rotation
 QEImage, 160
 ROTATION_0
 QEImage, 147
 ROTATION_180
 QEImage, 147
 ROTATION_90_LEFT
 QEImage, 147
 ROTATION_90_RIGHT
 QEImage, 147
 RotationOptions
 QEImage, 147
 rotationOptions
 QEImage, 147

Scale
 QEAnalogIndicator, 58
 scale2
 QEShape, 262
 scale3
 QEShape, 262
 scale4
 QEShape, 262
 scale5
 QEShape, 262
 scale6
 QEShape, 262

Scientific
 QEAnalogProgressBar, 65
 QECheckBox, 81
 QELabel, 168
 QELineEdit, 177
 QEPushButton, 213
 QERadioButton, 231

Scientist

QEAnalogProgressBar, 65
QEBitStatus, 73
QECheckBox, 82
QEComboBox, 95
QEConfiguredLayout, 100
QEFileBrowser, 106
QEFrame, 112
QEGenericEdit, 119
QEGroupBox, 126
QEImage, 148
QELabel, 169
QELog, 185
QEPeriodic, 196
QEPlot, 204
QEPushButton, 213
QERadioButton, 232
QEScript, 249
QEShape, 256
QESlider, 268
QESpinBox, 273
screenSelectDialog, 287
selected
 QEFileBrowser, 107
selectMenu, 288
selectOptions
 QEImage, 147
setConfirmWrite
 QEGenericEdit, 120
setSubscribe
 QEGenericEdit, 120
setWriteOnEnter
 QEGenericEdit, 121
setWriteOnFinish
 QEGenericEdit, 121
setWriteOnLoseFocus
 QEGenericEdit, 121
shapeOptions
 QEShape, 256
showScale
 QEAnalogIndicator, 60
showText
 QEAnalogIndicator, 60
showTime
 QEImage, 160
SO_AREA4
 QEImage, 148
SO_BEAM
 QEImage, 148
SO_HSLICE
 QEImage, 148
SO_NONE
 QEImage, 148
SO_PANNING
 QEImage, 148
SO_PROFILE
 QEImage, 148
SO_TARGET
 QEImage, 148
SO_VSLICE
 QEImage, 148
SolidSmallCrosshair
 QEImage, 148
spanAngle
 QEAnalogIndicator, 60
State
 QECheckBox, 82
 QEPushButton, 213
 QERadioButton, 232
StdOutput
 QECheckBox, 81
 QEImage, 147
 QEPushButton, 213
 QERadioButton, 231
subscribe
 QECheckBox, 89
 QEComboBox, 96
 QEGenericEdit, 122
 QEPeriodic, 197
 QEPushButton, 221
 QERadioButton, 239
 QESlider, 269
 QESpinBox, 274
substitutedTitle
 QEGroupBox, 127
targetColor
 QEImage, 160
TargetOptions
 QEImage, 148
targetTriggerVariable
 QEImage, 160
targetXVariable
 QEImage, 160
targetYVariable
 QEImage, 161
Terminal
 QECheckBox, 81
 QEImage, 146
 QEPushButton, 213
 QERadioButton, 231

Text
 QECheckBox, 82
 QELabel, 169
 QEPushButton, 213
 QERadioButton, 232

TextAndIcon
 QECheckBox, 82
 QEPushButton, 213
 QERadioButton, 232

textSubstitutions
 QEGroupBox, 127
 QESubstitutedLabel, 286

Time
 QEAnalogProgressBar, 65
 QECheckBox, 81
 QELabel, 168
 QELineEdit, 177
 QEPushButton, 212
 QERadioButton, 231

timeColor
 QEImage, 161

Top_To_Bottom
 QEAnalogIndicator, 59

trace, 288

trailingZeros
 QEAnalogProgressBar, 68
 QECheckBox, 89
 QELabel, 173
 QELineEdit, 180
 QEPushButton, 221
 QERadioButton, 239

UnsignedInteger
 QEAnalogProgressBar, 65
 QECheckBox, 81
 QELabel, 168
 QELineEdit, 177
 QEPushButton, 212
 QERadioButton, 231

UPDATE_PIXMAP
 QELabel, 169

UPDATE_TEXT
 QELabel, 169

updateImage
 mpegSource, 50

updateOption
 QECheckBox, 89
 QELabel, 173
 QEPushButton, 221
 QERadioButton, 239

UpdateOptions
 QECheckBox, 81
 QELabel, 169
 QEPushButton, 213
 QERadioButton, 231

updateOptions
 QELabel, 168

URL
 QEImage, 161

useDbDisplayLimits
 QEAnalogProgressBar, 68

useDbEnumerations
 QEComboBox, 95

useDbPrecision
 QEAnalogProgressBar, 68
 QECheckBox, 89
 QELabel, 173
 QELineEdit, 180
 QEPushButton, 221
 QERadioButton, 239

useFalseColour
 QEImage, 161

User
 QEAnalogProgressBar, 65
 QEBitStatus, 73
 QECheckBox, 82
 QEComboBox, 95
 QEConfiguredLayout, 100
 QEFileBrowser, 106
 QEFrame, 112
 QEGenericEdit, 119
 QEGroupBox, 126
 QEImage, 148
 QELabel, 169
 QELog, 185
 QEPeriodic, 196
 QEPlot, 204
 QEPushButton, 213
 QERadioButton, 232
 QEScript, 249
 QEShape, 256
 QESlider, 268
 QESpinBox, 273
 userInfoStruct, 288
 userLevelEnabled
 QEAnalogProgressBar, 69
 QEBitStatus, 74
 QECheckBox, 89
 QEComboBox, 96
 QEConfiguredLayout, 101

QEFileBrowser, 107
QEFrame, 113
QEGenericEdit, 122
QEGroupBox, 127
QEImage, 161
QELabel, 173
QELog, 186
QEPeriodic, 197
QEPlot, 205
QEPushButton, 221
QERadioButton, 239
QEScript, 250
QEShape, 263
QESlider, 269
QESpinBox, 275
userLevelEngineerStyle
QEAnalogProgressBar, 69
QEBitStatus, 74
QECheckBox, 90
QEComboBox, 96
QEConfiguredLayout, 101
QEFileBrowser, 108
QEFrame, 113
QEGenericEdit, 122
QEGroupBox, 127
QEImage, 161
QELabel, 174
QELog, 186
QEPeriodic, 197
QEPlot, 205
QEPushButton, 221
QERadioButton, 240
QEScript, 250
QEShape, 263
QESlider, 269
QESpinBox, 275
userLevelUserStyle
QEAnalogProgressBar, 69
QEBitStatus, 75
QECheckBox, 90
QEComboBox, 97
QEConfiguredLayout, 102
QEFileBrowser, 108
QEFrame, 114
QEGenericEdit, 123
QEGroupBox, 127
QEImage, 162
QELabel, 174
QELog, 186
QEPeriodic, 198
QEPlot, 206
QEPushButton, 222
QERadioButton, 240
QEScript, 250
QEShape, 263
QESlider, 270
QESpinBox, 275
userLevelVisibility

QEAnalogProgressBar, 69
 QEBitStatus, 75
 QECheckBox, 90
 QEComboBox, 97
 QEConfiguredLayout, 102
 QEFileBrowser, 108
 QEFrame, 114
 QEGenericEdit, 123
 QEGroupBox, 128
 QEImage, 162
 QELabel, 174
 QELog, 187
 QEPeriodic, 198
 QEPlot, 206
 QEPushButton, 222
 QERadioButton, 240
 QEScript, 251
 QEShape, 263
 QESlider, 270
 QESpinBox, 275

value
 QEAnalogIndicator, 60
 QEAnalogProgressBar, 70

ValueScaling, 289

variable
 QEAnalogProgressBar, 70
 QEBitStatus, 75
 QECheckBox, 90
 QEComboBox, 97
 QEFileBrowser, 108
 QEGenericEdit, 123
 QELabel, 174
 QEPushButton, 222
 QEProperties, 225
 QERadioButton, 240
 QESlider, 270
 QESpinBox, 276

variable1
 QEPlot, 206
 QEShape, 264

variable2
 QEPlot, 206
 QEShape, 264

variable3
 QEPlot, 206
 QEShape, 264

variable4
 QEPlot, 207
 QEShape, 264

variable5
 QEShape, 264

variable6
 QEShape, 264

variableAsToolTip
 QEAnalogProgressBar, 70
 QEBitStatus, 75
 QECheckBox, 90
 QEComboBox, 97
 QEConfiguredLayout, 102
 QEFileBrowser, 109
 QEFrame, 114
 QEGenericEdit, 123
 QEGroupBox, 128
 QEImage, 162
 QELabel, 174
 QELog, 187
 QEPeriodic, 198
 QEPlot, 207
 QEPushButton, 222
 QERadioButton, 240
 QEScript, 251
 QEShape, 264
 QESlider, 270
 QESpinBox, 276

variableSubstitutions
 QEAnalogProgressBar, 70
 QEBitStatus, 75
 QECheckBox, 91
 QEComboBox, 97
 QEFileBrowser, 109
 QEGenericEdit, 123
 QEImage, 162
 QELabel, 175
 QEPeriodic, 198
 QEPlot, 207
 QEPushButton, 222
 QEProperties, 225
 QERadioButton, 241
 QEShape, 265
 QESlider, 270
 QESpinBox, 276
 QEStripChart, 278

verticalFlip
 QEImage, 162

vertSliceColor
 QEImage, 162

VideoWidget, 289

visible
 QEAnalogProgressBar, 70

QEBitStatus, 75
QECheckBox, 91
QEComboBox, 98
QEConfiguredLayout, 102
QEFileBrowser, 109
QEFrame, 114
QEGenericEdit, 123
QEGroupBox, 128
QEImage, 162
QELabel, 175
QELog, 187
QEPeriodic, 198
QEPlot, 207
QEPushButton, 222
QERadioButton, 241
QEScript, 251
QEShape, 265
QESlider, 270
QESpinBox, 276

WhenInAlarm
QEAnalogProgressBar, 64
QEBitStatus, 73
QECheckBox, 80
QEComboBox, 94
QEConfiguredLayout, 100
QEFileBrowser, 106
QEFrame, 112
QEGenericEdit, 119
QEGroupBox, 126
QEImage, 145
QELabel, 168
QELog, 185
QEPeriodic, 195
QEPlot, 204
QEPushButton, 212
QERadioButton, 230
QEScript, 249
QEShape, 256
QESlider, 267
QESpinBox, 273

widthVariable
QEImage, 163

writeButtonVariable1
QEPeriodic, 199

writeButtonVariable2
QEPeriodic, 199

writeOnChange
QEComboBox, 95
QESlider, 268

writeOnClick
QECheckBox, 91
QEPushButton, 223
QERadioButton, 241

writeOnEnter
QEGenericEdit, 124

writeOnFinish
QEGenericEdit, 124

writeOnLoseFocus
QEGenericEdit, 124

writeOnPress
QECheckBox, 91
QEPushButton, 223
QERadioButton, 241

writeOnRelease
QECheckBox, 91
QEPushButton, 223
QERadioButton, 241

yuv422
QEImage, 146

yuv444
QEImage, 146

Zoom
QEImage, 147

zoomMenu, 291