

EPICS QT Framework

3.1.0

Generated by Doxygen 1.7.4

Mon May 18 2015 15:23:07

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	imageMarkupLegendSetText Class Reference	40
9.25	imageProcessor Class Reference	41
9.25.1	Detailed Description	43
9.25.2	Member Function Documentation	44
9.25.2.1	getPixelValueFromData	44
9.26	imageProperties Class Reference	44
9.26.1	Detailed Description	46
9.26.2	Member Enumeration Documentation	46
9.26.2.1	rotationOptions	46
9.26.3	Constructor & Destructor Documentation	46
9.26.3.1	imageProperties	46
9.27	imagePropertiesCore Class Reference	46

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

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

9.51.4.2	alarmSeverityDisplayStyle	74
9.51.4.3	allowDrop	74
9.51.4.4	arrayAction	74
9.51.4.5	defaultStyle	74
9.51.4.6	displayAlarmState	75
9.51.4.7	displayAlarmStateOption	75
9.51.4.8	format	75
9.51.4.9	int	75
9.51.4.10	leadingZero	75
9.51.4.11	localEnumeration	75
9.51.4.12	notation	76
9.51.4.13	precision	76
9.51.4.14	styleSheet	76
9.51.4.15	trailingZeros	76
9.51.4.16	useDbDisplayLimits	77
9.51.4.17	useDbPrecision	77
9.51.4.18	userLevelEnabled	77
9.51.4.19	userLevelEngineerStyle	77
9.51.4.20	userLevelScientistStyle	77
9.51.4.21	userLevelUserStyle	77
9.51.4.22	userLevelVisibility	78
9.51.4.23	value	78
9.51.4.24	variable	78
9.51.4.25	variableAsToolTip	78
9.51.4.26	variableSubstitutions	78
9.51.4.27	visible	78
9.52	QEBitStatus Class Reference	78
9.52.1	Member Enumeration Documentation	81
9.52.1.1	DisplayAlarmStateOptions	81
9.52.1.2	UserLevels	81
9.52.2	Member Function Documentation	81
9.52.2.1	dbValueChanged	81
9.52.2.2	setManagedVisible	81
9.52.3	Property Documentation	82

9.52.3.1	allowDrop	82
9.52.3.2	defaultStyle	82
9.52.3.3	displayAlarmState	82
9.52.3.4	displayAlarmStateOption	82
9.52.3.5	int	82
9.52.3.6	styleSheet	82
9.52.3.7	userLevelEnabled	83
9.52.3.8	userLevelEngineerStyle	83
9.52.3.9	userLevelScientistStyle	83
9.52.3.10	userLevelUserStyle	83
9.52.3.11	userLevelVisibility	83
9.52.3.12	variable	84
9.52.3.13	variableAsToolTip	84
9.52.3.14	variableSubstitutions	84
9.52.3.15	visible	84
9.53	QECheckBox Class Reference	84
9.53.1	Member Enumeration Documentation	88
9.53.1.1	ArrayActions	88
9.53.1.2	CreationOptionNames	88
9.53.1.3	DisplayAlarmStateOptions	89
9.53.1.4	Formats	89
9.53.1.5	Notations	90
9.53.1.6	ProgramStartupOptionNames	90
9.53.1.7	UpdateOptions	90
9.53.1.8	UserLevels	90
9.53.2	Constructor & Destructor Documentation	91
9.53.2.1	QECheckBox	91
9.53.2.2	QECheckBox	91
9.53.3	Member Function Documentation	91
9.53.3.1	clicked	91
9.53.3.2	dbValueChanged	91
9.53.3.3	pressed	91
9.53.3.4	released	91
9.53.3.5	requestAction	91

9.53.3.6	setManagedVisible	92
9.53.4	Property Documentation	92
9.53.4.1	addUnits	92
9.53.4.2	alignment	92
9.53.4.3	allowDrop	92
9.53.4.4	arguments	92
9.53.4.5	arrayAction	92
9.53.4.6	clickCheckedText	93
9.53.4.7	clickText	93
9.53.4.8	confirmAction	93
9.53.4.9	confirmText	93
9.53.4.10	creationOption	93
9.53.4.11	customisationName	93
9.53.4.12	defaultStyle	94
9.53.4.13	displayAlarmState	94
9.53.4.14	displayAlarmStateOption	94
9.53.4.15	format	94
9.53.4.16	guiFile	94
9.53.4.17	int	95
9.53.4.18	labelText	95
9.53.4.19	leadingZero	95
9.53.4.20	localEnumeration	95
9.53.4.21	notation	96
9.53.4.22	password	96
9.53.4.23	pixmap0	96
9.53.4.24	pixmap1	96
9.53.4.25	pixmap2	96
9.53.4.26	pixmap3	96
9.53.4.27	pixmap4	97
9.53.4.28	pixmap5	97
9.53.4.29	pixmap6	97
9.53.4.30	pixmap7	97
9.53.4.31	precision	97
9.53.4.32	pressText	97

9.53.4.33 prioritySubstitutions	97
9.53.4.34 program	98
9.53.4.35 programStartupOption	98
9.53.4.36 releaseText	98
9.53.4.37 styleSheet	98
9.53.4.38 subscribe	98
9.53.4.39 trailingZeros	98
9.53.4.40 updateOption	98
9.53.4.41 useDbPrecision	98
9.53.4.42 userLevelEnabled	99
9.53.4.43 userLevelEngineerStyle	99
9.53.4.44 userLevelScientistStyle	99
9.53.4.45 userLevelUserStyle	99
9.53.4.46 userLevelVisibility	99
9.53.4.47 variable	100
9.53.4.48 variableAsToolTip	100
9.53.4.49 variableSubstitutions	100
9.53.4.50 visible	100
9.53.4.51 writeOnClick	100
9.53.4.52 writeOnPress	100
9.53.4.53 writeOnRelease	100
9.54 QECheckBoxManager Class Reference	101
9.55 QEComboBox Class Reference	101
9.55.1 Member Enumeration Documentation	103
9.55.1.1 DisplayAlarmStateOptions	103
9.55.1.2 UserLevels	104
9.55.2 Member Function Documentation	104
9.55.2.1 dbValueChanged	104
9.55.2.2 setManagedVisible	104
9.55.3 Member Data Documentation	104
9.55.3.1 useDbEnumerations	104
9.55.3.2 writeOnChange	104
9.55.4 Property Documentation	105
9.55.4.1 allowDrop	105

9.55.4.2	defaultStyle	105
9.55.4.3	displayAlarmState	105
9.55.4.4	displayAlarmStateOption	105
9.55.4.5	int	105
9.55.4.6	localEnumeration	105
9.55.4.7	styleSheet	106
9.55.4.8	subscribe	106
9.55.4.9	userLevelEnabled	106
9.55.4.10	userLevelEngineerStyle	106
9.55.4.11	userLevelScientistStyle	106
9.55.4.12	userLevelUserStyle	106
9.55.4.13	userLevelVisibility	107
9.55.4.14	variable	107
9.55.4.15	variableAsToolTip	107
9.55.4.16	variableSubstitutions	107
9.55.4.17	visible	107
9.56	QEConfiguredLayout Class Reference	107
9.56.1	Member Enumeration Documentation	110
9.56.1.1	DisplayAlarmStateOptions	110
9.56.1.2	UserLevels	110
9.56.2	Member Function Documentation	110
9.56.2.1	setManagedVisible	110
9.56.3	Property Documentation	110
9.56.3.1	allowDrop	110
9.56.3.2	defaultStyle	110
9.56.3.3	displayAlarmState	111
9.56.3.4	displayAlarmStateOption	111
9.56.3.5	int	111
9.56.3.6	styleSheet	111
9.56.3.7	userLevelEnabled	111
9.56.3.8	userLevelEngineerStyle	111
9.56.3.9	userLevelScientistStyle	112
9.56.3.10	userLevelUserStyle	112
9.56.3.11	userLevelVisibility	112

9.56.3.12 variableAsToolTip	112
9.56.3.13 visible	112
9.57 QEConfiguredLayoutManager Class Reference	113
9.58 QEFileBrowser Class Reference	113
9.58.1 Detailed Description	116
9.58.2 Member Enumeration Documentation	116
9.58.2.1 DisplayAlarmStateOptions	116
9.58.2.2 UserLevels	116
9.58.3 Member Function Documentation	117
9.58.3.1 selected	117
9.58.3.2 setManagedVisible	117
9.58.4 Property Documentation	117
9.58.4.1 allowDrop	117
9.58.4.2 defaultStyle	117
9.58.4.3 displayAlarmState	117
9.58.4.4 displayAlarmStateOption	118
9.58.4.5 int	118
9.58.4.6 styleSheet	118
9.58.4.7 userLevelEnabled	118
9.58.4.8 userLevelEngineerStyle	118
9.58.4.9 userLevelScientistStyle	118
9.58.4.10 userLevelUserStyle	119
9.58.4.11 userLevelVisibility	119
9.58.4.12 variable	119
9.58.4.13 variableAsToolTip	119
9.58.4.14 variableSubstitutions	119
9.58.4.15 visible	119
9.59 QEForm Class Reference	120
9.59.1 Member Data Documentation	122
9.59.1.1 handleGuiLaunchRequests	122
9.59.1.2 resizeContents	122
9.59.2 Property Documentation	122
9.59.2.1 allowDrop	122
9.59.2.2 displayAlarmStateOption	122

9.59.2.3 int	123
9.59.2.4 messageFormFilter	123
9.59.2.5 messageSourceFilter	123
9.59.2.6 uiFile	123
9.59.2.7 variableAsToolTip	123
9.59.2.8 variableSubstitutions	123
9.60 QEFrame Class Reference	124
9.60.1 Member Enumeration Documentation	125
9.60.1.1 DisplayAlarmStateOptions	125
9.60.1.2 UserLevels	125
9.60.2 Member Function Documentation	126
9.60.2.1 setManagedVisible	126
9.60.3 Property Documentation	126
9.60.3.1 allowDrop	126
9.60.3.2 defaultStyle	126
9.60.3.3 displayAlarmState	126
9.60.3.4 displayAlarmStateOption	126
9.60.3.5 int	126
9.60.3.6 styleSheet	127
9.60.3.7 userLevelEnabled	127
9.60.3.8 userLevelEngineerStyle	127
9.60.3.9 userLevelScientistStyle	127
9.60.3.10 userLevelUserStyle	127
9.60.3.11 userLevelVisibility	128
9.60.3.12 variableAsToolTip	128
9.60.3.13 visible	128
9.61 QEGenericButton Class Reference	128
9.62 QEGenericEdit Class Reference	130
9.62.1 Member Enumeration Documentation	133
9.62.1.1 DisplayAlarmStateOptions	133
9.62.1.2 UserLevels	133
9.62.2 Constructor & Destructor Documentation	133
9.62.2.1 QEGenericEdit	133
9.62.2.2 QEGenericEdit	134

9.62.3 Member Function Documentation	134
9.62.3.1 getConfirmWrite	134
9.62.3.2 getSubscribe	134
9.62.3.3 getWriteOnEnter	134
9.62.3.4 getWriteOnFinish	134
9.62.3.5 getWriteOnLoseFocus	134
9.62.3.6 setConfirmWrite	134
9.62.3.7 setManagedVisible	134
9.62.3.8 setSubscribe	135
9.62.3.9 setWriteOnEnter	135
9.62.3.10 setWriteOnFinish	135
9.62.3.11 setWriteOnLoseFocus	135
9.62.4 Property Documentation	135
9.62.4.1 allowDrop	135
9.62.4.2 confirmWrite	135
9.62.4.3 defaultStyle	135
9.62.4.4 displayAlarmState	135
9.62.4.5 displayAlarmStateOption	136
9.62.4.6 int	136
9.62.4.7 styleSheet	136
9.62.4.8 subscribe	136
9.62.4.9 userLevelEnabled	136
9.62.4.10 userLevelEngineerStyle	137
9.62.4.11 userLevelScientistStyle	137
9.62.4.12 userLevelUserStyle	137
9.62.4.13 userLevelVisibility	137
9.62.4.14 variable	137
9.62.4.15 variableAsToolTip	137
9.62.4.16 variableSubstitutions	138
9.62.4.17 visible	138
9.62.4.18 writeOnEnter	138
9.62.4.19 writeOnFinish	138
9.62.4.20 writeOnLoseFocus	138
9.63 QEGroupBox Class Reference	138

9.63.1 Member Enumeration Documentation	140
9.63.1.1 DisplayAlarmStateOptions	140
9.63.1.2 UserLevels	140
9.63.2 Member Function Documentation	140
9.63.2.1 setManagedVisible	140
9.63.3 Property Documentation	140
9.63.3.1 allowDrop	140
9.63.3.2 defaultStyle	141
9.63.3.3 displayAlarmState	141
9.63.3.4 displayAlarmStateOption	141
9.63.3.5 int	141
9.63.3.6 styleSheet	141
9.63.3.7 substitutedTitle	141
9.63.3.8 textSubstitutions	142
9.63.3.9 userLevelEnabled	142
9.63.3.10 userLevelEngineerStyle	142
9.63.3.11 userLevelScientistStyle	142
9.63.3.12 userLevelUserStyle	142
9.63.3.13 userLevelVisibility	142
9.63.3.14 variableAsToolTip	143
9.63.3.15 visible	143
9.64 QEImage Class Reference	143
9.64.1 Detailed Description	166
9.64.2 Member Enumeration Documentation	167
9.64.2.1 DisplayAlarmStateOptions	167
9.64.2.2 EllipseVariableDefinitions	167
9.64.2.3 ellipseVariableDefinitions	167
9.64.2.4 FormatOptions	167
9.64.2.5 ProgramStartupOptionNames	168
9.64.2.6 ResizeOptions	168
9.64.2.7 resizeOptions	168
9.64.2.8 RotationOptions	168
9.64.2.9 selectOptions	169
9.64.2.10 TargetOptions	169

9.64.2.11	UserLevels	169
9.64.3	Constructor & Destructor Documentation	170
9.64.3.1	QEImage	170
9.64.3.2	QEImage	170
9.64.4	Member Function Documentation	170
9.64.4.1	dbValueChanged	170
9.64.4.2	setImageFile	170
9.64.4.3	setManagedVisible	170
9.64.5	Member Data Documentation	170
9.64.5.1	displayButtonBar	170
9.64.5.2	initialVertScrollPos	171
9.64.6	Property Documentation	171
9.64.6.1	allowDrop	171
9.64.6.2	areaColor	171
9.64.6.3	arguments1	171
9.64.6.4	arguments2	171
9.64.6.5	autoBrightnessContrast	171
9.64.6.6	beamColor	171
9.64.6.7	beamXVariable	171
9.64.6.8	beamYVariable	171
9.64.6.9	bitDepthVariable	172
9.64.6.10	briefInfoArea	172
9.64.6.11	clippingHighVariable	172
9.64.6.12	clippingLowVariable	172
9.64.6.13	clippingOnOffVariable	172
9.64.6.14	contrastReversal	172
9.64.6.15	defaultStyle	172
9.64.6.16	dimension1Variable	172
9.64.6.17	dimension2Variable	172
9.64.6.18	dimension3Variable	173
9.64.6.19	dimensionsVariable	173
9.64.6.20	displayAlarmState	173
9.64.6.21	displayAlarmStateOption	173
9.64.6.22	displayArea1Selection	173

9.64.6.23 displayArea2Selection	173
9.64.6.24 displayArea3Selection	173
9.64.6.25 displayArea4Selection	174
9.64.6.26 displayBeamSelection	174
9.64.6.27 displayCursorPixelInfo	174
9.64.6.28 displayEllipse	174
9.64.6.29 displayHozSlice1Selection	174
9.64.6.30 displayHozSlice2Selection	174
9.64.6.31 displayHozSlice3Selection	174
9.64.6.32 displayHozSlice4Selection	174
9.64.6.33 displayHozSlice5Selection	174
9.64.6.34 displayProfileSelection	175
9.64.6.35 displayTargetSelection	175
9.64.6.36 displayVertSliceSelection	175
9.64.6.37 ellipseColor	175
9.64.6.38 ellipseHVariable	175
9.64.6.39 ellipseWVariable	175
9.64.6.40 ellipseXVariable	175
9.64.6.41 ellipseYVariable	176
9.64.6.42 enableArea1Selection	176
9.64.6.43 enableArea2Selection	176
9.64.6.44 enableArea3Selection	176
9.64.6.45 enableArea4Selection	176
9.64.6.46 enableBeamSelection	176
9.64.6.47 enableHozSlice1Selection	176
9.64.6.48 enableHozSlice2Selection	176
9.64.6.49 enableHozSlice3Selection	177
9.64.6.50 enableHozSlice4Selection	177
9.64.6.51 enableHozSlice5Selection	177
9.64.6.52 enableProfileSelection	177
9.64.6.53 enableTargetSelection	177
9.64.6.54 enableVertSlice1Selection	177
9.64.6.55 enableVertSlice2Selection	177
9.64.6.56 enableVertSlice3Selection	178

9.64.6.57 enableVertSlice4Selection	178
9.64.6.58 enableVertSlice5Selection	178
9.64.6.59 externalControls	178
9.64.6.60 formatOption	178
9.64.6.61 formatVariable	178
9.64.6.62 heightVariable	178
9.64.6.63 horizontalFlip	178
9.64.6.64 hozSlice1Color	179
9.64.6.65 hozSlice2Color	179
9.64.6.66 hozSlice3Color	179
9.64.6.67 hozSlice4Color	179
9.64.6.68 hozSlice5Color	179
9.64.6.69 imageVariable	179
9.64.6.70 initialHosScrollPos	179
9.64.6.71 int	179
9.64.6.72 lineProfileArrayVariable	180
9.64.6.73 lineProfileThicknessVariable	180
9.64.6.74 lineProfileX1Variable	180
9.64.6.75 lineProfileX2Variable	180
9.64.6.76 lineProfileY1Variable	180
9.64.6.77 lineProfileY2Variable	180
9.64.6.78 logBrightness	180
9.64.6.79 profileColor	180
9.64.6.80 profileHoz1ThicknessVariable	180
9.64.6.81 profileHoz1Variable	181
9.64.6.82 profileHoz2ThicknessVariable	181
9.64.6.83 profileHoz2Variable	181
9.64.6.84 profileHoz3ThicknessVariable	181
9.64.6.85 profileHoz3Variable	181
9.64.6.86 profileHoz4ThicknessVariable	181
9.64.6.87 profileHoz4Variable	181
9.64.6.88 profileHoz5ThicknessVariable	181
9.64.6.89 profileHoz5Variable	181
9.64.6.90 profileHozArrayVariable	182

9.64.6.91 profileVert1ThicknessVariable	182
9.64.6.92 profileVert1Variable	182
9.64.6.93 profileVert2ThicknessVariable	182
9.64.6.94 profileVert2Variable	182
9.64.6.95 profileVert3ThicknessVariable	182
9.64.6.96 profileVert3Variable	182
9.64.6.97 profileVert4ThicknessVariable	182
9.64.6.98 profileVert4Variable	182
9.64.6.99 profileVert5ThicknessVariable	183
9.64.6.100profileVert5Variable	183
9.64.6.101profileVertArrayVariable	183
9.64.6.102program1	183
9.64.6.103program2	183
9.64.6.104programStartupOption1	183
9.64.6.105programStartupOption2	183
9.64.6.106regionOfInterest1HVariable	184
9.64.6.107regionOfInterest1WVariable	184
9.64.6.108regionOfInterest1XVariable	184
9.64.6.109regionOfInterest1YVariable	184
9.64.6.110regionOfInterest2HVariable	184
9.64.6.111regionOfInterest2WVariable	184
9.64.6.112regionOfInterest2XVariable	184
9.64.6.113regionOfInterest2YVariable	184
9.64.6.114regionOfInterest3HVariable	184
9.64.6.115regionOfInterest3WVariable	185
9.64.6.116regionOfInterest3XVariable	185
9.64.6.117regionOfInterest3YVariable	185
9.64.6.118regionOfInterest4HVariable	185
9.64.6.119regionOfInterest4WVariable	185
9.64.6.120regionOfInterest4XVariable	185
9.64.6.121regionOfInterest4YVariable	185
9.64.6.122resizeOption	185
9.64.6.123rotation	185
9.64.6.124showTime	186

9.64.6.125styleSheet	186
9.64.6.126targetColor	186
9.64.6.127targetTriggerVariable	186
9.64.6.128targetXVariable	186
9.64.6.129targetYVariable	186
9.64.6.130timeColor	186
9.64.6.131URL	186
9.64.6.132useFalseColour	186
9.64.6.133userLevelEnabled	187
9.64.6.134userLevelEngineerStyle	187
9.64.6.135userLevelScientistStyle	187
9.64.6.136userLevelUserStyle	187
9.64.6.137userLevelVisibility	187
9.64.6.138variableAsToolTip	188
9.64.6.139variableSubstitutions	188
9.64.6.140verticalFlip	188
9.64.6.141vertSlice1Color	188
9.64.6.142vertSlice2Color	188
9.64.6.143vertSlice3Color	188
9.64.6.144vertSlice4Color	188
9.64.6.145vertSlice5Color	188
9.64.6.146visible	188
9.64.6.147widthVariable	189
9.65 QEImageMarkupThickness Class Reference	189
9.66 QEImageOptionsDialog Class Reference	189
9.67 QELabel Class Reference	190
9.67.1 Detailed Description	193
9.67.2 Member Enumeration Documentation	194
9.67.2.1 ArrayActions	194
9.67.2.2 DisplayAlarmStateOptions	194
9.67.2.3 Formats	194
9.67.2.4 Notations	194
9.67.2.5 updateOptions	195
9.67.2.6 UpdateOptions	195

9.67.2.7	UserLevels	195
9.67.3	Constructor & Destructor Documentation	195
9.67.3.1	QELabel	195
9.67.3.2	QELabel	195
9.67.4	Member Function Documentation	196
9.67.4.1	dbValueChanged	196
9.67.4.2	setManagedVisible	196
9.67.5	Property Documentation	196
9.67.5.1	addUnits	196
9.67.5.2	allowDrop	196
9.67.5.3	arrayAction	196
9.67.5.4	defaultStyle	196
9.67.5.5	displayAlarmState	197
9.67.5.6	displayAlarmStateOption	197
9.67.5.7	format	197
9.67.5.8	int	197
9.67.5.9	leadingZero	197
9.67.5.10	localEnumeration	197
9.67.5.11	notation	198
9.67.5.12	pixmap0	198
9.67.5.13	pixmap1	198
9.67.5.14	pixmap2	198
9.67.5.15	pixmap3	199
9.67.5.16	pixmap4	199
9.67.5.17	pixmap5	199
9.67.5.18	pixmap6	199
9.67.5.19	pixmap7	199
9.67.5.20	precision	199
9.67.5.21	styleSheet	199
9.67.5.22	trailingZeros	199
9.67.5.23	updateOption	199
9.67.5.24	useDbPrecision	200
9.67.5.25	userLevelEnabled	200
9.67.5.26	userLevelEngineerStyle	200

9.67.5.27 userLevelScientistStyle	200
9.67.5.28 userLevelUserStyle	200
9.67.5.29 userLevelVisibility	201
9.67.5.30 variable	201
9.67.5.31 variableAsToolTip	201
9.67.5.32 variableSubstitutions	201
9.67.5.33 visible	201
9.68 QELineEdit Class Reference	201
9.68.1 Member Enumeration Documentation	203
9.68.1.1 ArrayActions	203
9.68.1.2 Formats	203
9.68.1.3 Notations	204
9.68.2 Constructor & Destructor Documentation	204
9.68.2.1 QELineEdit	204
9.68.2.2 QELineEdit	204
9.68.3 Member Function Documentation	204
9.68.3.1 dbValueChanged	204
9.68.4 Property Documentation	204
9.68.4.1 addUnits	204
9.68.4.2 arrayAction	204
9.68.4.3 format	205
9.68.4.4 int	205
9.68.4.5 leadingZero	205
9.68.4.6 localEnumeration	205
9.68.4.7 notation	206
9.68.4.8 precision	206
9.68.4.9 trailingZeros	206
9.68.4.10 useDbPrecision	206
9.69 QELineEditManager Class Reference	206
9.70 QELink Class Reference	207
9.71 QELog Class Reference	209
9.71.1 Member Enumeration Documentation	211
9.71.1.1 DisplayAlarmStateOptions	211
9.71.1.2 UserLevels	212

9.71.2 Member Function Documentation	212
9.71.2.1 setManagedVisible	212
9.71.3 Property Documentation	212
9.71.3.1 allowDrop	212
9.71.3.2 defaultStyle	212
9.71.3.3 displayAlarmState	212
9.71.3.4 displayAlarmStateOption	212
9.71.3.5 int	213
9.71.3.6 styleSheet	213
9.71.3.7 userLevelEnabled	213
9.71.3.8 userLevelEngineerStyle	213
9.71.3.9 userLevelScientistStyle	213
9.71.3.10 userLevelUserStyle	214
9.71.3.11 userLevelVisibility	214
9.71.3.12 variableAsToolTip	214
9.71.3.13 visible	214
9.72 QELogin Class Reference	214
9.73 QELoginDialog Class Reference	215
9.74 QENumericEdit Class Reference	215
9.74.1 Detailed Description	217
9.74.2 Constructor & Destructor Documentation	217
9.74.2.1 QENumericEdit	217
9.74.2.2 QENumericEdit	217
9.74.3 Member Function Documentation	218
9.74.3.1 dbValueChanged	218
9.74.4 Property Documentation	218
9.74.4.1 addUnits	218
9.74.4.2 autoScale	218
9.74.4.3 leadingZeros	218
9.74.4.4 maximum	218
9.74.4.5 minimum	218
9.74.4.6 precision	218
9.75 QENumericEditManager Class Reference	219
9.76 QEPeriodic Class Reference	219

9.76.1 Member Enumeration Documentation	222
9.76.1.1 DisplayAlarmStateOptions	222
9.76.1.2 UserLevels	223
9.76.2 Member Function Documentation	223
9.76.2.1 dbElementChanged	223
9.76.2.2 dbValueChanged	223
9.76.3 Member Data Documentation	223
9.76.3.1 allowDrop	223
9.76.4 Property Documentation	223
9.76.4.1 displayAlarmState	223
9.76.4.2 displayAlarmStateOption	224
9.76.4.3 int	224
9.76.4.4 readbackLabelVariable1	224
9.76.4.5 readbackLabelVariable2	224
9.76.4.6 subscribe	224
9.76.4.7 userLevelEnabled	224
9.76.4.8 userLevelEngineerStyle	225
9.76.4.9 userLevelScientistStyle	225
9.76.4.10 userLevelUserStyle	225
9.76.4.11 userLevelVisibility	225
9.76.4.12 variableAsToolTip	225
9.76.4.13 variableSubstitutions	225
9.76.4.14 visible	226
9.76.4.15 writeButtonVariable1	226
9.76.4.16 writeButtonVariable2	226
9.77 QEPeriodicComponentData Class Reference	226
9.78 QEPeriodicTaskMenu Class Reference	226
9.79 QEPeriodicTaskMenuFactory Class Reference	227
9.80 QEPlot Class Reference	227
9.80.1 Member Enumeration Documentation	231
9.80.1.1 DisplayAlarmStateOptions	231
9.80.1.2 UserLevels	231
9.80.2 Member Function Documentation	231
9.80.2.1 dbValueChanged	231

9.80.2.2	dbValueChanged	231
9.80.2.3	setManagedVisible	232
9.80.3	Member Data Documentation	232
9.80.3.1	allowDrop	232
9.80.4	Property Documentation	232
9.80.4.1	defaultStyle	232
9.80.4.2	displayAlarmState	232
9.80.4.3	displayAlarmStateOption	232
9.80.4.4	int	232
9.80.4.5	styleSheet	233
9.80.4.6	userLevelEnabled	233
9.80.4.7	userLevelEngineerStyle	233
9.80.4.8	userLevelScientistStyle	233
9.80.4.9	userLevelUserStyle	233
9.80.4.10	userLevelVisibility	234
9.80.4.11	variable1	234
9.80.4.12	variable2	234
9.80.4.13	variable3	234
9.80.4.14	variable4	234
9.80.4.15	variableAsToolTip	234
9.80.4.16	variableSubstitutions	234
9.80.4.17	visible	234
9.81	QEPushButton Class Reference	235
9.81.1	Member Enumeration Documentation	239
9.81.1.1	ArrayActions	239
9.81.1.2	CreationOptionNames	239
9.81.1.3	DisplayAlarmStateOptions	239
9.81.1.4	Formats	240
9.81.1.5	Notations	240
9.81.1.6	ProgramStartupOptionNames	240
9.81.1.7	UpdateOptions	240
9.81.1.8	UserLevels	241
9.81.2	Constructor & Destructor Documentation	241
9.81.2.1	QEPushButton	241

9.81.2.2	QEPushButton	241
9.81.3	Member Function Documentation	241
9.81.3.1	clicked	241
9.81.3.2	dbValueChanged	241
9.81.3.3	pressed	242
9.81.3.4	released	242
9.81.3.5	requestAction	242
9.81.3.6	setManagedVisible	242
9.81.4	Property Documentation	242
9.81.4.1	addUnits	242
9.81.4.2	alignment	242
9.81.4.3	allowDrop	242
9.81.4.4	altReadbackVariable	243
9.81.4.5	arguments	243
9.81.4.6	arrayAction	243
9.81.4.7	clickCheckedText	243
9.81.4.8	clickText	243
9.81.4.9	confirmAction	244
9.81.4.10	confirmText	244
9.81.4.11	creationOption	244
9.81.4.12	customisationName	244
9.81.4.13	defaultStyle	244
9.81.4.14	displayAlarmState	244
9.81.4.15	displayAlarmStateOption	245
9.81.4.16	format	245
9.81.4.17	guiFile	245
9.81.4.18	int	245
9.81.4.19	labelText	245
9.81.4.20	leadingZero	246
9.81.4.21	localEnumeration	246
9.81.4.22	notation	246
9.81.4.23	password	246
9.81.4.24	pixmap0	247
9.81.4.25	pixmap1	247

9.81.4.26 pixmap2	247
9.81.4.27 pixmap3	247
9.81.4.28 pixmap4	247
9.81.4.29 pixmap5	247
9.81.4.30 pixmap6	247
9.81.4.31 pixmap7	247
9.81.4.32 precision	247
9.81.4.33 pressText	248
9.81.4.34 prioritySubstitutions	248
9.81.4.35 program	248
9.81.4.36 programStartupOption	248
9.81.4.37 releaseText	248
9.81.4.38 styleSheet	248
9.81.4.39 subscribe	248
9.81.4.40 trailingZeros	249
9.81.4.41 updateOption	249
9.81.4.42 useDbPrecision	249
9.81.4.43 userLevelEnabled	249
9.81.4.44 userLevelEngineerStyle	249
9.81.4.45 userLevelScientistStyle	249
9.81.4.46 userLevelUserStyle	250
9.81.4.47 userLevelVisibility	250
9.81.4.48 variable	250
9.81.4.49 variableAsToolTip	250
9.81.4.50 variableSubstitutions	250
9.81.4.51 visible	250
9.81.4.52 writeOnClick	250
9.81.4.53 writeOnPress	251
9.81.4.54 writeOnRelease	251
9.82 QEPVNameLists Class Reference	251
9.83 QEPvProperties Class Reference	251
9.83.1 Property Documentation	252
9.83.1.1 variable	252
9.83.1.2 variableSubstitutions	253

9.84 QEPvPropertiesManager Class Reference	253
9.85 QERadioButton Class Reference	253
9.85.1 Member Enumeration Documentation	257
9.85.1.1 ArrayActions	257
9.85.1.2 CreationOptionNames	258
9.85.1.3 DisplayAlarmStateOptions	258
9.85.1.4 Formats	258
9.85.1.5 Notations	259
9.85.1.6 ProgramStartupOptionNames	259
9.85.1.7 UpdateOptions	259
9.85.1.8 UserLevels	260
9.85.2 Constructor & Destructor Documentation	260
9.85.2.1 QERadioButton	260
9.85.2.2 QERadioButton	260
9.85.3 Member Function Documentation	260
9.85.3.1 clicked	260
9.85.3.2 dbValueChanged	260
9.85.3.3 pressed	260
9.85.3.4 released	261
9.85.3.5 requestAction	261
9.85.3.6 setManagedVisible	261
9.85.4 Property Documentation	261
9.85.4.1 addUnits	261
9.85.4.2 alignment	261
9.85.4.3 allowDrop	261
9.85.4.4 arguments	261
9.85.4.5 arrayAction	262
9.85.4.6 clickCheckedText	262
9.85.4.7 clickText	262
9.85.4.8 confirmAction	262
9.85.4.9 confirmText	262
9.85.4.10 creationOption	263
9.85.4.11 customisationName	263
9.85.4.12 defaultStyle	263

9.85.4.13 displayAlarmState	263
9.85.4.14 displayAlarmStateOption	263
9.85.4.15 format	264
9.85.4.16 guiFile	264
9.85.4.17 int	264
9.85.4.18 labelText	264
9.85.4.19 leadingZero	264
9.85.4.20 localEnumeration	264
9.85.4.21 notation	265
9.85.4.22 password	265
9.85.4.23 pixmap0	265
9.85.4.24 pixmap1	265
9.85.4.25 pixmap2	266
9.85.4.26 pixmap3	266
9.85.4.27 pixmap4	266
9.85.4.28 pixmap5	266
9.85.4.29 pixmap6	266
9.85.4.30 pixmap7	266
9.85.4.31 precision	266
9.85.4.32 pressText	266
9.85.4.33 prioritySubstitutions	267
9.85.4.34 program	267
9.85.4.35 programStartupOption	267
9.85.4.36 releaseText	267
9.85.4.37 styleSheet	267
9.85.4.38 subscribe	267
9.85.4.39 trailingZeros	267
9.85.4.40 updateOption	268
9.85.4.41 useDbPrecision	268
9.85.4.42 userLevelEnabled	268
9.85.4.43 userLevelEngineerStyle	268
9.85.4.44 userLevelScientistStyle	268
9.85.4.45 userLevelUserStyle	268
9.85.4.46 userLevelVisibility	269

9.85.4.47 variable	269
9.85.4.48 variableAsToolTip	269
9.85.4.49 variableSubstitutions	269
9.85.4.50 visible	269
9.85.4.51 writeOnClick	269
9.85.4.52 writeOnPress	269
9.85.4.53 writeOnRelease	270
9.86 QERecipe Class Reference	270
9.87 QERecordFieldName Class Reference	272
9.88 QERecordSpec Class Reference	272
9.89 QERecordSpecList Class Reference	273
9.90 QEScript Class Reference	273
9.90.1 Detailed Description	277
9.90.2 Member Enumeration Documentation	277
9.90.2.1 DisplayAlarmStateOptions	277
9.90.2.2 UserLevels	278
9.90.3 Member Function Documentation	278
9.90.3.1 setManagedVisible	278
9.90.4 Property Documentation	278
9.90.4.1 allowDrop	278
9.90.4.2 defaultStyle	278
9.90.4.3 displayAlarmState	278
9.90.4.4 displayAlarmStateOption	279
9.90.4.5 int	279
9.90.4.6 styleSheet	279
9.90.4.7 userLevelEnabled	279
9.90.4.8 userLevelEngineerStyle	279
9.90.4.9 userLevelScientistStyle	279
9.90.4.10 userLevelUserStyle	280
9.90.4.11 userLevelVisibility	280
9.90.4.12 variableAsToolTip	280
9.90.4.13 visible	280
9.91 QEShape Class Reference	280
9.91.1 Detailed Description	285

9.91.2 Member Enumeration Documentation	285
9.91.2.1 animationOptions	285
9.91.2.2 DisplayAlarmStateOptions	285
9.91.2.3 shapeOptions	285
9.91.2.4 UserLevels	286
9.91.3 Constructor & Destructor Documentation	286
9.91.3.1 QEShape	286
9.91.3.2 QEShape	286
9.91.4 Member Function Documentation	286
9.91.4.1 dbValueChanged1	286
9.91.4.2 dbValueChanged2	286
9.91.4.3 dbValueChanged3	286
9.91.4.4 dbValueChanged4	287
9.91.4.5 dbValueChanged5	287
9.91.4.6 dbValueChanged6	287
9.91.4.7 setManagedVisible	287
9.91.5 Property Documentation	287
9.91.5.1 allowDrop	287
9.91.5.2 animation1	287
9.91.5.3 animation2	287
9.91.5.4 animation3	288
9.91.5.5 animation4	288
9.91.5.6 animation5	288
9.91.5.7 animation6	288
9.91.5.8 color1	288
9.91.5.9 color10	288
9.91.5.10 color2	288
9.91.5.11 color3	288
9.91.5.12 color4	288
9.91.5.13 color5	289
9.91.5.14 color6	289
9.91.5.15 color7	289
9.91.5.16 color8	289
9.91.5.17 color9	289

9.91.5.18 defaultStyle	289
9.91.5.19 displayAlarmState	289
9.91.5.20 displayAlarmStateOption	289
9.91.5.21 int	290
9.91.5.22 offset1	290
9.91.5.23 offset2	290
9.91.5.24 offset3	290
9.91.5.25 offset4	290
9.91.5.26 offset5	290
9.91.5.27 offset6	290
9.91.5.28 point1	291
9.91.5.29 point10	291
9.91.5.30 point2	291
9.91.5.31 point3	291
9.91.5.32 point4	291
9.91.5.33 point5	291
9.91.5.34 point6	291
9.91.5.35 point7	291
9.91.5.36 point8	292
9.91.5.37 point9	292
9.91.5.38 scale2	292
9.91.5.39 scale3	292
9.91.5.40 scale4	292
9.91.5.41 scale5	292
9.91.5.42 scale6	292
9.91.5.43 styleSheet	292
9.91.5.44 userLevelEnabled	292
9.91.5.45 userLevelEngineerStyle	293
9.91.5.46 userLevelScientistStyle	293
9.91.5.47 userLevelUserStyle	293
9.91.5.48 userLevelVisibility	293
9.91.5.49 variable1	293
9.91.5.50 variable2	294
9.91.5.51 variable3	294

9.91.5.52 variable4	294
9.91.5.53 variable5	294
9.91.5.54 variable6	294
9.91.5.55 variableAsToolTip	294
9.91.5.56 variableSubstitutions	294
9.91.5.57 visible	295
9.92 QESlider Class Reference	295
9.92.1 Member Enumeration Documentation	297
9.92.1.1 DisplayAlarmStateOptions	297
9.92.1.2 UserLevels	297
9.92.2 Member Function Documentation	298
9.92.2.1 dbValueChanged	298
9.92.2.2 setManagedVisible	298
9.92.3 Member Data Documentation	298
9.92.3.1 writeOnChange	298
9.92.4 Property Documentation	298
9.92.4.1 allowDrop	298
9.92.4.2 defaultStyle	298
9.92.4.3 displayAlarmState	298
9.92.4.4 displayAlarmStateOption	299
9.92.4.5 int	299
9.92.4.6 styleSheet	299
9.92.4.7 subscribe	299
9.92.4.8 userLevelEnabled	299
9.92.4.9 userLevelEngineerStyle	299
9.92.4.10 userLevelScientistStyle	300
9.92.4.11 userLevelUserStyle	300
9.92.4.12 userLevelVisibility	300
9.92.4.13 variable	300
9.92.4.14 variableAsToolTip	300
9.92.4.15 variableSubstitutions	300
9.92.4.16 visible	301
9.93 QESpinBox Class Reference	301
9.93.1 Member Enumeration Documentation	303

9.93.1.1	DisplayAlarmStateOptions	303
9.93.1.2	UserLevels	304
9.93.2	Member Function Documentation	304
9.93.2.1	dbValueChanged	304
9.93.2.2	setManagedVisible	304
9.93.3	Property Documentation	304
9.93.3.1	allowDrop	304
9.93.3.2	defaultStyle	304
9.93.3.3	displayAlarmState	304
9.93.3.4	displayAlarmStateOption	305
9.93.3.5	int	305
9.93.3.6	styleSheet	305
9.93.3.7	subscribe	305
9.93.3.8	userLevelEnabled	305
9.93.3.9	userLevelEngineerStyle	305
9.93.3.10	userLevelScientistStyle	306
9.93.3.11	userLevelUserStyle	306
9.93.3.12	userLevelVisibility	306
9.93.3.13	variable	306
9.93.3.14	variableAsToolTip	306
9.93.3.15	variableSubstitutions	306
9.93.3.16	visible	307
9.94	QEStripChart Class Reference	307
9.94.1	Property Documentation	309
9.94.1.1	variableSubstitutions	309
9.95	QEStripChartAdjustPVDialog Class Reference	309
9.96	QEStripChartContextMenu Class Reference	309
9.96.1	Constructor & Destructor Documentation	310
9.96.1.1	QEStripChartContextMenu	310
9.97	QEStripChartDurationDialog Class Reference	310
9.98	QEStripChartItem Class Reference	310
9.99	QEStripChartNames Class Reference	311
9.100	QEStripChartPushButtonSpecifications Struct Reference	313
9.101	QEStripChartRangeDialog Class Reference	313

9.102QEStripChartState Class Reference	313
9.103QEStripChartStateList Class Reference	314
9.104QEStripChartStatistics Class Reference	314
9.105QEStripChartTimeDialog Class Reference	315
9.106QEStripChartToolBar Class Reference	315
9.106.1 Detailed Description	316
9.107QESubstitutedLabel Class Reference	316
9.107.1 Member Data Documentation	316
9.107.1.1 labelText	316
9.107.2 Property Documentation	317
9.107.2.1 textSubstitutions	317
9.108recording Class Reference	317
9.109imageDisplayProperties::rgbPixel Struct Reference	317
9.110screenSelectDialog Class Reference	318
9.111selectMenu Class Reference	318
9.112trace Class Reference	318
9.113userInfoStruct Class Reference	319
9.114QEPeriodic::userInfoStructArray Struct Reference	319
9.115ValueScaling Class Reference	319
9.116VideoWidget Class Reference	320
9.117zoomMenu Class Reference	321

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

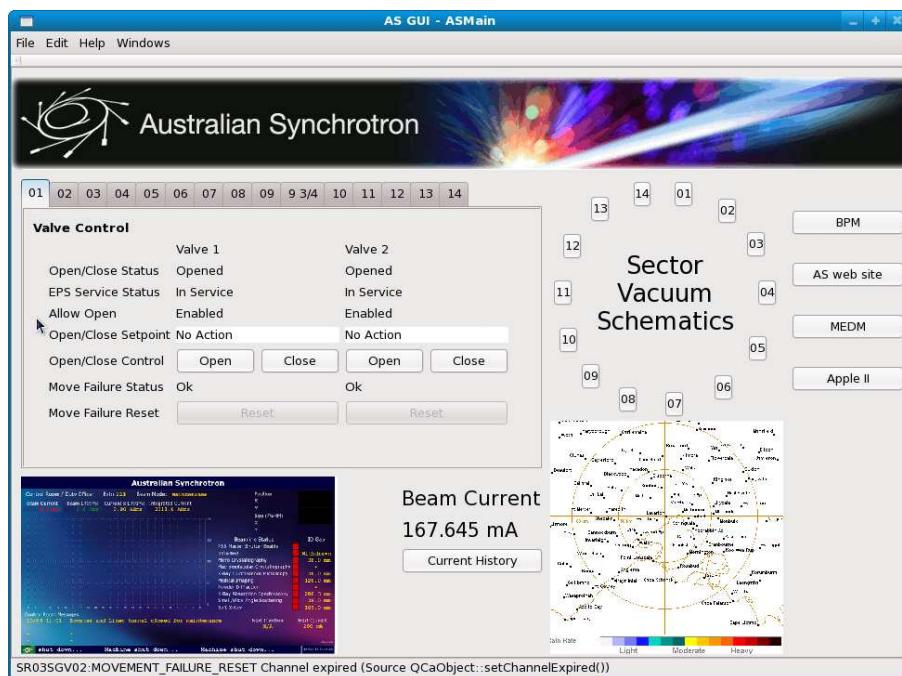


Figure 3.1: Australian Synchrotron mock up

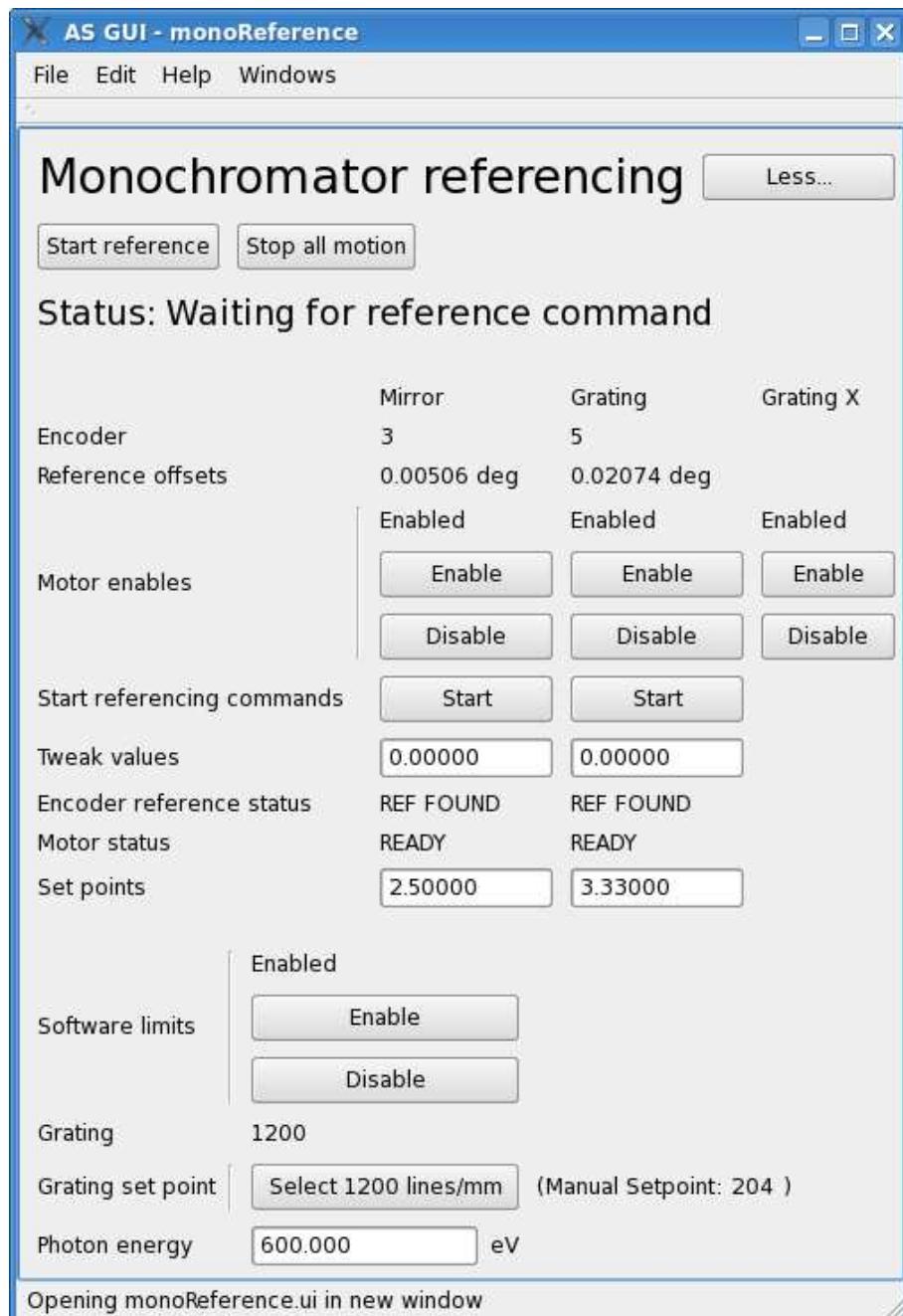


Figure 3.2: Monochromator referencing

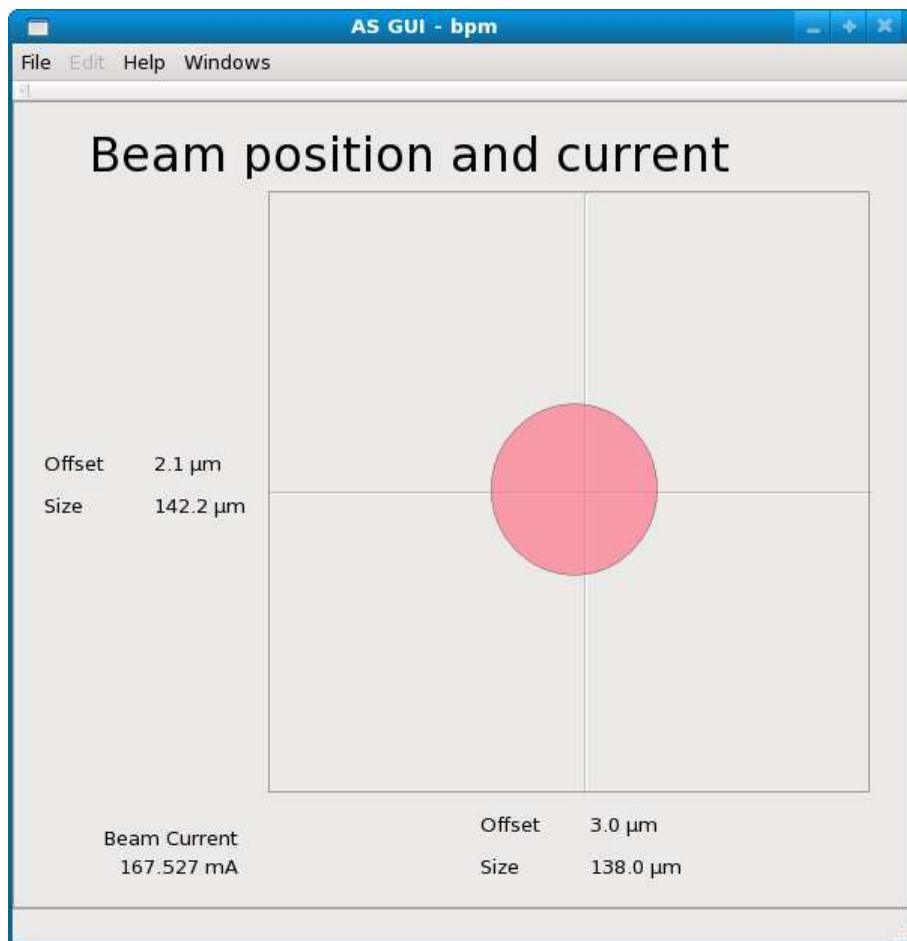


Figure 3.3: Beam position monitor

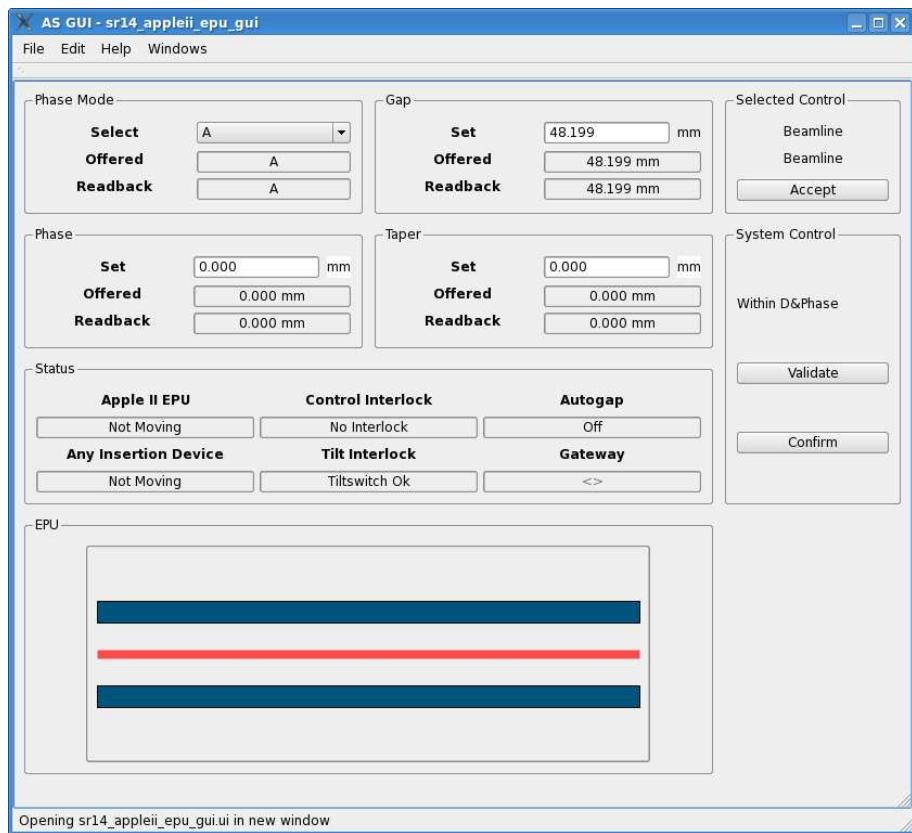


Figure 3.4: Insertion device



Figure 3.5: Injection efficiency monitor

Chapter 4

other applications using epicsqt widgets

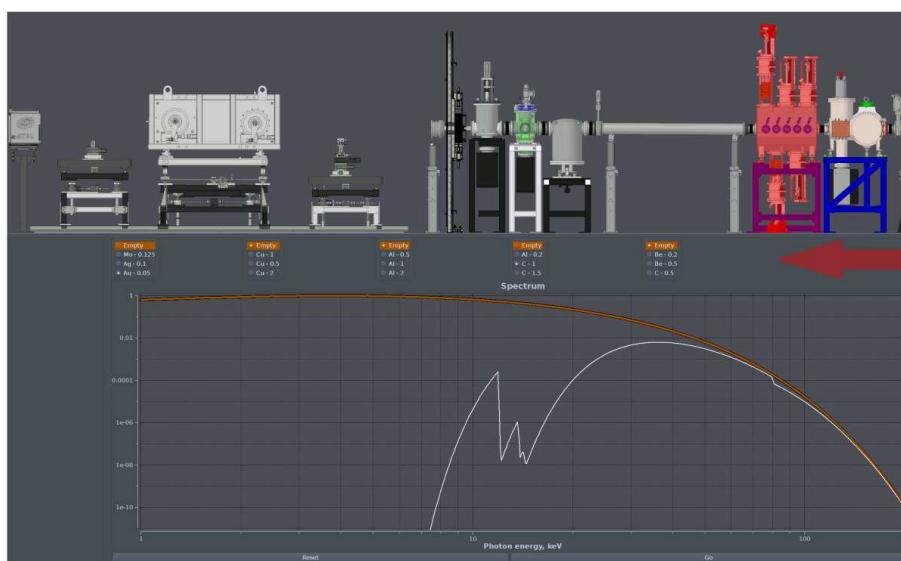


Figure 4.1: Medical Imaging beamline

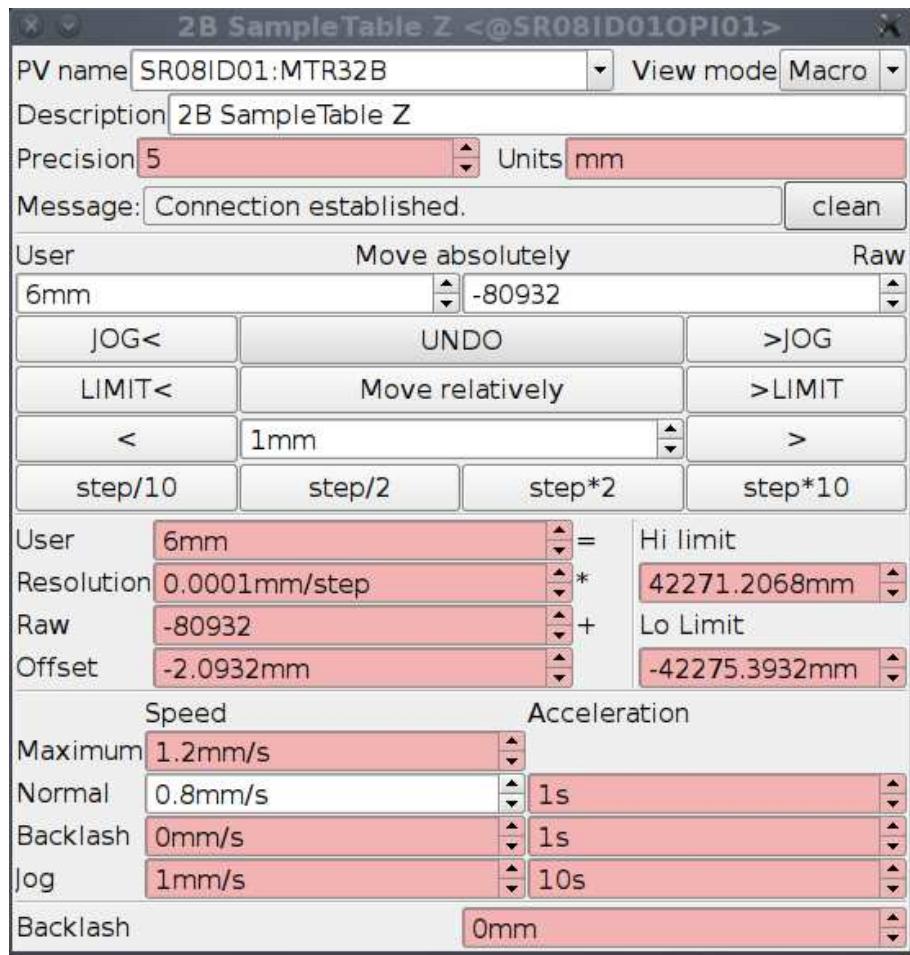


Figure 4.2: Motor controller

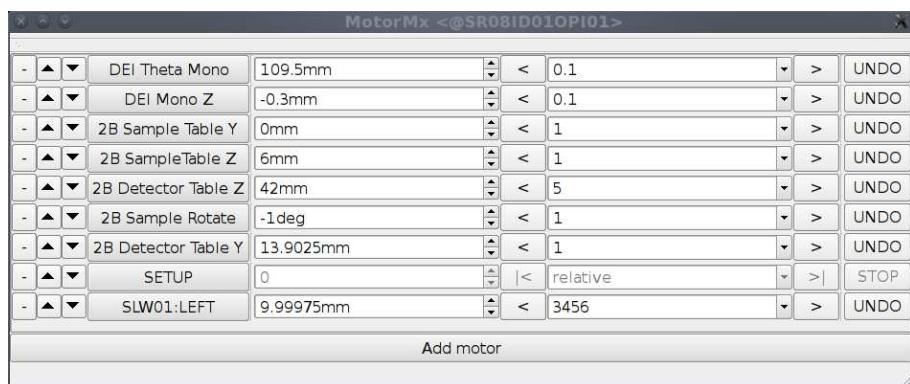


Figure 4.3: Motor controller

Chapter 5

Qt Designer

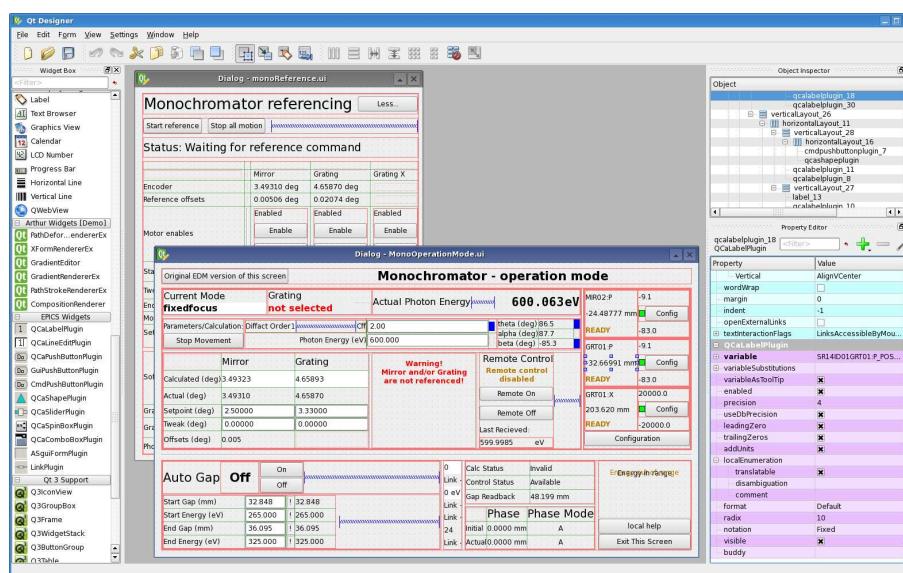


Figure 5.1: Editing multiple GUIs

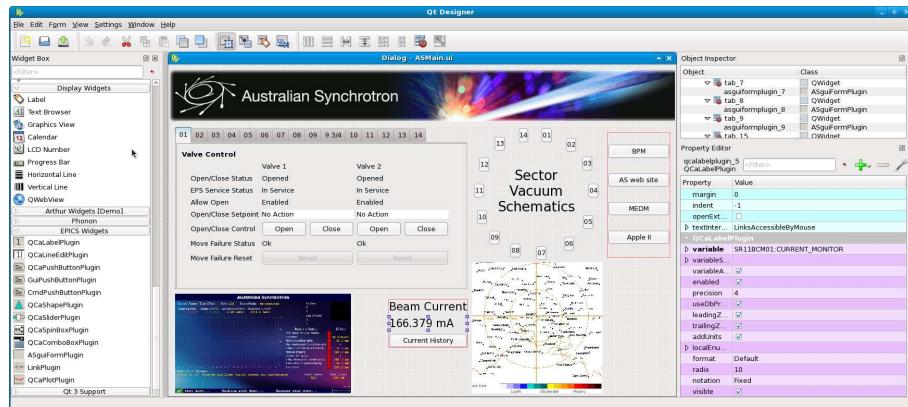


Figure 5.2: Editing a GUI

Chapter 6

Qt Creator

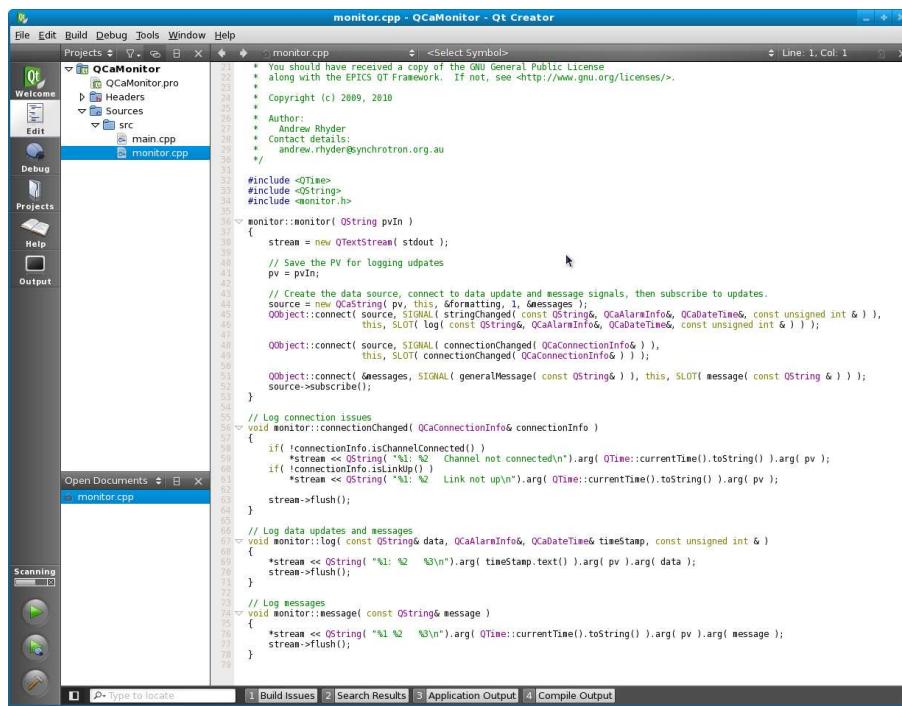


Figure 6.1: Application using epicsqt data source classes

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
FFThread	32
flipRotateMenu	33
fullScreenWindow	33
histogram	34
histogramScroll	34
historicImage	34
imageContextMenu	35
imageDisplayProperties	36
imageInfo	37
QEImage	143
imageMarkup	38
VideoWidget	320
imageMarkupLegendSetText	40
imageProperties	44
imageProcessor	41

imagePropertiesCore	46
imageUpdateIndicator	47
loginWidget	47
markupDisplayMenu	49
markupItem	51
markupCrosshair1	48
markupCrosshair2	48
markupEllipse	49
markupHLine	50
markupLine	54
markupRegion	54
markupText	55
markupVLine	56
mpegSource	57
QEImage	143
mpegSourceObject	57
QEStripChartToolBar::OwnWidgets	58
PeriodicDialog	58
PeriodicElementSetupForm	59
PeriodicSetupDialog	59
playbackTimer	59
pointInfo	60
profilePlot	60
QBitStatus	61
QEBitStatus	78
QEAnalogIndicator	63
QEAnalogProgressBar	68
QECheckBoxManager	101
QEComboBox	101
QEConfiguredLayout	107
QEConfiguredLayoutManager	113
QEFileBrowser	113
QEForm	120
QEFrame	124
QEPvProperties	251
QEStripChart	307
QEGenericButton	128
QECheckBox	84
QEPushButton	235
QERadioButton	253
QEGenericEdit	130
QELineEdit	201
QENumericEdit	215
QEGroupBox	138
QEImageMarkupThickness	189
QEImageOptionsDialog	189
QELabel	190
QELineEditManager	206

QELink	207
QELog	209
QELogin	214
QELoginDialog	215
QENumericEditManager	219
QEPeriodic	219
QEPeriodicComponentData	226
QEPeriodicTaskMenu	226
QEPeriodicTaskMenuFactory	227
QEPlot	227
QEPVNameLists	251
QEPvPropertiesManager	253
QERecipe	270
QERecordFieldName	272
QERecordSpec	272
QERecordSpecList	273
QEScript	273
QEShape	280
QESlider	295
QESpinBox	301
QEStripChartAdjustPVDialog	309
QEStripChartContextMenu	309
QEStripChartDurationDialog	310
QEStripChartItem	310
QEStripChartNames	311
QEStripChartPushButtonSpecifications	313
QEStripChartRangeDialog	313
QEStripChartState	313
QEStripChartStateList	314
QEStripChartStatistics	314
QEStripChartTimeDialog	315
QEStripChartToolBar	315
QESubstitutedLabel	316
recording	317
imageDisplayProperties::rgbPixel	317
screenSelectDialog	318
selectMenu	318
trace	318
userInfoStruct	319
QEPeriodic::userInfoStructArray	319
ValueScaling	319
zoomMenu	321

Chapter 8

Class Index

8.1 Class List

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

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

markupCrosshair1	48
markupCrosshair2	48
markupDisplayMenu	49
markupEllipse	49
markupHLine	50
markupItem	51
markupLine	54
markupRegion	54
markupText	55
markupVLine	56
mpegSource	57
mpegSourceObject	57
QEStripChartToolBar::OwnWidgets	58
PeriodicDialog	58
PeriodicElementSetupForm	59
PeriodicSetupDialog	59
playbackTimer	59
pointInfo	60
profilePlot	60
QBitStatus	61
QEAnalogIndicator	63
QEAnalogProgressBar	68
QEBitStatus	78
QECheckBox	84
QECheckBoxManager	101
QEComboBox	101
QEConfiguredLayout	107
QEConfiguredLayoutManager	113
QEFileBrowser	113
QEForm	120
QEFrame	124
QEGenericButton	128
QEGenericEdit	130
QEGroupBox	138
QEImage	143
QEImageMarkupThickness	189
QEImageOptionsDialog	189
QELabel	190
QELineEdit	201
QELineEditManager	206
QELink	207
QELog	209
QELogin	214
QELoginDialog	215
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)	215
QENumericEditManager	219
QEPeriodic	219
QEPeriodicComponentData	226

QEPeriodicTaskMenu	226
QEPeriodicTaskMenuFactory	227
QEPlot	227
QEPushButton	235
QEPVNameLists	251
QEPvProperties	251
QEPvPropertiesManager	253
QERadioButton	253
QERecipe	270
QERecordFieldName	272
QERecordSpec	272
QERecordSpecList	273
QEScript	273
QEShape	280
QESlider	295
QESpinBox	301
QEStripChart	307
QEStripChartAdjustPVDialo	309
QEStripChartContextMenu	309
QEStripChartDurationDialog	310
QEStripChartItem	310
QEStripChartNames	311
QEStripChartPushButtonSpecifications	313
QEStripChartRangeDialog	313
QEStripChartState	313
QEStripChartStateList	314
QEStripChartStatistics	314
QEStripChartTimeDialog	315
QEStripChartToolBar (This class holds all the StripChart tool bar widgets)	315
QESubstitutedLabel	316
recording	317
imageDisplayProperties::rgbPixel	317
screenSelectDialog	318
selectMenu	318
trace	318
userInfoStruct	319
QEPeriodic::userInfoStructArray	319
ValueScaling	319
VideoWidget	320
zoomMenu	321

Chapter 9

Class Documentation

9.1 _CopyPaste Class Reference

Public Member Functions

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

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

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

9.2 _Field Class Reference

Public Member Functions

- QEWidget * **getWidget** ()

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

Public Attributes

- QEWidget * **qeWidget**

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

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

9.3 Item Class Reference

Public Member Functions

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

Public Attributes

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

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

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

9.4 _QDialogItem Class Reference

Public Member Functions

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

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

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

9.5 _QPushButtonGroup Class Reference

Public Slots

- void **buttonGroupClicked** ()

Public Member Functions

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

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

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

9.6 _QTableWidgetFileBrowser Class Reference

Public Member Functions

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

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

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

9.7 _QTableWidgetLog Class Reference

Public Member Functions

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

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

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

9.8 _QTableWidgetScript Class Reference

Public Member Functions

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

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

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

9.9 arealInfo Class Reference

Public Member Functions

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

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

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

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

9.10 QEAnalogIndicator::Band Struct Reference

Public Attributes

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

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

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

9.11 QEAnalogIndicator::BandList Class Reference

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

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

9.12 QEPeriodic::elementInfoStruct Struct Reference

Public Attributes

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

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

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

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

9.13 FFBuffer Class Reference

Public Member Functions

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

Public Attributes

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

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

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

9.14 FFThread Class Reference

Public Slots

- void **stopGracefully** ()

Signals

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

Public Member Functions

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

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

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

9.15 flipRotateMenu Class Reference

Public Member Functions

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

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

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

9.16 fullScreenWindow Class Reference

Signals

- void **fullScreenResize** ()

Public Member Functions

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

Protected Member Functions

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

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

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

9.17 histogram Class Reference

Public Member Functions

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

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

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

9.18 histogramScroll Class Reference

Public Member Functions

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

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

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

9.19 historicImage Class Reference

Public Member Functions

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

Public Attributes

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

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

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

9.20 imageContextMenu Class Reference

Public Types

- enum **imageContextMenuOptions** {
 ICM_NONE = contextMenu::CM_SPECIFIC_WIDGETS_START_HERE, **ICM_SAVE**,
 ICM_PAUSE, **ICM_ENABLE_TIME**,
 ICM_ENABLE_CURSOR_PIXEL, **ICM_ABOUT_IMAGE**, **ICM_ENABLE_VERT1**,
 ICM_ENABLE_VERT2,
 ICM_ENABLE_VERT3, **ICM_ENABLE_VERT4**, **ICM_ENABLE_VERT5**, **ICM_ENABLE_HOZ1**,
 ICM_ENABLE_HOZ2, **ICM_ENABLE_HOZ3**, **ICM_ENABLE_HOZ4**, **ICM_ENABLE_HOZ5**,
 ICM_ENABLE_AREA1, **ICM_ENABLE_AREA2**, **ICM_ENABLE_AREA3**, **ICM_ENABLE_AREA4**,
 ICM_ENABLE_LINE, **ICM_ENABLE_TARGET**, **ICM_ENABLE_BEAM**, **ICM_DISPLAY_BUTTON_BAR**,
 ICM_DISPLAY_IMAGE_DISPLAY_PROPERTIES, **ICM_DISPLAY_RECORDER**,
 ICM_ZOOM_SELECTED, **ICM_ZOOM_FIT**,
 ICM_ZOOM_PLUS, **ICM_ZOOM_MINUS**, **ICM_ZOOM_10**, **ICM_ZOOM_25**,
 ICM_ZOOM_50, **ICM_ZOOM_75**, **ICM_ZOOM_100**, **ICM_ZOOM_150**,
 ICM_ZOOM_200, **ICM_ZOOM_300**, **ICM_ZOOM_400**, **ICM_ROTATE_NONE**,
 ICM_ROTATE_RIGHT, **ICM_ROTATE_LEFT**, **ICM_ROTATE_180**, **ICM_FLIP_HORIZONTAL**,
 ICM_FLIP_VERTICAL, **ICM_SELECT_PAN**, **ICM_SELECT_HSLICE1**, **ICM_SELECT_HSLICE2**,
 ICM_SELECT_HSLICE3, **ICM_SELECT_HSLICE4**, **ICM_SELECT_HSLICE5**, **ICM_SELECT_VSLICE1**,
 ICM_SELECT_VSLICE2, **ICM_SELECT_VSLICE3**, **ICM_SELECT_VSLICE4**, **ICM_SELECT_VSLICE5**,
 ICM_SELECT_AREA1, **ICM_SELECT_AREA2**, **ICM_SELECT_AREA3**, **ICM_SELECT_AREA4**,
 ICM_SELECT_PROFILE, **ICM_SELECT_TARGET**, **ICM_SELECT_BEAM**, **ICM_CLEAR_MARKUP**,
 ICM_SET_LEGEND, **ICM_THICKNESS_ONE_MARKUP**, **ICM_THICKNESS_SELECT_MARKUP**, **ICM_COPY_PLOT_DATA**,
 ICM_FULL_SCREEN, **ICM_DISPLAY_HSLICE1**, **ICM_DISPLAY_HSLICE2**, **ICM_DISPLAY_HSLICE3**,
 ICM_DISPLAY_HSLICE4, **ICM_DISPLAY_HSLICE5**, **ICM_DISPLAY_VSLICE1**,
 ICM_DISPLAY_VSLICE2,
 ICM_DISPLAY_VSLICE3, **ICM_DISPLAY_VSLICE4**, **ICM_DISPLAY_VSLICE5**,
 ICM_DISPLAY_AREA1,
 ICM_DISPLAY_AREA2, **ICM_DISPLAY_AREA3**, **ICM_DISPLAY_AREA4**, **ICM_DISPLAY_PROFILE**,

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

Public Member Functions

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

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

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

9.21 imageDisplayProperties Class Reference

Classes

- struct [rgbPixel](#)

Signals

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

Public Member Functions

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

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

Public Attributes

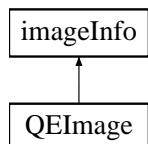
- int **zeroValue**
- int **fullValue**
- bool **defaultFullValue**
- unsigned int **range**
- unsigned int **maxP**
- unsigned int **minP**
- unsigned int **depth**
- unsigned int **bins** [HISTOGRAM_BINS]
- bool **statisticsSet**
- **rgbPixel** * **pixelLookup**
- QLabel * **histXLabel**

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

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

9.22 imageInfo Class Reference

Inheritance diagram for imageInfo:



Public Member Functions

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

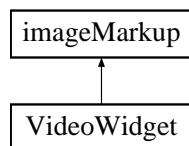
- void **infoUpdateHozProfile** ()
- void **infoUpdateHozProfile** (const int y, const unsigned int thickness)
- void **infoUpdateProfile** ()
- void **infoUpdateProfile** (const QPoint start, const QPoint end, const unsigned int thickness)
- void **infoUpdateRegion** (const unsigned int region)
- void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

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

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

9.23 imageMarkup Class Reference

Inheritance diagram for imageMarkup:



Public Types

- enum **markupIds** {
 MARKUP_ID_REGION1, **MARKUP_ID_REGION2**, **MARKUP_ID_REGION3**, **MARKUP_ID_REGION4**,
 MARKUP_ID_H1_SLICE, **MARKUP_ID_H2_SLICE**, **MARKUP_ID_H3_SLICE**,
 MARKUP_ID_H4_SLICE,
 MARKUP_ID_H5_SLICE, **MARKUP_ID_V1_SLICE**, **MARKUP_ID_V2_SLICE**, **MARKUP_ID_V3_SLICE**,
 MARKUP_ID_V4_SLICE, **MARKUP_ID_V5_SLICE**, **MARKUP_ID_LINE**, **MARKUP_ID_TARGET**,
 }

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

Public Member Functions

- void **setShowTime** (bool visibleIn)
- bool **getShowTime** ()
- markupIds **getMode** ()
- void **setMode** (markupIds modeIn)
- void **setMarkupColor** (markupIds mode, QColor markupColorIn)
- QColor **getMarkupColor** (markupIds mode)
- bool **showMarkupMenu** (const QPoint &pos, const QPoint &globalPos)
- void **markupRegionValueChange** (int arealIndex, QRect area, bool displayMarkups)

- void **markupH1ProfileChange** (int y, bool displayMarkups)
- void **markupH2ProfileChange** (int y, bool displayMarkups)
- void **markupH3ProfileChange** (int y, bool displayMarkups)
- void **markupH4ProfileChange** (int y, bool displayMarkups)
- void **markupH5ProfileChange** (int y, bool displayMarkups)
- void **markupV1ProfileChange** (int x, bool displayMarkups)
- void **markupV2ProfileChange** (int x, bool displayMarkups)
- void **markupV3ProfileChange** (int x, bool displayMarkups)
- void **markupV4ProfileChange** (int x, bool displayMarkups)
- void **markupV5ProfileChange** (int x, bool displayMarkups)
- void **markupLineProfileChange** (QPoint start, QPoint end, bool displayMarkups)

- void **markupTargetValueChange** (QPoint point, bool displayMarkups)
- void **markupBeamValueChange** (QPoint point, bool displayMarkups)
- void **markupEllipseValueChange** (QPoint point1, QPoint point2, bool displayMarkups)
- void **markupValueChange** (int markup, bool displayMarkups, QPoint p1, QPoint p2=QPoint())
- QCursor **getCircleCursor** ()
- QCursor **getTargetCursor** ()
- QCursor **getVLineCursor** ()
- QCursor **getHLineCursor** ()
- QCursor **getLineCursor** ()
- QCursor **getRegionCursor** ()
- virtual void **markupSetCursor** (QCursor cursor)=0
- void **setMarkupLegend** (markupIds mode, QString legend)
- QString **getMarkupLegend** (markupIds mode)
- void **clearMarkup** (markupIds markupId)
- void **showMarkup** (markupIds markupId)
- void **displayMarkup** (markupIds markupId, bool state)

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

Public Attributes

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

Protected Member Functions

- void **drawMarkups** (QPainter &p, const QRect &rect)
- bool **anyVisibleMarkups** ()
- QCursor **getDefaultMarkupCursor** ()
- void **setMarkupTime** (QCaDateTime &time)
- bool **markupMousePressEvent** (QMouseEvent *event, bool panning)
- bool **markupMouseReleaseEvent** (QMouseEvent *event, bool panning)
- bool **markupMouseMoveEvent** (QMouseEvent *event, bool panning)
- void **markupResize** (const double scale)
- virtual void **markupChange** (QVector< QRect > &changedAreas)=0
- virtual void **markupAction** (markupIds mode, bool complete, bool clearing, QPoint point1, QPoint point2, unsigned int thickness)=0

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

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

9.24 imageMarkupLegendSetText Class Reference

Public Member Functions

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

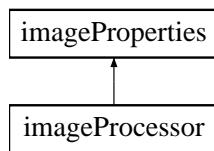
The documentation for this class was generated from the following files:

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

9.25 imageProcessor Class Reference

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

Inheritance diagram for imageProcessor:



Signals

- void [imageBuilt](#) (QImage imageData, QString error)

An image has been generated from image data and is now ready for presentation.

Public Member Functions

- [imageProcessor](#) ()

Constructor.

- [~imageProcessor](#) ()

Destructor.

- void [setImageBuff](#) ()

Ensure the image buffer used to process images is appropriately sized. This is called whenever an image attribute changes that may affect the buffer size required.

- void [setImage](#) (const QByteArray &imageIn, unsigned long dataSize)

Save the image data for analysis processing and display.

- void [buildImage](#) ()

Generate a new image.

- bool [setWidth](#) (unsigned long uValue)

Set the image width.

- bool [setHeight](#) (unsigned long uValue)

Set the image height.

- bool [setNumDimensions](#) (unsigned long uValue)

Set the number of dimensions.

- bool [setDimension0](#) (unsigned long uValue)

Set the first dimension (width if two dimensions, bytes per element if three dimensions)

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

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

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

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

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

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

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

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

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

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

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

Starts the processing loop.

Public Attributes

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

9.25.1 Detailed Description

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

9.25.2 Member Function Documentation

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

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

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

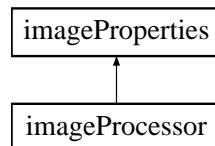
The documentation for this class was generated from the following files:

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

9.26 imageProperties Class Reference

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

Inheritance diagram for imageProperties:



Public Types

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

Public Member Functions

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

- void **setImageBuffWidth** (unsigned long imageBuffWidthIn)
- void **setImageBuffHeight** (unsigned long imageBuffHeightIn)
- unsigned long **getImageBuffWidth** ()
- unsigned long **getImageBuffHeight** ()
- imageDataFormats::formatOptions **getFormat** ()
- void **setFormat** (imageDataFormats::formatOptions formatIn)
- bool **setFormat** (const QString &text)
- void **setBitDepth** (unsigned int bitDepthIn)
- unsigned int **getBitDepth** ()
- void **setElementsPerPixel** (unsigned long elementsPerPixelIn)
- void **setImageDisplayProperties** (imageDisplayProperties *imageDisplayPropsIn)

- void **setWidthHeightFromDimensions** ()
 // Update the image dimensions (width and height) from the area detector dimension variables.
- void **invalidatePixelLookup** ()
 recalculate (when next required) pixel summary information
- QString **getInfoText** ()
 Generate textual information regarding the current image.

Protected Attributes

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

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

9.26.1 Detailed Description

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

9.26.2 Member Enumeration Documentation

9.26.2.1 enum imageProperties::rotationOptions

Image rotation options

Enumerator:

- ROTATION_0** No image rotation.
- ROTATION_90_RIGHT** Rotate image 90 degrees clockwise.
- ROTATION_90_LEFT** Rotate image 90 degrees anticlockwise.
- ROTATION_180** Rotate image 180 degrees.

9.26.3 Constructor & Destructor Documentation

9.26.3.1 imageProperties::imageProperties()

Constructor.

Construction. Set all image attributes to sensible defaults.

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

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

9.27 imagePropertiesCore Class Reference

Public Member Functions

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

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

9.27.1 Member Function Documentation

9.27.1.1 QImage imagePropertiesCore::buildImageCore ()

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

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

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

9.28 imageUpdateIndicator Class Reference

Public Member Functions

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

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

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

9.29 loginWidget Class Reference

Public Member Functions

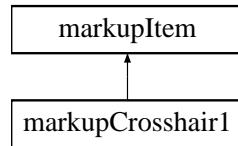
- **loginWidget** (QELogin *ownerIn)
- userLevelTypes::userLevels **getUserType** ()
- QString **getPassword** ()
- void **clearPassword** ()

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

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

9.30 markupCrosshair1 Class Reference

Inheritance diagram for markupCrosshair1:



Public Member Functions

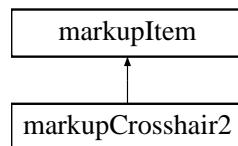
- **markupCrosshair1** ([imageMarkup](#) *ownerIn, const bool interactiveIn, const bool reportOnMoveIn, const QString legendIn)
- void **startDrawing** (const QPoint pos)
- void **setArea** ()
- void **drawMarkup** (QPainter &p)
- void **moveTo** (const QPoint pos)
- bool **isOver** (const QPoint point,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

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

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

9.31 markupCrosshair2 Class Reference

Inheritance diagram for markupCrosshair2:



Public Member Functions

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

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

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

9.32 markupDisplayMenu Class Reference

Public Member Functions

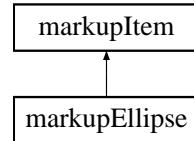
- **markupDisplayMenu** (QWidget *parent=0)
- void **setDisplay** (imageContextMenu::imageContextMenuOptions option, bool state)
- void **setItemText** (imageContextMenu::imageContextMenuOptions option, QString title)
- bool **isDisplayed** (imageContextMenu::imageContextMenuOptions option)
- void **enable**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

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

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

9.33 markupEllipse Class Reference

Inheritance diagram for markupEllipse:



Public Member Functions

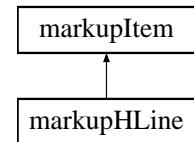
- **markupEllipse** ([imageMarkup](#) *ownerIn, const bool interactiveIn, const bool reportOnMoveIn, const QString legendIn)
- void **startDrawing** (const QPoint pos)
- void **setArea** ()
- void **drawMarkup** (QPainter &p)
- void **moveTo** (const QPoint pos)
- bool **isOver** (const QPoint point,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

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

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

9.34 markupHLine Class Reference

Inheritance diagram for markupHLine:



Public Member Functions

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

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

9.34.1 Member Function Documentation

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

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

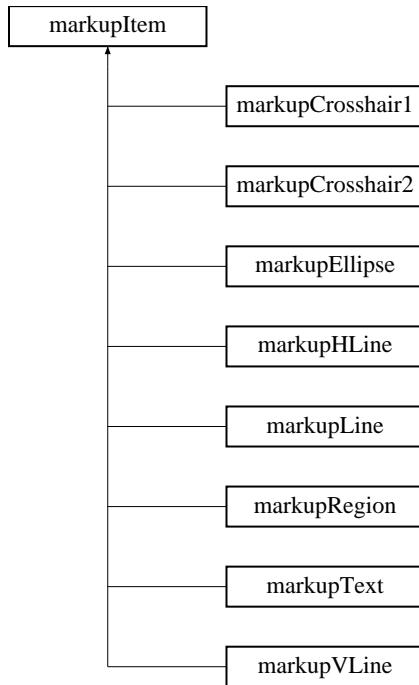
Implements [markupItem](#).

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

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

9.35 markupItem Class Reference

Inheritance diagram for markupItem:



Public Types

- enum **markupHandles** {

 MARKUP_HANDLE_NONE, **MARKUP_HANDLE_START**, **MARKUP_HANDLE_END**, **MARKUP_HANDLE_CENTER**,

 MARKUP_HANDLE_TL, **MARKUP_HANDLE_TR**, **MARKUP_HANDLE_BL**, **MARKUP_HANDLE_BR**,

 MARKUP_HANDLE_T, **MARKUP_HANDLE_B**, **MARKUP_HANDLE_L**, **MARKUP_HANDLE_R** }

Public Member Functions

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

- virtual QPoint **getPoint1** ()=0
- virtual QPoint **getPoint2** ()=0
- virtual QCursors **defaultCursor** ()=0

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

Public Attributes

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

Protected Types

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

Protected Member Functions

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

Protected Attributes

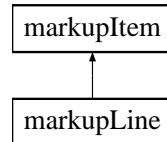
- markupHandles **activeHandle**
- [imageMarkup](#) * **owner**
- unsigned int **thickness**
- unsigned int **maxThickness**

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

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

9.36 markupLine Class Reference

Inheritance diagram for markupLine:



Public Member Functions

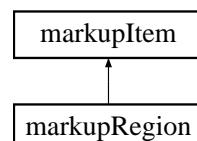
- **markupLine** ([imageMarkup](#) *ownerIn, const bool interactiveIn, const bool reportOnMoveIn, const QString legendIn)
- void **startDrawing** (const QPoint pos)
- void **setArea** ()
- void **drawMarkup** (QPainter &p)
- void **moveTo** (const QPoint pos)
- bool **isOver** (const QPoint point, QCursor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

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

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

9.37 markupRegion Class Reference

Inheritance diagram for markupRegion:



Public Member Functions

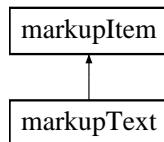
- **markupRegion** (*imageMarkup* *ownerIn, const bool interactiveIn, const bool reportOnMoveIn, const QString legendIn)
- void **startDrawing** (const QPoint pos)
- void **setArea** ()
- void **drawMarkup** (QPainter &p)
- void **moveTo** (const QPoint pos)
- bool **isOver** (const QPoint point, QCursor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

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

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

9.38 markupText Class Reference

Inheritance diagram for markupText:



Public Member Functions

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

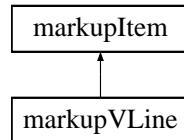
- `QPoint getPoint2 ()`
- `QCursor defaultCursor ()`

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

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

9.39 markupVLine Class Reference

Inheritance diagram for markupVLine:



Public Member Functions

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

9.39.1 Member Function Documentation

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

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

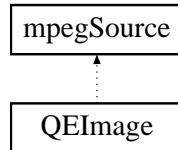
Implements [markupItem](#).

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

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

9.40 mpegSource Class Reference

Inheritance diagram for mpegSource:



Public Member Functions

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

Protected Member Functions

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

9.40.1 Member Function Documentation

9.40.1.1 void mpegSource::updateImage (FFBuffer * buf)

!???: * 3 for color only

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

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

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

9.41 mpegSourceObject Class Reference

Public Slots

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

Signals

- void **aboutToQuit** ()

Public Member Functions

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

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

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

9.42 QEStripChartToolBar::OwnWidgets Class Reference

Public Member Functions

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

Public Attributes

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

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

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

9.43 PeriodicDialog Class Reference

Public Member Functions

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

Protected Member Functions

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

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

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

9.44 PeriodicElementSetupForm Class Reference

Public Member Functions

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

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

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

9.45 PeriodicSetupDialog Class Reference

Public Member Functions

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

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

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

9.46 playbackTimer Class Reference

Public Member Functions

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

Public Attributes

- `recording` * **recorder**

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

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

9.47 pointInfo Class Reference

Public Member Functions

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

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

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

9.48 profilePlot Class Reference

Public Types

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

Public Member Functions

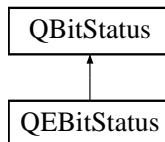
- **profilePlot** (plotDirections plotDirectionIn)
- void **setProfile** (QVector< QPointF > *profile, double minX, double maxX, double minY, double maxY, QString title, QPoint start, QPoint end, unsigned int thicknessIn)
- void **clearProfile** ()

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

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

9.49 QBitStatus Class Reference

Inheritance diagram for QBitStatus:



Public Types

- enum **Orientations** { **LSB_On_Right**, **LSB_On_Bottom**, **LSB_On_Left**, **LSB_On_Top** }
- enum **Shapes** { **Rectangle**, **Circle** }

Public Slots

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

Public Member Functions

- **QBitStatus** (QWidget *parent=0)
- virtual QSize **sizeHint** () const
- void **setBorderColour** (const QColor value)
- QColor **getBorderColour** ()
- void **setOnColour** (const QColor value)
- QColor **getOnColour** ()
- void **setOffColour** (const QColor value)
- QColor **getOffColour** ()
- void **setInvalidColour** (const QColor value)
- QColor **getInvalidColour** ()
- void **setClearColour** (const QColor value)
- QColor **getClearColour** ()
- void **setDrawBorder** (const bool value)
- bool **getDrawBorder** ()
- void **setNumberOfBits** (const int value)
- int **getNumberOfBits** ()
- void **setGap** (const int value)
- int **getGap** ()
- void **setShift** (const int value)
- int **getShift** ()
- void **setOnClearMask** (const QString value)
- QString **getOnClearMask** ()

- void **setOffClearMask** (const QString value)
- QString **getOffClearMask** ()
- void **setReversePolarityMask** (const QString value)
- QString **getReversePolarityMask** ()
- void **setisValid** (const bool value)
- bool **getisValid** ()
- void **setOrientation** (const enum Orientations value)
- enum Orientations **getOrientation** ()
- void **setShape** (const enum Shapes value)
- enum Shapes **getShape** ()
- int **getValue** ()

Protected Member Functions

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

Properties

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

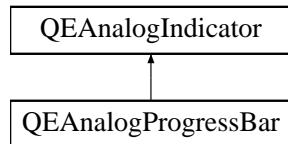
The documentation for this class was generated from the following files:

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

9.50 QEAnalogIndicator Class Reference

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

Inheritance diagram for QEAnalogIndicator:



Classes

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

Public Types

- enum [Orientations](#) { [Left_To_Right](#), [Top_To_Bottom](#), [Right_To_Left](#), [Bottom_To_Top](#) }
- enum [Modes](#) { [Bar](#), [Scale](#), [Meter](#) }

Public Slots

- void [setRange](#) (const double MinimumIn, const double MaximumIn)
- void [setValue](#) (const double ValueIn)

Public Member Functions

- [QEAnalogIndicator](#) (QWidget *parent=0)
Constructor.
- virtual [~QEAnalogIndicator](#) ()
Destructor.
- virtual QSize [sizeHint](#) () const
Size hint.
- double [getValue](#) () const
Access function for [value](#) property - refer to [value](#) property for details.
- void [setMinimum](#) (const double value)
Access function for [minimum](#) - refer to [minimum](#) property for details.
- double [getMinimum](#) () const
Access function for [minimum](#) - refer to [minimum](#) property for details.
- void [setMaximum](#) (const double value)

- Access function for `maximum` - refer to `maximum` property for details.
- `double getMaximum () const`
Access function for `maximum` - refer to `maximum` property for details.
- `void setOrientation (const enum Orientations value)`
Access function for `orientation` - refer to `orientation` property for details.
- `enum Orientations getOrientation () const`
Access function for `orientation` - refer to `orientation` property for details.
- `void setMode (const enum Modes value)`
Access function for `mode` - refer to `mode` property for details.
- `enum Modes getMode () const`
Access function for `mode` - refer to `mode` property for details.
- `void setCentreAngle (const int value)`
Access function for `centreAngle` - refer to `centreAngle` property for details.
- `int getCentreAngle () const`
Access function for `centreAngle` - refer to `centreAngle` property for details.
- `void setSpanAngle (const int value)`
Access function for `spanAngle` - refer to `spanAngle` property for details.
- `int getSpanAngle () const`
Access function for `spanAngle` - refer to `spanAngle` property for details.
- `void setMinorInterval (const double value)`
Access function for `minorInterval` - refer to `minorInterval` property for details.
- `double getMinorInterval () const`
Access function for `minorInterval` - refer to `minorInterval` property for details.
- `void setMajorInterval (const double value)`
Access function for `majorInterval` - refer to `majorInterval` property for details.
- `double getMajorInterval () const`
Access function for `majorInterval` - refer to `majorInterval` property for details.
- `void setLogScaleInterval (const int value)`
Access function for `logScaleInterval` - refer to `logScaleInterval` property for details.
- `int getLogScaleInterval () const`
Access function for `logScaleInterval` - refer to `logScaleInterval` property for details.
- `void setBorderColour (const QColor value)`
Access function for `borderColour` - refer to `borderColour` property for details.
- `QColor getBorderColour () const`
Access function for `borderColour` - refer to `borderColour` property for details.
- `void setForegroundColour (const QColor value)`
Access function for `foregroundColour` - refer to `foregroundColour` property for details.
- `QColor getForegroundColour () const`
Access function for `foregroundColour` - refer to `foregroundColour` property for details.
- `void setBackgroundColour (const QColor value)`
Access function for `backgroundColour` - refer to `backgroundColour` property for details.
- `QColor getBackgroundColour () const`
Access function for `backgroundColour` - refer to `backgroundColour` property for details.

- void **setFontColour** (const QColor value)
Access function for `fontColour` - refer to `fontColour` property for details.
- QColor **getFontColour** () const
Access function for `fontColour` - refer to `fontColour` property for details.
- void **setShowText** (const bool value)
Access function for `showText` - refer to `showText` property for details.
- bool **getShowText** () const
Access function for `showText` - refer to `showText` property for details.
- void **setShowScale** (const bool value)
Access function for `showScale` - refer to `showScale` property for details.
- bool **getShowScale** () const
Access function for `showScale` - refer to `showScale` property for details.
- void **setLogScale** (const bool value)
Access function for `logScale` - refer to `logScale` property for details.
- bool **getLogScale** () const
Access function for `logScale` - refer to `logScale` property for details.

Protected Member Functions

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

Properties

- double `value`
- double `minimum`
- double `maximum`
- double `minorInterval`
- double `majorInterval`
- int `logScaleInterval`
- bool `showText`
- bool `showScale`
- bool `logScale`
- Modes `mode`
- Orientations `orientation`
- int `centreAngle`
- int `spanAngle`
- QColor `borderColour`
- QColor `backgroundColour`
- QColor `foregroundColour`
- QColor `fontColour`
- bool `isActive`

Alternative to `isEnabled`. Default is true.

9.50.1 Detailed Description

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

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

9.50.2 Member Enumeration Documentation

9.50.2.1 enum QEAnalogIndicator::Modes

The type of analog indicator used to represent the value

Enumerator:

Bar Bar (solid bar from minimum up to current value)

Scale Scale (diamond marker tracks current value)

Meter Meter (Needle moving across an arc scale)

9.50.2.2 enum QEAnalogIndicator::Orientations

The orientation of Bar and Scale indicators

Enumerator:

Left_To_Right Left to right.

Top_To_Bottom Top to bottom.

Right_To_Left Right to left.

Bottom_To_Top Bottom to top.

9.50.3 Property Documentation

9.50.3.1 QColor QEAnalogIndicator::backgroundColour [read, write]

Background colour

9.50.3.2 QColor QEAnalogIndicator::borderColour [read, write]

Border colour

9.50.3.3 int QEAnalogIndicator::centreAngle [read, write]

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

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

Font colour

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

Foreground colour

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

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

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

Log scale interval.

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

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

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

Maximum indicated value.

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

Minimum indicated value.

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

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

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

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

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

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

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

If set, show the scale

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

If set, show textual representation of value on the indicator

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

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

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

Current indicated value.

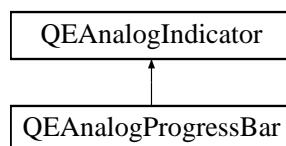
Reimplemented in [QEAnalogProgressBar](#).

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

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

9.51 QEAnalogProgressBar Class Reference

Inheritance diagram for QEAnalogProgressBar:



Public Types

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

- enum `Formats` {
 `Default` = `QQStringFormatting::FORMAT_DEFAULT`, `Floating` = `QQStringFormatting::FORMAT_FLOATING`, `Integer` = `QQStringFormatting::FORMAT_INTEGER`, `UnsignedInteger` = `QQStringFormatting::FORMAT_UNSIGNEDINTEGER`,
 `Time` = `QQStringFormatting::FORMAT_TIME`, `LocalEnumeration` = `QQStringFormatting::FORMAT_LOCAL_ENUMERATE` }
• enum `Notations` { `Fixed` = `QQStringFormatting::NOTATION_FIXED`, `Scientific` = `QQStringFormatting::NOTATION_SCIENTIFIC`, `Automatic` = `QQStringFormatting::NOTATION_AUTOMATIC` }
• enum `ArrayActions` { `Append` = `QQStringFormatting::APPEND`, `Ascii` = `QQStringFormatting::ASCII`, `Index` = `QQStringFormatting::INDEX` }

Public Slots

- void `setManagedVisible` (bool v)

Signals

- void `dbValueChanged` (const 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` ()

- 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 styleSheet`
- `QString defaultStyle`
- `QString userLevelUserStyle`
- `QString userLevelScientistStyle`
- `QString userLevelEngineerStyle`
- `UserLevels userLevelVisibility`
- `UserLevels userLevelEnabled`
- `bool displayAlarmState`
- `DisplayAlarmStateOptions displayAlarmStateOption`
- `AlarmSeverityDisplayModes alarmSeverityDisplayStyle`
- `bool useDbDisplayLimits`
- `int value`
- `bool isActive`

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

- [QString localEnumeration](#)
- [Formats format](#)
- [Notations notation](#)
- [ArrayActions arrayAction](#)

9.51.1 Member Enumeration Documentation

9.51.1.1 enum QEAnalogProgressBar::ArrayActions

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

Enumerator:

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

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

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

9.51.1.2 enum QEAnalogProgressBar::DisplayAlarmStateOptions

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

Enumerator:

Never Refer to [DISPLAY_ALARM_STATE_NEVER](#) for details.

Always Refer to [DISPLAY_ALARM_STATE_ALWAYS](#) for details.

WhenInAlarm Refer to [DISPLAY_ALARM_STATE_WHEN_IN_ALARM](#) for details.

9.51.1.3 enum QEAnalogProgressBar::Formats

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

Enumerator:

Default Format as best appropriate for the data type.

Floating Format as a floating point number.

Integer Format as an integer.

UnsignedInteger Format as an unsigned integer.

Time Format as a time.

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

9.51.1.4 enum QEAnalogProgressBar::Notations

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

Enumerator:

Fixed Refer to QEStringFormatting::NOTATION_FIXED for details.

Scientific Refer to QEStringFormatting::NOTATION_SCIENTIFIC for details.

Automatic Refer to QEStringFormatting::NOTATION_AUTOMATIC for details.

9.51.1.5 enum QEAnalogProgressBar::UserLevels

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

Enumerator:

User Refer to USERLEVEL_USER for details.

Scientist Refer to USERLEVEL_SCIENTIST for details.

Engineer Refer to USERLEVEL_ENGINEER for details.

9.51.2 Constructor & Destructor Documentation

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

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

9.51.2.2 QEAnalogProgressBar::QEAnalogProgressBar (const QString & *variableName*, QWidget * *parent* = 0)

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

9.51.3 Member Function Documentation

9.51.3.1 void QEAnalogProgressBar::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.

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

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

9.51.4 Property Documentation**9.51.4.1 bool QEAnalogProgressBar::addUnits [read, write]**

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

9.51.4.2 AlarmSeverityDisplayModes QEAnalogProgressBar::alarmSeverityDisplayStyle [read, write]

Visualise the EPICS alarm severity

9.51.4.3 bool QEAnalogProgressBar::allowDrop [read, write]

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

9.51.4.4 ArrayActions QEAnalogProgressBar::arrayAction [read, write]

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

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

9.51.4.5 QString QEAnalogProgressBar::defaultStyle [read, write]

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

9.51.4.6 bool QEAnalogProgressBar::displayAlarmState [read, write]

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

9.51.4.7 DisplayAlarmStateOptions QEAnalogProgressBar::displayAlarmStateOption [read, write]

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

9.51.4.8 Formats QEAnalogProgressBar::format [read, write]

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

9.51.4.9 unsigned QEAnalogProgressBar::int [read, write]

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

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.

9.51.4.10 bool QEAnalogProgressBar::leadingZero [read, write]

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

9.51.4.11 QString QEAnalogProgressBar::localEnumeration [read, write]

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

Format is:

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

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

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

may be included in a string to indicate a line break

Examples are:

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

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

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

9.51.4.12 Notations QEAnalogProgressBar::notation [read, write]

Notation used for numerical formatting. Default is fixed.

9.51.4.13 int QEAnalogProgressBar::precision [read, write]

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

9.51.4.14 QString QEAnalogProgressBar::styleSheet [read, write]

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

9.51.4.15 bool QEAnalogProgressBar::trailingZeros [read, write]

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

9.51.4.16 bool QEAnalogProgressBar::useDbDisplayLimits [read, write]

Use the EPICS database display limits

9.51.4.17 bool QEAnalogProgressBar::useDbPrecision [read, write]

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

9.51.4.18 UserLevels QEAnalogProgressBar::userLevelEnabled [read, write]

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

9.51.4.19 QString QEAnalogProgressBar::userLevelEngineerStyle [read, write]

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

9.51.4.20 QString QEAnalogProgressBar::userLevelScientistStyle [read, write]

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

9.51.4.21 QString QEAnalogProgressBar::userLevelUserStyle [read, write]

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

9.51.4.22 UserLevels QEAnalogProgressBar::userLevelVisibility [read, write]

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

9.51.4.23 int QEAnalogProgressBar::value [read, write]

Current indicated value.

Reimplemented from [QEAnalogIndicator](#).

9.51.4.24 QString QEAnalogProgressBar::variable [read, write]

EPICS variable name (CA PV)

9.51.4.25 bool QEAnalogProgressBar::variableAsToolTip [read, write]

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

9.51.4.26 QString QEAnalogProgressBar::variableSubstitutions [read, write]

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

9.51.4.27 bool QEAnalogProgressBar::visible [read, write]

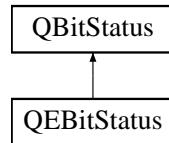
Display the widget. Default is true. Setting this property false is useful if widget is only used to provide a signal - for example, when supplying data to a [QELink](#) widget. Note, when false the widget will still be visible in Qt Designer.

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

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

9.52 QEBitStatus Class Reference

Inheritance diagram for QEBitStatus:



Public Types

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

Public Slots

- void `setManagedVisible` (bool v)

Signals

- void `dbValueChanged` (const 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 **styleSheet**
- QString **defaultStyle**
- QString **userLevelUserStyle**
- QString **userLevelScientistStyle**
- QString **userLevelEngineerStyle**
- UserLevels **userLevelVisibility**
- UserLevels **userLevelEnabled**
- bool **displayAlarmState**
- DisplayAlarmStateOptions **displayAlarmStateOption**

- double **value**
- bool **isActive**
- bool **isValid**

9.52.1 Member Enumeration Documentation

9.52.1.1 enum QEBitStatus::DisplayAlarmStateOptions

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

Enumerator:

Never Refer to DISPLAY_ALARM_STATE_NEVER for details.

Always Refer to DISPLAY_ALARM_STATE_ALWAYS for details.

WhenInAlarm Refer to DISPLAY_ALARM_STATE_WHEN_IN_ALARM for details.

9.52.1.2 enum QEBitStatus::UserLevels

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

Enumerator:

User Refer to USERLEVEL_USER for details.

Scientist Refer to USERLEVEL_SCIENTIST for details.

Engineer Refer to USERLEVEL_ENGINEER for details.

9.52.2 Member Function Documentation

9.52.2.1 void QEBitStatus::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.

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

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

9.52.3 Property Documentation

9.52.3.1 bool QEBitStatus::allowDrop [read, write]

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

9.52.3.2 QString QEBitStatus::defaultStyle [read, write]

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

9.52.3.3 bool QEBitStatus::displayAlarmState [read, write]

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

9.52.3.4 DisplayAlarmStateOptions QEBitStatus::displayAlarmStateOption [read, write]

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

9.52.3.5 unsigned QEBitStatus::int [read, write]

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

9.52.3.6 QString QEBitStatus::styleSheet [read, write]

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

9.52.3.7 UserLevels QEBitStatus::userLevelEnabled [read, write]

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

9.52.3.8 QString QEBitStatus::userLevelEngineerStyle [read, write]

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

9.52.3.9 QString QEBitStatus::userLevelScientistStyle [read, write]

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

9.52.3.10 QString QEBitStatus::userLevelUserStyle [read, write]

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

9.52.3.11 UserLevels QEBitStatus::userLevelVisibility [read, write]

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

9.52.3.12 QString QEBitStatus::variable [read, write]

EPICS variable name (CA PV)

9.52.3.13 bool QEBitStatus::variableAsToolTip [read, write]

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

9.52.3.14 QString QEBitStatus::variableSubstitutions [read, write]

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

9.52.3.15 bool QEBitStatus::visible [read, write]

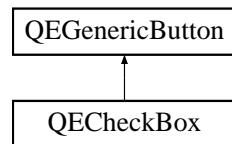
Display the widget. Default is true. Setting this property false is useful if widget is only used to provide a signal - for example, when supplying data to a [QELink](#) widget. Note, when false the widget will still be visible in Qt Designer.

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

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

9.53 QECheckBox Class Reference

Inheritance diagram for QECheckBox:



Public Types

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

- enum `Formats` {

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

`Time` = QEStringFormatting::FORMAT_TIME, `LocalEnumeration` = QEStringFormatting::FORMAT_LOCAL_ENUMERATE }
 - enum `Notations` { `Fixed` = QEStringFormatting::NOTATION_FIXED, `Scientific` = QEStringFormatting::NOTATION_SCIENTIFIC, `Automatic` = QEStringFormatting::NOTATION_AUTOMATIC }
 - enum `ArrayActions` { `Append` = QEStringFormatting::APPEND, `Ascii` = QEStringFormatting::ASCII, `Index` = QEStringFormatting::INDEX }
 - enum `UpdateOptions` { `Text` = QEGenericButton::UPDATE_TEXT, `Icon` = QEGenericButton::UPDATE_ICON, `TextAndIcon` = QEGenericButton::UPDATE_TEXT_AND_ICON, `State` = QEGenericButton::UPDATE_STATE }
- User friendly enumerations for updateOption property - refer to QEGenericButton::updateOptions for details.*
- enum `ProgramStartupOptionNames` { `None` = applicationLauncher::PSO_NONE, `Terminal` = applicationLauncher::PSO_TERMINAL, `LogOutput` = applicationLauncher::PSO_LOGOUTPUT, `StdOutput` = applicationLauncher::PSO_STDOUPUT }
 - enum `CreationOptionNames` {

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

`DockBottom` = QEActionRequests::OptionBottomDockWindow, `DockLeft` = QEActionRequests::OptionLeftDockWindow, `DockRight` = QEActionRequests::OptionRightDockWindow, `DockTopTabbed` = QEActionRequests::OptionTopDockWindowTabbed,

`DockBottomTabbed` = QEActionRequests::OptionBottomDockWindowTabbed, `DockLeftTabbed` = QEActionRequests::OptionLeftDockWindowTabbed, `DockRightTabbed` = QEActionRequests::OptionRightDockWindowTabbed, `DockFloating` = QEActionRequests::OptionFloatingDockWindow }
- Creation options. Used to indicate how to present a GUI when requesting a new GUI be created. Open a new window, open a new tab, or replace the current window.*

Public Slots

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

Update the default style applied to this widget.
- void `setManagedVisible` (bool v)

Signals

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

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

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

Internal use only. Request a new GUI is created. Typically, this is caught by the QEGui application.

- void [pressed](#) (int value)
- void [released](#) (int value)
- void [clicked](#) (int value)
- void [programCompleted](#) ()

Program started by button has completed.

Public Member Functions

- [QECheckBox](#) (QWidget *parent=0)
- [QECheckBox](#) (const QString &variableName, QWidget *parent=0)
- void [writeNow](#) ()
- 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 styleSheet`
- `QString defaultStyle`
- `QString userLevelUserStyle`
- `QString userLevelScientistStyle`
- `QString userLevelEngineerStyle`
- `UserLevels userLevelVisibility`
- `UserLevels userLevelEnabled`
- `bool displayAlarmState`
- `DisplayAlarmStateOptions displayAlarmStateOption`
- `int precision`
- `bool useDbPrecision`
- `bool leadingZero`
- `bool trailingZeros`
- `bool addUnits`
- `QString localEnumeration`
- `Formats format`
- `Notations notation`
- `ArrayActions arrayAction`
- `Qt::Alignment alignment`
- `UpdateOptions updateOption`
- `QPixmap pixmap0`

- `QPixmap pixmap1`
- `QPixmap pixmap2`
- `QPixmap pixmap3`
- `QPixmap pixmap4`
- `QPixmap pixmap5`
- `QPixmap pixmap6`
- `QPixmap pixmap7`
- `QString password`
- `bool confirmAction`
- `QString confirmText`
- `bool writeOnPress`
- `bool writeOnRelease`
- `bool writeOnClick`
- `QString pressText`
- `QString releaseText`
- `QString clickText`
- `QString clickCheckedText`
- `QString labelText`
- `QString program`
- `QStringList arguments`
- `ProgramStartupOptionNames programStartupOption`
- `QString guiFile`
- `CreationOptionNames creationOption`
- `QString prioritySubstitutions`
- `QString customisationName`

9.53.1 Member Enumeration Documentation

9.53.1.1 enum QECheckBox::ArrayActions

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

Enumerator:

Append Refer to `QEStringFormatting::APPEND` for details.

Ascii Refer to `QEStringFormatting::ASCII` for details.

Index Refer to `QEStringFormatting::INDEX` for details.

9.53.1.2 enum QECheckBox::CreationOptionNames

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

Enumerator:

Open Replace the current GUI with the new GUI.

NewTab Open new GUI in a new tab.

NewWindow Open new GUI in a new window.

DockTop Open new GUI in a top dock window.

DockBottom Open new GUI in a bottom dock window.

DockLeft Open new GUI in a left dock window.

DockRight Open new GUI in a right dock window.

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

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

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

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

DockFloating Open new GUI in a floating dock window.

9.53.1.3 enum QECheckBox::DisplayAlarmStateOptions

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

Enumerator:

Never Refer to DISPLAY_ALARM_STATE_NEVER for details.

Always Refer to DISPLAY_ALARM_STATE_ALWAYS for details.

WhenInAlarm Refer to DISPLAY_ALARM_STATE_WHEN_IN_ALARM for details.

9.53.1.4 enum QECheckBox::Formats

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

Enumerator:

Default Format as best appropriate for the data type.

Floating Format as a floating point number.

Integer Format as an integer.

UnsignedInteger Format as an unsigned integer.

Time Format as a time.

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

9.53.1.5 enum QECheckBox::Notations

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

Enumerator:

Fixed Refer to QEStringFormatting::NOTATION_FIXED for details.

Scientific Refer to QEStringFormatting::NOTATION_SCIENTIFIC for details.

Automatic Refer to QEStringFormatting::NOTATION_AUTOMATIC for details.

9.53.1.6 enum QECheckBox::ProgramStartupOptionNames

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

Enumerator:

None Just run the program.

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

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

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

9.53.1.7 enum QECheckBox::UpdateOptions

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

Enumerator:

Text Data updates will update the button text.

Icon Data updates will update the button icon.

TextAndIcon Data updates will update the button text and icon.

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

9.53.1.8 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.

9.53.2 Constructor & Destructor Documentation

9.53.2.1 `QECheckBox::QECheckBox (QWidget * parent = 0)`

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

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

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

9.53.3 Member Function Documentation

9.53.3.1 `void QECheckBox::clicked (int value) [signal]`

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

9.53.3.2 `void QECheckBox::dbValueChanged (const QString & out) [signal]`

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

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

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

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

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

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

Default slot used to create a new GUI if there is no slot indicated in the ContainerProfile class. This slot is typically used when the button is pressed within the Designer preview window to allow the operation of the button to be tested. If an application does not specify a slot to use for creating new windows (through the ContainerProfile class) a window will still be created through this slot, but it will not respect the window creation

options or any other window related application constraints. For example, the QEGui application does provide a slot for creating new GUIs in the ContainerProfile class which respects the creation options, knows how to add tabs in the application, and extend the application's window menu in the menu bar.

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

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

9.53.4 Property Documentation

9.53.4.1 bool QECheckBox::addUnits [read, write]

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

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

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

9.53.4.3 bool QECheckBox::allowDrop [read, write]

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

9.53.4.4 QStringList QECheckBox::arguments [read, write]

Arguments for program specified in the 'program' property.

9.53.4.5 ArrayActions QECheckBox::arrayAction [read, write]

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

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

9.53.4.6 QString QECheckBox::clickCheckedText [read, write]

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

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

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

Reimplemented from [QEGenericButton](#).

9.53.4.7 QString QECheckBox::clickText [read, write]

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

Reimplemented from [QEGenericButton](#).

9.53.4.8 bool QECheckBox::confirmAction [read, write]

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

9.53.4.9 QString QECheckBox::confirmText [read, write]

Text used to confirm action if confirmation dialog is presented

Reimplemented from [QEGenericButton](#).

9.53.4.10 CreationOptionNames QECheckBox::creationOption [read, write]

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

Reimplemented from [QEGenericButton](#).

9.53.4.11 QString QECheckBox::customisationName [read, write]

Window customisation name. This name will be used to select a set of window customisations including menu items and tool bar buttons. Applications such as QEGui

can load .xml files containing named sets of window customisations. This property is used to select a set loaded from these files. The selected set of customisations will be applied to the main window containing the new GUI.

Reimplemented from [QEGenericButton](#).

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

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

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

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

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

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

9.53.4.15 **Formats QECheckBox::format** [read, write]

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

9.53.4.16 **QString QECheckBox::guiFile** [read, write]

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

9.53.4.17 unsigned QECheckBox::int [read, write]

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

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.

9.53.4.18 QString QECheckBox::labelText [read, write]

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

Reimplemented from [QEGenericButton](#).

9.53.4.19 bool QECheckBox::leadingZero [read, write]

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

9.53.4.20 QString QECheckBox::localEnumeration [read, write]

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

Format is:

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

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

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

may be included in a string to indicate a line break

Examples are:

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

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

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

9.53.4.21 Notations QECheckBox::notation [read, write]

Notation used for numerical formatting. Default is fixed.

9.53.4.22 QString QECheckBox::password [read, write]

Password user will need to enter before any action is taken

Reimplemented from [QEGenericButton](#).

9.53.4.23 QPixmap QECheckBox:: pixmap [read, write]

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

9.53.4.24 QPixmap QECheckBox:: pixmap1 [read, write]

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

9.53.4.25 QPixmap QECheckBox:: pixmap2 [read, write]

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

9.53.4.26 QPixmap QECheckBox:: pixmap3 [read, write]

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

9.53.4.27 QPixmap QECheckBox:: pixmap4 [read, write]

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

9.53.4.28 QPixmap QECheckBox:: pixmap5 [read, write]

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

9.53.4.29 QPixmap QECheckBox:: pixmap6 [read, write]

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

9.53.4.30 QPixmap QECheckBox:: pixmap7 [read, write]

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

9.53.4.31 int QECheckBox:: precision [read, write]

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

9.53.4.32 QString QECheckBox:: pressText [read, write]

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

Reimplemented from [QEGenericButton](#).

9.53.4.33 QString QECheckBox:: prioritySubstitutions [read, write]

Overriding macro substitutions. These macro 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](#).

9.53.4.34 QString QECheckBox::program [read, write]

Program to run when the button is clicked. No attempt to run a program is made if this property is empty. Example: firefox

9.53.4.35 ProgramStartupOptionNames QECheckBox::programStartupOption [read, write]

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

9.53.4.36 QString QECheckBox::releaseText [read, write]

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

Reimplemented from [QEGenericButton](#).

9.53.4.37 QString QECheckBox::styleSheet [read, write]

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

9.53.4.38 bool QECheckBox::subscribe [read, write]

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

9.53.4.39 bool QECheckBox::trailingZeros [read, write]

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

9.53.4.40 UpdateOptions QECheckBox::updateOption [read, write]

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

Reimplemented from [QEGenericButton](#).

9.53.4.41 bool QECheckBox::useDbPrecision [read, write]

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

9.53.4.42 UserLevels QECheckBox::userLevelEnabled [read, write]

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

9.53.4.43 QString QECheckBox::userLevelEngineerStyle [read, write]

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

9.53.4.44 QString QECheckBox::userLevelScientistStyle [read, write]

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

9.53.4.45 QString QECheckBox::userLevelUserStyle [read, write]

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

9.53.4.46 UserLevels QECheckBox::userLevelVisibility [read, write]

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

9.53.4.47 **QString QECheckBox::variable** [read, write]

EPICS variable name (CA PV)

9.53.4.48 **bool QECheckBox::variableAsToolTip** [read, write]

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

9.53.4.49 **QString QECheckBox::variableSubstitutions** [read, write]

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

9.53.4.50 **bool QECheckBox::visible** [read, write]

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

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

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

Reimplemented from [QEGenericButton](#).

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

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

Reimplemented from [QEGenericButton](#).

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

If true, the 'releaseText' property is written when the button is released. Default is false

Reimplemented from [QEGenericButton](#).

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

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

9.54 QECheckBoxManager Class Reference

Public Member Functions

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

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

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

9.55 QEComboBox Class Reference

Public Types

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

Public Slots

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

Signals

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

Protected Member Functions

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

Protected Attributes

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

Properties

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

9.55.1 Member Enumeration Documentation

9.55.1.1 enum QEComboBox::DisplayAlarmStateOptions

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

Enumerator:

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

9.55.1.2 enum QEComboBox::UserLevels

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

Enumerator:

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

9.55.2 Member Function Documentation**9.55.2.1 void QEComboBox::dbValueChanged (const qlonglong & out) [signal]**

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

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

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

9.55.3 Member Data Documentation**9.55.3.1 bool QEComboBox::useDbEnumerations [read, write, protected]**

Use database enumerations - defaults to true

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

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

9.55.4 Property Documentation

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

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

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

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

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

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

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

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

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

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

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

Enumerations values used when `useDbEnumerations` is false.

9.55.4.7 QString QEComboBox::styleSheet [read, write]

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

9.55.4.8 bool QEComboBox::subscribe [read, write]

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

9.55.4.9 UserLevels QEComboBox::userLevelEnabled [read, write]

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

9.55.4.10 QString QEComboBox::userLevelEngineerStyle [read, write]

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

9.55.4.11 QString QEComboBox::userLevelScientistStyle [read, write]

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

9.55.4.12 QString QEComboBox::userLevelUserStyle [read, write]

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

9.55.4.13 UserLevels QEComboBox::userLevelVisibility [read, write]

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

9.55.4.14 QString QEComboBox::variable [read, write]

EPICS variable name (CA PV)

9.55.4.15 bool QEComboBox::variableAsToolTip [read, write]

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

9.55.4.16 QString QEComboBox::variableSubstitutions [read, write]

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

9.55.4.17 bool QEComboBox::visible [read, write]

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

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

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

9.56 QEConfiguredLayout Class Reference

Public Types

- enum **configurationTypesProperty** { **File** = FROM_FILE, **Text** = FROM_TEXT }
- enum **optionsLayoutProperty** { **Top** = TOP, **Bottom** = BOTTOM, **Left** = LEFT, **Right** = RIGHT }
- enum **UserLevels** { **User** = userLevelTypes::USERLEVEL_USER, **Scientist** = userLevelTypes::USERLEVEL_SCIENTIST, **Engineer** = userLevelTypes::USERLEVEL_ENGINEER }

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

Public Slots

- void **setManagedVisible** (bool v)

Public Member Functions

- **QEConfiguredLayout** (QWidget *pParent=0, bool pSubscription=true)
- void **setItemDescription** (QString pValue)
- QString **getItemDescription** ()
- void **setShowItemList** (bool pValue)
- bool **getShowItemList** ()
- void **setConfigurationType** (int pValue)
- int **getConfigurationType** ()
- void **setConfigurationFile** (QString pValue)
- QString **getConfigurationFile** ()
- void **setConfigurationText** (QString pValue)
- QString **getConfigurationText** ()
- void **setOptionsLayout** (int pValue)
- int **getOptionsLayout** ()
- void **setCurrentUserType** (int pValue)
- int **getCurrentUserType** ()
- void **refreshFields** ()
- void **userLevelChanged** (userLevelTypes::userLevels pValue)
- void **setConfigurationTypeProperty** (configurationTypesProperty pConfigurationType)
 - configurationTypesProperty **getConfigurationTypeProperty** ()
 - void **setOptionsLayoutProperty** (optionsLayoutProperty pOptionsLayout)
 - optionsLayoutProperty **getOptionsLayoutProperty** ()
 - UserLevels **getUserLevelVisibilityProperty** ()
 - Access function for userLevelVisibility property - refer to userLevelVisibility property for details.*
 - void **setUserLevelVisibilityProperty** (UserLevels level)
 - Access function for userLevelVisibility property - refer to userLevelVisibility property for details.*
 - UserLevels **getUserLevelEnabledProperty** ()
 - Access function for userLevelEnabled property - refer to userLevelEnabled property for details.*
 - void **setUserLevelEnabledProperty** (UserLevels level)
 - Access function for userLevelEnabled property - refer to userLevelEnabled property for details.*
 - **DisplayAlarmStateOptions getDisplayAlarmStateOptionProperty** ()

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

- void `setDisplayAlarmStateOptionProperty (DisplayAlarmStateOptions option)`

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

Public Attributes

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

Protected Attributes

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

Properties

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

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

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

9.56.1 Member Enumeration Documentation

9.56.1.1 enum QEConfiguredLayout::DisplayAlarmStateOptions

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

Enumerator:

Never Refer to DISPLAY_ALARM_STATE_NEVER for details.

Always Refer to DISPLAY_ALARM_STATE_ALWAYS for details.

WhenInAlarm Refer to DISPLAY_ALARM_STATE_WHEN_IN_ALARM for details.

9.56.1.2 enum QEConfiguredLayout::UserLevels

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

Enumerator:

User Refer to USERLEVEL_USER for details.

Scientist Refer to USERLEVEL_SCIENTIST for details.

Engineer Refer to USERLEVEL_ENGINEER for details.

9.56.2 Member Function Documentation

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

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

9.56.3 Property Documentation

9.56.3.1 bool QEConfiguredLayout::allowDrop [read, write]

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

9.56.3.2 QString QEConfiguredLayout::defaultStyle [read, write]

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

9.56.3.3 bool QEConfiguredLayout::displayAlarmState [read, write]

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

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

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

9.56.3.5 unsigned QEConfiguredLayout::int [read, write]

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

9.56.3.6 QString QEConfiguredLayout::styleSheet [read, write]

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

9.56.3.7 UserLevels QEConfiguredLayout::userLevelEnabled [read, write]

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

9.56.3.8 QString QEConfiguredLayout::userLevelEngineerStyle [read, write]

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

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

9.56.3.9 `QString QEConfiguredLayout::userLevelScientistStyle [read, write]`

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

9.56.3.10 `QString QEConfiguredLayout::userLevelUserStyle [read, write]`

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

9.56.3.11 `UserLevels QEConfiguredLayout::userLevelVisibility [read, write]`

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

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

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

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

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

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

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

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

9.57 QEConfiguredLayoutManager Class Reference

Public Member Functions

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

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

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

9.58 QEFileBrowser Class Reference

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

Public Types

- enum **optionsLayoutProperty** { **Top** = TOP, **Bottom** = BOTTOM, **Left** = LEFT, **Right** = RIGHT }
- enum **UserLevels** { **User** = userLevelTypes::USERLEVEL_USER, **Scientist** = userLevelTypes::USERLEVEL_SCIENTIST, **Engineer** = userLevelTypes::USERLEVEL_ENGINEER }
- enum **DisplayAlarmStateOptions** { **Never** = standardProperties::DISPLAY_ALARM_STATE_NEVER, **Always** = standardProperties::DISPLAY_ALARM_STATE_ALWAYS, **WhenInAlarm** = standardProperties::DISPLAY_ALARM_STATE_WHEN_IN_ALARM }

Public Slots

- void **setManagedVisible** (bool v)

Signals

- void **selected** (QString pFilename)

Public Member Functions

- **QEFileBrowser** (QWidget *pParent=0)
- void **setVariableName** (QString pValue)
- QString **getVariableName** ()
- void **setVariableNameSubstitutions** (QString pValue)
- QString **getVariableNameSubstitutions** ()
- void **setDirectoryPath** (QString pValue)
- QString **getDirectoryPath** ()
- void **setShowDirectoryPath** (bool pValue)
- bool **getShowDirectoryPath** ()
- void **setShowDirectoryBrowser** (bool pValue)
- bool **getShowDirectoryBrowser** ()
- void **setShowRefresh** (bool pValue)
- bool **getShowRefresh** ()
- void **setShowTable** (bool pValue)
- bool **getShowTable** ()
- void **setShowColumnTime** (bool pValue)
- bool **getShowColumnTime** ()
- void **setShowColumnSize** (bool pValue)
- bool **getShowColumnSize** ()
- void **setShowColumnFilename** (bool pValue)
- bool **getShowColumnFilename** ()
- void **setShowFileDialogExtension** (bool pValue)
- bool **getShowFileDialogExtension** ()
- 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 RIG.

- bool [variableAsToolTip](#)

- bool [allowDrop](#)

- bool [visible](#)

- unsigned int

- QString [styleSheet](#)

- QString [defaultStyle](#)

- QString [userLevelUserStyle](#)

- QString [userLevelScientistStyle](#)

- QString [userLevelEngineerStyle](#)

- [UserLevels userLevelVisibility](#)

- [UserLevels userLevelEnabled](#)

- bool [displayAlarmState](#)

- [DisplayAlarmStateOptions displayAlarmStateOption](#)

9.58.1 Detailed Description

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

9.58.2 Member Enumeration Documentation

9.58.2.1 enum QEFileBrowser::DisplayAlarmStateOptions

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

Enumerator:

Never Refer to DISPLAY_ALARM_STATE_NEVER for details.

Always Refer to DISPLAY_ALARM_STATE_ALWAYS for details.

WhenInAlarm Refer to DISPLAY_ALARM_STATE_WHEN_IN_ALARM for details.

9.58.2.2 enum QEFileBrowser::UserLevels

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

Enumerator:

User Refer to USERLEVEL_USER for details.

Scientist Refer to USERLEVEL_SCIENTIST for details.

Engineer Refer to USERLEVEL_ENGINEER for details.

9.58.3 Member Function Documentation

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

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

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

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

9.58.4 Property Documentation

9.58.4.1 bool QEFileBrowser::allowDrop [read, write]

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

9.58.4.2 QString QEFileBrowser::defaultStyle [read, write]

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

9.58.4.3 bool QEFileBrowser::displayAlarmState [read, write]

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

9.58.4.4 DisplayAlarmStateOptions QEFileBrowser::displayAlarmStateOption [read, write]

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

9.58.4.5 unsigned QEFileBrowser::int [read, write]

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

9.58.4.6 QString QEFileBrowser::styleSheet [read, write]

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

9.58.4.7 UserLevels QEFileBrowser::userLevelEnabled [read, write]

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

9.58.4.8 QString QEFileBrowser::userLevelEngineerStyle [read, write]

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

9.58.4.9 QString QEFileBrowser::userLevelScientistStyle [read, write]

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

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

9.58.4.10 **QString QEFileBrowser::userLevelUserStyle** [read, write]

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

9.58.4.11 **UserLevels QEFileBrowser::userLevelVisibility** [read, write]

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

9.58.4.12 **QString QEFileBrowser::variable** [read, write]

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

9.58.4.13 **bool QEFileBrowser::variableAsToolTip** [read, write]

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

9.58.4.14 **QString QEFileBrowser::variableSubstitutions** [read, write]

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

9.58.4.15 **bool QEFileBrowser::visible** [read, write]

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

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

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

9.59 QEForm Class Reference

Public Types

- enum **MessageFilterOptions** { **Match** = UserMessage::MESSAGE_FILTER_MATCH, **None** = UserMessage::MESSAGE_FILTER_NONE }

Public Slots

- bool **readUiFile** ()

*Find a widget within the ui loaded by the **QEForm**. Returns NULL if no UI is loaded yet or if the named widget can't be found.*
- void **requestAction** (const QEActionRequests &request)

*Read a .ui file and present it within this **QEForm**.*

Signals

- void **formLoaded** (bool fileLoaded)

Public Member Functions

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

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

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

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

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

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

Get flag indicating if form should take account of file monitoring.

- bool [getHandleGuiLaunchRequests \(\)](#)
Set flag indicating form should handle gui form launch requests.
- void [setResizeContents \(bool resizeContentsIn\)](#)
Get flag indicating form should handle gui form launch requests.
- bool [getResizeContents \(\)](#)
Set flag indicating form should resize contents to match form size (otherwise resize form to match contents)
- QString [getContainedFrameworkVersion \(\)](#)
Get flag indicating form should resize contents to match form size (otherwise resize form to match contents)
- QString [getUniqueId \(\)](#)
Get the version of the first QE widget (if any) of QE widgets by QUILoader.
- void [setUniqueId \(QString name\)](#)
Get a unique identifier string for this form. This identifier should be persistant across application runs as it is based on the QEForm's position in the widget hierarchy. The same widget will generate the same identifier when opened within the same GUI.
- int [getDisconnectedCount \(\)](#)
Set a unique identifier string for this form. This identifier should be persistant across application runs as it is based on the QEForm's position in the widget hierarchy. The same widget will generate the same identifier when opened within the same GUI.
- int [getConnectedCount \(\)](#)
Return the count of disconnected variables.
- QWidget * [getChild \(QString name\)](#)
Return the count of connected variables.
- void [setUiFileNameProperty \(QString uiFileName\)](#)
- QString [getUiFileNameProperty \(\)](#)
- void [setVariableNameSubstitutionsProperty \(QString variableNameSubstitutions\)](#)
- QString [getVariableNameSubstitutionsProperty \(\)](#)
- MessageFilterOptions [getMessageFormFilter \(\)](#)
- void [setMessageFormFilter \(MessageFilterOptions messageFormFilter\)](#)
- MessageFilterOptions [getMessageSourceFilter \(\)](#)
- void [setMessageSourceFilter \(MessageFilterOptions messageSourceFilter\)](#)

Protected Attributes

- QString **uiFileName**
- QString **fullUiFileName**
- bool [handleGuiLaunchRequests](#)
- bool [resizeContents](#)

Properties

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

9.59.1 Member Data Documentation

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

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

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

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

9.59.2 Property Documentation

9.59.2.1 bool QEForm::allowDrop [read, write]

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

9.59.2.2 DisplayAlarmStateOptions QEForm::displayAlarmStateOption [read, write]

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

9.59.2.3 unsigned QEForm::int [read, write]

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

9.59.2.4 MessageFilterOptions QEForm::messageFormFilter [read, write]

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

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

9.59.2.5 MessageFilterOptions QEForm::messageSourceFilter [read, write]

!!?? is this a valid property. Resending messages based on the source ID is unnecessary as they will be sent on with the same source ID? Message filter that attempts to match messages sent through the QE message logging system based on the messageSourceld of the widget that generatedd the messge. This filter will match message message source ID of the message to the message source ID of this QEform as follows:

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

9.59.2.6 QString QEForm::uiFile [read, write]

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

9.59.2.7 bool QEForm::variableAsToolTip [read, write]

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

9.59.2.8 QString QEForm::variableSubstitutions [read, write]

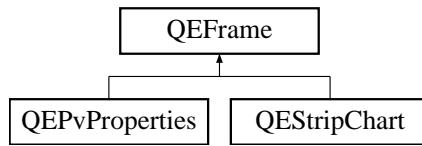
Macro substitutions to be applied to this widget, and all QE widgets that are opened when the .ui file is presented. Note, despite the name, the macro substitutions are general macro substitutions, and do not just apply to a variable name (in fact a [QEForm](#) widget does not even have a variable name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

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

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

9.60 QEFrame Class Reference

Inheritance diagram for QEFrame:



Public Types

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

Public Slots

- void `setManagedVisible` (bool v)

Public Member Functions

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

9.60.1 Member Enumeration Documentation

9.60.1.1 enum QEFrame::DisplayAlarmStateOptions

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

Enumerator:

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

9.60.1.2 enum QEFrame::UserLevels

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

Enumerator:

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

9.60.2 Member Function Documentation

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

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

9.60.3 Property Documentation

9.60.3.1 bool QEFrame::allowDrop [read, write]

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

9.60.3.2 QString QEFrame::defaultStyle [read, write]

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

9.60.3.3 bool QEFrame::displayAlarmState [read, write]

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

9.60.3.4 DisplayAlarmStateOptions QEFrame::displayAlarmStateOption [read, write]

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

9.60.3.5 unsigned QEFrame::int [read, write]

Set the ID used by the message filtering system. Default is zero. Widgets or applications that use messages from the framework have the option of filtering on this ID. For

example, by using a unique message source ID a [QELog](#) widget may be set up to only log messages from a select set of widgets.

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

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

9.60.3.7 `UserLevels QEFrame::userLevelEnabled [read, write]`

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

9.60.3.8 `QString QEFrame::userLevelEngineerStyle [read, write]`

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

9.60.3.9 `QString QEFrame::userLevelScientistStyle [read, write]`

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

9.60.3.10 `QString QEFrame::userLevelUserStyle [read, write]`

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

9.60.3.11 **UserLevels** QEFrame::userLevelVisibility [read, write]

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

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

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

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

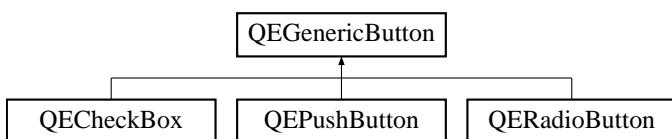
Display the widget. Default is true. Setting this property false is usefull if widget is only used to provide a signal - for example, when supplying data to a [QELink](#) widget. Note, when false the widget will still be visible in Qt Designer.

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

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

9.61 QEGenericButton Class Reference

Inheritance diagram for QEGenericButton:



Public Types

- enum **updateOptions** { **UPDATE_TEXT**, **UPDATE_ICON**, **UPDATE_TEXT_AND_ICON**, **UPDATE_STATE** }

Public Member Functions

- **QEGenericButton** (QWidget *owner)
- void **setSubscribe** (bool subscribe)

- bool **getSubscribe** ()
- void **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)
- void **processWriteNow** (const bool checked)
- virtual updateOptions **getDefaultValue** ()=0
- void **setup** ()
- void **establishConnection** (unsigned int variableIndex)
- void **calcStyleOption** ()

Protected Attributes

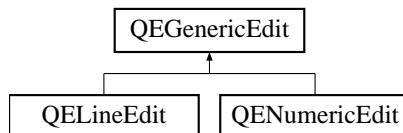
- applicationLauncher **programLauncher**

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

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

9.62 QEGenericEdit Class Reference

Inheritance diagram for QEGenericEdit:



Public Types

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

Public Slots

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

Update the default style applied to this widget.

Signals

- void `userChange` (const QVariant &oldValue, const QVariant &newValue, const QVariant &lastValue)
Internal use only. Used by `QEConfiguredLayout` to be notified when one of its widgets has written something.
- void `requestResend` ()
Internal use only. Used when changing a property value to force a re-display to reflect the new property value.

Public Member Functions

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

Property access function for `variableSubstitutions` property. This has special behaviour to work well within designer.
- QString `getVariableNameSubstitutionsProperty` ()
Property access function for `variableSubstitutions` property. This has special behaviour to work well within designer.
- UserLevels `getUserLevelVisibilityProperty` ()
Access function for `userLevelVisibility` property - refer to `userLevelVisibility` property for details.
- void `setUserLevelVisibilityProperty` (UserLevels level)
Access function for `userLevelVisibility` property - refer to `userLevelVisibility` property for details.
- UserLevels `getUserLevelEnabledProperty` ()
Access function for `userLevelEnabled` property - refer to `userLevelEnabled` property for details.
- void `setUserLevelEnabledProperty` (UserLevels level)
Access function for `userLevelEnabled` property - refer to `userLevelEnabled` property for details.
- DisplayAlarmStateOptions `getDisplayAlarmStateOptionProperty` ()
Access function for `displayAlarmStateOption` property - refer to `displayAlarmStateOption` property for details.

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

Protected Member Functions

- void **setDataIfNoFocus** (const QVariant &value, QCaAlarmInfo &alarmInfo, QCaDateTime &dateTime)
- bool **getIsConnected** ()
- 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 `styleSheet`
- QString `defaultStyle`
- QString `userLevelUserStyle`
- QString `userLevelScientistStyle`
- QString `userLevelEngineerStyle`
- UserLevels `userLevelVisibility`
- UserLevels `userLevelEnabled`
- bool `displayAlarmState`
- DisplayAlarmStateOptions `displayAlarmStateOption`

9.62.1 Member Enumeration Documentation

9.62.1.1 enum QEGenericEdit::DisplayAlarmStateOptions

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

Enumerator:

Never Refer to `DISPLAY_ALARM_STATE_NEVER` for details.

Always Refer to `DISPLAY_ALARM_STATE_ALWAYS` for details.

WhenInAlarm Refer to `DISPLAY_ALARM_STATE_WHEN_IN_ALARM` for details.

9.62.1.2 enum QEGenericEdit::UserLevels

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

Enumerator:

User Refer to `USERLEVEL_USER` for details.

Scientist Refer to `USERLEVEL_SCIENTIST` for details.

Engineer Refer to `USERLEVEL_ENGINEER` for details.

9.62.2 Constructor & Destructor Documentation

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

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

9.62.2.2 `QEGenericEdit::QEGenericEdit(const QString & variableName, QWidget * parent = 0)`

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

9.62.3 Member Function Documentation

9.62.3.1 `bool QEGenericEdit::getConfirmWrite()`

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

9.62.3.2 `bool QEGenericEdit::getSubscribe()`

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

9.62.3.3 `bool QEGenericEdit::getWriteOnEnter()`

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

9.62.3.4 `bool QEGenericEdit::getWriteOnFinish()`

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

9.62.3.5 `bool QEGenericEdit::getWriteOnLoseFocus()`

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

9.62.3.6 `void QEGenericEdit::setConfirmWrite(bool confirmWrite)`

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

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

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

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

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

9.62.3.9 void QEGenericEdit::setWriteOnEnter (bool *writeOnEnter*)

Sets if this widget writes any changes when the user presses 'enter'. Note, the current value will be written even if the user has not changed it. Default is 'true' (writes any changes when the user presses 'enter').

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

Sets if this widget writes any changes when the user finished editing (the QLineEdit 'editingFinished' signal is emitted). No writing occurs if no changes were made. Default is 'true' (writes any changes when the QLineEdit 'editingFinished' signal is emitted).

9.62.3.11 void QEGenericEdit::setWriteOnLoseFocus (bool *writeOnLoseFocus*)

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

9.62.4 Property Documentation

9.62.4.1 bool QEGenericEdit::allowDrop [read, write]

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

9.62.4.2 bool QEGenericEdit::confirmWrite [read, write]

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

9.62.4.3 QString QEGenericEdit::defaultStyle [read, write]

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

9.62.4.4 bool QEGenericEdit::displayAlarmState [read, write]

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

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

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

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

9.62.4.6 `unsigned QEGenericEdit::int` [read, write]

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

Reimplemented in [QELineEdit](#).

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

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

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

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

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

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

9.62.4.10 QString QEGenericEdit::userLevelEngineerStyle [read, write]

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

9.62.4.11 QString QEGenericEdit::userLevelScientistStyle [read, write]

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

9.62.4.12 QString QEGenericEdit::userLevelUserStyle [read, write]

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

9.62.4.13 UserLevels QEGenericEdit::userLevelVisibility [read, write]

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

9.62.4.14 QString QEGenericEdit::variable [read, write]

EPICS variable name (CA PV)

9.62.4.15 bool QEGenericEdit::variableAsToolTip [read, write]

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

9.62.4.16 QString QEGenericEdit::variableSubstitutions [read, write]

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

9.62.4.17 bool QEGenericEdit::visible [read, write]

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

9.62.4.18 bool QEGenericEdit::writeOnEnter [read, write]

Sets if this widget writes any changes when the user presses 'enter'. Note, the current value will be written even if the user has not changed it. Default is 'true' (writes any changes when the user presses 'enter').

9.62.4.19 bool QEGenericEdit::writeOnFinish [read, write]

Sets if this widget writes any changes when the user finished editing (the QLineEdit 'editingFinished' signal is emitted). No writing occurs if no changes were made. Default is 'true' (writes any changes when the QLineEdit 'editingFinished' signal is emitted).

9.62.4.20 bool QEGenericEdit::writeOnLoseFocus [read, write]

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

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

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

9.63 QEGroupBox Class Reference

Public Types

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

Public Slots

- void `setManagedVisible` (bool v)

Public Member Functions

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

Protected Member Functions

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

Properties

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

- [UserLevels userLevelEnabled](#)
- [bool displayAlarmState](#)
- [DisplayAlarmStateOptions displayAlarmStateOption](#)
- [QString substitutedTitle](#)
- [QString textSubstitutions](#)

9.63.1 Member Enumeration Documentation

9.63.1.1 enum QEGroupBox::DisplayAlarmStateOptions

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

Enumerator:

Never Refer to [DISPLAY_ALARM_STATE_NEVER](#) for details.

Always Refer to [DISPLAY_ALARM_STATE_ALWAYS](#) for details.

WhenInAlarm Refer to [DISPLAY_ALARM_STATE_WHEN_IN_ALARM](#) for details.

9.63.1.2 enum QEGroupBox::UserLevels

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

Enumerator:

User Refer to [USERLEVEL_USER](#) for details.

Scientist Refer to [USERLEVEL_SCIENTIST](#) for details.

Engineer Refer to [USERLEVEL_ENGINEER](#) for details.

9.63.2 Member Function Documentation

9.63.2.1 void QEGroupBox::setManagedVisible (bool v) [inline, slot]

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

9.63.3 Property Documentation

9.63.3.1 bool QEGroupBox::allowDrop [read, write]

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

9.63.3.2 QString QEGroupBox::defaultStyle [read, write]

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

9.63.3.3 bool QEGroupBox::displayAlarmState [read, write]

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

9.63.3.4 DisplayAlarmStateOptions QEGroupBox::displayAlarmStateOption [read, write]

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

9.63.3.5 unsigned QEGroupBox::int [read, write]

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

9.63.3.6 QString QEGroupBox::styleSheet [read, write]

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

9.63.3.7 QString QEGroupBox::substitutedTitle [read, write]

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

9.63.3.8 QString QEGroupBox::textSubstitutions [read, write]

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

9.63.3.9 UserLevels QEGroupBox::userLevelEnabled [read, write]

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

9.63.3.10 QString QEGroupBox::userLevelEngineerStyle [read, write]

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

9.63.3.11 QString QEGroupBox::userLevelScientistStyle [read, write]

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

9.63.3.12 QString QEGroupBox::userLevelUserStyle [read, write]

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

9.63.3.13 UserLevels QEGroupBox::userLevelVisibility [read, write]

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

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

9.63.3.14 bool QEGroupBox::variableAsToolTip [read, write]

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

9.63.3.15 bool QEGroupBox::visible [read, write]

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

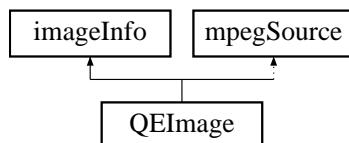
The documentation for this class was generated from the following files:

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

9.64 QEImage Class Reference

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

Inheritance diagram for QEImage:



Public Types

- enum [selectOptions](#) {
 [SO_NONE](#), [SO_PANNING](#), [SO_VSLICE1](#), [SO_VSLICE2](#),
[SO_VSLICE3](#), [SO_VSLICE4](#), [SO_VSLICE5](#), [SO_HSLICE1](#),
[SO_HSLICE2](#), [SO_HSLICE3](#), [SO_HSLICE4](#), [SO_HSLICE5](#),
[SO_AREA1](#), [SO_AREA2](#), [SO_AREA3](#), [SO_AREA4](#),
[SO_PROFILE](#), [SO_TARGET](#), [SO_BEAM](#) }
- enum [imageUses](#) { [IMAGE_USE_DISPLAY](#), [IMAGE_USE_SAVE](#), [IMAGE_USE_DISPLAY_AND_SAVE](#) }
- enum [resizeOptions](#) { [RESIZE_OPTION_ZOOM](#), [RESIZE_OPTION_FIT](#) }

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

`Mono` = `imageDataFormats::MONO`, `Bayer` = `imageDataFormats::BAYERRG`, `BayerGB` = `imageDataFormats::BAYERGB`, `BayerBG` = `imageDataFormats::BAYERBG`,

`BayerGR` = `imageDataFormats::BAYERGR`, `BayerRG` = `imageDataFormats::BAYERRG`, `rgb1` = `imageDataFormats::RGB1`, `rgb2` = `imageDataFormats::RGB2`,

`rgb3` = `imageDataFormats::RGB3`, `yuv444` = `imageDataFormats::YUV444`, `yuv422` = `imageDataFormats::YUV422`, `yuv421` = `imageDataFormats::YUV421` }
- enum `EllipseVariableDefinitions` { `BoundingRectangle` = `BOUNDRING_RECTANGLE`, `CenterAndSize` = `CENTRE_AND_SIZE` }
- enum `TargetOptions` { `DottedFullCrosshair` = `VideoWidget::CROSSHAIR1`, `SolidSmallCrosshair` = `VideoWidget::CROSSHAIR2` }
- enum `ResizeOptions` { `Zoom` = `QEImage::RESIZE_OPTION_ZOOM`, `Fit` = `QEImage::RESIZE_OPTION_FIT` }
- enum `RotationOptions` { `NoRotation` = `imageProperties::ROTATION_0`, `Rotate90Right` = `imageProperties::ROTATION_90_RIGHT`, `Rotate90Left` = `imageProperties::ROTATION_90_LEFT`, `Rotate180` = `imageProperties::ROTATION_180` }
- enum `ProgramStartupOptionNames` { `None` = `applicationLauncher::PSO_NONE`, `Terminal` = `applicationLauncher::PSO_TERMINAL`, `LogOutput` = `applicationLauncher::PSO_LOGOUTPUT`, `StdOutput` = `applicationLauncher::PSO_STDOUPUT` }

Public Slots

- void `setImageFile` (QString name)

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

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

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

Framework use only. Slot to allow external setting of selection menu options.
- void `setSelectArea1Mode` ()

Framework use only. Slot to allow external setting of selection menu options.
- void `setSelectArea2Mode` ()

Framework use only. Slot to allow external setting of selection menu options.
- void `setSelectArea3Mode` ()

Framework use only. Slot to allow external setting of selection menu options.
- void `setSelectArea4Mode` ()

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

- void `setSelectProfileMode ()`
Framework use only. Slot to allow external setting of selection menu options.
- void `setSelectTargetMode ()`
Framework use only. Slot to allow external setting of selection menu options.
- void `setSelectBeamMode ()`
Framework use only. Slot to allow external setting of selection menu options.
- void `setSelectVSlice1Mode ()`
Framework use only. Slot to allow external setting of selection menu options.
- void `setSelectVSlice2Mode ()`
Framework use only. Slot to allow external setting of selection menu options.
- void `setSelectVSlice3Mode ()`
Framework use only. Slot to allow external setting of selection menu options.
- void `setSelectVSlice4Mode ()`
Framework use only. Slot to allow external setting of selection menu options.
- void `setSelectVSlice5Mode ()`
Framework use only. Slot to allow external setting of selection menu options.
- void `setSelectHSlice1Mode ()`
Framework use only. Slot to allow external setting of selection menu options.
- void `setSelectHSlice2Mode ()`
Framework use only. Slot to allow external setting of selection menu options.
- void `setSelectHSlice3Mode ()`
Framework use only. Slot to allow external setting of selection menu options.
- void `setSelectHSlice4Mode ()`
Framework use only. Slot to allow external setting of selection menu options.
- void `setSelectHSlice5Mode ()`
Framework use only. Slot to allow external setting of selection menu options.
- void `pauseClicked ()`
Framework use only. Slot to allow external setting of selection menu options.
- void `saveClicked ()`
Framework use only. Slot to allow external setting of selection menu options.
- void `targetClicked ()`
Framework use only. Slot to allow external setting of selection menu options.
- void `imageDisplayPropsDestroyed (QObject *)`
Framework use only. Slot to catch deletion of components (such as profile plots) that have been passed to the application for presentation.
- void `vSliceDisplayDestroyed (QObject *)`
Framework use only. Slot to catch deletion of components (such as profile plots) that have been passed to the application for presentation.
- void `hSliceDisplayDestroyed (QObject *)`
Framework use only. Slot to catch deletion of components (such as profile plots) that have been passed to the application for presentation.
- void `profileDisplayDestroyed (QObject *)`
Framework use only. Slot to catch deletion of components (such as profile plots) that have been passed to the application for presentation.

- void **recorderDestroyed** (QObject *)
Framework use only. Slot to catch deletion of components (such as profile plots) that have been passed to the application for presentation.
- void **showProfile** ()
Show the arbitrary line (profile) markup. --refer to - refer to [enableProfileSelection](#) property and #displayMarkups property for details.
- void **hideProfile** ()
Hide the arbitrary line (profile) markup but note that if its PV changes it will reshown unless DisplayMarkups has been set to off - refer to [enableProfileSelection](#) property and #displayMarkups property for details.
- void **showArea1** ()
Show the area1 markup - refer to [enableArea1Selection](#) property and #displayMarkups property for details.
- void **hideArea1** ()
Hide the area1 markup but note that if its PV changes it will reshown unless DisplayMarkups has been set to off - refer to [enableArea1Selection](#) property and #displayMarkups property for details.
- void **setDisplayMarkupsOn** ()
Set markup display to on to show all markups that change either due to user or PV activity, even if their setDisplay????Selection is off - refer to #displayMarkups property for details.
- void **setDisplayMarkupsOff** ()
Set markup display to off to stop PV controlled pvs from showing even if they change, unless their setDisplay????Selection is on - refer to #displayMarkups property for details.
- void **setManagedVisible** (bool v)

Signals

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

Public Member Functions

- **QEImage** (QWidget *parent=0)
- **QEImage** (const QString &variableName, QWidget *parent=0)
- **~QEImage** ()
Destructor.
- **selectOptions** **getSelectionOption** ()
- void **setBitDepth** (unsigned int bitDepthIn)
Access function for #bitDepth property - refer to #bitDepth property for details.
- unsigned int **getBitDepth** ()

- `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 ()`
Access function for `zoom` property - refer to `zoom` property for details.
- `void setRotation (imageProperties::rotationOptions rotationIn)`
Access function for `rotation` property - refer to `rotation` property for details.
- `imageProperties::rotationOptions getRotation ()`
Access function for `rotation` property - refer to `rotation` property for details.
- `void setHorizontalFlip (bool flipHozIn)`
Access function for `horizontalFlip` property - refer to `horizontalFlip` property for details.
- `bool getHorizontalFlip ()`
Access function for `horizontalFlip` property - refer to `horizontalFlip` property for details.
- `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 `initialHozScrollPos` property - refer to `initialHozScrollPos` property for details.
- `int getInitialHozScrollPos ()`
Access function for `initialHozScrollPos` property - refer to `initialHozScrollPos` property for details.
- `void setInitialVertScrollPos (int initialVertScrollPosIn)`
Access function for `initialVertScrollPos` property - refer to `initialVertScrollPos` property for details.
- `int getInitialVertScrollPos ()`
Access function for `initialVertScrollPos` property - refer to `initialVertScrollPos` property for details.
- `void setDisplayButtonBar (bool displayButtonBarIn)`
Access function for `displayButtonBar` property - refer to `displayButtonBar` property for details.
- `bool getDisplayButtonBar ()`
Access function for `displayButtonBar` property - refer to `displayButtonBar` property for details.
- `void setShowTime (bool pValue)`

- Access function for `showTime` property - refer to `showTime` property for details.
 - `bool getShowTime ()`
Access function for `showTime` property - refer to `showTime` property for details.
 - `void setUseFalseColour (bool pValue)`
Access function for `useFalseColour` property - refer to `useFalseColour` property for details.
 - `bool getUseFalseColour ()`
Access function for `useFalseColour` property - refer to `useFalseColour` property for details.
 - `void setVertSlice1MarkupColor (QColor pValue)`
Access function for `#vertSliceColor` property - refer to `#vertSliceColor` property for details.
 - `QColor getVertSlice1MarkupColor ()`
Access function for `#vertSliceColor` property - refer to `#vertSliceColor` property for details.
 - `void setVertSlice2MarkupColor (QColor pValue)`
Access function for `vertSlice2Color` property - refer to `vertSlice2Color` property for details.
 - `QColor getVertSlice2MarkupColor ()`
Access function for `vertSlice2Color` property - refer to `vertSlice2Color` property for details.
 - `void setVertSlice3MarkupColor (QColor pValue)`
Access function for `vertSlice3Color` property - refer to `vertSlice3Color` property for details.
 - `QColor getVertSlice3MarkupColor ()`
Access function for `vertSlice3Color` property - refer to `vertSlice3Color` property for details.
 - `void setVertSlice4MarkupColor (QColor pValue)`
Access function for `vertSlice4Color` property - refer to `vertSlice4Color` property for details.
 - `QColor getVertSlice4MarkupColor ()`
Access function for `vertSlice4Color` property - refer to `vertSlice4Color` property for details.
 - `void setVertSlice5MarkupColor (QColor pValue)`
Access function for `vertSlice5Color` property - refer to `vertSlice5Color` property for details.
 - `QColor getVertSlice5MarkupColor ()`
Access function for `vertSlice5Color` property - refer to `vertSlice5Color` property for details.
- `void setHozSlice1MarkupColor (QColor pValue)`
Access function for `#hozSliceColor` property - refer to `#hozSliceColor` property for details.
- `QColor getHozSlice1MarkupColor ()`
Access function for `#hozSliceColor` property - refer to `#hozSliceColor` property for details.
- `void setHozSlice2MarkupColor (QColor pValue)`

- `QColor getHozSlice2MarkupColor ()`
Access function for `hozSlice2Color` property - refer to `hozSlice2Color` property for details.
- `void setHozSlice3MarkupColor (QColor pValue)`
Access function for `hozSlice3Color` property - refer to `hozSlice3Color` property for details.
- `QColor getHozSlice3MarkupColor ()`
Access function for `hozSlice3Color` property - refer to `hozSlice3Color` property for details.
- `void setHozSlice4MarkupColor (QColor pValue)`
Access function for `hozSlice4Color` property - refer to `hozSlice4Color` property for details.
- `QColor getHozSlice4MarkupColor ()`
Access function for `hozSlice4Color` property - refer to `hozSlice4Color` property for details.
- `void setHozSlice5MarkupColor (QColor pValue)`
Access function for `hozSlice5Color` property - refer to `hozSlice5Color` property for details.
- `QColor getHozSlice5MarkupColor ()`
Access function for `hozSlice5Color` property - refer to `hozSlice5Color` property for details.
- `void setProfileMarkupColor (QColor pValue)`
Access function for `profileColor` property - refer to `profileColor` property for details.
- `QColor getProfileMarkupColor ()`
Access function for `profileColor` property - refer to `profileColor` property for details.
- `void setAreaMarkupColor (QColor pValue)`
Access function for `areaColor` property - refer to `areaColor` property for details.
- `QColor getAreaMarkupColor ()`
Access function for `areaColor` property - refer to `areaColor` property for details.
- `void setTargetMarkupColor (QColor pValue)`
Access function for `targetColor` property - refer to `targetColor` property for details.
- `QColor getTargetMarkupColor ()`
Access function for `targetColor` property - refer to `targetColor` property for details.
- `void setBeamMarkupColor (QColor pValue)`
Access function for `beamColor` property - refer to `beamColor` property for details.
- `QColor getBeamMarkupColor ()`
Access function for `beamColor` property - refer to `beamColor` property for details.
- `void setTimeMarkupColor (QColor pValue)`
Access function for `timeColor` property - refer to `timeColor` property for details.
- `QColor getTimeMarkupColor ()`
Access function for `timeColor` property - refer to `timeColor` property for details.
- `void setEllipseMarkupColor (QColor markupColor)`
Access function for `ellipseColor` property - refer to `ellipseColor` property for details.

- QColor **getEllipseMarkupColor ()**
Access function for *ellipseColor* property - refer to *ellipseColor* property for details.
- void **setDisplayCursorPixelInfo (bool displayCursorPixelInfo)**
Access function for *displayCursorPixelInfo* property - refer to *displayCursorPixelInfo* property for details.
- bool **getDisplayCursorPixelInfo ()**
Access function for *displayCursorPixelInfo* property - refer to *displayCursorPixelInfo* property for details.
- void **setContrastReversal (bool contrastReversalIn)**
Access function for *contrastReversal* property - refer to *contrastReversal* property for details.
- bool **getContrastReversal ()**
Access function for *contrastReversal* property - refer to *contrastReversal* property for details.
- void **setLog (bool log)**
Access function for *logBrightness* property - refer to *logBrightness* property for details.
- bool **getLog ()**
Access function for *logBrightness* property - refer to *logBrightness* property for details.
- void **setEnableVertSlice1Selection (bool enableVSliceSelection)**
Access function for *enableVertSlice1Selection* property - refer to *enableVertSlice1Selection* property for details.
- bool **getEnableVertSlice1Selection ()**
Access function for *enableVertSlice1Selection* property - refer to *enableVertSlice1Selection* property for details.
- void **setEnableVertSlice2Selection (bool enableVSliceSelection)**
Access function for *enableVertSlice2Selection* property - refer to *enableVertSlice2Selection* property for details.
- bool **getEnableVertSlice2Selection ()**
Access function for *enableVertSlice2Selection* property - refer to *enableVertSlice2Selection* property for details.
- void **setEnableVertSlice3Selection (bool enableVSliceSelection)**
Access function for *enableVertSlice3Selection* property - refer to *enableVertSlice3Selection* property for details.
- bool **getEnableVertSlice3Selection ()**
Access function for *enableVertSlice3Selection* property - refer to *enableVertSlice3Selection* property for details.
- void **setEnableVertSlice4Selection (bool enableVSliceSelection)**
Access function for *enableVertSlice4Selection* property - refer to *enableVertSlice4Selection* property for details.
- bool **getEnableVertSlice4Selection ()**
Access function for *enableVertSlice4Selection* property - refer to *enableVertSlice4Selection* property for details.
- void **setEnableVertSlice5Selection (bool enableVSliceSelection)**
Access function for *enableVertSlice5Selection* property - refer to *enableVertSlice5Selection* property for details.
- bool **getEnableVertSlice5Selection ()**

- void `setEnableHozSlice1Selection` (bool enableHSliceSelection)
Access function for `enableHozSlice1Selection` property - refer to `enableHozSlice1Selection` property for details.
- bool `getEnableHozSlice1Selection` ()
Access function for `enableHozSlice1Selection` property - refer to `enableHozSlice1Selection` property for details.
- void `setEnableHozSlice2Selection` (bool enableHSliceSelection)
Access function for `enableHozSlice2Selection` property - refer to `enableHozSlice2Selection` property for details.
- bool `getEnableHozSlice2Selection` ()
Access function for `enableHozSlice2Selection` property - refer to `enableHozSlice2Selection` property for details.
- void `setEnableHozSlice3Selection` (bool enableHSliceSelection)
Access function for `enableHozSlice3Selection` property - refer to `enableHozSlice3Selection` property for details.
- bool `getEnableHozSlice3Selection` ()
Access function for `enableHozSlice3Selection` property - refer to `enableHozSlice3Selection` property for details.
- void `setEnableHozSlice4Selection` (bool enableHSliceSelection)
Access function for `enableHozSlice4Selection` property - refer to `enableHozSlice4Selection` property for details.
- bool `getEnableHozSlice4Selection` ()
Access function for `enableHozSlice4Selection` property - refer to `enableHozSlice4Selection` property for details.
- void `setEnableHozSlice5Selection` (bool enableHSliceSelection)
Access function for `enableHozSlice5Selection` property - refer to `enableHozSlice5Selection` property for details.
- bool `getEnableHozSlice5Selection` ()
Access function for `enableHozSlice5Selection` property - refer to `enableHozSlice5Selection` property for details.
- void `setEnableArea1Selection` (bool enableAreaSelectionIn)
Access function for `enableArea1Selection` property - refer to `enableArea1Selection` property for details.
- bool `getEnableArea1Selection` ()
Access function for `enableArea1Selection` property - refer to `enableArea1Selection` property for details.
- void `setEnableArea2Selection` (bool enableAreaSelectionIn)
Access function for `enableArea2Selection` property - refer to `enableArea2Selection` property for details.
- bool `getEnableArea2Selection` ()
Access function for `enableArea2Selection` property - refer to `enableArea2Selection` property for details.
- void `setEnableArea3Selection` (bool enableAreaSelectionIn)
Access function for `enableArea3Selection` property - refer to `enableArea3Selection` property for details.

- `bool getEnableArea3Selection ()`
Access function for `enableArea3Selection` property - refer to `enableArea3Selection` property for details.
- `void setEnableArea4Selection (bool enableAreaSelectionIn)`
Access function for `enableArea4Selection` property - refer to `enableArea4Selection` property for details.
- `bool getEnableArea4Selection ()`
Access function for `enableArea4Selection` property - refer to `enableArea4Selection` property for details.
- `void setEnableProfileSelection (bool enableProfileSelectionIn)`
Access function for `enableProfileSelection` property - refer to `enableProfileSelection` property for details.
- `bool getEnableProfileSelection ()`
Access function for `enableProfileSelection` property - refer to `enableProfileSelection` property for details.
- `void setEnableTargetSelection (bool enableTargetSelectionIn)`
Access function for `enableTargetSelection` property - refer to `enableTargetSelection` property for details.
- `bool getEnableTargetSelection ()`
Access function for `enableTargetSelection` property - refer to `enableTargetSelection` property for details.
- `void setEnableBeamSelection (bool enableBeamSelectionIn)`
Access function for `enableBeamSelection` property - refer to `enableBeamSelection` property for details.
- `bool getEnableBeamSelection ()`
Access function for `enableBeamSelection` property - refer to `enableBeamSelection` property for details.
- `void setEnableImageDisplayProperties (bool enableImageDisplayPropertiesIn)`
Access function for `enableImageDisplayProperties` property - refer to `enableImageDisplayProperties` property for details.
- `bool getEnableImageDisplayProperties ()`
Access function for `enableImageDisplayProperties` property - refer to `enableImageDisplayProperties` property for details.
- `void setEnableRecording (bool enableRecordingIn)`
Access function for `enableRecording` property - refer to `enableRecording` property for details.
- `bool getEnableRecording ()`
Access function for `enableRecording` property - refer to `enableRecording` property for details.
- `void setAutoBrightnessContrast (bool autoBrightnessContrastIn)`
Access function for `autoBrightnessContrast` property - refer to `autoBrightnessContrast` property for details.
- `bool getAutoBrightnessContrast ()`
Access function for `autoBrightnessContrast` property - refer to `autoBrightnessContrast` property for details.
- `void setExternalControls (bool externalControlsIn)`

- Access function for `externalControls` property - refer to `externalControls` property for details.
 - `bool getExternalControls ()`
Access function for `externalControls` property - refer to `externalControls` property for details.
 - `void setFullContextMenu (bool fullContextMenuIn)`
Access function for `#fullContextMenu` property - refer to `#fullContextMenu` property for details.
 - `bool getFullContextMenu ()`
Access function for `#fullContextMenu` property - refer to `#fullContextMenu` property for details.
 - `void setEnableProfilePresentation (bool enableProfilePresentationIn)`
Access function for `#enableProfilePresentation` property - refer to `#enableProfilePresentation` property for details.
 - `bool getEnableProfilePresentation ()`
Access function for `#enableProfilePresentation` property - refer to `#enableProfilePresentation` property for details.
 - `void setEnableHozSlicePresentation (bool enableHozSlicePresentationIn)`
Access function for `#enableHozSlicePresentation` property - refer to `#enableHozSlicePresentation` property for details.
 - `bool getEnableHozSlicePresentation ()`
Access function for `#enableHozSlicePresentation` property - refer to `#enableHozSlicePresentation` property for details.
 - `void setEnableVertSlicePresentation (bool enableVertSlicePresentationIn)`
Access function for `#enableVertSlicePresentation` property - refer to `#enableVertSlicePresentation` property for details.
 - `bool getEnableVertSlicePresentation ()`
Access function for `#enableVertSlicePresentation` property - refer to `#enableVertSlicePresentation` property for details.
 - `void setDisplayVertSlice1Selection (bool displayVSliceSelection)`
Access function for `#displayVertSlice1Selection` property - refer to `#displayVertSlice1Selection` property for details.
 - `bool getDisplayVertSlice1Selection ()`
Access function for `#displayVertSlice1Selection` property - refer to `#displayVertSlice1Selection` property for details.
 - `void setDisplayVertSlice2Selection (bool displayVSliceSelection)`
Access function for `#displayVertSlice2Selection` property - refer to `#displayVertSlice2Selection` property for details.
 - `bool getDisplayVertSlice2Selection ()`
Access function for `#displayVertSlice2Selection` property - refer to `#displayVertSlice2Selection` property for details.
 - `void setDisplayVertSlice3Selection (bool displayVSliceSelection)`
Access function for `#displayVertSlice3Selection` property - refer to `#displayVertSlice3Selection` property for details.
 - `bool getDisplayVertSlice3Selection ()`
Access function for `#displayVertSlice3Selection` property - refer to `#displayVertSlice3Selection` property for details.

- void `setDisplayVertSlice4Selection` (bool displayVSliceSelection)
Access function for #displayVertSlice4Selection property - refer to #displayVertSlice4Selection property for details.
- bool `getDisplayVertSlice4Selection` ()
Access function for #displayVertSlice4Selection property - refer to #displayVertSlice4Selection property for details.
- void `setDisplayVertSlice5Selection` (bool displayVSliceSelection)
Access function for #displayVertSlice5Selection property - refer to #displayVertSlice5Selection property for details.
- bool `getDisplayVertSlice5Selection` ()
Access function for #displayVertSlice5Selection property - refer to #displayVertSlice5Selection property for details.
- void `setDisplayHozSlice1Selection` (bool displayHSliceSelection)
Access function for `displayHozSlice1Selection` property - refer to `displayHozSlice1Selection` property for details.
- bool `getDisplayHozSlice1Selection` ()
Access function for `displayHozSlice1Selection` property - refer to `displayHozSlice1Selection` property for details.
- void `setDisplayHozSlice2Selection` (bool displayHSliceSelection)
Access function for `displayHozSlice2Selection` property - refer to `displayHozSlice2Selection` property for details.
- bool `getDisplayHozSlice2Selection` ()
Access function for `displayHozSlice2Selection` property - refer to `displayHozSlice2Selection` property for details.
- void `setDisplayHozSlice3Selection` (bool displayHSliceSelection)
Access function for `displayHozSlice3Selection` property - refer to `displayHozSlice3Selection` property for details.
- bool `getDisplayHozSlice3Selection` ()
Access function for `displayHozSlice3Selection` property - refer to `displayHozSlice3Selection` property for details.
- void `setDisplayHozSlice4Selection` (bool displayHSliceSelection)
Access function for `displayHozSlice4Selection` property - refer to `displayHozSlice4Selection` property for details.
- bool `getDisplayHozSlice4Selection` ()
Access function for `displayHozSlice4Selection` property - refer to `displayHozSlice4Selection` property for details.
- void `setDisplayHozSlice5Selection` (bool displayHSliceSelection)
Access function for `displayHozSlice5Selection` property - refer to `displayHozSlice5Selection` property for details.
- bool `getDisplayHozSlice5Selection` ()
Access function for `displayHozSlice5Selection` property - refer to `displayHozSlice5Selection` property for details.
- void `setDisplayArea1Selection` (bool displayAreaSelection)
Access function for `displayArea1Selection` property - refer to `displayArea1Selection` property for details.
- bool `getDisplayArea1Selection` ()

- Access function for `displayArea1Selection` property - refer to `displayArea1Selection` property for details.
 - void `setDisplayArea2Selection` (bool displayAreaSelection)
Access function for `displayArea2Selection` property - refer to `displayArea2Selection` property for details.
 - bool `getDisplayArea2Selection` ()
Access function for `displayArea2Selection` property - refer to `displayArea2Selection` property for details.
 - void `setDisplayArea3Selection` (bool displayAreaSelection)
Access function for `displayArea3Selection` property - refer to `displayArea3Selection` property for details.
 - bool `getDisplayArea3Selection` ()
Access function for `displayArea3Selection` property - refer to `displayArea3Selection` property for details.
 - void `setDisplayArea4Selection` (bool displayAreaSelection)
Access function for `displayArea4Selection` property - refer to `displayArea4Selection` property for details.
 - bool `getDisplayArea4Selection` ()
Access function for `displayArea4Selection` property - refer to `displayArea4Selection` property for details.
- void `setDisplayProfileSelection` (bool displayProfileSelection)
Access function for `displayProfileSelection` property - refer to `displayProfileSelection` property for details.
- bool `getDisplayProfileSelection` ()
Access function for `displayProfileSelection` property - refer to `displayProfileSelection` property for details.
- void `setDisplayTargetSelection` (bool displayTargetSelection)
Access function for `displayTargetSelection` property - refer to `displayTargetSelection` property for details.
- bool `getDisplayTargetSelection` ()
Access function for `displayTargetSelection` property - refer to `displayTargetSelection` property for details.
- void `setDisplayBeamSelection` (bool displayBeamSelection)
Access function for `displayBeamSelection` property - refer to `displayBeamSelection` property for details.
- bool `getDisplayBeamSelection` ()
Access function for `displayBeamSelection` property - refer to `displayBeamSelection` property for details.
- void `setDisplayEllipse` (bool displayEllipse)
Access function for `displayEllipse` property - refer to `displayEllipse` property for details.
- bool `getDisplayEllipse` ()
Access function for `displayEllipse` property - refer to `displayEllipse` property for details.
- `ellipseVariableDefinitions` `getEllipseVariableDefinition` ()
Access function for `ellipseVariableDefinition` property - refer to `ellipseVariableDefinition` property for details.
- void `setEllipseVariableDefinition` (`ellipseVariableDefinitions` def)

- Access function for `ellipseVariableDefinition` property - refer to `ellipseVariableDefinition` property for details.*
- void `setDisplayMarkups` (bool displayMarkupsIn)
Access function for `#displayMarkups` property - refer to `#displayMarkups` property for details.
 - bool `getDisplayMarkups` ()
Access function for `#displayMarkups` property - refer to `#displayMarkups` property for details.
 - void `setName` (QString nameIn)
Access function for `name` property - refer to `#name` property for details.
 - QString `getName` ()
Access function for `name` property - refer to `#name` property for details.
 - void `setProgram1` (QString program)
Access function for `program1` property - refer to `program1` property for details.
 - QString `getProgram1` ()
Access function for `program1` property - refer to `program1` property for details.
 - void `setProgram2` (QString program)
Access function for `program2` property - refer to `program2` property for details.
 - QString `getProgram2` ()
Access function for `program2` property - refer to `program2` property for details.
 - void `setArguments1` (QStringList arguments)
Access function for `arguments1` property - refer to `arguments1` property for details.
 - QStringList `getArguments1` ()
Access function for `arguments1` property - refer to `arguments1` property for details.
 - void `setArguments2` (QStringList arguments)
Access function for `arguments2` property - refer to `arguments2` property for details.
 - QStringList `getArguments2` ()
Access function for `arguments2` property - refer to `arguments2` property for details.
 - void `setProgramStartupOption1` (applicationLauncher::programStartupOptions programStartupOption)
Access function for `programStartupOption1` property - refer to `programStartupOption1` property for details.
 - applicationLauncher::programStartupOptions `getProgramStartupOption1` ()
Access function for `programStartupOption1` property - refer to `programStartupOption1` property for details.
 - void `setProgramStartupOption2` (applicationLauncher::programStartupOptions programStartupOption)
Access function for `programStartupOption2` property - refer to `programStartupOption2` property for details.
 - applicationLauncher::programStartupOptions `getProgramStartupOption2` ()
Access function for `programStartupOption2` property - refer to `programStartupOption2` property for details.
 - QString `getHozSlice1Legend` ()
Access function for `hozSlice1Legend` property - refer to `hozSlice1Legend` property for details.

- void `setHozSlice1Legend` (QString legend)
Access function for `hozSlice1Legend` property - refer to `hozSlice1Legend` property for details.
- QString `getHozSlice2Legend` ()
Access function for `hozSlice2Legend` property - refer to `hozSlice2Legend` property for details.
- void `setHozSlice2Legend` (QString legend)
Access function for `hozSlice2Legend` property - refer to `hozSlice2Legend` property for details.
- QString `getHozSlice3Legend` ()
Access function for `hozSlice3Legend` property - refer to `hozSlice3Legend` property for details.
- void `setHozSlice3Legend` (QString legend)
Access function for `hozSlice3Legend` property - refer to `hozSlice3Legend` property for details.
- QString `getHozSlice4Legend` ()
Access function for `hozSlice4Legend` property - refer to `hozSlice4Legend` property for details.
- void `setHozSlice4Legend` (QString legend)
Access function for `hozSlice4Legend` property - refer to `hozSlice4Legend` property for details.
- QString `getHozSlice5Legend` ()
Access function for `hozSlice5Legend` property - refer to `hozSlice5Legend` property for details.
- void `setHozSlice5Legend` (QString legend)
Access function for `hozSlice5Legend` property - refer to `hozSlice5Legend` property for details.
- QString `getVertSlice1Legend` ()
Access function for `vertSlice1Legend` property - refer to `vertSlice1Legend` property for details.
- void `setVertSlice1Legend` (QString legend)
Access function for `vertSlice1Legend` property - refer to `vertSlice1Legend` property for details.
- QString `getVertSlice2Legend` ()
Access function for `vertSlice2Legend` property - refer to `vertSlice2Legend` property for details.
- void `setVertSlice2Legend` (QString legend)
Access function for `vertSlice2Legend` property - refer to `vertSlice2Legend` property for details.
- QString `getVertSlice3Legend` ()
Access function for `vertSlice3Legend` property - refer to `vertSlice3Legend` property for details.
- void `setVertSlice3Legend` (QString legend)
Access function for `vertSlice3Legend` property - refer to `vertSlice3Legend` property for details.
- QString `getVertSlice4Legend` ()

- Access function for `vertSlice4Legend` property - refer to `vertSlice4Legend` property for details.
 - void `setVertSlice4Legend` (QString legend)
Access function for `vertSlice4Legend` property - refer to `vertSlice4Legend` property for details.
 - QString `getVertSlice5Legend` ()
Access function for `vertSlice5Legend` property - refer to `vertSlice5Legend` property for details.
 - void `setVertSlice5Legend` (QString legend)
Access function for `vertSlice5Legend` property - refer to `vertSlice5Legend` property for details.
 - QString `getprofileLegend` ()
Access function for `profileLegend` property - refer to `profileLegend` property for details.
 - void `setProfileLegend` (QString legend)
Access function for `profileLegend` property - refer to `profileLegend` property for details.
- QString `getAreaSelection1Legend` ()
Access function for `areaSelection1Legend` property - refer to `areaSelection1Legend` property for details.
- void `setAreaSelection1Legend` (QString legend)
Access function for `areaSelection1Legend` property - refer to `areaSelection1Legend` property for details.
- QString `getAreaSelection2Legend` ()
Access function for `areaSelection2Legend` property - refer to `areaSelection2Legend` property for details.
- void `setAreaSelection2Legend` (QString legend)
Access function for `areaSelection2Legend` property - refer to `areaSelection2Legend` property for details.
- QString `getAreaSelection3Legend` ()
Access function for `areaSelection3Legend` property - refer to `areaSelection3Legend` property for details.
- void `setAreaSelection3Legend` (QString legend)
Access function for `areaSelection3Legend` property - refer to `areaSelection3Legend` property for details.
- QString `getAreaSelection4Legend` ()
Access function for `areaSelection4Legend` property - refer to `areaSelection4Legend` property for details.
- void `setAreaSelection4Legend` (QString legend)
Access function for `areaSelection4Legend` property - refer to `areaSelection4Legend` property for details.
- QString `getTargetLegend` ()
Access function for `targetLegend` property - refer to `targetLegend` property for details.
- void `setTargetLegend` (QString legend)
Access function for `targetLegend` property - refer to `targetLegend` property for details.
- QString `getBeamLegend` ()
Access function for `beamLegend` property - refer to `beamLegend` property for details.
- void `setBeamLegend` (QString legend)

- `QString getEllipseLegend ()`
Access function for `ellipseLegend` property - refer to `ellipseLegend` property for details.
 - `void setEllipseLegend (QString legend)`
Access function for `ellipseLegend` property - refer to `ellipseLegend` property for details.
 - `bool getFullScreen ()`
Access function for `#fullScreen` property - refer to `#fullScreen` property for details.
 - `void setFullScreen (bool fullScreenIn)`
Access function for `#fullScreen` property - refer to `#fullScreen` property for details.
 - `void setSubstitutedUrl (QString urlIn)`
Access function for `URL` property - refer to `URL` property for details.
 - `QString getSubstitutedUrl ()`
Access function for `URL` property - refer to `URL` property for details.
 - `void setVariableNameSubstitutionsProperty (QString variableNameSubstitutions)`
- Property access function for `variableSubstitutions` property. This has special behaviour to work well within designer.*
- `QString getVariableNameSubstitutionsProperty ()`
Property access function for `variableSubstitutions` property. This has special behaviour to work well within designer.
 - `UserLevels getUserLevelVisibilityProperty ()`
Access function for `userLevelVisibility` property - refer to `userLevelVisibility` property for details.
 - `void setUserLevelVisibilityProperty (UserLevels level)`
Access function for `userLevelVisibility` property - refer to `userLevelVisibility` property for details.
 - `UserLevels getUserLevelEnabledProperty ()`
Access function for `userLevelEnabled` property - refer to `userLevelEnabled` property for details.
 - `void setUserLevelEnabledProperty (UserLevels level)`
Access function for `userLevelEnabled` property - refer to `userLevelEnabled` property for details.
 - `DisplayAlarmStateOptions getDisplayAlarmStateOptionProperty ()`
Access function for `displayAlarmStateOption` property - refer to `displayAlarmStateOption` property for details.
 - `void setDisplayAlarmStateOptionProperty (DisplayAlarmStateOptions option)`
Access function for `displayAlarmStateOption` property - refer to `displayAlarmStateOption` property for details.
 - `void setFormatOptionProperty (FormatOptions formatOption)`
Access function for `formatOption` property - refer to `formatOption` property for details.
 - `FormatOptions getFormatOptionProperty ()`
Access function for `formatOption` property - refer to `formatOption` property for details.
 - `void setBitDepthProperty (unsigned int bitDepth)`
Access function for `#bitDepth` property - refer to `#bitDepth` property for details.
 - `unsigned int getBitDepthProperty ()`

- Access function for `#bitDepth` property - refer to `#bitDepth` property for details.
- [EllipseVariableDefinitions getEllipseVariableDefinitionProperty \(\)](#)
 - Access function for `#EllipseVariableDefinition` property - refer to `#EllipseVariableDefinition` property for details.
- [void setEllipseVariableDefinitionProperty \(EllipseVariableDefinitions variableUsage\)](#)

- Access function for `EllipseVariableDefinitions` property - refer to `EllipseVariableDefinitions` property for details.
- [TargetOptions getTargetOptionProperty \(\)](#)
 - Access function for `targetOption` property - refer to `targetOption` property for details.
- [void setTargetOptionProperty \(TargetOptions option\)](#)
 - Access function for `targetOption` property - refer to `targetOption` property for details.
- [TargetOptions getBeamOptionProperty \(\)](#)
 - Access function for `beamOption` property - refer to `beamOption` property for details.
- [void setBeamOptionProperty \(TargetOptions option\)](#)
 - Access function for `beamOption` property - refer to `beamOption` property for details.
- [void setResizeOptionProperty \(ResizeOptions resizeMode\)](#)
 - Access function for `#resizeOption` property - refer to `#resizeOption` property for details.
- [ResizeOptions getResizeOptionProperty \(\)](#)
 - Access function for `#resizeOption` property - refer to `#resizeOption` property for details.
- [void setRotationProperty \(RotationOptions rotation\)](#)
 - Access function for `rotation` property - refer to `rotation` property for details.
- [RotationOptions getRotationProperty \(\)](#)
 - Access function for `rotation` property - refer to `rotation` property for details.
- [void setProgramStartupOptionProperty1 \(ProgramStartupOptionNames programStartupOption\)](#)
 - Access function for `#ProgramStartupOptionNames1` property - refer to `#ProgramStartupOptionNames1` property for details.
- [ProgramStartupOptionNames getProgramStartupOptionProperty1 \(\)](#)
 - Access function for `#ProgramStartupOptionNames1` property - refer to `#ProgramStartupOptionNames1` property for details.
- [void setProgramStartupOptionProperty2 \(ProgramStartupOptionNames programStartupOption\)](#)
 - Access function for `#ProgramStartupOptionNames2` property - refer to `#ProgramStartupOptionNames2` property for details.
- [ProgramStartupOptionNames getProgramStartupOptionProperty2 \(\)](#)
 - Access function for `#ProgramStartupOptionNames2` property - refer to `#ProgramStartupOptionNames2` property for details.

Protected Types

- enum **variableIndexes** {
IMAGE_VARIABLE, FORMAT_VARIABLE, BIT_DEPTH_VARIABLE, WIDTH_VARIABLE,

```

HEIGHT_VARIABLE, NUM_DIMENSIONS_VARIABLE, DIMENSION_0_VARIABLE,
DIMENSION_1_VARIABLE,
DIMENSION_2_VARIABLE, ROI1_X_VARIABLE, ROI1_Y_VARIABLE, ROI1_-
W_VARIABLE,
ROI1_H_VARIABLE, ROI2_X_VARIABLE, ROI2_Y_VARIABLE, ROI2_W_VARIABLE,
ROI2_H_VARIABLE, ROI3_X_VARIABLE, ROI3_Y_VARIABLE, ROI3_W_VARIABLE,
ROI3_H_VARIABLE, ROI4_X_VARIABLE, ROI4_Y_VARIABLE, ROI4_W_VARIABLE,
ROI4_H_VARIABLE, TARGET_X_VARIABLE, TARGET_Y_VARIABLE, BEAM_-
X_VARIABLE,
BEAM_Y_VARIABLE, TARGET_TRIGGER_VARIABLE, CLIPPING_ONOFF_-
VARIABLE, CLIPPING_LOW_VARIABLE,
CLIPPING_HIGH_VARIABLE, PROFILE_H1_VARIABLE, PROFILE_H1_THICKNESS_-_
VARIABLE, PROFILE_H2_VARIABLE,
PROFILE_H2_THICKNESS_VARIABLE, PROFILE_H3_VARIABLE, PROFILE_-_
H3_THICKNESS_VARIABLE, PROFILE_H4_VARIABLE,
PROFILE_H4_THICKNESS_VARIABLE, PROFILE_H5_VARIABLE, PROFILE_-_
H5_THICKNESS_VARIABLE, PROFILE_V1_VARIABLE,
PROFILE_V1_THICKNESS_VARIABLE, PROFILE_V2_VARIABLE, PROFILE_-_
V2_THICKNESS_VARIABLE, PROFILE_V3_VARIABLE,
PROFILE_V3_THICKNESS_VARIABLE, PROFILE_V4_VARIABLE, PROFILE_-_
V4_THICKNESS_VARIABLE, PROFILE_V5_VARIABLE,
PROFILE_V5_THICKNESS_VARIABLE, LINE_PROFILE_X1_VARIABLE, LINE_-
PROFILE_Y1_VARIABLE, LINE_PROFILE_X2_VARIABLE,
LINE_PROFILE_Y2_VARIABLE, LINE_PROFILE_THICKNESS_VARIABLE, PROFILE_-_
H_ARRAY, PROFILE_V_ARRAY,
PROFILE_LINE_ARRAY, ELLIPSE_X_VARIABLE, ELLIPSE_Y_VARIABLE, ELLIPSE_-_
W_VARIABLE,
ELLIPSE_H_VARIABLE, QEIMAGE_NUM_VARIABLES }

```

Protected Member Functions

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

Protected Attributes

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

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

Properties

- QString **imageVariable**
- QString **formatVariable**
- QString **bitDepthVariable**
- QString **widthVariable**
- QString **heightVariable**
- QString **dimensionsVariable**
- QString **dimension1Variable**
- QString **dimension2Variable**
- QString **dimension3Variable**
- QString **regionOfInterest1XVariable**
- QString **regionOfInterest1YVariable**
- QString **regionOfInterest1WVariable**
- QString **regionOfInterest1HVariable**
- QString **regionOfInterest2XVariable**
- QString **regionOfInterest2YVariable**
- QString **regionOfInterest2WVariable**
- QString **regionOfInterest2HVariable**
- QString **regionOfInterest3XVariable**
- QString **regionOfInterest3YVariable**
- QString **regionOfInterest3WVariable**
- QString **regionOfInterest3HVariable**
- QString **regionOfInterest4XVariable**
- QString **regionOfInterest4YVariable**
- QString **regionOfInterest4WVariable**
- QString **regionOfInterest4HVariable**
- QString **targetXVariable**
- QString **targetYVariable**
- QString **beamXVariable**
- QString **beamYVariable**
- QString **targetTriggerVariable**
- QString **clippingOnOffVariable**
- QString **clippingLowVariable**

- QString `clippingHighVariable`
- QString `profileHozVariable`
- QString `profileHoz1Variable`
- QString `profileHozThicknessVariable`
- QString `profileHoz1ThicknessVariable`
- QString `profileHoz2Variable`
- QString `profileHoz2ThicknessVariable`
- QString `profileHoz3Variable`
- QString `profileHoz3ThicknessVariable`
- QString `profileHoz4Variable`
- QString `profileHoz4ThicknessVariable`
- QString `profileHoz5Variable`
- QString `profileHoz5ThicknessVariable`
- QString `profileVertVariable`
- QString `profileVert1Variable`
- QString `profileVertThicknessVariable`
- QString `profileVert1ThicknessVariable`
- QString `profileVert2Variable`
- QString `profileVert2ThicknessVariable`
- QString `profileVert3Variable`
- QString `profileVert3ThicknessVariable`
- QString `profileVert4Variable`
- QString `profileVert4ThicknessVariable`
- QString `profileVert5Variable`
- QString `profileVert5ThicknessVariable`
- QString `lineProfileX1Variable`
- QString `lineProfileY1Variable`
- QString `lineProfileX2Variable`
- QString `lineProfileY2Variable`
- QString `lineProfileThicknessVariable`
- QString `profileHozArrayVariable`
- QString `profileVertArrayVariable`
- QString `lineProfileArrayVariable`
- QString `ellipseXVariable`
- QString `ellipseYVariable`
- QString `ellipseWVariable`
- QString `ellipseHVariable`
- QString `variableSubstitutions`
- bool `variableAsToolTip`
- bool `allowDrop`
- bool `visible`
- unsigned `int`
- QString `styleSheet`
- QString `defaultStyle`
- QString `userLevelUserStyle`
- QString `userLevelScientistStyle`

- `QString userLevelEngineerStyle`
- `UserLevels userLevelVisibility`
- `UserLevels userLevelEnabled`
- `bool displayAlarmState`
- `DisplayAlarmStateOptions displayAlarmStateOption`
- `FormatOptions formatOption`
- `bool enableVertSliceSelection`
- `bool enableVertSlice1Selection`
- `bool enableVertSlice2Selection`
- `bool enableVertSlice3Selection`
- `bool enableVertSlice4Selection`
- `bool enableVertSlice5Selection`
- `bool enableHozSliceSelection`
- `bool enableHozSlice1Selection`
- `bool enableHozSlice2Selection`
- `bool enableHozSlice3Selection`
- `bool enableHozSlice4Selection`
- `bool enableHozSlice5Selection`
- `bool enableProfileSelection`
- `bool enableArea1Selection`
- `bool enableArea2Selection`
- `bool enableArea3Selection`
- `bool enableArea4Selection`
- `bool enableTargetSelection`
- `bool enableBeamSelection`
- `QString hozSliceLegend`
- `QString hozSlice1Legend`

Name of horizontal slice 1 markup.
- `QString hozSlice2Legend`

Name of horizontal slice 2 markup.
- `QString hozSlice3Legend`

Name of horizontal slice 3 markup.
- `QString hozSlice4Legend`

Name of horizontal slice 4 markup.
- `QString hozSlice5Legend`

Name of horizontal slice 5 markup.
- `QString vertSliceLegend`
- `QString vertSlice1Legend`

Name of vertical slice 1 markup.
- `QString vertSlice2Legend`

Name of vertical slice 2 markup.
- `QString vertSlice3Legend`

Name of vertical slice 3 markup.
- `QString vertSlice4Legend`

Name of vertical slice 4 markup.

- **QString vertSlice5Legend**
Name of vertical slice 5 markup.
- **QString profileLegend**
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 displayHozSlice1Selection**
- **bool displayHozSlice2Selection**
- **bool displayHozSlice3Selection**
- **bool displayHozSlice4Selection**
- **bool displayHozSlice5Selection**
- **bool displayProfileSelection**
- **bool displayArea1Selection**
- **bool displayArea2Selection**
- **bool displayArea3Selection**
- **bool displayArea4Selection**
- **bool displayTargetSelection**
- **bool displayBeamSelection**
- **bool displayEllipse**
- **EllipseVariableDefinitions ellipseVariableDefinition**
Definition of how ellipse variables are to be used.
- **TargetOptions targetOption**
Definition of target markup options.
- **TargetOptions beamOption**
Definition of beam markup options.
- **bool displayCursorPixelInfo**
- **bool contrastReversal**
- **bool logBrightness**
- **bool showTime**
- **bool useFalseColour**

- QColor **vertSliceColor**
- QColor **vertSlice1Color**
- QColor **vertSlice2Color**
- QColor **vertSlice3Color**
- QColor **vertSlice4Color**
- QColor **vertSlice5Color**
- QColor **hozSliceColor**
- QColor **hozSlice1Color**
- QColor **hozSlice2Color**
- QColor **hozSlice3Color**
- QColor **hozSlice4Color**
- QColor **hozSlice5Color**
- QColor **profileColor**
- QColor **areaColor**
- QColor **beamColor**
- QColor **targetColor**
- QColor **timeColor**
- QColor **ellipseColor**
- **ResizeOptions** **resizeOption**
- **RotationOptions** **rotation**
- bool **verticalFlip**
- bool **horizontalFlip**
- int **initialHosScrollPos**
- bool **enableImageDisplayProperties**
 - If true, the local Image Display Properties controls are displayed.*
- bool **enableRecording**
 - If true, the recording controls are displayed.*
- bool **autoBrightnessContrast**
- bool **externalControls**
- bool **briefInfoArea**
- QString **program1**
- QStringList **arguments1**
- **ProgramStartupOptionNames** **programStartupOption1**
- QString **program2**
- QStringList **arguments2**
- **ProgramStartupOptionNames** **programStartupOption2**
- QString **URL**

9.64.1 Detailed Description

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

9.64.2 Member Enumeration Documentation

9.64.2.1 enum QEImage::DisplayAlarmStateOptions

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

Enumerator:

Never Refer to DISPLAY_ALARM_STATE_NEVER for details.

Always Refer to DISPLAY_ALARM_STATE_ALWAYS for details.

WhenInAlarm Refer to DISPLAY_ALARM_STATE_WHEN_IN_ALARM for details.

9.64.2.2 enum QEImage::EllipseVariableDefinitions

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

Enumerator:

BoundingRectangle Refer to BOUNDING_RECTANGLE for details.

CenterAndSize Refer to CENTRE_AND_SIZE for details.

9.64.2.3 enum QEImage::ellipseVariableDefinitions

Options for the use of ellipse markup variables.

Enumerator:

BOUNDING_RECTANGLE Variables define bounding rectangle of ellipse.

9.64.2.4 enum QEImage::FormatOptions

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

Enumerator:

Mono Grey scale.

Bayer Colour (Bayer Red Green)

BayerGB Colour (Bayer Green Blue)

BayerBG Colour (Bayer Blue Green)

BayerGR Colour (Bayer Green Red)

BayerRG Colour (Bayer Red Green)

rgb1 Colour (24 bit RGB)

rgb2 Colour (??? bit RGB)

rgb3 Colour (??? bit RGB)

yuv444 Colour (???)

yuv422 Colour (???)

9.64.2.5 enum QEImage::ProgramStartupOptionNames

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

Enumerator:

None Just run the program.

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

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

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

9.64.2.6 enum QEImage::ResizeOptions

User friendly enumerations for #resizeOption property

Enumerator:

Zoom Zoom to selected percentage.

Fit Zoom to fit the current window size.

9.64.2.7 enum QEImage::resizeOptions

Image resize options

Enumerator:

RESIZE_OPTION_ZOOM Zoom to selected percentage.

RESIZE_OPTION_FIT Zoom to fit the current window size.

9.64.2.8 enum QEImage::RotationOptions

User friendly enumerations for [rotation](#) property

Enumerator:

NoRotation No image rotation.

Rotate90Right Rotate image 90 degrees clockwise.

Rotate90Left Rotate image 90 degrees anticlockwise.

Rotate180 Rotate image 180 degrees.

9.64.2.9 enum QEImage::selectOptions

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

Enumerator:

SO_NONE Do nothing.

SO_PANNING User is panning.

SO_VSLICE1 Select the vertical slice 1 point.

SO_VSLICE2 Select the vertical slice 2 point.

SO_VSLICE3 Select the vertical slice 3 point.

SO_VSLICE4 Select the vertical slice 4 point.

SO_VSLICE5 Select the vertical slice 5 point.

SO_HSLICE1 Select the horizontal slice 1 point.

SO_HSLICE2 Select the horizontal slice 2 point.

SO_HSLICE3 Select the horizontal slice 3 point.

SO_HSLICE4 Select the horizontal slice 4 point.

SO_HSLICE5 Select the horizontal slice 5 point.

SO_AREA4 User is selecting an area (for region of interest)

SO_PROFILE Select an arbitrary line across the image (to determine a profile)

SO_TARGET Mark the target point.

SO_BEAM Mark the current beam location.

9.64.2.10 enum QEImage::TargetOptions

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

Enumerator:

DottedFullCrosshair Refer to CROSSHAIR1 for details.

SolidSmallCrosshair Refer to CROSSHAIR2 for details.

9.64.2.11 enum QEImage::UserLevels

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

Enumerator:

User Refer to `USERLEVEL_USER` for details.

Scientist Refer to `USERLEVEL_SCIENTIST` for details.

Engineer Refer to `USERLEVEL_ENGINEER` for details.

9.64.3 Constructor & Destructor Documentation

9.64.3.1 `QEImage::QEImage (QWidget * parent = 0)`

Create without a variable. Use `setVariableName'n'Property()` - where 'n' is a number from 0 to 40 - and `setSubstitutionsProperty()` to define variables and, optionally, macro substitutions later. Note, each variable property is named by function (such as `imageVariable` and `widthVariable`) but given a numeric get and set property access function such as `setVariableName22Property()`. Refer to the property definitions to determine what 'set' and 'get' function is used for each variable, or use Qt library functions to set or get the variable names by name.

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

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

9.64.4 Member Function Documentation

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

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

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

!! memcpy will be more efficient.

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

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

9.64.5 Member Data Documentation

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

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

9.64.5.2 int QEImage::initialVertScrollPos [read, write, protected]

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

9.64.6 Property Documentation

9.64.6.1 bool QEImage::allowDrop [read, write]

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

9.64.6.2 QColor QEImage::areaColor [read, write]

Used to select the color of the area selection markups.

9.64.6.3 QStringList QEImage::arguments1 [read, write]

Arguments for program specified in the 'program1' property.

9.64.6.4 QStringList QEImage::arguments2 [read, write]

Arguments for program specified in the 'program2' property.

9.64.6.5 bool QEImage::autoBrightnessContrast [read, write]

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

9.64.6.6 QColor QEImage::beamColor [read, write]

Used to select the color of the beam marker.

9.64.6.7 QString QEImage::beamXVariable [read, write]

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

9.64.6.8 QString QEImage::beamYVariable [read, write]

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

9.64.6.9 QString QEImage::bitDepthVariable [read, write]

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

9.64.6.10 bool QEImage::briefInfoArea [read, write]

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

9.64.6.11 QString QEImage::clippingHighVariable [read, write]

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

9.64.6.12 QString QEImage::clippingLowVariable [read, write]

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

9.64.6.13 QString QEImage::clippingOnOffVariable [read, write]

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

9.64.6.14 bool QEImage::contrastReversal [read, write]

If true, the image will undergo contrast reversal.

9.64.6.15 QString QEImage::defaultStyle [read, write]

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

9.64.6.16 QString QEImage::dimension1Variable [read, write]

EPICS variable name (CA PV). This variable is used to read the first area detector dimension of the image. If there are 2 dimensions, this will be the image width. If there are 3 dimensions, this will be the number of elements per pixel.

9.64.6.17 QString QEImage::dimension2Variable [read, write]

EPICS variable name (CA PV). This variable is used to read the second area detector dimension of the image. If there are 2 dimensions, this will be the image height. If there are 3 dimensions, this will be the image width.

9.64.6.18 QString QEImage::dimension3Variable [read, write]

EPICS variable name (CA PV). This variable is used to read the third area detector dimension of the image. If there are 3 dimensions, this will be the image height.

9.64.6.19 QString QEImage::dimensionsVariable [read, write]

EPICS variable name (CA PV). This variable is used to read the number of area detector dimensions of the image. If used, this will be 2 (one element per pixel arranged by width and height) or 3 (multiple elements per pixel arranged by pixel, width and height)

9.64.6.20 bool QEImage::displayAlarmState [read, write]

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

9.64.6.21 DisplayAlarmStateOptions QEImage::displayAlarmStateOption [read, write]

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

9.64.6.22 bool QEImage::displayArea1Selection [read, write]

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

9.64.6.23 bool QEImage::displayArea2Selection [read, write]

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

9.64.6.24 bool QEImage::displayArea3Selection [read, write]

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

9.64.6.25 bool QEImage::displayArea4Selection [read, write]

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

9.64.6.26 bool QEImage::displayBeamSelection [read, write]

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

9.64.6.27 bool QEImage::displayCursorPixelInfo [read, write]

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

9.64.6.28 bool QEImage::displayEllipse [read, write]

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

9.64.6.29 bool QEImage::displayHozSlice1Selection [read, write]

If true, the selected horizontal slice will be displayed on the image. Note, this property is ignored unless the [enableHozSlice1Selection](#) property is true.

9.64.6.30 bool QEImage::displayHozSlice2Selection [read, write]

If true, the selected horizontal slice will be displayed on the image. Note, this property is ignored unless the [enableHozSlice2Selection](#) property is true.

9.64.6.31 bool QEImage::displayHozSlice3Selection [read, write]

If true, the selected horizontal slice will be displayed on the image. Note, this property is ignored unless the [enableHozSlice3Selection](#) property is true.

9.64.6.32 bool QEImage::displayHozSlice4Selection [read, write]

If true, the selected horizontal slice will be displayed on the image. Note, this property is ignored unless the [enableHozSlice4Selection](#) property is true.

9.64.6.33 bool QEImage::displayHozSlice5Selection [read, write]

If true, the selected horizontal slice will be displayed on the image. Note, this property is ignored unless the [enableHozSlice5Selection](#) property is true.

9.64.6.34 bool QEImage::displayProfileSelection [read, write]

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

9.64.6.35 bool QEImage::displayTargetSelection [read, write]

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

9.64.6.36 bool QEImage::displayVertSliceSelection [read, write]

If true, the selected vertical slice 1 will be displayed on the image. Note, this property is ignored unless the [enableVertSlice1Selection](#) property is true.

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

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

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

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

9.64.6.37 QColor QEImage::ellipseColor [read, write]

Used to select the color of the ellipse marker.

9.64.6.38 QString QEImage::ellipseHVariable [read, write]

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

9.64.6.39 QString QEImage::ellipseWVariable [read, write]

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

9.64.6.40 QString QEImage::ellipseXVariable [read, write]

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

9.64.6.41 QString QEImage::ellipseYVariable [read, write]

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

9.64.6.42 bool QEImage::enableArea1Selection [read, write]

If true, the user will be able to select area 1. These are used for selection of Region of Interests, and for zooming to area 1

9.64.6.43 bool QEImage::enableArea2Selection [read, write]

If true, the user will be able to select area 2. These are used for selection of Region of Interests, and for zooming to area 2

9.64.6.44 bool QEImage::enableArea3Selection [read, write]

If true, the user will be able to select area 3. These are used for selection of Region of Interests, and for zooming to area 3

9.64.6.45 bool QEImage::enableArea4Selection [read, write]

If true, the user will be able to select area 4. These are used for selection of Region of Interests, and for zooming to area 4

9.64.6.46 bool QEImage::enableBeamSelection [read, write]

If true, the user will be able to select points on the image to mark a beam position. This can be used for automatic beam positioning.

9.64.6.47 bool QEImage::enableHozSlice1Selection [read, write]

If true, the option to select a horizontal slice through the image will be available to the user. This will be used to generate a horizontal pixel profile, and write the position of the slice to the optional variable specified by the `profileHoz1Variable` property. The profile will only be presented to the user if `enableHozSlicePresentation` property is true.

9.64.6.48 bool QEImage::enableHozSlice2Selection [read, write]

If true, the option to select a second horizontal slice through the image will be available to the user. This will be used to write the position of the slice to the optional variable specified by the `profileHoz2Variable` property.

9.64.6.49 bool QEImage::enableHozSlice3Selection [read, write]

If true, the option to select a third horizontal slice through the image will be available to the user. This will be used to write the position of the slice to the optional variable specified by the [profileHoz3Variable](#) property.

9.64.6.50 bool QEImage::enableHozSlice4Selection [read, write]

If true, the option to select a fourth horizontal slice through the image will be available to the user. This will be used to write the position of the slice to the optional variable specified by the [profileHoz4Variable](#) property.

9.64.6.51 bool QEImage::enableHozSlice5Selection [read, write]

If true, the option to select a fifth horizontal slice through the image will be available to the user. This will be used to write the position of the slice to the optional variable specified by the [profileHoz5Variable](#) property.

9.64.6.52 bool QEImage::enableProfileSelection [read, write]

If true, the option to select an arbitrary line through any part of the image will be available to the user. This will be used to generate a pixel profile.

9.64.6.53 bool QEImage::enableTargetSelection [read, write]

If true, the user will be able to select points on the image to mark a target position. This can be used for automatic beam positioning.

9.64.6.54 bool QEImage::enableVertSlice1Selection [read, write]

If true, the option to select a vertical slice through the image will be available to the user. This will be used to generate a horizontal pixel profile, and write the position of the slice to the optional variable specified by the [profileVert1Variable](#) property. The profile will only be presented to the user if #enableVertSlicePresentation property is true.

9.64.6.55 bool QEImage::enableVertSlice2Selection [read, write]

If true, the option to select a second vertical slice through the image will be available to the user. This will be used to write the position of the slice to the optional variable specified by the [profileVert2Variable](#) property.

9.64.6.56 bool QEImage::enableVertSlice3Selection [read, write]

If true, the option to select a third vertical slice through the image will be available to the user. This will be used to write the position of the slice to the optional variable specified by the [profileVert3Variable](#) property.

9.64.6.57 bool QEImage::enableVertSlice4Selection [read, write]

If true, the option to select a fourth vertical slice through the image will be available to the user. This will be used to write the position of the slice to the optional variable specified by the [profileVert4Variable](#) property.

9.64.6.58 bool QEImage::enableVertSlice5Selection [read, write]

If true, the option to select a fifth vertical slice through the image will be available to the user. This will be used to write the position of the slice to the optional variable specified by the [profileVert5Variable](#) property.

9.64.6.59 bool QEImage::externalControls [read, write]

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

9.64.6.60 FormatOptions QEImage::formatOption [read, write]

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

9.64.6.61 QString QEImage::formatVariable [read, write]

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

9.64.6.62 QString QEImage::heightVariable [read, write]

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

9.64.6.63 bool QEImage::horizontalFlip [read, write]

If true, flip image horizontally.

9.64.6.64 `QColor QEImage::hozSlice1Color [read, write]`

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

9.64.6.65 `QColor QEImage::hozSlice2Color [read, write]`

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

9.64.6.66 `QColor QEImage::hozSlice3Color [read, write]`

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

9.64.6.67 `QColor QEImage::hozSlice4Color [read, write]`

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

9.64.6.68 `QColor QEImage::hozSlice5Color [read, write]`

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

9.64.6.69 `QString QEImage::imageVariable [read, write]`

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

9.64.6.70 `int QEImage::initialHosScrollPos [read, write]`

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

9.64.6.71 `unsigned QEImage::int [read, write]`

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

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

9.64.6.72 QString QEImage::lineProfileArrayVariable [read, write]

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

9.64.6.73 QString QEImage::lineProfileThicknessVariable [read, write]

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

9.64.6.74 QString QEImage::lineProfileX1Variable [read, write]

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

9.64.6.75 QString QEImage::lineProfileX2Variable [read, write]

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

9.64.6.76 QString QEImage::lineProfileY1Variable [read, write]

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

9.64.6.77 QString QEImage::lineProfileY2Variable [read, write]

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

9.64.6.78 bool QEImage::logBrightness [read, write]

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

9.64.6.79 QColor QEImage::profileColor [read, write]

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

9.64.6.80 QString QEImage::profileHoz1ThicknessVariable [read, write]

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

9.64.6.81 QString QEImage::profileHoz1Variable [read, write]

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

9.64.6.82 QString QEImage::profileHoz2ThicknessVariable [read, write]

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

9.64.6.83 QString QEImage::profileHoz2Variable [read, write]

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

9.64.6.84 QString QEImage::profileHoz3ThicknessVariable [read, write]

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

9.64.6.85 QString QEImage::profileHoz3Variable [read, write]

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

9.64.6.86 QString QEImage::profileHoz4ThicknessVariable [read, write]

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

9.64.6.87 QString QEImage::profileHoz4Variable [read, write]

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

9.64.6.88 QString QEImage::profileHoz5ThicknessVariable [read, write]

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

9.64.6.89 QString QEImage::profileHoz5Variable [read, write]

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

9.64.6.90 QString QEImage::profileHozArrayVariable [read, write]

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

9.64.6.91 QString QEImage::profileVert1ThicknessVariable [read, write]

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

9.64.6.92 QString QEImage::profileVert1Variable [read, write]

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

9.64.6.93 QString QEImage::profileVert2ThicknessVariable [read, write]

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

9.64.6.94 QString QEImage::profileVert2Variable [read, write]

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

9.64.6.95 QString QEImage::profileVert3ThicknessVariable [read, write]

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

9.64.6.96 QString QEImage::profileVert3Variable [read, write]

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

9.64.6.97 QString QEImage::profileVert4ThicknessVariable [read, write]

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

9.64.6.98 QString QEImage::profileVert4Variable [read, write]

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

9.64.6.99 QString QEImage::profileVert5ThicknessVariable [read, write]

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

9.64.6.100 QString QEImage::profileVert5Variable [read, write]

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

9.64.6.101 QString QEImage::profileVertArrayVariable [read, write]

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

9.64.6.102 QString QEImage::program1 [read, write]

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

9.64.6.103 QString QEImage::program2 [read, write]

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

9.64.6.104 ProgramStartupOptionNames QEImage::programStartupOption1 [read, write]

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

9.64.6.105 ProgramStartupOptionNames QEImage::programStartupOption2 [read, write]

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

9.64.6.106 QString QEImage::regionOfInterest1HVariable [read, write]

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

9.64.6.107 QString QEImage::regionOfInterest1WVariable [read, write]

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

9.64.6.108 QString QEImage::regionOfInterest1XVariable [read, write]

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

9.64.6.109 QString QEImage::regionOfInterest1YVariable [read, write]

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

9.64.6.110 QString QEImage::regionOfInterest2HVariable [read, write]

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

9.64.6.111 QString QEImage::regionOfInterest2WVariable [read, write]

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

9.64.6.112 QString QEImage::regionOfInterest2XVariable [read, write]

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

9.64.6.113 QString QEImage::regionOfInterest2YVariable [read, write]

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

9.64.6.114 QString QEImage::regionOfInterest3HVariable [read, write]

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

9.64.6.115 QString QEImage::regionOfInterest3WVariable [read, write]

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

9.64.6.116 QString QEImage::regionOfInterest3XVariable [read, write]

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

9.64.6.117 QString QEImage::regionOfInterest3YVariable [read, write]

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

9.64.6.118 QString QEImage::regionOfInterest4HVariable [read, write]

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

9.64.6.119 QString QEImage::regionOfInterest4WVariable [read, write]

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

9.64.6.120 QString QEImage::regionOfInterest4XVariable [read, write]

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

9.64.6.121 QString QEImage::regionOfInterest4YVariable [read, write]

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

9.64.6.122 ResizeOptions QEImage::resizeOption [read, write]

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

9.64.6.123 RotationOptions QEImage::rotation [read, write]

Image rotation option.

9.64.6.124 `bool QEImage::showTime [read, write]`

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

9.64.6.125 `QString QEImage::styleSheet [read, write]`

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

9.64.6.126 `QColor QEImage::targetColor [read, write]`

Used to select the color of the target marker.

9.64.6.127 `QString QEImage::targetTriggerVariable [read, write]`

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

9.64.6.128 `QString QEImage::targetXVariable [read, write]`

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

9.64.6.129 `QString QEImage::targetYVariable [read, write]`

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

9.64.6.130 `QColor QEImage::timeColor [read, write]`

Used to select the color of the timestamp.

9.64.6.131 `QString QEImage::URL [read, write]`

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

9.64.6.132 `bool QEImage::useFalseColour [read, write]`

If true, the apply false colour to the image.

9.64.6.133 UserLevels QEImage::userLevelEnabled [read, write]

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

9.64.6.134 QString QEImage::userLevelEngineerStyle [read, write]

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

9.64.6.135 QString QEImage::userLevelScientistStyle [read, write]

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

9.64.6.136 QString QEImage::userLevelUserStyle [read, write]

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

9.64.6.137 UserLevels QEImage::userLevelVisibility [read, write]

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

9.64.6.138 bool QEImage::variableAsToolTip [read, write]

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

9.64.6.139 QString QEImage::variableSubstitutions [read, write]

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

9.64.6.140 bool QEImage::verticalFlip [read, write]

If true, flip image vertically.

9.64.6.141 QColor QEImage::vertSlice1Color [read, write]

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

9.64.6.142 QColor QEImage::vertSlice2Color [read, write]

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

9.64.6.143 QColor QEImage::vertSlice3Color [read, write]

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

9.64.6.144 QColor QEImage::vertSlice4Color [read, write]

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

9.64.6.145 QColor QEImage::vertSlice5Color [read, write]

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

9.64.6.146 bool QEImage::visible [read, write]

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

9.64.6.147 QString QEImage::widthVariable [read, write]

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

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

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

9.65 QEImageMarkupThickness Class Reference

Public Member Functions

- **QEImageMarkupThickness** (QWidget *parent=0)
- void **setThickness** (unsigned int thicknessIn)
- unsigned int **getThickness** ()

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

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

9.66 QEImageOptionsDialog Class Reference

Signals

- void **optionChange** (imageContextMenu::imageContextMenuOptions option, bool checked)

Public Member Functions

- **QEImageOptionsDialog** (QWidget *parent=0)
- void **initialise** ()
- void **optionSet** (imageContextMenu::imageContextMenuOptions option, bool checked)
- bool **optionGet** (imageContextMenu::imageContextMenuOptions option)

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

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

9.67 QELabel Class Reference

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

Public Types

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

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

Public Slots

- void `setDefaultStyle` (const QString &style)

Update the default style applied to this widget.
- void `setManagedVisible` (bool v)

Signals

- void `dbValueChanged` (const QString &out)
 - void `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 StyleSheet`
- `QString defaultStyle`
- `QString userLevelUserStyle`
- `QString userLevelScientistStyle`
- `QString userLevelEngineerStyle`
- `UserLevels userLevelVisibility`
- `UserLevels userLevelEnabled`
- `bool displayAlarmState`
- `DisplayAlarmStateOptions displayAlarmStateOption`
- `int precision`
- `bool useDbPrecision`
- `bool leadingZero`
- `bool trailingZeros`
- `bool addUnits`
- `QString localEnumeration`
- `Formats format`
- `Notations notation`
- `ArrayActions arrayAction`
- `QString text`
- `UpdateOptions updateOption`
- `QPixmap pixmap0`
- `QPixmap pixmap1`
- `QPixmap pixmap2`
- `QPixmap pixmap3`
- `QPixmap pixmap4`
- `QPixmap pixmap5`
- `QPixmap pixmap6`
- `QPixmap pixmap7`

9.67.1 Detailed Description

This class is a EPICS aware label widget based on the Qt label widget. When a variable is defined, the label text (or optionally the background pixmap) will be updated. The label will be disabled if the variable is invalid. It is tightly integrated with the base class QEWidget which provides generic support such as macro substitutions, drag/drop, and standard properties.

9.67.2 Member Enumeration Documentation

9.67.2.1 enum QELabel::ArrayActions

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

Enumerator:

Append Refer to `QQStringFormatting::APPEND` for details.

Ascii Refer to `QQStringFormatting::ASCII` for details.

Index Refer to `QQStringFormatting::INDEX` for details.

9.67.2.2 enum QELabel::DisplayAlarmStateOptions

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

Enumerator:

Never Refer to `DISPLAY_ALARM_STATE_NEVER` for details.

Always Refer to `DISPLAY_ALARM_STATE_ALWAYS` for details.

WhenInAlarm Refer to `DISPLAY_ALARM_STATE_WHEN_IN_ALARM` for details.

9.67.2.3 enum QELabel::Formats

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

Enumerator:

Default Format as best appropriate for the data type.

Floating Format as a floating point number.

Integer Format as an integer.

UnsignedInteger Format as an unsigned integer.

Time Format as a time.

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

9.67.2.4 enum QELabel::Notations

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

Enumerator:

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

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

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

9.67.2.5 enum QELabel::updateOptions

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

Enumerator:

UPDATE_TEXT Update the label text.

UPDATE_PIXMAP Update the label background pixmap.

9.67.2.6 enum QELabel::UpdateOptions

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

Enumerator:

Text Data updates will update the label text.

Picture Data updates will update the label icon.

9.67.2.7 enum QELabel::UserLevels

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

Enumerator:

User Refer to USERLEVEL_USER for details.

Scientist Refer to USERLEVEL_SCIENTIST for details.

Engineer Refer to USERLEVEL_ENGINEER for details.

9.67.3 Constructor & Destructor Documentation

9.67.3.1 QELabel::QELabel (QWidget * *parent* = 0)

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

9.67.3.2 QELabel::QELabel (const QString & *variableName*, QWidget * *parent* = 0)

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

9.67.4 Member Function Documentation

9.67.4.1 void QELabel::dbValueChanged (const QString & *out*) [signal]

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

9.67.4.2 void QELabel::setManagedVisible (bool *v*) [inline, slot]

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

9.67.5 Property Documentation

9.67.5.1 bool QELabel::addUnits [read, write]

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

9.67.5.2 bool QELabel::allowDrop [read, write]

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

9.67.5.3 ArrayActions QELabel::arrayAction [read, write]

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

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

9.67.5.4 QString QELabel::defaultStyle [read, write]

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

9.67.5.5 bool QELabel::displayAlarmState [read, write]

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

9.67.5.6 DisplayAlarmStateOptions QELabel::displayAlarmStateOption [read, write]

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

9.67.5.7 Formats QELabel::format [read, write]

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

9.67.5.8 unsigned QELabel::int [read, write]

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

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

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

9.67.5.9 bool QELabel::leadingZero [read, write]

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

9.67.5.10 QString QELabel::localEnumeration [read, write]

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

Format is:

```
[[<|<=|=|=|>|=]>]value1|*]: string1 , [[<|<=|=|=|>|=]>]value2|*]: string2 , [[<|<=|=|=|>|=]>]value3|*]
: string3 , ...
```

Where: < Less than <= Less than or equal = Equal (default if no operator specified)
>= Greater than or equal > Greater than Always match (used to specify default text)

Values may be numeric or textual Values do not have to be in any order, but first match wins Values may be quoted Strings may be quoted Consecutive values do not have to be present. Operator is assumed to be equality if not present. White space is ignored except within quoted strings.

may be included in a string to indicate a line break

Examples are:

```
0:Off,1:On 0 : "Pump Running", 1 : "Pump not running" 0:"", 1:"Warning!\nAlarm"
<2:"Value is less than two", =2:"Value is equal to two", >2:"Value is grater than 2"
3:"Beamline Available", *:"" "Pump Off":"OH NO!, the pump is OFF!","Pump On":"It's
OK, the pump is on"
```

The data value is converted to a string if no enumeration for that value is available. For example, if the local enumeration is '0:off,1:on', and a value of 10 is processed, the text generated is '10'. If a blank string is required, this should be explicit. for example, '0:off,1:on,10:'''

A range of numbers can be covered by a pair of values as in the following example:
>=4:"Between 4 and 8",<=8:"Between 4 and 8"

9.67.5.11 Notations QELabel::notation [read, write]

Notation used for numerical formatting. Default is fixed.

9.67.5.12 QPixmap QELabel::pixmap0 [read, write]

Pixmap displayed when updateOption property is 'Picture' and data is interpreted as 0.

9.67.5.13 QPixmap QELabel::pixmap1 [read, write]

Pixmap displayed when updateOption property is 'Picture' and data is interpreted as 1.

9.67.5.14 QPixmap QELabel::pixmap2 [read, write]

Pixmap displayed when updateOption property is 'Picture' and data is interpreted as 2.

9.67.5.15 QPixmap QELabel:: pixmap3 [read, write]

Pixmap displayed when updateOption property is 'Picture' and data is interpreted as 3.

9.67.5.16 QPixmap QELabel:: pixmap4 [read, write]

Pixmap displayed when updateOption property is 'Picture' and data is interpreted as 4.

9.67.5.17 QPixmap QELabel:: pixmap5 [read, write]

Pixmap displayed when updateOption property is 'Picture' and data is interpreted as 5.

9.67.5.18 QPixmap QELabel:: pixmap6 [read, write]

Pixmap displayed when updateOption property is 'Picture' and data is interpreted as 6.

9.67.5.19 QPixmap QELabel:: pixmap7 [read, write]

Pixmap displayed when updateOption property is 'Picture' and data is interpreted as 7.

9.67.5.20 int QELabel:: precision [read, write]

Precision used when formatting floating point numbers. The default is 4. This is only used if useDbPrecision is false.

9.67.5.21 QString QELabel:: styleSheet [read, write]

Hide style sheet from designer as style calculation by the styleManager and not directly setable per se. This also stops transient styles being saved to the ui file.

9.67.5.22 bool QELabel:: trailingZeros [read, write]

If true (default), always remove any trailing zeros when formatting numbers.

9.67.5.23 UpdateOptions QELabel:: updateOption [read, write]

Determines if data updates the label text, or the label pixmap. For both options all normal string formatting is applied. If Text, the formatted text is simply presented as the

label text. If Picture, the FORMATTED text is then interpreted as an integer and used to select one of the pixmaps specified by properties pixmap0 through to pixmap7.

9.67.5.24 bool QELabel::useDbPrecision [read, write]

If true (default), format floating point numbers using the precision supplied with the data. If false, the precision property is used.

9.67.5.25 UserLevels QELabel::userLevelEnabled [read, write]

Lowest user level at which the widget is enabled. Default is 'User'. Used when designing GUIs that allow access to more and more detail according to the user mode. The user mode is set application wide through the [QELogin](#) widget, or programmatically through `setUserLevel()`. Widgets that are always accessible should be visible at 'User'. Widgets that are only accessible to scientists managing the facility should be visible at 'Scientist'. Widgets that are only accessible to engineers maintaining the facility should be visible at 'Engineer'.

9.67.5.26 QString QELabel::userLevelEngineerStyle [read, write]

Style Sheet string to be applied when the widget is displayed in 'Engineer' mode. Default is an empty string. The syntax is the standard Qt Style Sheet syntax. For example, 'background-color: red' This Style Sheet string will be applied by the styleManager class. Refer to the styleManager class for details about how this Style Sheet string will be merged with any pre-existing Style Sheet string and any Style Sheet strings generated during the display of data.

9.67.5.27 QString QELabel::userLevelScientistStyle [read, write]

Style Sheet string to be applied when the widget is displayed in 'Scientist' mode. Default is an empty string. The syntax is the standard Qt Style Sheet syntax. For example, 'background-color: red' This Style Sheet string will be applied by the styleManager class. Refer to the styleManager class for details about how this Style Sheet string will be merged with any pre-existing Style Sheet string and any Style Sheet strings generated during the display of data.

9.67.5.28 QString QELabel::userLevelUserStyle [read, write]

Style Sheet string to be applied when the widget is displayed in 'User' mode. Default is an empty string. The syntax is the standard Qt Style Sheet syntax. For example, 'background-color: red' This Style Sheet string will be applied by the styleManager class. Refer to the styleManager class for details about how this Style Sheet string will be merged with any pre-existing Style Sheet string and any Style Sheet strings generated during the display of data.

9.67.5.29 UserLevels QELabel::userLevelVisibility [read, write]

Lowest user level at which the widget is visible. Default is 'User'. Used when designing GUIs that display more and more detail according to the user mode. The user mode is set application wide through the [QELogin](#) widget, or programmatically through setUserLevel(). Widgets that are always visible should be visible at 'User'. Widgets that are only used by scientists managing the facility should be visible at 'Scientist'. Widgets that are only used by engineers maintaining the facility should be visible at 'Engineer'.

9.67.5.30 QString QELabel::variable [read, write]

EPICS variable name (CA PV)

9.67.5.31 bool QELabel::variableAsToolTip [read, write]

Use the variable as the tool tip. Default is true. Tool tip property will be overwritten by the variable name.

9.67.5.32 QString QELabel::variableSubstitutions [read, write]

Macro substitutions. The default is no substitutions. The format is NAME1=VALUE1[,] NAME2=VALUE2... Values may be quoted strings. For example, 'PUMP=PMP3, NAME = "My Pump"' These substitutions are applied to variable names for all QE widgets. In some widgets are also used for other purposes.

9.67.5.33 bool QELabel::visible [read, write]

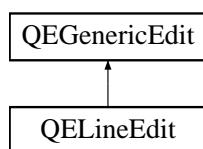
Display the widget. Default is true. Setting this property false is useful if widget is only used to provide a signal - for example, when supplying data to a [QELink](#) widget. Note, when false the widget will still be visible in Qt Designer.

The documentation for this class was generated from the following files:

- /tmp/epicsqt/trunk/framework/widgets/QELabel/QELabel.h
- /tmp/epicsqt/trunk/framework/widgets/QELabel/QELabel.cpp

9.68 QELlineEdit Class Reference

Inheritance diagram for QELlineEdit:



Public Types

- enum `Formats` {

`Default` = QEStringFormatting::FORMAT_DEFAULT, `Floating` = QEStringFormatting::FORMAT_FLOATING, `Integer` = QEStringFormatting::FORMAT_INTEGER, `UnsignedInteger` = QEStringFormatting::FORMAT_UNSIGNEDINTEGER,

`Time` = QEStringFormatting::FORMAT_TIME, `LocalEnumeration` = QEStringFormatting::FORMAT_LOCAL_ENUMERATE }
- enum `Notations` { `Fixed` = QEStringFormatting::NOTATION_FIXED, `Scientific` = QEStringFormatting::NOTATION_SCIENTIFIC, `Automatic` = QEStringFormatting::NOTATION_AUTOMATIC }
- enum `ArrayActions` { `Append` = QEStringFormatting::APPEND, `Ascii` = QEStringFormatting::ASCII, `Index` = QEStringFormatting::INDEX }

Signals

- void `dbValueChanged` (const QString &out)
- void `userChange` (const QString &oldValue, const QString &newValue, const QString &lastValue)

Internal use only. Used by `QEConfiguredLayout` to be notified when one of its widgets has written something.
- void `requestResend` ()

Internal use only. Used when changing a property value to force a re-display to reflect the new property value.

Public Member Functions

- void `setFormatProperty` (`Formats` format)

Access function for `format` property - refer to `format` property for details.
- `Formats getFormatProperty` ()

Access function for `format` property - refer to `format` property for details.
- void `setNotationProperty` (`Notations` notation)

Access function for `notation` property - refer to `notation` property for details.
- `Notations getNotationProperty` ()

Access function for `notation` property - refer to `notation` property for details.
- void `setArrayActionProperty` (`ArrayActions` arrayAction)

Access function for `arrayAction` property - refer to `arrayAction` property for details.
- `ArrayActions getArrayActionProperty` ()

Access function for `arrayAction` property - refer to `arrayAction` property for details.
- `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](#)

9.68.1 Member Enumeration Documentation

9.68.1.1 enum QELineEdit::ArrayActions

User friendly enumerations for arrayAction property - refer to QEStringFormatting::arrayActions for details.

Enumerator:

Append Refer to QEStringFormatting::APPEND for details.

Ascii Refer to QEStringFormatting::ASCII for details.

Index Refer to QEStringFormatting::INDEX for details.

9.68.1.2 enum QELineEdit::Formats

User friendly enumerations for format property - refer to QEStringFormatting::formats for details.

Enumerator:

Default Format as best appropriate for the data type.

Floating Format as a floating point number.

Integer Format as an integer.

UnsignedInteger Format as an unsigned integer.

Time Format as a time.

LocalEnumeration Format as a selection from the [localEnumeration](#) property.

9.68.1.3 enum QELineEdit::Notations

User friendly enumerations for notation property - refer to QEStringFormatting::notations for details.

Enumerator:

Fixed Refer to QEStringFormatting::NOTATION_FIXED for details.

Scientific Refer to QEStringFormatting::NOTATION_SCIENTIFIC for details.

Automatic Refer to QEStringFormatting::NOTATION_AUTOMATIC for details.

9.68.2 Constructor & Destructor Documentation

9.68.2.1 QELineEdit::QELineEdit (QWidget * *parent* = 0)

Create without a variable. Use [setVariableNameProperty\(\)](#) and [setSubstitutionsProperty\(\)](#) to define a variable and, optionally, macro substitutions later.

9.68.2.2 QELineEdit::QELineEdit (const QString & *variableName*, QWidget * *parent* = 0)

Create with a variable. A connection is automatically established. If macro substitutions are required, create without a variable and set the variable and macro substitutions after creation.

9.68.3 Member Function Documentation

9.68.3.1 void QELineEdit::dbValueChanged (const QString & *out*) [signal]

Sent when the widget is updated following a data change Can be used to pass on EPICS data (as presented in this widget) to other widgets. For example a QList widget could log updates from this widget.

9.68.4 Property Documentation

9.68.4.1 bool QELineEdit::addUnits [read, write]

If true (default), add engineering units supplied with the data.

9.68.4.2 ArrayActions QELineEdit::arrayAction [read, write]

Text formatting option for array data. Default is ASCII. Options are:

- ASCII - treat array as a single text string. For example an array of three characters 'a' 'b' 'c' will be formatted as 'abc'.

- APPEND - treat array as an array of numbers and format a string containing them all with a space between each. For example, an array of three numbers 10, 11 and 12 will be formatted as '10 11 12'.
- INDEX - Extract a single item from the array. The item is then formatted as any other non array data would be. The item selected is determined by the arrayIndex property. For example, if arrayIndex property is 1, an array of three numbers 10, 11 and 12 will be formatted as '11'.

9.68.4.3 Formats QELineEdit::format [read, write]

Format to apply to data. Default is 'Default' in which case the data type supplied with the data determines how the data is formatted. For all other options, an attempt is made to format the data as requested (whatever its native form).

9.68.4.4 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](#).

9.68.4.5 bool QELineEdit::leadingZero [read, write]

If true (default), always add a leading zero when formatting numbers.

9.68.4.6 QString QELineEdit::localEnumeration [read, write]

An enumeration list used to data values. Used only when the formatting option is 'local enumeration'. Value is converted to an integer and used to select a string from this list.

Format is:

```
[[<|<=|=|=|>=|>]value1|*] : string1 , [[<|<=|=|=|>=|>]value2|*] : string2 , [[<|<=|=|=|>=|>]value3|*]
: string3 , ...
```

Where: < Less than <= Less than or equal = Equal (default if no operator specified)
>= Greater than or equal > Greater than Always match (used to specify default text)

Values may be numeric or textual Values do not have to be in any order, but first match wins Values may be quoted Strings may be quoted Consecutive values do not have to be present. Operator is assumed to be equality if not present. White space is ignored except within quoted strings.

may be included in a string to indicate a line break

Examples are:

0:Off,1:On 0 : "Pump Running", 1 : "Pump not running" 0:"", 1:"Warning!\nAlarm" <2:"Value is less than two", =2:"Value is equal to two", >2:"Value is grater than 2" 3:"Beamline Available", *:"" "Pump Off":"OH NO!, the pump is OFF!","Pump On":"It's OK, the pump is on"

The data value is converted to a string if no enumeration for that value is available. For example, if the local enumeration is '0:off,1:on', and a value of 10 is processed, the text generated is '10'. If a blank string is required, this should be explicit. for example, '0:off,1:on,10:'''

A range of numbers can be covered by a pair of values as in the following example:
>=4:"Between 4 and 8", <=8:"Between 4 and 8"

9.68.4.7 Notations QELineEdit::notation [read, write]

Notation used for numerical formatting. Default is fixed.

9.68.4.8 int QELineEdit::precision [read, write]

Precision used when formatting floating point numbers. The default is 4. This is only used if useDbPrecision is false.

9.68.4.9 bool QELineEdit::trailingZeros [read, write]

If true (default), always remove any trailing zeros when formatting numbers.

9.68.4.10 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

9.69 QELineEditManager Class Reference

Public Member Functions

- **QELineEditManager** (QObject *parent=0)
- bool **isContainer** () const
- bool **isInitialized** () const
- QIcon **icon** () const
- QString **group** () const
- QString **includeFile** () const

- `QString name () const`
- `QString toolTip () const`
- `QString whatsThis () const`
- `QWidget * createWidget (QWidget *parent)`
- `void initialize (QDesignerFormEditorInterface *core)`

The documentation for this class was generated from the following file:

- `/tmp/epicsqt/trunk/framework/widgets/QELineEdit/QELineEditManager.h`

9.70 QELink Class Reference

Public Types

- `enum conditions {
 CONDITION_EQ, CONDITION_NE, CONDITION_GT, CONDITION_GE,
 CONDITION_LT, CONDITION_LE }`
- `enum ConditionNames {
 Equal = QELink::CONDITION_EQ, NotEqual = QELink::CONDITION_NE, GreaterThan
 = QELink::CONDITION_GT, GreaterThanOrEqual = QELink::CONDITION_GE,
 LessThan = QELink::CONDITION_LT, LessThanOrEqual = QELink::CONDITION_-
 LE }`

Public Slots

- `void in (const bool &in)`
- `void in (const 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

9.71 QELog Class Reference

Public Types

- enum **optionsLayoutProperty** { **Top** = TOP, **Bottom** = BOTTOM, **Left** = LEFT, **Right** = RIGHT }
- enum **MessageFilterOptions** { **Any** = UserMessage::MESSAGE_FILTER_ANY, **Match** = UserMessage::MESSAGE_FILTER_MATCH, **None** = UserMessage::MESSAGE_FILTER_NONE }
- enum **UserLevels** { **User** = userLevelTypes::USERLEVEL_USER, **Scientist** = userLevelTypes::USERLEVEL_SCIENTIST, **Engineer** = userLevelTypes::USERLEVEL_ENGINEER }
- enum **DisplayAlarmStateOptions** { **Never** = standardProperties::DISPLAY_ALARM_STATE_NEVER, **Always** = standardProperties::DISPLAY_ALARM_STATE_ALWAYS, **WhenInAlarm** = standardProperties::DISPLAY_ALARM_STATE_WHEN_IN_ALARM }

Public Slots

- void **setManagedVisible** (bool v)

Public Member Functions

- **QELog** (QWidget *pParent=0)
- void **setShowColumnType** (bool pValue)
- bool **getShowColumnType** ()
- void **setShowColumnMessage** (bool pValue)
- bool **getShowColumnMessage** ()
- void **setShowMessageFilter** (bool pValue)
- bool **getShowMessageFilter** ()
- void **setShowClear** (bool pValue)
- bool **getShowClear** ()
- void **setShowSave** (bool pValue)
- bool **getShowSave** ()
- void **setOptionsLayout** (int pValue)
- int **getOptionsLayout** ()
- void **setScrollToBottom** (bool pValue)
- bool **getScrollToBottom** ()
- void **setInfoColor** (QColor pValue)
- QColor **getInfoColor** ()
- void **setWarningColor** (QColor pValue)
- QColor **getWarningColor** ()
- void **setErrorColor** (QColor pValue)
- QColor **getErrorColor** ()
- void **clearLog** ()

- void **addLog** (int pType, QString pMessage)
- void **refreshLog** ()
- void **setOptionsLayoutProperty** (optionsLayoutProperty pOptionsLayout)
- optionsLayoutProperty **getOptionsLayoutProperty** ()
- MessageFilterOptions **getMessageFormFilter** ()
- void **setMessageFormFilter** (MessageFilterOptions messageFormFilter)
- MessageFilterOptions **getMessageSourceFilter** ()
- void **setMessageSourceFilter** (MessageFilterOptions messageSourceFilter)
- **UserLevels getUserLevelVisibilityProperty** ()

Access function for `userLevelVisibility` property - refer to `userLevelVisibility` property for details.
- void **setUserLevelVisibilityProperty** (UserLevels level)

Access function for `userLevelVisibility` property - refer to `userLevelVisibility` property for details.
- **UserLevels getUserLevelEnabledProperty** ()

Access function for `userLevelEnabled` property - refer to `userLevelEnabled` property for details.
- void **setUserLevelEnabledProperty** (UserLevels level)

Access function for `userLevelEnabled` property - refer to `userLevelEnabled` property for details.
- **DisplayAlarmStateOptions getDisplayAlarmStateOptionProperty** ()

Access function for `displayAlarmStateOption` property - refer to `displayAlarmStateOption` property for details.
- void **setDisplayAlarmStateOptionProperty** (DisplayAlarmStateOptions option)

Access function for `displayAlarmStateOption` property - refer to `displayAlarmStateOption` property for details.

Protected Attributes

- **_QTableWidgetLog * qTableWidgetLog**
- QCheckBox * **qCheckBoxInfoMessage**
- QCheckBox * **qCheckBoxWarningMessage**
- QCheckBox * **qCheckBoxErrorMessage**
- QPushButton * **qPushButtonClear**
- QPushButton * **qPushButtonSave**
- QColor **qColorInfo**
- QColor **qColorWarning**
- QColor **qColorError**
- bool **scrollToBottom**
- int **optionsLayout**

Properties

- bool **showColumnTime**
- bool **showColumnType**
- bool **showColumnMessage**
- bool **showMessageFilter**
- bool **showClear**
- bool **showSave**
- optionsLayoutProperty **optionsLayout**
- QColor **infoColor**
- QColor **warningColor**
- QColor **errorColor**
- MessageFilterOptions **messageFormFilter**
- MessageFilterOptions **messageSourceFilter**
- unsigned **int**
- bool **variableAsToolTip**
- bool **allowDrop**
- bool **visible**
- QString **styleSheet**
- QString **defaultStyle**
- QString **userLevelUserStyle**
- QString **userLevelScientistStyle**
- QString **userLevelEngineerStyle**
- UserLevels **userLevelVisibility**
- UserLevels **userLevelEnabled**
- bool **displayAlarmState**
- DisplayAlarmStateOptions **displayAlarmStateOption**

9.71.1 Member Enumeration Documentation

9.71.1.1 enum QELog::DisplayAlarmStateOptions

User friendly enumerations for [displayAlarmStateOption](#) property - refer to [displayAlarmStateOption](#) property and [displayAlarmStateOptions](#) enumeration for details.

Enumerator:

Never Refer to DISPLAY_ALARM_STATE_NEVER for details.

Always Refer to DISPLAY_ALARM_STATE_ALWAYS for details.

WhenInAlarm Refer to DISPLAY_ALARM_STATE_WHEN_IN_ALARM for details.

9.71.1.2 enum QELog::UserLevels

User friendly enumerations for [userLevelVisibility](#) and [userLevelEnabled](#) properties - refer to [userLevelVisibility](#) and [userLevelEnabled](#) properties and userLevel enumeration for details.

Enumerator:

User Refer to USERLEVEL_USER for details.

Scientist Refer to USERLEVEL_SCIENTIST for details.

Engineer Refer to USERLEVEL_ENGINEER for details.

9.71.2 Member Function Documentation

9.71.2.1 void QELog::setManagedVisible(bool v) [inline, slot]

Slot to set the visibility of a QE widget, taking into account the user level. Widget will be hidden if hidden by a call this slot, by will only be made visible by a call to this slot if the user level allows.

9.71.3 Property Documentation

9.71.3.1 bool QELog::allowDrop [read, write]

Allow drag/drops operations to this widget. Default is false. Any dropped text will be used as a new variable name.

9.71.3.2 QString QELog::defaultStyle [read, write]

Style Sheet string to be applied before, i.e. lower priority than, any other style, e.g. alarm style and/or user level style. Default is an empty string.

9.71.3.3 bool QELog::displayAlarmState [read, write]

DEPRECATED. USE [displayAlarmStateOption](#) INSTEAD. If set (default) widget will indicate the alarm state of any variable data it is displaying. If clear widget will never indicate the alarm state of any variable data it is displaying. Typically the background colour is set to indicate the alarm state. Note, this property is included in the set of standard properties as it applies to most widgets. It will do nothing for widgets that don't display data.

9.71.3.4 DisplayAlarmStateOptions QELog::displayAlarmStateOption [read, write]

If 'Always' (default) widget will indicate the alarm state of any variable data it is displaying, including 'No Alarm' If 'Never' widget will never indicate the alarm state of any

variable data it is displaying. If 'WhenInAlarm' widget only indicate the alarm state of any variable data it is displaying if it is 'in alarm'. Typically the background colour is set to indicate the alarm state. Note, this property is included in the set of standard properties as it applies to most widgets. It will do nothing for widgets that don't display data.

9.71.3.5 `unsigned QELog::int [read, write]`

Set the ID used by the message filtering system. Default is zero. Widgets or applications that use messages from the framework have the option of filtering on this ID. For example, by using a unique message source ID a [QELog](#) widget may be set up to only log messages from a select set of widgets.

9.71.3.6 `QString QELog::styleSheet [read, write]`

Hide style sheet from designer as style calculation by the styleManager and not directly setable per se. This also stops transient styles being saved to the ui file.

9.71.3.7 `UserLevels QELog::userLevelEnabled [read, write]`

Lowest user level at which the widget is enabled. Default is 'User'. Used when designing GUIs that allow access to more and more detail according to the user mode. The user mode is set application wide through the [QELogin](#) widget, or programmatically through `setUserLevel()`. Widgets that are always accessible should be visible at 'User'. Widgets that are only accessible to scientists managing the facility should be visible at 'Scientist'. Widgets that are only accessible to engineers maintaining the facility should be visible at 'Engineer'.

9.71.3.8 `QString QELog::userLevelEngineerStyle [read, write]`

Style Sheet string to be applied when the widget is displayed in 'Engineer' mode. Default is an empty string. The syntax is the standard Qt Style Sheet syntax. For example, 'background-color: red' This Style Sheet string will be applied by the styleManager class. Refer to the styleManager class for details about how this Style Sheet string will be merged with any pre-existing Style Sheet string and any Style Sheet strings generated during the display of data.

9.71.3.9 `QString QELog::userLevelScientistStyle [read, write]`

Style Sheet string to be applied when the widget is displayed in 'Scientist' mode. Default is an empty string. The syntax is the standard Qt Style Sheet syntax. For example, 'background-color: red' This Style Sheet string will be applied by the styleManager class. Refer to the styleManager class for details about how this Style Sheet string will be merged with any pre-existing Style Sheet string and any Style Sheet strings generated during the display of data.

9.71.3.10 QString QELog::userLevelUserStyle [read, write]

Style Sheet string to be applied when the widget is displayed in 'User' mode. Default is an empty string. The syntax is the standard Qt Style Sheet syntax. For example, 'background-color: red' This Style Sheet string will be applied by the styleManager class. Refer to the styleManager class for details about how this Style Sheet string will be merged with any pre-existing Style Sheet string and any Style Sheet strings generated during the display of data.

9.71.3.11 UserLevels QELog::userLevelVisibility [read, write]

Lowest user level at which the widget is visible. Default is 'User'. Used when designing GUIs that display more and more detail according to the user mode. The user mode is set application wide through the [QELogin](#) widget, or programmaticaly through setUserLevel() Widgets that are always visible should be visible at 'User'. Widgets that are only used by scientists managing the facility should be visible at 'Scientist'. Widgets that are only used by engineers maintaining the facility should be visible at 'Engineer'.

9.71.3.12 bool QELog::variableAsToolTip [read, write]

Use the variable as the tool tip. Default is true. Tool tip property will be overwritten by the variable name.

9.71.3.13 bool QELog::visible [read, write]

Display the widget. Default is true. Setting this property false is usefull if widget is only used to provide a signal - for example, when supplying data to a [QELink](#) widget. Note, when false the widget will still be visible in Qt Designer.

The documentation for this class was generated from the following files:

- /tmp/epicsqt/trunk/framework/widgets/QELog/QELog.h
- /tmp/epicsqt/trunk/framework/widgets/QELog/QELog.cpp

9.72 QELogin Class Reference

Signals

- void **login ()**

Public Member Functions

- **QELogin** (QWidget *pParent=0)
- bool **login** (userLevelTypes::userLevels level, QString password)
- QString **getPriorityUserPassword ()**

- `QString getPriorityScientistPassword ()`
- `QString getPriorityEngineerPassword ()`
- `void setUserPassword (QString pValue)`
- `QString getUserPassword ()`
- `void setScientistPassword (QString pValue)`
- `QString getScientistPassword ()`
- `void setEngineerPassword (QString pValue)`
- `QString getEngineerPassword ()`
- `void setCompactStyle (bool compactStyle)`
- `bool getCompactStyle ()`
- `void setStatusOnly (bool statusOnlyIn)`
- `bool getStatusOnly ()`
- `QString getUserTypeName (userLevelTypes::userLevels type)`

The documentation for this class was generated from the following files:

- `/tmp/epicsqt/trunk/framework/widgets/QELogin/QELogin.h`
- `/tmp/epicsqt/trunk/framework/widgets/QELogin/QELogin.cpp`

9.73 QELoginDialog Class Reference

Public Member Functions

- `QELoginDialog (QELogin *ownerIn)`

The documentation for this class was generated from the following files:

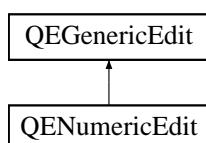
- `/tmp/epicsqt/trunk/framework/widgets/QELogin/QELogin.h`
- `/tmp/epicsqt/trunk/framework/widgets/QELogin/QELogin.cpp`

9.74 QENumericEdit Class Reference

The `QENumericEdit` class This class is similar to `QLineEdit` (both of which are derived from `QLineEdit`). However this class is tailored specifically for editing numerical values.

```
#include <QENumericEdit.h>
```

Inheritance diagram for QENumericEdit:



Signals

- void **dbValueChanged** (const double &out)

Public Member Functions

- **QENumericEdit** (QWidget *parent=0)
- **QENumericEdit** (const QString &variableName, QWidget *parent=0)
- virtual ~**QENumericEdit** ()

Destruction.
- double **getNumericValue** ()
- void **setNumericValue** (const double value, const bool isUserUpdate=false)
- 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 * **createQcaItem** (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**

9.74.1 Detailed Description

The **QENumericEdit** class This class is similar to **QLineEdit** (both of which are derived from **QLineEdit**). However this class is tailored specifically for editing numerical values.

Note: this class based on thumb_wheel_edits.pas by same author.

9.74.2 Constructor & Destructor Documentation

9.74.2.1 QENumericEdit::QENumericEdit (QWidget * *parent* = 0)

Create without a variable. Use **setVariableNameProperty()** and **setSubstitutionsProperty()** to define a variable and, optionally, macro substitutions later.

9.74.2.2 QENumericEdit::QENumericEdit (const QString & *variableName*, QWidget * *parent* = 0)

Create with a variable. A connection is automatically established. If macro substitutions are required, create without a variable and set the variable and macro substitutions after creation.

9.74.3 Member Function Documentation

9.74.3.1 void QENumericEdit::dbValueChanged (const double & *out*) [signal]

Sent when the widget is updated following a data change. Can be used to pass on EPICS data (as presented in this widget) to other widgets. For example a QList widget could log updates from this widget.

9.74.4 Property Documentation

9.74.4.1 bool QENumericEdit::addUnits [read, write]

If true (default), add engineering units supplied with the data.

9.74.4.2 bool QENumericEdit::autoScale [read, write]

If true (default), display and editing of numbers using the precision, and control limits supplied with the data. If false, the precision, leadingZeros, minimum and maximum properties are used.

9.74.4.3 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.

9.74.4.4 double QENumericEdit::maximum [read, write]

Specifies the maximum allowed value. This is only used if autoScale is false.

9.74.4.5 double QENumericEdit::minimum [read, write]

Specifies the mimimum allowed value. This is only used if autoScale is false.

9.74.4.6 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

9.75 QENumericEditManager Class Reference

Public Member Functions

- **QENumericEditManager** (QObject *parent=0)
- bool **isContainer** () const
- bool **isInitialized** () const
- QIcon **icon** () const
- QString **group** () const
- QString **includeFile** () const
- QString **name** () const
- QString **toolTip** () const
- QString **whatsThis** () const
- QWidget * **createWidget** (QWidget *parent)
- void **initialize** (QDesignerFormEditorInterface *core)

The documentation for this class was generated from the following files:

- /tmp/epicsqt/trunk/framework/widgets/QELineEdit/QENumericEditManager.h
- /tmp/epicsqt/trunk/framework/widgets/QELineEdit/QENumericEditManager.cpp

9.76 QEPeriodic Class Reference

Classes

- struct [elementInfoStruct](#)
- struct [userInfoStructArray](#)

Public Types

- enum **variableTypes** {
 VARIABLE_TYPE_NUMBER, **VARIABLE_TYPE_ATOMIC_WEIGHT**, **VARIABLE_TYPE_MELTING_POINT**, **VARIABLE_TYPE_BOILING_POINT**,
 VARIABLE_TYPE_DENSITY, **VARIABLE_TYPE_GROUP**, **VARIABLE_TYPE_IONIZATION_ENERGY**, **VARIABLE_TYPE_USER_VALUE_1**,
 VARIABLE_TYPE_USER_VALUE_2 }
- enum **presentationOptions** { **PRESENTATION_BUTTON_AND_LABEL**, **PRESENTATION_BUTTON_ONLY**, **PRESENTATION_LABEL_ONLY** }
- enum **UserLevels** { **User** = userLevelTypes::USERLEVEL_USER, **Scientist** = userLevelTypes::USERLEVEL_SCIENTIST, **Engineer** = userLevelTypes::USERLEVEL_ENGINEER }
- enum **DisplayAlarmStateOptions** { **Never** = standardProperties::DISPLAY_ALARM_STATE_NEVER, **Always** = standardProperties::DISPLAY_ALARM_STATE_ALWAYS, **WhenInAlarm** = standardProperties::DISPLAY_ALARM_STATE_WHEN_IN_ALARM }

```

• enum PresentationOptions { buttonAndLabel = QEPeriodic::PRESENTATION_-
  BUTTON_AND_LABEL, buttonOnly = QEPeriodic::PRESENTATION_BUTTON_-
  ONLY, labelOnly = QEPeriodic::PRESENTATION_LABEL_ONLY }

• enum VariableTypes {

  Number = QEPeriodic::VARIABLE_TYPE_NUMBER, atomicWeight = QEPeriodic::VARIABLE_-
  TYPE_ATOMIC_WEIGHT, meltingPoint = QEPeriodic::VARIABLE_TYPE_MELTING_-
  POINT, boilingPoint = QEPeriodic::VARIABLE_TYPE_BOILING_POINT,
  density = QEPeriodic::VARIABLE_TYPE_DENSITY, group = QEPeriodic::VARIABLE_-
  TYPE_GROUP, ionizationEnergy = QEPeriodic::VARIABLE_TYPE_IONIZATION_-
  ENERGY, userValue1 = QEPeriodic::VARIABLE_TYPE_USER_VALUE_1,
  userValue2 = QEPeriodic::VARIABLE_TYPE_USER_VALUE_2 }

```

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**

9.76.1 Member Enumeration Documentation

9.76.1.1 enum QEPeriodic::DisplayAlarmStateOptions

User friendly enumerations for **displayAlarmStateOption** property - refer to **displayAlarmStateOption** property and **displayAlarmStateOptions** enumeration for details.

Enumerator:

Never Refer to DISPLAY_ALARM_STATE_NEVER for details.

Always Refer to DISPLAY_ALARM_STATE_ALWAYS for details.

WhenInAlarm Refer to DISPLAY_ALARM_STATE_WHEN_IN_ALARM for details.

9.76.1.2 enum QEPeriodic::UserLevels

User friendly enumerations for `userLevelVisibility` and `userLevelEnabled` properties - refer to `userLevelVisibility` and `userLevelEnabled` properties and userLevel enumeration for details.

Enumerator:

User Refer to USERLEVEL_USER for details.

Scientist Refer to USERLEVEL_SCIENTIST for details.

Engineer Refer to USERLEVEL_ENGINEER for details.

9.76.2 Member Function Documentation

9.76.2.1 void QEPeriodic::dbElementChanged (const QString & out) [signal]

Sent when the widget is updated following a data change Can be used to pass on EPICS data (as presented in this widget) to other widgets. For example a QList widget could log updates from this widget.

9.76.2.2 void QEPeriodic::dbValueChanged (const double & out) [signal]

Sent when the widget is updated following a data change Can be used to pass on EPICS data (as presented in this widget) to other widgets. For example a QList widget could log updates from this widget.

9.76.3 Member Data Documentation

9.76.3.1 bool QEPeriodic::allowDrop [read, write, protected]

Allow drag/drops operations to this widget. Default is false. Any dropped text will be used as a new variable name.

9.76.4 Property Documentation

9.76.4.1 bool QEPeriodic::displayAlarmState [read, write]

DEPRECATED. USE `displayAlarmStateOption` INSTEAD. If set (default) widget will indicate the alarm state of any variable data it is displaying. If clear widget will never indicate the alarm state of any variable data it is displaying. Typically the background colour is set to indicate the alarm state. Note, this property is included in the set of standard properties as it applies to most widgets. It will do nothing for widgets that don't display data.

9.76.4.2 DisplayAlarmStateOptions `QEPeriodic::displayAlarmStateOption` [read, write]

If 'Always' (default) widget will indicate the alarm state of any variable data it is displaying, including 'No Alarm'. If 'Never' widget will never indicate the alarm state of any variable data it is displaying. If 'WhenInAlarm' widget only indicate the alarm state of any variable data it is displaying if it is 'in alarm'. Typically the background colour is set to indicate the alarm state. Note, this property is included in the set of standard properties as it applies to most widgets. It will do nothing for widgets that don't display data.

9.76.4.3 unsigned `QEPeriodic::int` [read, write]

Set the ID used by the message filtering system. Default is zero. Widgets or applications that use messages from the framework have the option of filtering on this ID. For example, by using a unique message source ID a [QELog](#) widget may be set up to only log messages from a select set of widgets.

9.76.4.4 QString `QEPeriodic::readbackLabelVariable1` [read, write]

EPICS variable name (CA PV). This variable is used to read the value to the first of two positioners to determine which (if any) element is currently selected.

9.76.4.5 QString `QEPeriodic::readbackLabelVariable2` [read, write]

EPICS variable name (CA PV). This variable is used to read the value to the second of two positioners to determine which (if any) element is currently selected.

9.76.4.6 bool `QEPeriodic::subscribe` [read, write]

Sets if this widget subscribes for data updates and displays current data. Default is 'true' (subscribes for and displays data updates)

9.76.4.7 UserLevels `QEPeriodic::userLevelEnabled` [read, write]

Lowest user level at which the widget is enabled. Default is 'User'. Used when designing GUIs that allow access to more and more detail according to the user mode. The user mode is set application wide through the [QELogin](#) widget, or programmatically through `setUserLevel()`. Widgets that are always accessible should be visible at 'User'. Widgets that are only accessible to scientists managing the facility should be visible at 'Scientist'. Widgets that are only accessible to engineers maintaining the facility should be visible at 'Engineer'.

9.76.4.8 QString QEPeriodic::userLevelEngineerStyle [read, write]

Style Sheet string to be applied when the widget is displayed in 'Engineer' mode. Default is an empty string. The syntax is the standard Qt Style Sheet syntax. For example, 'background-color: red' This Style Sheet string will be applied by the styleManager class. Refer to the styleManager class for details about how this Style Sheet string will be merged with any pre-existing Style Sheet string and any Style Sheet strings generated during the display of data.

9.76.4.9 QString QEPeriodic::userLevelScientistStyle [read, write]

Style Sheet string to be applied when the widget is displayed in 'Scientist' mode. Default is an empty string. The syntax is the standard Qt Style Sheet syntax. For example, 'background-color: red' This Style Sheet string will be applied by the styleManager class. Refer to the styleManager class for details about how this Style Sheet string will be merged with any pre-existing Style Sheet string and any Style Sheet strings generated during the display of data.

9.76.4.10 QString QEPeriodic::userLevelUserStyle [read, write]

Style Sheet string to be applied when the widget is displayed in 'User' mode. Default is an empty string. The syntax is the standard Qt Style Sheet syntax. For example, 'background-color: red' This Style Sheet string will be applied by the styleManager class. Refer to the styleManager class for details about how this Style Sheet string will be merged with any pre-existing Style Sheet string and any Style Sheet strings generated during the display of data.

9.76.4.11 UserLevels QEPeriodic::userLevelVisibility [read, write]

Lowest user level at which the widget is visible. Default is 'User'. Used when designing GUIs that display more and more detail according to the user mode. The user mode is set application wide through the [QELogin](#) widget, or programmatically through setUserLevel() Widgets that are always visible should be visible at 'User'. Widgets that are only used by scientists managing the facility should be visible at 'Scientist'. Widgets that are only used by engineers maintaining the facility should be visible at 'Engineer'.

9.76.4.12 bool QEPeriodic::variableAsToolTip [read, write]

Use the variable as the tool tip. Default is true. Tool tip property will be overwritten by the variable name.

9.76.4.13 QString QEPeriodic::variableSubstitutions [read, write]

Macro substitutions. The default is no substitutions. The format is NAME1=VALUE1[,] NAME2=VALUE2... Values may be quoted strings. For example, 'SAMPLE=SAM1, NAME = "Ref foil"' These substitutions are applied to all the variable names.

9.76.4.14 bool QEPeriodic::visible [read, write]

Display the widget. Default is true. Setting this property false is useful if widget is only used to provide a signal - for example, when supplying data to a [QELink](#) widget. Note, when false the widget will still be visible in Qt Designer.

9.76.4.15 QString QEPeriodic::writeButtonVariable1 [read, write]

EPICS variable name (CA PV). This variable is used to write a value to the first of two positioners that will position the select element.

9.76.4.16 QString QEPeriodic::writeButtonVariable2 [read, write]

EPICS variable name (CA PV). This variable is used to write a value to the second of two positioners that will position the select element.

The documentation for this class was generated from the following files:

- /tmp/epicsqt/trunk/framework/widgets/QEPeriodic/QEPeriodic.h
- /tmp/epicsqt/trunk/framework/widgets/QEPeriodic/QEPeriodic.cpp

9.77 QEPeriodicComponentData Class Reference

Public Attributes

- unsigned int **variableIndex1**
- double **lastData1**
- bool **haveLastData1**
- unsigned int **variableIndex2**
- double **lastData2**
- bool **haveLastData2**

The documentation for this class was generated from the following file:

- /tmp/epicsqt/trunk/framework/widgets/QEPeriodic/QEPeriodic.h

9.78 QEPeriodicTaskMenu Class Reference

Public Member Functions

- **QEPeriodicTaskMenu** ([QEPeriodic](#) *periodic, [QObject](#) *parent)
- [QAction](#) * **preferredEditAction** () const
- [QList< QAction * >](#) **taskActions** () const

The documentation for this class was generated from the following files:

- /tmp/epicsqt/trunk/framework/widgets/QEPeriodic/QEPeriodicTaskMenu.h
- /tmp/epicsqt/trunk/framework/widgets/QEPeriodic/QEPeriodicTaskMenuExtension.cpp

9.79 QEPeriodicTaskMenuFactory Class Reference

Public Member Functions

- **QEPeriodicTaskMenuFactory** (QExtensionManager *parent=0)

Protected Member Functions

- QObject * **createExtension** (QObject *object, const QString &iid, QObject *parent)
const

The documentation for this class was generated from the following files:

- /tmp/epicsqt/trunk/framework/widgets/QEPeriodic/QEPeriodicTaskMenu.h
- /tmp/epicsqt/trunk/framework/widgets/QEPeriodic/QEPeriodicTaskMenuExtension.cpp

9.80 QEPlot Class Reference

Public Types

- enum **UserLevels** { **User** = userLevelTypes::USERLEVEL_USER, **Scientist** = userLevelTypes::USERLEVEL_SCIENTIST, **Engineer** = userLevelTypes::USERLEVEL_ENGINEER }
- enum **DisplayAlarmStateOptions** { **Never** = standardProperties::DISPLAY_ALARM_STATE_NEVER, **Always** = standardProperties::DISPLAY_ALARM_STATE_ALWAYS, **WhenInAlarm** = standardProperties::DISPLAY_ALARM_STATE_WHEN_IN_ALARM }
- enum **TraceStyles** { **Lines** = QwtPlotCurve::Lines, **Sticks** = QwtPlotCurve::Sticks, **Steps** = QwtPlotCurve::Steps, **Dots** = QwtPlotCurve::Dots }

Public Slots

- void **setManagedVisible** (bool v)

Signals

- void **dbValueChanged** (const double &out)
- void **dbValueChanged** (const QVector< double > &out)

Public Member Functions

- **QEPlot** (QWidget *parent=0)
- **QEPlot** (const QString &variableName, QWidget *parent=0)
- QSize **sizeHint** () const
- void **setYMin** (double yMin)
- double **getYMin** ()
- void **setYMax** (double yMax)
- double **getYMax** ()
- void **setAutoScale** (bool autoScale)
- bool **getAutoScale** ()
- void **setAxisEnableX** (bool axisEnableXIn)
- bool **getAxisEnableX** ()
- void **setAxisEnableY** (bool axisEnableYIn)
- bool **getAxisEnableY** ()
- QString **getTitle** ()
- void **setBackgroundColor** (QColor backgroundColor)
- QColor **getBackgroundColor** ()
- void **setTraceStyle** (QwtPlotCurve::CurveStyle traceStyle, const unsigned int variableIndex)
- QwtPlotCurve::CurveStyle **getTraceStyle** (const unsigned int variableIndex)
- void **setTraceColor** (QColor traceColor, const unsigned int variableIndex)
- void **setTraceColor1** (QColor traceColor)
- void **setTraceColor2** (QColor traceColor)
- void **setTraceColor3** (QColor traceColor)
- void **setTraceColor4** (QColor traceColor)
- QColor **getTraceColor** (const unsigned int variableIndex)
- QColor **getTraceColor1** ()
- QColor **getTraceColor2** ()
- QColor **getTraceColor3** ()
- QColor **getTraceColor4** ()
- void **setTraceLegend1** (QString traceLegend)
- void **setTraceLegend2** (QString traceLegend)
- void **setTraceLegend3** (QString traceLegend)
- void **setTraceLegend4** (QString traceLegend)
- QString **getTraceLegend1** ()
- QString **getTraceLegend2** ()
- QString **getTraceLegend3** ()
- QString **getTraceLegend4** ()
- void **setXUnit** (QString xUnit)
- QString **getXUnit** ()
- void **setYUnit** (QString yUnit)
- QString **getYUnit** ()
- void **setGridEnableMajorX** (bool gridEnableMajorXIn)
- void **setGridEnableMajorY** (bool gridEnableMajorYIn)
- void **setGridEnableMinorX** (bool gridEnableMinorXIn)
- void **setGridEnableMinorY** (bool gridEnableMinorYIn)

- bool **getGridEnableMajorX** ()
- bool **getGridEnableMajorY** ()
- bool **getGridEnableMinorX** ()
- bool **getGridEnableMinorY** ()
- void **setGridMajorColor** (QColor gridMajorColorIn)
- void **setGridMinorColor** (QColor gridMinorColorIn)
- QColor **getGridMajorColor** ()
- QColor **getGridMinorColor** ()
- void **setXStart** (double xStart)
- double **getXStart** ()
- void **setXIncrement** (double xIncrement)
- double **getXIncrement** ()
- void **setTimeSpan** (unsigned int timeSpan)
- unsigned int **getTimeSpan** ()
- void **setTickRate** (unsigned int tickRate)
- unsigned int **getTickRate** ()
- void **setVariableNameSubstitutionsProperty** (QString variableNameSubstitutions)

Property access function for `variableSubstitutions` property. This has special behaviour to work well within designer.

- QString **getVariableNameSubstitutionsProperty** ()
Property access function for `variableSubstitutions` property. This has special behaviour to work well within designer.
- UserLevels **getUserLevelVisibilityProperty** ()
Access function for `userLevelVisibility` property - refer to `userLevelVisibility` property for details.
- void **setUserLevelVisibilityProperty** (UserLevels level)
Access function for `userLevelVisibility` property - refer to `userLevelVisibility` property for details.
- UserLevels **getUserLevelEnabledProperty** ()
Access function for `userLevelEnabled` property - refer to `userLevelEnabled` property for details.
- void **setUserLevelEnabledProperty** (UserLevels level)
Access function for `userLevelEnabled` property - refer to `userLevelEnabled` property for details.
- DisplayAlarmStateOptions **getDisplayAlarmStateOptionProperty** ()
Access function for `displayAlarmStateOption` property - refer to `displayAlarmStateOption` property for details.
- void **setDisplayAlarmStateOptionProperty** (DisplayAlarmStateOptions option)
Access function for `displayAlarmStateOption` property - refer to `displayAlarmStateOption` property for details.
- void **setTraceStyle1** (TraceStyles traceStyle)
- void **setTraceStyle2** (TraceStyles traceStyle)
- void **setTraceStyle3** (TraceStyles traceStyle)
- void **setTraceStyle4** (TraceStyles traceStyle)
- TraceStyles **getTraceStyle1** ()
- TraceStyles **getTraceStyle2** ()
- TraceStyles **getTraceStyle3** ()
- TraceStyles **getTraceStyle4** ()

Protected Member Functions

- void **establishConnection** (unsigned int variableIndex)
- void **dragEnterEvent** (QDragEnterEvent *event)
- void **dropEvent** (QDropEvent *event)
- void **mousePressEvent** (QMouseEvent *event)
- void **setDrop** (QVariant drop)
- QVariant **getDrop** ()
- QString **copyVariable** ()
- QVariant **copyData** ()
- void **paste** (QVariant s)

Protected Attributes

- QEFloatingFormatting **floatingFormatting**
- bool **localEnabled**
- bool **allowDrop**

Properties

- QString **variable1**
- QString **variable2**
- QString **variable3**
- QString **variable4**
- QString **variableSubstitutions**
- bool **variableAsToolTip**
- bool **visible**
- unsigned **int**
- QString **styleSheet**
- QString **defaultStyle**
- QString **userLevelUserStyle**
- QString **userLevelScientistStyle**
- QString **userLevelEngineerStyle**
- **UserLevels userLevelVisibility**
- **UserLevels userLevelEnabled**
- bool **displayAlarmState**
- **DisplayAlarmStateOptions displayAlarmStateOption**
- QColor **traceColor1**
- QColor **traceColor2**
- QColor **traceColor3**
- QColor **traceColor4**
- TraceStyles **traceStyle1**
- TraceStyles **traceStyle2**
- TraceStyles **traceStyle3**
- TraceStyles **traceStyle4**
- QString **traceLegend1**

- `QString traceLegend2`
- `QString traceLegend3`
- `QString traceLegend4`
- `QString title`
- `QColor backgroundColor`
- `QString xUnit`
- `QString yUnit`

9.80.1 Member Enumeration Documentation

9.80.1.1 enum QEPlot::DisplayAlarmStateOptions

User friendly enumerations for `displayAlarmStateOption` property - refer to `displayAlarmStateOption` property and `displayAlarmStateOptions` enumeration for details.

Enumerator:

- Never** Refer to `DISPLAY_ALARM_STATE_NEVER` for details.
Always Refer to `DISPLAY_ALARM_STATE_ALWAYS` for details.
WhenInAlarm Refer to `DISPLAY_ALARM_STATE_WHEN_IN_ALARM` for details.

9.80.1.2 enum QEPlot::UserLevels

User friendly enumerations for `userLevelVisibility` and `userLevelEnabled` properties - refer to `userLevelVisibility` and `userLevelEnabled` properties and `userLevel` enumeration for details.

Enumerator:

- User** Refer to `USERLEVEL_USER` for details.
Scientist Refer to `USERLEVEL_SCIENTIST` for details.
Engineer Refer to `USERLEVEL_ENGINEER` for details.

9.80.2 Member Function Documentation

9.80.2.1 void QEPlot::dbValueChanged (const double & out) [signal]

Sent when the widget is updated following a data change Can be used to pass on EPICS data (as presented in this widget) to other widgets. For example a QList widget could log updates from this widget.

9.80.2.2 void QEPlot::dbValueChanged (const QVector< double > & out) [signal]

Sent when the widget is updated following a data change Can be used to pass on EPICS data (as presented in this widget) to other widgets. For example a QList widget could log updates from this widget.

9.80.2.3 void QEPlot::setManagedVisible (bool v) [inline, slot]

Slot to set the visibility of a QE widget, taking into account the user level. Widget will be hidden if hidden by a call this slot, by will only be made visible by a call to this slot if the user level allows.

9.80.3 Member Data Documentation**9.80.3.1 bool QEPlot::allowDrop [read, write, protected]**

Allow drag/drops operations to this widget. Default is false. Any dropped text will be used as a new variable name.

9.80.4 Property Documentation**9.80.4.1 QString QEPlot::defaultStyle [read, write]**

Style Sheet string to be applied before, i.e. lower priority than, any other style, e.g. alarm style and/or user level style. Default is an empty string.

9.80.4.2 bool QEPlot::displayAlarmState [read, write]

DEPRECATED. USE displayAlarmStateOption INSTEAD. If set (default) widget will indicate the alarm state of any variable data it is displaying. If clear widget will never indicate the alarm state of any variable data it is displaying. Typically the background colour is set to indicate the alarm state. Note, this property is included in the set of standard properties as it applies to most widgets. It will do nothing for widgets that don't display data.

9.80.4.3 DisplayAlarmStateOptions QEPlot::displayAlarmStateOption [read, write]

If 'Always' (default) widget will indicate the alarm state of any variable data it is displaying, including 'No Alarm'. If 'Never' widget will never indicate the alarm state of any variable data it is displaying. If 'WhenInAlarm' widget only indicate the alarm state of any variable data it is displaying if it is 'in alarm'. Typically the background colour is set to indicate the alarm state. Note, this property is included in the set of standard properties as it applies to most widgets. It will do nothing for widgets that don't display data.

9.80.4.4 unsigned QEPlot::int [read, write]

Set the ID used by the message filtering system. Default is zero. Widgets or applications that use messages from the framework have the option of filtering on this ID. For

example, by using a unique message source ID a [QELog](#) widget may be set up to only log messages from a select set of widgets.

9.80.4.5 `QString QEPlot::styleSheet [read, write]`

Hide style sheet from designer as style calculation by the styleManager and not directly setable per se. This also stops transient styles being saved to the ui file.

9.80.4.6 `UserLevels QEPlot::userLevelEnabled [read, write]`

Lowest user level at which the widget is enabled. Default is 'User'. Used when designing GUIs that allow access to more and more detail according to the user mode. The user mode is set application wide through the [QELogin](#) widget, or programmatically through `setUserLevel()`. Widgets that are always accessible should be visible at 'User'. Widgets that are only accessible to scientists managing the facility should be visible at 'Scientist'. Widgets that are only accessible to engineers maintaining the facility should be visible at 'Engineer'.

9.80.4.7 `QString QEPlot::userLevelEngineerStyle [read, write]`

Style Sheet string to be applied when the widget is displayed in 'Engineer' mode. Default is an empty string. The syntax is the standard Qt Style Sheet syntax. For example, 'background-color: red' This Style Sheet string will be applied by the styleManager class. Refer to the styleManager class for details about how this Style Sheet string will be merged with any pre-existing Style Sheet string and any Style Sheet strings generated during the display of data.

9.80.4.8 `QString QEPlot::userLevelScientistStyle [read, write]`

Style Sheet string to be applied when the widget is displayed in 'Scientist' mode. Default is an empty string. The syntax is the standard Qt Style Sheet syntax. For example, 'background-color: red' This Style Sheet string will be applied by the styleManager class. Refer to the styleManager class for details about how this Style Sheet string will be merged with any pre-existing Style Sheet string and any Style Sheet strings generated during the display of data.

9.80.4.9 `QString QEPlot::userLevelUserStyle [read, write]`

Style Sheet string to be applied when the widget is displayed in 'User' mode. Default is an empty string. The syntax is the standard Qt Style Sheet syntax. For example, 'background-color: red' This Style Sheet string will be applied by the styleManager class. Refer to the styleManager class for details about how this Style Sheet string will be merged with any pre-existing Style Sheet string and any Style Sheet strings generated during the display of data.

9.80.4.10 UserLevels QEPlot::userLevelVisibility [read, write]

Lowest user level at which the widget is visible. Default is 'User'. Used when designing GUIs that display more and more detail according to the user mode. The user mode is set application wide through the [QELogin](#) widget, or programmatically through setUserLevel() Widgets that are always visible should be visible at 'User'. Widgets that are only used by scientists managing the facility should be visible at 'Scientist'. Widgets that are only used by engineers maintaining the facility should be visible at 'Engineer'.

9.80.4.11 QString QEPlot::variable1 [read, write]

EPICS variable name (CA PV). This variable is used to read updating values or waveforms for plotting in the first trace.

9.80.4.12 QString QEPlot::variable2 [read, write]

EPICS variable name (CA PV). This variable is used to read updating values or waveforms for plotting in the second trace.

9.80.4.13 QString QEPlot::variable3 [read, write]

EPICS variable name (CA PV). This variable is used to read updating values or waveforms for plotting in the third trace.

9.80.4.14 QString QEPlot::variable4 [read, write]

EPICS variable name (CA PV). This variable is used to read updating values or waveforms for plotting in the fourth trace.

9.80.4.15 bool QEPlot::variableAsToolTip [read, write]

Use the variable as the tool tip. Default is true. Tool tip property will be overwritten by the variable name.

9.80.4.16 QString QEPlot::variableSubstitutions [read, write]

Macro substitutions. The default is no substitutions. The format is NAME1=VALUE1[,] NAME2=VALUE2... Values may be quoted strings. For example, 'SAMPLE=SAM1, NAME = "Ref foil"' These substitutions are applied to all the variable names.

9.80.4.17 bool QEPlot::visible [read, write]

Display the widget. Default is true. Setting this property false is useful if widget is only used to provide a signal - for example, when supplying data to a [QELink](#) widget. Note,

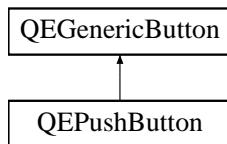
when false the widget will still be visible in Qt Designer.

The documentation for this class was generated from the following files:

- /tmp/epicsqt/trunk/framework/widgets/QEPlot/QEPlot.h
- /tmp/epicsqt/trunk/framework/widgets/QEPlot/QEPlot.cpp

9.81 QEPushButton Class Reference

Inheritance diagram for QEPushButton:



Public Types

- enum `UserLevels` { `User` = userLevelTypes::USERLEVEL_USER, `Scientist` = userLevelTypes::USERLEVEL_SCIENTIST, `Engineer` = userLevelTypes::USERLEVEL_ENGINEER }
 - enum `DisplayAlarmStateOptions` { `Never` = standardProperties::DISPLAY_ALARM_STATE_NEVER, `Always` = standardProperties::DISPLAY_ALARM_STATE_ALWAYS, `WhenInAlarm` = standardProperties::DISPLAY_ALARM_STATE_WHEN_IN_ALARM }
 - enum `Formats` {
 `Default` = QQStringFormatting::FORMAT_DEFAULT, `Floating` = QQStringFormatting::FORMAT_FLOATING, `Integer` = QQStringFormatting::FORMAT_INTEGER, `UnsignedInteger` = QQStringFormatting::FORMAT_UNSIGNEDINTEGER,
 `Time` = QQStringFormatting::FORMAT_TIME, `LocalEnumeration` = QQStringFormatting::FORMAT_LOCAL_ENUMERATE }
 - enum `Notations` { `Fixed` = QQStringFormatting::NOTATION_FIXED, `Scientific` = QQStringFormatting::NOTATION_SCIENTIFIC, `Automatic` = QQStringFormatting::NOTATION_AUTOMATIC }
 - enum `ArrayActions` { `Append` = QQStringFormatting::APPEND, `Ascii` = QQStringFormatting::ASCII, `Index` = QQStringFormatting::INDEX }
 - enum `UpdateOptions` { `Text` = QEGenericButton::UPDATE_TEXT, `Icon` = QEGenericButton::UPDATE_ICON, `TextAndIcon` = QEGenericButton::UPDATE_TEXT_AND_ICON, `State` = QEGenericButton::UPDATE_STATE }
- User friendly enumerations for updateOption property - refer to QEGenericButton::updateOptions for details.*
- enum `ProgramStartupOptionNames` { `None` = applicationLauncher::PSO_NONE, `Terminal` = applicationLauncher::PSO_TERMINAL, `LogOutput` = applicationLauncher::PSO_LOGOUTPUT, `StdOutput` = applicationLauncher::PSO_STDOUTPUT }

- enum `CreationOptionNames` {

`Open` = QEActionRequests::OptionOpen, `NewTab` = QEActionRequests::OptionNewTab,

`NewWindow` = QEActionRequests::OptionNewWindow, `DockTop` = QEActionRequests::OptionTopDockWindow,

`DockBottom` = QEActionRequests::OptionBottomDockWindow, `DockLeft` = QEActionRequests::OptionLeftDockWindow, `DockRight` = QEActionRequests::OptionRightDockWindow,

`DockTopTabbed` = QEActionRequests::OptionTopDockWindowTabbed,

`DockBottomTabbed` = QEActionRequests::OptionBottomDockWindowTabbed, `DockLeftTabbed` = QEActionRequests::OptionLeftDockWindowTabbed, `DockRightTabbed` = QEActionRequests::OptionRightDockWindowTabbed, `DockFloating` = QEActionRequests::OptionFloatingDockWindow }

Creation options. Used to indicate how to present a GUI when requesting a new GUI be created. Open a new window, open a new tab, or replace the current window.

Public Slots

- void `requestAction` (const QEActionRequests &request)
- void `setDefaultStyle` (const QString &style)

Update the default style applied to this widget.
- void `setManagedVisible` (bool v)

Signals

- void `dbValueChanged` (const QString &out)
- void `requestResend` ()

Internal use only. Used when changing a property value to force a re-display to reflect the new property value.
- void `newGui` (const QEActionRequests &request)

Internal use only. Request a new GUI is created. Typically, this is caught by the QEGui application.
- void `pressed` (int value)
- void `released` (int value)
- void `clicked` (int value)
- void `programCompleted` ()

Program started by button has completed.

Public Member Functions

- `QEPushButton` (QWidget *parent=0)
- `QEPushButton` (const QString &variableName, QWidget *parent=0)
- void `writeNow` ()
- void `setVariableNameSubstitutionsProperty` (QString variableNameSubstitutions)

Property access function for `variableSubstitutions` property. This has special behaviour to work well within designer.

- **QString getVariableNameSubstitutionsProperty ()**
Property access function for `variableSubstitutions` property. This has special behaviour to work well within designer.
- **UserLevels getUserLevelVisibilityProperty ()**
Access function for `userLevelVisibility` property - refer to `userLevelVisibility` property for details.
- **void setUserLevelVisibilityProperty (UserLevels level)**
Access function for `userLevelVisibility` property - refer to `userLevelVisibility` property for details.
- **UserLevels getUserLevelEnabledProperty ()**
Access function for `userLevelEnabled` property - refer to `userLevelEnabled` property for details.
- **void setUserLevelEnabledProperty (UserLevels level)**
Access function for `userLevelEnabled` property - refer to `userLevelEnabled` property for details.
- **DisplayAlarmStateOptions getDisplayAlarmStateOptionProperty ()**
Access function for `displayAlarmStateOption` property - refer to `displayAlarmStateOption` property for details.
- **void setDisplayAlarmStateOptionProperty (DisplayAlarmStateOptions option)**
Access function for `displayAlarmStateOption` property - refer to `displayAlarmStateOption` property for details.
- **void setFormatProperty (Formats format)**
Access function for `format` property - refer to `format` property for details.
- **Formats getFormatProperty ()**
Access function for `format` property - refer to `format` property for details.
- **void setNotationProperty (Notations notation)**
Access function for `notation` property - refer to `notation` property for details.
- **Notations getNotationProperty ()**
Access function for `notation` property - refer to `notation` property for details.
- **void setArrayActionProperty (ArrayActions arrayAction)**
Access function for `arrayAction` property - refer to `arrayAction` property for details.
- **ArrayActions getArrayActionProperty ()**
Access function for `arrayAction` property - refer to `arrayAction` property for details.

Properties

- **QString variable**
- **QString altReadbackVariable**
- **QString variableSubstitutions**
- **bool subscribe**
- **bool variableAsToolTip**
- **bool allowDrop**
- **bool visible**
- **unsigned int**
- **QString styleSheet**

- `QString defaultStyle`
- `QString userLevelUserStyle`
- `QString userLevelScientistStyle`
- `QString userLevelEngineerStyle`
- `UserLevels userLevelVisibility`
- `UserLevels userLevelEnabled`
- `bool displayAlarmState`
- `DisplayAlarmStateOptions displayAlarmStateOption`
- `int precision`
- `bool useDbPrecision`
- `bool leadingZero`
- `bool trailingZeros`
- `bool addUnits`
- `QString localEnumeration`
- `Formats format`
- `Notations notation`
- `ArrayActions arrayAction`
- `Qt::Alignment alignment`
- `UpdateOptions updateOption`
- `QPixmap pixmap0`
- `QPixmap pixmap1`
- `QPixmap pixmap2`
- `QPixmap pixmap3`
- `QPixmap pixmap4`
- `QPixmap pixmap5`
- `QPixmap pixmap6`
- `QPixmap pixmap7`
- `QString password`
- `bool confirmAction`
- `QString confirmText`
- `bool writeOnPress`
- `bool writeOnRelease`
- `bool writeOnClick`
- `QString pressText`
- `QString releaseText`
- `QString clickText`
- `QString clickCheckedText`
- `QString labelText`
- `QString program`
- `QStringList arguments`
- `ProgramStartupOptionNames programStartupOption`
- `QString guiFile`
- `CreationOptionNames creationOption`
- `QString prioritySubstitutions`
- `QString customisationName`

9.81.1 Member Enumeration Documentation

9.81.1.1 enum QEPushButton::ArrayActions

User friendly enumerations for arrayAction property - refer to `QStringFormatting::arrayActions` for details.

Enumerator:

Append Refer to `QStringFormatting::APPEND` for details.

Ascii Refer to `QStringFormatting::ASCII` for details.

Index Refer to `QStringFormatting::INDEX` for details.

9.81.1.2 enum QEPushButton::CreationOptionNames

Creation options. Used to indicate how to present a GUI when requesting a new GUI be created. Open a new window, open a new tab, or replace the current window.

Enumerator:

Open Replace the current GUI with the new GUI.

NewTab Open new GUI in a new tab.

NewWindow Open new GUI in a new window.

DockTop Open new GUI in a top dock window.

DockBottom Open new GUI in a bottom dock window.

DockLeft Open new GUI in a left dock window.

DockRight Open new GUI in a right dock window.

DockTopTabbed Open new GUI in a top dock window (tabbed with any existing dock in that area)

DockBottomTabbed Open new GUI in a bottom dock window (tabbed with any existing dock in that area)

DockLeftTabbed Open new GUI in a left dock window (tabbed with any existing dock in that area)

DockRightTabbed Open new GUI in a right dock window (tabbed with any existing dock in that area)

DockFloating Open new GUI in a floating dock window.

9.81.1.3 enum QEPushButton::DisplayAlarmStateOptions

User friendly enumerations for `displayAlarmStateOption` property - refer to `displayAlarmStateOption` property and `displayAlarmStateOptions` enumeration for details.

Enumerator:

Never Refer to `DISPLAY_ALARM_STATE_NEVER` for details.

Always Refer to `DISPLAY_ALARM_STATE_ALWAYS` for details.

WhenInAlarm Refer to `DISPLAY_ALARM_STATE_WHEN_IN_ALARM` for details.

9.81.1.4 enum QEPushButton::Formats

User friendly enumerations for format property - refer to QEStringFormatting::formats for details.

Enumerator:

Default Format as best appropriate for the data type.

Floating Format as a floating point number.

Integer Format as an integer.

UnsignedInteger Format as an unsigned integer.

Time Format as a time.

LocalEnumeration Format as a selection from the [localEnumeration](#) property.

9.81.1.5 enum QEPushButton::Notations

User friendly enumerations for notation property - refer to QEStringFormatting::notations for details.

Enumerator:

Fixed Refer to QEStringFormatting::NOTATION_FIXED for details.

Scientific Refer to QEStringFormatting::NOTATION_SCIENTIFIC for details.

Automatic Refer to QEStringFormatting::NOTATION_AUTOMATIC for details.

9.81.1.6 enum QEPushButton::ProgramStartupOptionNames

Startup options. Just run the command, run the command within a terminal, or display the output in QE message system.

Enumerator:

None Just run the program.

Terminal Run the program in a terminal (in Windows a command interpreter will also be started, so the program may be a built-in command like 'dir')

LogOutput Run the program, and log the output in the QE message system.

StdOutput Run the program, and send output to standard output and standard error.

9.81.1.7 enum QEPushButton::UpdateOptions

User friendly enumerations for updateOption property - refer to QEGenericButton::updateOptions for details.

Enumerator:

- Text** Data updates will update the button text.
- Icon** Data updates will update the button icon.
- TextAndIcon** Data updates will update the button text and icon.
- State** Data updates will update the button state (checked or unchecked)

9.81.1.8 enum QEPushButton::UserLevels

User friendly enumerations for [userLevelVisibility](#) and [userLevelEnabled](#) properties - refer to [userLevelVisibility](#) and [userLevelEnabled](#) properties and userLevel enumeration for details.

Enumerator:

- User** Refer to USERLEVEL_USER for details.
- Scientist** Refer to USERLEVEL_SCIENTIST for details.
- Engineer** Refer to USERLEVEL_ENGINEER for details.

9.81.2 Constructor & Destructor Documentation**9.81.2.1 QEPushButton::QEPushButton (QWidget * *parent* = 0)**

Create without a variable. Use [setVariableNameProperty\(\)](#) and [setSubstitutionsProperty\(\)](#) to define a variable and, optionally, macro substitutions later.

9.81.2.2 QEPushButton::QEPushButton (const QString & *variableName*, QWidget * *parent* = 0)

Create with a variable. A connection is automatically established. If macro substitutions are required, create without a variable and set the variable and macro substitutions after creation.

9.81.3 Member Function Documentation**9.81.3.1 void QEPushButton::clicked (int *value*) [signal]**

Button has been Clicked. The value emitted is the integer interpretation of the [clickText](#) property (or the [clickCheckedText](#) property if the button was checked)

9.81.3.2 void QEPushButton::dbValueChanged (const QString & *out*) [signal]

Sent when the widget is updated following a data change Can be used to pass on EPICS data (as presented in this widget) to other widgets. For example a QList widget could log updates from this widget.

9.81.3.3 void QEPushButton::pressed (int value) [signal]

Button has been Pressed. The value emitted is the integer interpretation of the press-Text property

9.81.3.4 void QEPushButton::released (int value) [signal]

Button has been Released The value emitted is the integer interpretation of the release-Text property

**9.81.3.5 void QEPushButton::requestAction (const QEActionRequests & request)
[inline, slot]**

Default slot used to create a new GUI if there is no slot indicated in the ContainerProfile class. This slot is typically used when the button is pressed within the Designer preview window to allow the operation of the button to be tested. If an application does not specify a slot to use for creating new windows (through the ContainerProfile class) a window will still be created through this slot, but it will not respect the window creation options or any other window related application constraints. For example, the QEGui application does provide a slot for creating new GUIs in the ContainerProfile class which respects the creation options, knows how to add tabs in the application, and extend the application's window menu in the menu bar.

9.81.3.6 void QEPushButton::setManagedVisible (bool v) [inline, slot]

Slot to set the visibility of a QE widget, taking into account the user level. Widget will be hidden if hidden by a call this slot, by will only be made visible by a call to this slot if the user level allows.

9.81.4 Property Documentation

9.81.4.1 bool QEPushButton::addUnits [read, write]

If true (default), add engineering units supplied with the data.

9.81.4.2 Qt::Alignment QEPushButton::alignment [read, write]

Set the buttons text alignment. Left justification is particularly useful when displaying quickly changing numeric data updates.

9.81.4.3 bool QEPushButton::allowDrop [read, write]

Allow drag/drops operations to this widget. Default is false. Any dropped text will be used as a new variable name.

9.81.4.4 QString QEPushButton::altReadbackVariable [read, write]

EPICS variable name (CA PV). This variable is used to provide a readback value when different to the variable written to by a button press.

9.81.4.5 QStringList QEPushButton::arguments [read, write]

Arguments for program specified in the 'program' property.

9.81.4.6 ArrayActions QEPushButton::arrayAction [read, write]

Text formatting option for array data. Default is ASCII. Options are:

- ASCII - treat array as a single text string. For example an array of three characters 'a' 'b' 'c' will be formatted as 'abc'.
- APPEND - treat array as an array of numbers and format a string containing them all with a space between each. For example, an array of three numbers 10, 11 and 12 will be formatted as '10 11 12'.
- INDEX - Extract a single item from the array. The item is then formatted as any other non array data would be. The item selected is determined by the arrayIndex property. For example, if arrayIndex property is 1, an array of three numbers 10, 11 and 12 will be formatted as '11'.

9.81.4.7 QString QEPushButton::clickCheckedText [read, write]

Text used to compare with text written or read to determine if push button should be marked as checked. Note, must be an exact match following formatting of data updates. When writing values, the 'pressText', 'ReleaseText', or 'clickedtext' must match this property to cause the button to be checked when the write occurs.

Good example: formatting set to display a data value of '1' as 'On', clickCheckedText is 'On', clickText is 'On'. In this example, the push button will be checked when a data update occurs with a value of 1 or when the button is clicked.

Bad example: formatting set to display a data value of '1' as 'On', clickCheckedText is 'On', clickText is '1'. In this example, the push button will be checked when a data update occurs with a value of 1 but, although a valid value will be written when clicked, the button will not be checked when clicked as '1' is not the same as 'On'.

Reimplemented from [QEGenericButton](#).

9.81.4.8 QString QEPushButton::clickText [read, write]

Value written when user clicks button if 'writeOnClick' property is true

Reimplemented from [QEGenericButton](#).

9.81.4.9 bool QEPushButton::confirmAction [read, write]

If true, a dialog will be presented asking the user to confirm if the button action should be carried out

9.81.4.10 QString QEPushButton::confirmText [read, write]

Text used to confirm action if confirmation dialog is presented

Reimplemented from [QEGenericButton](#).

9.81.4.11 CreationOptionNames QEPushButton::creationOption [read, write]

Creation options when opening a new GUI. Open a new window, open a new tab, or replace the current window. The creation option is supplied when the button generates a newGui signal. Application code connected to this signal should honour this request if possible. When used within the QEGui application, the QEGui application creates a new window, new tab, or replaces the current window as appropriate.

Reimplemented from [QEGenericButton](#).

9.81.4.12 QString QEPushButton::customisationName [read, write]

Window customisation name. This name will be used to select a set of window customisations including menu items and tool bar buttons. Applications such as QEGui can load .xml files containing named sets of window customisations. This property is used to select a set loaded from these files. The selected set of customisations will be applied to the main window containing the new GUI. Customisations are not applied if the GUI is opened as a dock.

Reimplemented from [QEGenericButton](#).

9.81.4.13 QString QEPushButton::defaultStyle [read, write]

Style Sheet string to be applied before, i.e. lower priority than, any other style, e.g. alarm style and/or user level style. Default is an empty string.

9.81.4.14 bool QEPushButton::displayAlarmState [read, write]

DEPRECATED. USE displayAlarmStateOption INSTEAD. If set (default) widget will indicate the alarm state of any variable data it is displaying. If clear widget will never indicate the alarm state of any variable data it is displaying. Typically the background colour is set to indicate the alarm state. Note, this property is included in the set of standard properties as it applies to most widgets. It will do nothing for widgets that don't display data.

9.81.4.15 DisplayAlarmStateOptions QEPushButton::displayAlarmStateOption [read, write]

If 'Always' (default) widget will indicate the alarm state of any variable data it is displaying, including 'No Alarm'. If 'Never' widget will never indicate the alarm state of any variable data it is displaying. If 'WhenInAlarm' widget only indicate the alarm state of any variable data it is displaying if it is 'in alarm'. Typically the background colour is set to indicate the alarm state. Note, this property is included in the set of standard properties as it applies to most widgets. It will do nothing for widgets that don't display data.

9.81.4.16 Formats QEPushButton::format [read, write]

Format to apply to data. Default is 'Default' in which case the data type supplied with the data determines how the data is formatted. For all other options, an attempt is made to format the data as requested (whatever its native form).

9.81.4.17 QString QEPushButton::guiFile [read, write]

File name of GUI to be presented on button click. File name can be absolute, relative to the path of the QEform in which the [QEPushButton](#) is located, relative to the any path in the path list published in the ContainerProfile class, or relative to the current path. See [QEWidget::openQEFfile\(\)](#) in [QEWidget.cpp](#) for details.

9.81.4.18 unsigned QEPushButton::int [read, write]

Set the ID used by the message filtering system. Default is zero. Widgets or applications that use messages from the framework have the option of filtering on this ID. For example, by using a unique message source ID a [QELog](#) widget may be set up to only log messages from a select set of widgets.

Base used for when formatting integers. Default is 10 (duh!)

Index used to select a single item of data for formatting from an array of data. Default is 0. Only used when the arrayAction property is INDEX. Refer to the arrayAction property for more details.

9.81.4.19 QString QEPushButton::labelText [read, write]

Button label text (prior to substitution). Macro substitutions will be applied to this text and the result will be set as the button text. Used when data updates are not being represented in the button text. IF NOT LEFT EMPTY, THIS TEXT WILL TAKE PRIORITY OVER THE PUSH BUTTON 'text' PROPERTY! For example, a button in a sub form may have a 'labelText' property of 'Turn Pump On'. When the sub form is used twice in a main form with substitutions PUMPNUM=1 and PUMPNUM=2 respectively, the two identical buttons in the sub forms will have the labels 'Turn Pump 1 On' and 'Turn Pump 2 On' respectively.

Reimplemented from [QEGenericButton](#).

9.81.4.20 bool QEPushButton::leadingZero [read, write]

If true (default), always add a leading zero when formatting numbers.

9.81.4.21 QString QEPushButton::localEnumeration [read, write]

An enumeration list used to data values. Used only when the formatting option is 'local enumeration'. Value is converted to an integer and used to select a string from this list.

Format is:

```
[[<|<=|=|=|>|=|>]value1|*] : string1 , [[<|<=|=|=|>|=|>]value2|*] : string2 , [[<|<=|=|=|>|=|>]value3|*]
: string3 , ...
```

Where: < Less than <= Less than or equal = Equal (default if no operator specified)
 >= Greater than or equal > Greater than Always match (used to specify default text)

Values may be numeric or textual Values do not have to be in any order, but first match wins Values may be quoted Strings may be quoted Consecutive values do not have to be present. Operator is assumed to be equality if not present. White space is ignored except within quoted strings.

may be included in a string to indicate a line break

Examples are:

```
0:Off,1:On 0 : "Pump Running", 1 : "Pump not running" 0:"", 1:"Warning!\nAlarm"
<2:"Value is less than two", =2:"Value is equal to two", >2:"Value is grater than 2"
3:"Beamline Available", *:"" "Pump Off":"OH NO!, the pump is OFF!","Pump On":"It's
OK, the pump is on"
```

The data value is converted to a string if no enumeration for that value is available. For example, if the local enumeration is '0:off,1:on', and a value of 10 is processed, the text generated is '10'. If a blank string is required, this should be explicit. for example, '0:off,1:on,10:'''

A range of numbers can be covered by a pair of values as in the following example:
 >=4:"Between 4 and 8",<=8:"Between 4 and 8"

9.81.4.22 Notations QEPushButton::notation [read, write]

Notation used for numerical formatting. Default is fixed.

9.81.4.23 QString QEPushButton::password [read, write]

Password user will need to enter before any action is taken

Reimplemented from [QEGenericButton](#).

9.81.4.24 QPixmap QEPushButton:: pixmap0 [read, write]

Pixmap to display if updateOption is Icon or TextAndIcon and data value translates to an index of 0

9.81.4.25 QPixmap QEPushButton:: pixmap1 [read, write]

Pixmap to display if updateOption is Icon or TextAndIcon and data value translates to an index of 1

9.81.4.26 QPixmap QEPushButton:: pixmap2 [read, write]

Pixmap to display if updateOption is Icon or TextAndIcon and data value translates to an index of 2

9.81.4.27 QPixmap QEPushButton:: pixmap3 [read, write]

Pixmap to display if updateOption is Icon or TextAndIcon and data value translates to an index of 3

9.81.4.28 QPixmap QEPushButton:: pixmap4 [read, write]

Pixmap to display if updateOption is Icon or TextAndIcon and data value translates to an index of 4

9.81.4.29 QPixmap QEPushButton:: pixmap5 [read, write]

Pixmap to display if updateOption is Icon or TextAndIcon and data value translates to an index of 5

9.81.4.30 QPixmap QEPushButton:: pixmap6 [read, write]

Pixmap to display if updateOption is Icon or TextAndIcon and data value translates to an index of 6

9.81.4.31 QPixmap QEPushButton:: pixmap7 [read, write]

Pixmap to display if updateOption is Icon or TextAndIcon and data value translates to an index of 7

9.81.4.32 int QEPushButton:: precision [read, write]

Precision used when formatting floating point numbers. The default is 4. This is only used if useDbPrecision is false.

9.81.4.33 QString QEPushButton::pressText [read, write]

Value written when user presses button if 'writeOnPress' property is true

Reimplemented from [QEGenericButton](#).

9.81.4.34 QString QEPushButton::prioritySubstitutions [read, write]

Overriding macro substitutions. These macro substitions take precedence over any existing macro substitutions defined by the variableSubstitutions property, any parent forms, or the application containing the button. These macro substitutions are particularly usefull when the button's function is to reload the same form but with different macro substitutions. The variableSubstitutions property cannot be used for this since, although they are added to the list of macro substitutions applied to the new form, they are appended to the list and the existing macro substitutions take precedence.

Reimplemented from [QEGenericButton](#).

9.81.4.35 QString QEPushButton::program [read, write]

Program to run when the button is clicked. No attempt to run a program is made if this property is empty. Example: firefox

9.81.4.36 ProgramStartupOptionNames QEPushButton::programStartupOption [read, write]

Startup options. Just run the command, run the command within a terminal, or display the output in QE message system.

9.81.4.37 QString QEPushButton::releaseText [read, write]

Value written when user releases button if 'writeOnRelease' property is true

Reimplemented from [QEGenericButton](#).

9.81.4.38 QString QEPushButton::styleSheet [read, write]

Hide style sheet from designer as style calculation by the styleManager and not directly setable per se. This also stops transient styles being saved to the ui file.

9.81.4.39 bool QEPushButton::subscribe [read, write]

Sets if this widget subscribes for data updates and displays current data. Default is 'true' (subscribes for and displays data updates)

9.81.4.40 bool QEPushButton::trailingZeros [read, write]

If true (default), always remove any trailing zeros when formatting numbers.

9.81.4.41 UpdateOptions QEPushButton::updateOption [read, write]

Update options (text, pixmap, both, or state (checked or unchecked)

Reimplemented from [QEGenericButton](#).

9.81.4.42 bool QEPushButton::useDbPrecision [read, write]

If true (default), format floating point numbers using the precision supplied with the data.

If false, the precision property is used.

9.81.4.43 UserLevels QEPushButton::userLevelEnabled [read, write]

Lowest user level at which the widget is enabled. Default is 'User'. Used when designing GUIs that allow access to more and more detail according to the user mode. The user mode is set application wide through the [QELogin](#) widget, or programmatically through `setUserLevel()`. Widgets that are always accessible should be visible at 'User'. Widgets that are only accessible to scientists managing the facility should be visible at 'Scientist'. Widgets that are only accessible to engineers maintaining the facility should be visible at 'Engineer'.

9.81.4.44 QString QEPushButton::userLevelEngineerStyle [read, write]

Style Sheet string to be applied when the widget is displayed in 'Engineer' mode. Default is an empty string. The syntax is the standard Qt Style Sheet syntax. For example, 'background-color: red' This Style Sheet string will be applied by the styleManager class. Refer to the styleManager class for details about how this Style Sheet string will be merged with any pre-existing Style Sheet string and any Style Sheet strings generated during the display of data.

9.81.4.45 QString QEPushButton::userLevelScientistStyle [read, write]

Style Sheet string to be applied when the widget is displayed in 'Scientist' mode. Default is an empty string. The syntax is the standard Qt Style Sheet syntax. For example, 'background-color: red' This Style Sheet string will be applied by the styleManager class. Refer to the styleManager class for details about how this Style Sheet string will be merged with any pre-existing Style Sheet string and any Style Sheet strings generated during the display of data.

9.81.4.46 QString QEPushButton::userLevelUserStyle [read, write]

Style Sheet string to be applied when the widget is displayed in 'User' mode. Default is an empty string. The syntax is the standard Qt Style Sheet syntax. For example, 'background-color: red' This Style Sheet string will be applied by the styleManager class. Refer to the styleManager class for details about how this Style Sheet string will be merged with any pre-existing Style Sheet string and any Style Sheet strings generated during the display of data.

9.81.4.47 UserLevels QEPushButton::userLevelVisibility [read, write]

Lowest user level at which the widget is visible. Default is 'User'. Used when designing GUIs that display more and more detail according to the user mode. The user mode is set application wide through the [QELogin](#) widget, or programmatically through setUserLevel() Widgets that are always visible should be visible at 'User'. Widgets that are only used by scientists managing the facility should be visible at 'Scientist'. Widgets that are only used by engineers maintaining the facility should be visible at 'Engineer'.

9.81.4.48 QString QEPushButton::variable [read, write]

EPICS variable name (CA PV). This variable is used for both writing (on button press), and reading if subscribed and no alternate readback variable is provided.

9.81.4.49 bool QEPushButton::variableAsToolTip [read, write]

Use the variable as the tool tip. Default is true. Tool tip property will be overwritten by the variable name.

9.81.4.50 QString QEPushButton::variableSubstitutions [read, write]

Macro substitutions. The default is no substitutions. The format is NAME1=VALUE1[,] NAME2=VALUE2... Values may be quoted strings. For example, 'PUMP=PMP3, NAME = "My Pump"' These substitutions are applied to variable names for all QE widgets. In some widgets are also used for other purposes.

9.81.4.51 bool QEPushButton::visible [read, write]

Display the widget. Default is true. Setting this property false is useful if widget is only used to provide a signal - for example, when supplying data to a [QELink](#) widget. Note, when false the widget will still be visible in Qt Designer.

9.81.4.52 bool QEPushButton::writeOnClick [read, write]

If true, the 'clickText' property is written when the button is clicked. Default is true

Reimplemented from [QEGenericButton](#).

9.81.4.53 bool QEPushButton::writeOnPress [read, write]

If true, the 'pressText' property is written when the button is pressed. Default is false

Reimplemented from [QEGenericButton](#).

9.81.4.54 bool QEPushButton::writeOnRelease [read, write]

If true, the 'releaseText' property is written when the button is released. Default is false

Reimplemented from [QEGenericButton](#).

The documentation for this class was generated from the following files:

- /tmp/epicsqt/trunk/framework/widgets/QEButton/QEPushButton.h
- /tmp/epicsqt/trunk/framework/widgets/QEButton/QEPushButton.cpp

9.82 QEPVNameLists Class Reference

Public Member Functions

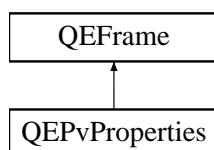
- void **prependOrMoveToFirst** (const QString &item)
- void **saveConfiguration** (PMElement &parentElement)
- void **restoreConfiguration** (PMElement &parentElement)

The documentation for this class was generated from the following file:

- /tmp/epicsqt/trunk/framework/widgets/QEStripChart/QEStripChart.cpp

9.83 QEPvProperties Class Reference

Inheritance diagram for QEPvProperties:



Signals

- void **setCurrentBoxIndex** (int index)

Public Member Functions

- void [setVariableNameProperty](#) (QString variableName)
Property access function for `variable` property. This has special behaviour to work well within designer.
- QString [getVariableNameProperty](#) ()
Property access function for `variable` property. This has special behaviour to work well within designer.
- void [setVariableNameSubstitutionsProperty](#) (QString variableNameSubstitutions)
Property access function for `variableSubstitutions` property. This has special behaviour to work well within designer.
- QString [getVariableNameSubstitutionsProperty](#) ()
Property access function for `variableSubstitutions` property. This has special behaviour to work well within designer.
- **QEPvProperties** (QWidget *parent=0)
- **QEPvProperties** (const QString &variableName, QWidget *parent=0)
- QSize [sizeHint](#) () const

Protected Member Functions

- void [resizeEvent](#) (QResizeEvent *event)
- void [establishConnection](#) (unsigned int variableIndex)
- qcaobject::QCaObject * [createQcalItem](#) (unsigned int variableIndex)
- void [mousePressEvent](#) (QMouseEvent *event)
- void [dragEnterEvent](#) (QDragEnterEvent *event)
- void [dropEvent](#) (QDropEvent *event)
- void [saveConfiguration](#) (PersistanceManager *pm)
- void [restoreConfiguration](#) (PersistanceManager *pm, restorePhases restorePhase)
- QString [copyVariable](#) ()
- QVariant [copyData](#) ()
- void [paste](#) (QVariant s)

Properties

- QString [variable](#)
- QString [variableSubstitutions](#)

9.83.1 Property Documentation

9.83.1.1 QString QEPvProperties::variable [read, write]

EPICS variable name (CA PV)

9.83.1.2 QString QEPvProperties::variableSubstitutions [read, write]

Macro substitutions. The default is no substitutions. The format is NAME1=VALUE1[,] NAME2=VALUE2... Values may be quoted strings. For example, 'PUMP=PMP3, NAME = "My Pump" These substitutions are applied to variable names for all QE widgets. In some widgets are also used for other purposes.

The documentation for this class was generated from the following files:

- /tmp/epicsqt/trunk/framework/widgets/QEPvProperties/QEPvProperties.h
- /tmp/epicsqt/trunk/framework/widgets/QEPvProperties/QEPvProperties.cpp

9.84 QEPvPropertiesManager Class Reference

Public Member Functions

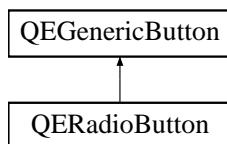
- **QEPvPropertiesManager** (QObject *parent=0)
- bool **isContainer** () const
- bool **isInitialized** () const
- QIcon **icon** () const
- QString **group** () const
- QString **includeFile** () const
- QString **name** () const
- QString **toolTip** () const
- QString **whatsThis** () const
- QWidget * **createWidget** (QWidget *parent)
- void **initialize** (QDesignerFormEditorInterface *core)

The documentation for this class was generated from the following files:

- /tmp/epicsqt/trunk/framework/widgets/QEPvProperties/QEPvPropertiesManager.h
- /tmp/epicsqt/trunk/framework/widgets/QEPvProperties/QEPvPropertiesManager.cpp

9.85 QERadioButton Class Reference

Inheritance diagram for QERadioButton:



Public Types

- enum `UserLevels` { `User` = userLevelTypes::USERLEVEL_USER, `Scientist` = userLevelTypes::USERLEVEL_SCIENTIST, `Engineer` = userLevelTypes::USERLEVEL_ENGINEER }
- enum `DisplayAlarmStateOptions` { `Never` = standardProperties::DISPLAY_ALARM_STATE_NEVER, `Always` = standardProperties::DISPLAY_ALARM_STATE_ALWAYS, `WhenInAlarm` = standardProperties::DISPLAY_ALARM_STATE_WHEN_IN_ALARM }
- enum `Formats` {

`Default` = QEStringFormatting::FORMAT_DEFAULT, `Floating` = QEStringFormatting::FORMAT_FLOATING, `Integer` = QEStringFormatting::FORMAT_INTEGER, `UnsignedInteger` = QEStringFormatting::FORMAT_UNSIGNEDINTEGER,

`Time` = QEStringFormatting::FORMAT_TIME, `LocalEnumeration` = QEStringFormatting::FORMAT_LOCAL_ENUMERATE }
- enum `Notations` { `Fixed` = QEStringFormatting::NOTATION_FIXED, `Scientific` = QEStringFormatting::NOTATION_SCIENTIFIC, `Automatic` = QEStringFormatting::NOTATION_AUTOMATIC }
- enum `ArrayActions` { `Append` = QEStringFormatting::APPEND, `Ascii` = QEStringFormatting::ASCII, `Index` = QEStringFormatting::INDEX }
- enum `UpdateOptions` { `Text` = QEGenericButton::UPDATE_TEXT, `Icon` = QEGenericButton::UPDATE_ICON, `TextAndIcon` = QEGenericButton::UPDATE_TEXT_AND_ICON, `State` = QEGenericButton::UPDATE_STATE }

User friendly enumerations for updateOption property - refer to QEGenericButton::updateOptions for details.
- enum `ProgramStartupOptionNames` { `None` = applicationLauncher::PSO_NONE, `Terminal` = applicationLauncher::PSO_TERMINAL, `LogOutput` = applicationLauncher::PSO_LOGOUTPUT, `StdOutput` = applicationLauncher::PSO_STDOUPUT }
- enum `CreationOptionNames` {

`Open` = QEActionRequests::OptionOpen, `NewTab` = QEActionRequests::OptionNewTab, `NewWindow` = QEActionRequests::OptionNewWindow, `DockTop` = QEActionRequests::OptionTopDockWindow,

`DockBottom` = QEActionRequests::OptionBottomDockWindow, `DockLeft` = QEActionRequests::OptionLeftDockWindow, `DockRight` = QEActionRequests::OptionRightDockWindow,
 `DockTopTabbed` = QEActionRequests::OptionTopDockWindowTabbed, `DockBottomTabbed` = QEActionRequests::OptionBottomDockWindowTabbed, `DockLeftTabbed` = QEActionRequests::OptionLeftDockWindowTabbed, `DockRightTabbed` = QEActionRequests::OptionRightDockWindowTabbed, `DockFloating` = QEActionRequests::OptionFloatingDockWindow }

Creation options. Used to indicate how to present a GUI when requesting a new GUI be created. Open a new window, open a new tab, or replace the current window.

Public Slots

- void `requestAction` (const QEActionRequests &request)
- void `setDefaultStyle` (const QString &style)

Update the default style applied to this widget.
- void `setManagedVisible` (bool v)

Signals

- void `dbValueChanged` (const QString &out)
- void `requestResend` ()
Internal use only. Used when changing a property value to force a re-display to reflect the new property value.
- void `newGui` (const QEActionRequests &request)
Internal use only. Request a new GUI is created. Typically, this is caught by the QEgui application.
- void `pressed` (int value)
- void `released` (int value)
- void `clicked` (int value)
- void `programCompleted` ()
Program started by button has completed.

Public Member Functions

- `QERadioButton` (QWidget *parent=0)
- `QERadioButton` (const QString &variableName, QWidget *parent=0)
- void `writeNow` ()
- 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 `styleSheet`
- QString `defaultStyle`
- QString `userLevelUserStyle`
- QString `userLevelScientistStyle`
- QString `userLevelEngineerStyle`
- `UserLevels userLevelVisibility`
- `UserLevels userLevelEnabled`
- bool `displayAlarmState`
- `DisplayAlarmStateOptions displayAlarmStateOption`
- int `precision`
- bool `useDbPrecision`
- bool `leadingZero`
- bool `trailingZeros`
- bool `addUnits`
- QString `localEnumeration`
- `Formats format`
- `Notations notation`
- `ArrayActions arrayAction`

- Qt::Alignment `alignment`
- `UpdateOptions updateOption`
- `QPixmap pixmap0`
- `QPixmap pixmap1`
- `QPixmap pixmap2`
- `QPixmap pixmap3`
- `QPixmap pixmap4`
- `QPixmap pixmap5`
- `QPixmap pixmap6`
- `QPixmap pixmap7`
- `QString password`
- `bool confirmAction`
- `QString confirmText`
- `bool writeOnPress`
- `bool writeOnRelease`
- `bool writeOnClick`
- `QString pressText`
- `QString releaseText`
- `QString clickText`
- `QString clickCheckedText`
- `QString labelText`
- `QString program`
- `QStringList arguments`
- `ProgramStartupOptionNames programStartupOption`
- `QString guiFile`
- `CreationOptionNames creationOption`
- `QString prioritySubstitutions`
- `QString customisationName`

9.85.1 Member Enumeration Documentation

9.85.1.1 enum QERadioButton::ArrayActions

User friendly enumerations for arrayAction property - refer to QEStringFormatting::arrayActions for details.

Enumerator:

- Append** Refer to QEStringFormatting::APPEND for details.
Ascii Refer to QEStringFormatting::ASCII for details.
Index Refer to QEStringFormatting::INDEX for details.

9.85.1.2 enum QERadioButton::CreationOptionNames

Creation options. Used to indicate how to present a GUI when requesting a new GUI be created. Open a new window, open a new tab, or replace the current window.

Enumerator:

- Open*** Replace the current GUI with the new GUI.
- NewTab*** Open new GUI in a new tab.
- NewWindow*** Open new GUI in a new window.
- DockTop*** Open new GUI in a top dock window.
- DockBottom*** Open new GUI in a bottom dock window.
- DockLeft*** Open new GUI in a left dock window.
- DockRight*** Open new GUI in a right dock window.
- DockTopTabbed*** Open new GUI in a top dock window (tabbed with any existing dock in that area)
- DockBottomTabbed*** Open new GUI in a bottom dock window (tabbed with any existing dock in that area)
- DockLeftTabbed*** Open new GUI in a left dock window (tabbed with any existing dock in that area)
- DockRightTabbed*** Open new GUI in a right dock window (tabbed with any existing dock in that area)
- DockFloating*** Open new GUI in a floating dock window.

9.85.1.3 enum QERadioButton::DisplayAlarmStateOptions

User friendly enumerations for [displayAlarmStateOption](#) property - refer to [displayAlarmStateOption](#) property and [displayAlarmStateOptions](#) enumeration for details.

Enumerator:

- Never*** Refer to DISPLAY_ALARM_STATE_NEVER for details.
- Always*** Refer to DISPLAY_ALARM_STATE_ALWAYS for details.
- WhenInAlarm*** Refer to DISPLAY_ALARM_STATE_WHEN_IN_ALARM for details.

9.85.1.4 enum QERadioButton::Formats

User friendly enumerations for format property - refer to [QEStringFormatting::formats](#) for details.

Enumerator:

- Default*** Format as best appropriate for the data type.
- Floating*** Format as a floating point number.

Integer Format as an integer.

UnsignedInteger Format as an unsigned integer.

Time Format as a time.

LocalEnumeration Format as a selection from the [localEnumeration](#) property.

9.85.1.5 enum QERadioButton::Notations

User friendly enumerations for notation property - refer to QEStringFormatting::notations for details.

Enumerator:

Fixed Refer to QEStringFormatting::NOTATION_FIXED for details.

Scientific Refer to QEStringFormatting::NOTATION_SCIENTIFIC for details.

Automatic Refer to QEStringFormatting::NOTATION_AUTOMATIC for details.

9.85.1.6 enum QERadioButton::ProgramStartupOptionNames

Startup options. Just run the command, run the command within a terminal, or display the output in QE message system.

Enumerator:

None Just run the program.

Terminal Run the program in a terminal (in Windows a command interpreter will also be started, so the program may be a built-in command like 'dir')

LogOutput Run the program, and log the output in the QE message system.

StdOutput Run the program, and send output to standard output and standard error.

9.85.1.7 enum QERadioButton::UpdateOptions

User friendly enumerations for updateOption property - refer to QEGenericButton::updateOptions for details.

Enumerator:

Text Data updates will update the button text.

Icon Data updates will update the button icon.

TextAndIcon Data updates will update the button text and icon.

State Data updates will update the button state (checked or unchecked)

9.85.1.8 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.

9.85.2 Constructor & Destructor Documentation

9.85.2.1 QERadioButton::QERadioButton (`QWidget * parent = 0`)

Create without a variable. Use `setVariableNameProperty()` and `setSubstitutionsProperty()` to define a variable and, optionally, macro substitutions later.

9.85.2.2 QERadioButton::QERadioButton (`const QString & variableName, QWidget * parent = 0`)

Create with a variable. A connection is automatically established. If macro substitutions are required, create without a variable and set the variable and macro substitutions after creation.

9.85.3 Member Function Documentation

9.85.3.1 void QERadioButton::clicked (`int value`) [signal]

Button has been Clicked. The value emitted is the integer interpretation of the `clickText` property (or the `clickCheckedText` property if the button was checked)

9.85.3.2 void QERadioButton::dbValueChanged (`const QString & out`) [signal]

Sent when the widget is updated following a data change Can be used to pass on EPICS data (as presented in this widget) to other widgets. For example a QList widget could log updates from this widget.

9.85.3.3 void QERadioButton::pressed (`int value`) [signal]

Button has been Pressed. The value emitted is the integer interpretation of the `pressText` property

9.85.3.4 void QERadioButton::released (int value) [signal]

Button has been Released The value emitted is the integer interpretation of the release-Text property

9.85.3.5 void QERadioButton::requestAction (const QEActionRequests & request) [inline, slot]

Default slot used to create a new GUI if there is no slot indicated in the ContainerProfile class. This slot is typically used when the button is pressed within the Designer preview window to allow the operation of the button to be tested. If an application does not specify a slot to use for creating new windows (through the ContainerProfile class) a window will still be created through this slot, but it will not respect the window creation options or any other window related application constraints. For example, the QEGui application does provide a slot for creating new GUIs in the ContainerProfile class which respects the creation options, knows how to add tabs in the application, and extend the application's window menu in the menu bar.

9.85.3.6 void QERadioButton::setManagedVisible (bool v) [inline, slot]

Slot to set the visibility of a QE widget, taking into account the user level. Widget will be hidden if hidden by a call this slot, by will only be made visible by a call to this slot if the user level allows.

9.85.4 Property Documentation

9.85.4.1 bool QERadioButton::addUnits [read, write]

If true (default), add engineering units supplied with the data.

9.85.4.2 Qt::Alignment QERadioButton::alignment [read, write]

Set the buttons text alignment. Left justification is particularly useful when displaying quickly changing numeric data updates.

9.85.4.3 bool QERadioButton::allowDrop [read, write]

Allow drag/drops operations to this widget. Default is false. Any dropped text will be used as a new variable name.

9.85.4.4 QStringList QERadioButton::arguments [read, write]

Arguments for program specified in the 'program' property.

9.85.4.5 ArrayActions QERadioButton::arrayAction [read, write]

Text formatting option for array data. Default is ASCII. Options are:

- ASCII - treat array as a single text string. For example an array of three characters 'a' 'b' 'c' will be formatted as 'abc'.
- APPEND - treat array as an array of numbers and format a string containing them all with a space between each. For example, an array of three numbers 10, 11 and 12 will be formatted as '10 11 12'.
- INDEX - Extract a single item from the array. The item is then formatted as any other non array data would be. The item selected is determined by the arrayIndex property. For example, if arrayIndex property is 1, an array of three numbers 10, 11 and 12 will be formatted as '11'.

9.85.4.6 QString QERadioButton::clickCheckedText [read, write]

Text used to compare with text written or read to determine if push button should be marked as checked. Note, must be an exact match following formatting of data updates. When writing values, the 'pressText', 'ReleaseText', or 'clickedtext' must match this property to cause the button to be checked when the write occurs.

Good example: formatting set to display a data value of '1' as 'On', clickCheckedText is 'On', clickText is 'On'. In this example, the push button will be checked when a data update occurs with a value of 1 or when the button is clicked.

Bad example: formatting set to display a data value of '1' as 'On', clickCheckedText is 'On', clickText is '1'. In this example, the push button will be checked when a data update occurs with a value of 1 but, although a valid value will be written when clicked, the button will not be checked when clicked as '1' is not the same as 'On'.

Reimplemented from [QEGenericButton](#).

9.85.4.7 QString QERadioButton::clickText [read, write]

Value written when user clicks button if 'writeOnClick' property is true

Reimplemented from [QEGenericButton](#).

9.85.4.8 bool QERadioButton::confirmAction [read, write]

If true, a dialog will be presented asking the user to confirm if the button action should be carried out

9.85.4.9 QString QERadioButton::confirmText [read, write]

Text used to confirm action if confirmation dialog is presented

Reimplemented from [QEGenericButton](#).

9.85.4.10 CreationOptionNames QERadioButton::creationOption [read, write]

Creation options when opening a new GUI. Open a new window, open a new tab, or replace the current window. the creation option is supplied when the button generates a newGui signal. Application code connected to this signal should honour this request if possible. When used within the QEgui application, the QEgui application creates a new window, new tab, or replaces the current window as appropriate.

Reimplemented from [QEGenericButton](#).

9.85.4.11 QString QERadioButton::customisationName [read, write]

Window customisation name. This name will be used to select a set of window customisations including menu items and tool bar buttons. Applications such as QEgui can load .xml files containing named sets of window customisations. This property is used to select a set loaded from these files. The selected set of customisations will be applied to the main window containing the new GUI.

Reimplemented from [QEGenericButton](#).

9.85.4.12 QString QERadioButton::defaultStyle [read, write]

Style Sheet string to be applied before, i.e. lower priority than, any other style, e.g. alarm style and/or user level style. Default is an empty string.

9.85.4.13 bool QERadioButton::displayAlarmState [read, write]

DEPRECATED. USE `displayAlarmStateOption` INSTEAD. If set (default) widget will indicate the alarm state of any variable data it is displaying. If clear widget will never indicate the alarm state of any variable data it is displaying. Typically the background colour is set to indicate the alarm state. Note, this property is included in the set of standard properties as it applies to most widgets. It will do nothing for widgets that don't display data.

**9.85.4.14 DisplayAlarmStateOptions QERadioButton::displayAlarmStateOption
[read, write]**

If 'Always' (default) widget will indicate the alarm state of any variable data it is displaying, including 'No Alarm'. If 'Never' widget will never indicate the alarm state of any variable data it is displaying. If 'WhenInAlarm' widget only indicate the alarm state of any variable data it is displaying if it is 'in alarm'. Typically the background colour is set to indicate the alarm state. Note, this property is included in the set of standard properties as it applies to most widgets. It will do nothing for widgets that don't display data.

9.85.4.15 Formats `QERadioButton::format` [read, write]

Format to apply to data. Default is 'Default' in which case the data type supplied with the data determines how the data is formatted. For all other options, an attempt is made to format the data as requested (whatever its native form).

9.85.4.16 QString `QERadioButton::guiFile` [read, write]

File name of GUI to be presented on button click. File name can be absolute, relative to the path of the QEform in which the [QEPushButton](#) is located, relative to the any path in the path list published in the ContainerProfile class, or relative to the current path. See `QEWidget::openQEFile()` in `QEWidget.cpp` for details.

9.85.4.17 unsigned `QERadioButton::int` [read, write]

Set the ID used by the message filtering system. Default is zero. Widgets or applications that use messages from the framework have the option of filtering on this ID. For example, by using a unique message source ID a [QELog](#) widget may be set up to only log messages from a select set of widgets.

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.

9.85.4.18 QString `QERadioButton::labelText` [read, write]

Button label text (prior to substitution). Macro substitutions will be applied to this text and the result will be set as the button text. Used when data updates are not being represented in the button text. IF NOT LEFT EMPTY, THIS TEXT WILL TAKE PRIORITY OVER THE PUSH BUTTON 'text' PROPERTY! For example, a button in a sub form may have a 'labelText' property of 'Turn Pump On'. When the sub form is used twice in a main form with substitutions PUMPNUM=1 and PUMPNUM=2 respectively, the two identical buttons in the sub forms will have the labels 'Turn Pump 1 On' and 'Turn Pump 2 On' respectively.

Reimplemented from [QEGenericButton](#).

9.85.4.19 bool `QERadioButton::leadingZero` [read, write]

If true (default), always add a leading zero when formatting numbers.

9.85.4.20 QString `QERadioButton::localEnumeration` [read, write]

An enumeration list used to data values. Used only when the formatting option is 'local enumeration'. Value is converted to an integer and used to select a string from this list.

Format is:

```
[[<|<=|=|=|>|=|>]value1|*]: string1 , [[<|<=|=|=|>|=|>]value2|*]: string2 , [[<|<=|=|=|>|=|>]value3|*]
: string3 , ...
```

Where: < Less than <= Less than or equal = Equal (default if no operator specified)
>= Greater than or equal > Greater than Always match (used to specify default text)

Values may be numeric or textual Values do not have to be in any order, but first match wins Values may be quoted Strings may be quoted Consecutive values do not have to be present. Operator is assumed to be equality if not present. White space is ignored except within quoted strings.

may be included in a string to indicate a line break

Examples are:

```
0:Off,1:On 0 : "Pump Running", 1 : "Pump not running" 0:"", 1:"Warning!\nAlarm"
<2:"Value is less than two", =2:"Value is equal to two", >2:"Value is grater than 2"
3:"Beamline Available", *:"" "Pump Off":"OH NO!, the pump is OFF!","Pump On":"It's
OK, the pump is on"
```

The data value is converted to a string if no enumeration for that value is available. For example, if the local enumeration is '0:off,1:on', and a value of 10 is processed, the text generated is '10'. If a blank string is required, this should be explicit. for example, '0:off,1:on,10:'''

A range of numbers can be covered by a pair of values as in the following example:
>=4:"Between 4 and 8",<=8:"Between 4 and 8"

9.85.4.21 Notations QERadioButton::notation [read, write]

Notation used for numerical formatting. Default is fixed.

9.85.4.22 QString QERadioButton::password [read, write]

Password user will need to enter before any action is taken

Reimplemented from [QEGenericButton](#).

9.85.4.23 QPixmap QERadioButton:: pixmap0 [read, write]

Pixmap to display if updateOption is Icon or TextAndIcon and data value translates to an index of 0

9.85.4.24 QPixmap QERadioButton:: pixmap1 [read, write]

Pixmap to display if updateOption is Icon or TextAndIcon and data value translates to an index of 1

9.85.4.25 QPixmap QERadioButton:: pixmap2 [read, write]

Pixmap to display if updateOption is Icon or TextAndIcon and data value translates to an index of 2

9.85.4.26 QPixmap QERadioButton:: pixmap3 [read, write]

Pixmap to display if updateOption is Icon or TextAndIcon and data value translates to an index of 3

9.85.4.27 QPixmap QERadioButton:: pixmap4 [read, write]

Pixmap to display if updateOption is Icon or TextAndIcon and data value translates to an index of 4

9.85.4.28 QPixmap QERadioButton:: pixmap5 [read, write]

Pixmap to display if updateOption is Icon or TextAndIcon and data value translates to an index of 5

9.85.4.29 QPixmap QERadioButton:: pixmap6 [read, write]

Pixmap to display if updateOption is Icon or TextAndIcon and data value translates to an index of 6

9.85.4.30 QPixmap QERadioButton:: pixmap7 [read, write]

Pixmap to display if updateOption is Icon or TextAndIcon and data value translates to an index of 7

9.85.4.31 int QERadioButton:: precision [read, write]

Precision used when formatting floating point numbers. The default is 4. This is only used if useDbPrecision is false.

9.85.4.32 QString QERadioButton:: pressText [read, write]

Value written when user presses button if 'writeOnPress' property is true

Reimplemented from [QEGenericButton](#).

9.85.4.33 QString QERadioButton::prioritySubstitutions [read, write]

Overriding macro substitutions. These macro substitions take precedence over any existing macro substitutions defined by the variableSubstitutions property, any parent forms, or the application containing the button. These macro substitutions are particularly usefull when the button's function is to reload the same form but with different macro substitutions. The variableSubstitutions property cannot be used for this since, although they are added to the list of macro substitions applied to the new form, they are appended to the list and the existing macro substitutions take precedence.

Reimplemented from [QEGenericButton](#).

9.85.4.34 QString QERadioButton::program [read, write]

Program to run when the button is clicked. No attempt to run a program is made if this property is empty. Example: firefox

9.85.4.35 ProgramStartupOptionNames QERadioButton::programStartupOption [read, write]

Startup options. Just run the command, run the command within a terminal, or display the output in QE message system.

9.85.4.36 QString QERadioButton::releaseText [read, write]

Value written when user releases button if 'writeOnRelease' property is true

Reimplemented from [QEGenericButton](#).

9.85.4.37 QString QERadioButton::styleSheet [read, write]

Hide style sheet from designer as style calculation by the styleManager and not directly setable per se. This also stops transient styles being saved to the ui file.

9.85.4.38 bool QERadioButton::subscribe [read, write]

Sets if this widget subscribes for data updates and displays current data. Default is 'true' (subscribes for and displays data updates)

9.85.4.39 bool QERadioButton::trailingZeros [read, write]

If true (default), always remove any trailing zeros when formatting numbers.

9.85.4.40 UpdateOptions QERadioButton::updateOption [read, write]

Update options (text, pixmap, both, or state (checked or unchecked)

Reimplemented from [QEGenericButton](#).

9.85.4.41 bool QERadioButton::useDbPrecision [read, write]

If true (default), format floating point numbers using the precision supplied with the data.

If false, the precision property is used.

9.85.4.42 UserLevels QERadioButton::userLevelEnabled [read, write]

Lowest user level at which the widget is enabled. Default is 'User'. Used when designing GUIs that allow access to more and more detail according to the user mode. The user mode is set application wide through the [QELogin](#) widget, or programmatically through `setUserLevel()`. Widgets that are always accessible should be visible at 'User'. Widgets that are only accessible to scientists managing the facility should be visible at 'Scientist'. Widgets that are only accessible to engineers maintaining the facility should be visible at 'Engineer'.

9.85.4.43 QString QERadioButton::userLevelEngineerStyle [read, write]

Style Sheet string to be applied when the widget is displayed in 'Engineer' mode. Default is an empty string. The syntax is the standard Qt Style Sheet syntax. For example, 'background-color: red' This Style Sheet string will be applied by the styleManager class. Refer to the styleManager class for details about how this Style Sheet string will be merged with any pre-existing Style Sheet string and any Style Sheet strings generated during the display of data.

9.85.4.44 QString QERadioButton::userLevelScientistStyle [read, write]

Style Sheet string to be applied when the widget is displayed in 'Scientist' mode. Default is an empty string. The syntax is the standard Qt Style Sheet syntax. For example, 'background-color: red' This Style Sheet string will be applied by the styleManager class. Refer to the styleManager class for details about how this Style Sheet string will be merged with any pre-existing Style Sheet string and any Style Sheet strings generated during the display of data.

9.85.4.45 QString QERadioButton::userLevelUserStyle [read, write]

Style Sheet string to be applied when the widget is displayed in 'User' mode. Default is an empty string. The syntax is the standard Qt Style Sheet syntax. For example, 'background-color: red' This Style Sheet string will be applied by the styleManager class. Refer to the styleManager class for details about how this Style Sheet string

will be merged with any pre-existing Style Sheet string and any Style Sheet strings generated during the display of data.

9.85.4.46 **UserLevels** `QERadioButton::userLevelVisibility` [read, write]

Lowest user level at which the widget is visible. Default is 'User'. Used when designing GUIs that display more and more detail according to the user mode. The user mode is set application wide through the [QELogin](#) widget, or programmatically through setUserLevel(). Widgets that are always visible should be visible at 'User'. Widgets that are only used by scientists managing the facility should be visible at 'Scientist'. Widgets that are only used by engineers maintaining the facility should be visible at 'Engineer'.

9.85.4.47 **QString** `QERadioButton::variable` [read, write]

EPICS variable name (CA PV)

9.85.4.48 **bool** `QERadioButton::variableAsToolTip` [read, write]

Use the variable as the tool tip. Default is true. Tool tip property will be overwritten by the variable name.

9.85.4.49 **QString** `QERadioButton::variableSubstitutions` [read, write]

Macro substitutions. The default is no substitutions. The format is NAME1=VALUE1[,] NAME2=VALUE2... Values may be quoted strings. For example, 'PUMP=PMP3, NAME = "My Pump"' These substitutions are applied to variable names for all QE widgets. In some widgets are also used for other purposes.

9.85.4.50 **bool** `QERadioButton::visible` [read, write]

Display the widget. Default is true. Setting this property false is useful if widget is only used to provide a signal - for example, when supplying data to a [QELink](#) widget. Note, when false the widget will still be visible in Qt Designer.

9.85.4.51 **bool** `QERadioButton::writeOnClick` [read, write]

If true, the 'clickText' property is written when the button is clicked. Default is true

Reimplemented from [QEGenericButton](#).

9.85.4.52 **bool** `QERadioButton::writeOnPress` [read, write]

If true, the 'pressText' property is written when the button is pressed. Default is false

Reimplemented from [QEGenericButton](#).

9.85.4.53 bool QERadioButton::writeOnRelease [read, write]

If true, the 'releaseText' property is written when the button is released. Default is false
Reimplemented from [QEGenericButton](#).

The documentation for this class was generated from the following files:

- /tmp/epicsqt/trunk/framework/widgets/QEButton/QERadioButton.h
- /tmp/epicsqt/trunk/framework/widgets/QEButton/QERadioButton.cpp

9.86 QERecipe Class Reference

Public Types

- enum **configurationTypesProperty** { **File** = FROM_FILE, **Text** = FROM_TEXT }
- enum **optionsLayoutProperty** { **Top** = TOP, **Bottom** = BOTTOM, **Left** = LEFT, **Right** = RIGHT }
- enum **userTypesProperty** { **User** = userLevelTypes::USERLEVEL_USER, **Scientist** = userLevelTypes::USERLEVEL_SCIENTIST, **Engineer** = userLevelTypes::USERLEVEL_ENGINEER }

Public Member Functions

- **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

9.87 QERecordFieldName Class Reference

Static Public Member Functions

- static QString **recordName** (const QString &pvName)
- static QString **fieldName** (const QString &pvName)
- static QString **fieldPvName** (const QString &pvName, const QString &field)
- static QString **rtypePvName** (const QString &pvName)
- static bool **pvNamesValid** (const QString &pvName)
- static bool **extractPvName** (const QString &item, QString &pvName)

The documentation for this class was generated from the following files:

- /tmp/epicsqt/trunk/framework/widgets/QEPvProperties/QEPvPropertiesUtilities.h
- /tmp/epicsqt/trunk/framework/widgets/QEPvProperties/QEPvPropertiesUtilities.cpp

9.88 QERecordSpec Class Reference

Public Member Functions

- **QERecordSpec** (const QString recordType)
- QString **getRecordType** ()
- QString **getFieldName** (const int index)

The documentation for this class was generated from the following files:

- /tmp/epicsqt/trunk/framework/widgets/QEPvProperties/QEPvPropertiesUtilities.h
- /tmp/epicsqt/trunk/framework/widgets/QEPvProperties/QEPvPropertiesUtilities.cpp

9.89 QERecordSpecList Class Reference

Public Member Functions

- `QERecordSpec * find (const QString recordType)`
- `void appendOrReplace (QERecordSpec *recordSpec)`
- `bool processRecordSpecFile (const QString &filename)`

The documentation for this class was generated from the following files:

- /tmp/epicsqt/trunk/framework/widgets/QEPvProperties/QEPvPropertiesUtilities.h
- /tmp/epicsqt/trunk/framework/widgets/QEPvProperties/QEPvPropertiesUtilities.cpp

9.90 QEScript Class Reference

```
#include <QEScript.h>
```

Public Types

- enum `scriptTypesProperty` { `File` = FROM_FILE, `Text` = FROM_TEXT }
- enum `optionsLayoutProperty` { `Top` = TOP, `Bottom` = BOTTOM, `Left` = LEFT, `Right` = RIGHT }
- enum `UserLevels` { `User` = userLevelTypes::USERLEVEL_USER, `Scientist` = userLevelTypes::USERLEVEL_SCIENTIST, `Engineer` = userLevelTypes::USERLEVEL_ENGINEER }
- enum `DisplayAlarmStateOptions` { `Never` = standardProperties::DISPLAY_ALARM_STATE_NEVER, `Always` = standardProperties::DISPLAY_ALARM_STATE_ALWAYS, `WhenInAlarm` = standardProperties::DISPLAY_ALARM_STATE_WHEN_IN_ALARM }

Public Slots

- void `setManagedVisible` (bool v)

Signals

- void `selected` (QString pFilename)

Public Member Functions

- `QEScript (QWidget *pParent=0)`
- void `setShowScriptList` (bool pValue)
- bool `getShowScriptList` ()
- void `setShowNew` (bool pValue)

- bool **getShowNew** ()
- void **setShowSave** (bool pValue)
- bool **getShowSave** ()
- void **setShowDelete** (bool pValue)
- bool **getShowDelete** ()
- void **setShowExecute** (bool pValue)
- bool **getShowExecute** ()
- void **setShowAbort** (bool pValue)
- bool **getShowAbort** ()
- void **setEditableTable** (bool pValue)
- bool **getEditableTable** ()
- void **setShowTable** (bool pValue)
- bool **getShowTable** ()
- void **setShowTableControl** (bool pValue)
- bool **getShowTableControl** ()
- void **setShowColumnNumber** (bool pValue)
- bool **getShowColumnNumber** ()
- void **setShowColumnEnable** (bool pValue)
- bool **getShowColumnEnable** ()
- void **setShowColumnProgram** (bool pValue)
- bool **getShowColumnProgram** ()
- void **setShowColumnParameters** (bool pValue)
- bool **getShowColumnParameters** ()
- void **setShowColumnWorkingDirectory** (bool pValue)
- bool **getShowColumnWorkingDirectory** ()
- void **setShowColumnTimeout** (bool pValue)
- bool **getShowColumnTimeout** ()
- void **setShowColumnStop** (bool pValue)
- bool **getShowColumnStop** ()
- void **setShowColumnLog** (bool pValue)
- bool **getShowColumnLog** ()
- void **setScriptType** (int pValue)
- int **getScriptType** ()
- void **setScriptFile** (QString pValue)
- QString **getScriptFile** ()
- void **setScriptText** (QString pValue)
- QString **getScriptText** ()
- void **setScriptDefault** (QString pValue)
- QString **getScriptDefault** ()
- void **setExecuteText** (QString pValue)
- QString **getExecuteText** ()
- void **setOptionsLayout** (int pValue)
- int **getOptionsLayout** ()
- void **insertRow** (bool pEnable, QString pProgram, QString pParameter, QString pWorkingDirectory, int pTimeOut, bool pStop, bool pLog)
- bool **saveScriptList** ()

- void **refreshScriptList** ()
- void **refreshWidgets** ()
- void **setScriptTypeProperty** (scriptTypesProperty pScriptType)
- scriptTypesProperty **getScriptTypeProperty** ()
- void **setOptionsLayoutProperty** (optionsLayoutProperty pOptionsLayout)
- optionsLayoutProperty **getOptionsLayoutProperty** ()
- UserLevels **getUserLevelVisibilityProperty** ()
Access function for userLevelVisibility property - refer to userLevelVisibility property for details.
- void **setUserLevelVisibilityProperty** (UserLevels level)
Access function for userLevelVisibility property - refer to userLevelVisibility property for details.
- UserLevels **getUserLevelEnabledProperty** ()
Access function for userLevelEnabled property - refer to userLevelEnabled property for details.
- void **setUserLevelEnabledProperty** (UserLevels level)
Access function for userLevelEnabled property - refer to userLevelEnabled property for details.
- DisplayAlarmStateOptions **getDisplayAlarmStateOptionProperty** ()
Access function for displayAlarmStateOption property - refer to displayAlarmStateOption property for details.
- void **setDisplayAlarmStateOptionProperty** (DisplayAlarmStateOptions option)
Access function for displayAlarmStateOption property - refer to displayAlarmStateOption property for details.

Protected Attributes

- QComboBox * **qComboBoxScriptList**
- QPushButton * **qPushButtonNew**
- QPushButton * **qPushButtonSave**
- QPushButton * **qPushButtonDelete**
- QPushButton * **qPushButtonExecute**
- QPushButton * **qPushButtonAbort**
- QPushButton * **qPushButtonAdd**
- QPushButton * **qPushButtonRemove**
- QPushButton * **qPushButtonUp**
- QPushButton * **qPushButtonDown**
- QPushButton * **qPushButtonCopy**
- QPushButton * **qPushButtonPaste**
- _QTableWidgetScript * **qTableWidgetScript**
- QString **scriptFile**
Define the file where to save the scripts (if not defined then the scripts will be saved in a file named "QEScript.xml")
- QString **scriptText**
Define the XML text that contains the scripts.
- QString **scriptDefault**

Define the script (previously saved by the user) that will be load as the default script when the widget starts.

- int **scriptType**
- int **optionsLayout**
- QDomDocument **document**
- QString **filename**
- QList< **_CopyPaste** * > **copyPasteList**
- bool **editableTable**
Enable/disable table edition.
- bool **isExecuting**

Properties

- bool **showScriptList**
Show/hide combobox that contains the list of existing scripts created by the user.
- bool **showNew**
Show/hide button to reset (initialize) the table that contains the sequence of programs to be executed.
- bool **showSave**
Show/hide button to save/overwrite a new/existing script.
- bool **showDelete**
Show/hide button to delete an existing script.
- bool **showExecute**
Show/hide button to execute a sequence of programs.
- bool **showAbort**
Show/hide button to abort the execution of a sequence of programs.
- bool **showTable**
Show/hide table that contains a sequence of programs to be executed.
- bool **showTableControl**
Show/hide the controls of the table that contains a sequence of programs to be executed.
- bool **showColumnName**
Show/hide the column '#' that displays the sequential number of programs.
- bool **showColumnEnable**
Show/hide the column 'Enable' that enables the execution of programs.
- bool **showColumnProgram**
Show/hide the column 'Program' that contains the external programs to be executed.
- bool **showColumnParameters**
Show/hide the column 'Parameters' that contains the parameters that are passed to external programs to be executed.
- bool **showColumnWorkingDirectory**
Show/hide the column 'Directory' that defines the working directory to be used when external programs are executed.
- bool **showColumnTimeout**

Show/hide the column 'Timeout' that defines a time out period in seconds (if equal to 0 then the program runs until it finishes; otherwise if greater than 0 then the program will only run during this amount of seconds and will be aborted beyond this time)

- bool [showColumnStop](#)

Show/hide the column 'Stop' that enables stopping the execution of subsequent programs when the current one exited with an error code different from 0.

- bool [showColumnLog](#)

Show/hide the column 'Log' that enables the generation of log messages (these messages may be displayed using the [QELog](#) widget)

- [scriptTypesProperty](#) [scriptType](#)

Select if the scripts are to be loaded/saved from an XML file or from an XML text.

- QString [executeText](#)

Define the caption of the button responsible for starting the execution of external programs (if not defined then the caption will be "Execute")

- [optionsLayoutProperty](#) [optionsLayout](#)

Change the order of the widgets. Valid orders are: TOP, BOTTOM, LEFT and RIGHT.

- bool [variableAsToolTip](#)
- bool [allowDrop](#)
- bool [visible](#)
- unsigned [int](#)
- QString [styleSheet](#)
- QString [defaultStyle](#)
- QString [userLevelUserStyle](#)
- QString [userLevelScientistStyle](#)
- QString [userLevelEngineerStyle](#)
- [UserLevels](#) [userLevelVisibility](#)
- [UserLevels](#) [userLevelEnabled](#)
- bool [displayAlarmState](#)
- [DisplayAlarmStateOptions](#) [displayAlarmStateOption](#)

9.90.1 Detailed Description

This class is a EPICS aware widget. The [QEScript](#) widget allows the user to define a certain sequence of external programs to be executed. This sequence may be saved, modified or loaded for future usage.

9.90.2 Member Enumeration Documentation

9.90.2.1 enum QEScript::DisplayAlarmStateOptions

User friendly enumerations for [displayAlarmStateOption](#) property - refer to [displayAlarmStateOption](#) property and [displayAlarmStateOptions](#) enumeration for details.

Enumerator:

Never Refer to DISPLAY_ALARM_STATE_NEVER for details.

Always Refer to DISPLAY_ALARM_STATE_ALWAYS for details.

WhenInAlarm Refer to DISPLAY_ALARM_STATE_WHEN_IN_ALARM for details.

9.90.2.2 enum QEScript::UserLevels

User friendly enumerations for [userLevelVisibility](#) and [userLevelEnabled](#) properties - refer to [userLevelVisibility](#) and [userLevelEnabled](#) properties and userLevel enumeration for details.

Enumerator:

User Refer to USERLEVEL_USER for details.

Scientist Refer to USERLEVEL_SCIENTIST for details.

Engineer Refer to USERLEVEL_ENGINEER for details.

9.90.3 Member Function Documentation

9.90.3.1 void QEScript::setManagedVisible(bool v) [inline, slot]

Slot to set the visibility of a QE widget, taking into account the user level. Widget will be hidden if hidden by a call this slot, by will only be made visible by a call to this slot if the user level allows.

9.90.4 Property Documentation

9.90.4.1 bool QEScript::allowDrop [read, write]

Allow drag/drops operations to this widget. Default is false. Any dropped text will be used as a new variable name.

9.90.4.2 QString QEScript::defaultStyle [read, write]

Style Sheet string to be applied before, i.e. lower priority than, any other style, e.g. alarm style and/or user level style. Default is an empty string.

9.90.4.3 bool QEScript::displayAlarmState [read, write]

DEPRECATED. USE displayAlarmStateOption INSTEAD. If set (default) widget will indicate the alarm state of any variable data it is displaying. If clear widget will never indicate the alarm state of any variable data it is displaying. Typically the background colour is set to indicate the alarm state. Note, this property is included in the set of standard properties as it applies to most widgets. It will do nothing for widgets that don't display data.

9.90.4.4 DisplayAlarmStateOptions QEScript::displayAlarmStateOption [read, write]

If 'Always' (default) widget will indicate the alarm state of any variable data it is displaying, including 'No Alarm'. If 'Never' widget will never indicate the alarm state of any variable data it is displaying. If 'WhenInAlarm' widget only indicate the alarm state of any variable data it is displaying if it is 'in alarm'. Typically the background colour is set to indicate the alarm state. Note, this property is included in the set of standard properties as it applies to most widgets. It will do nothing for widgets that don't display data.

9.90.4.5 unsigned QEScript::int [read, write]

Set the ID used by the message filtering system. Default is zero. Widgets or applications that use messages from the framework have the option of filtering on this ID. For example, by using a unique message source ID a [QELog](#) widget may be set up to only log messages from a select set of widgets.

9.90.4.6 QString QEScript::styleSheet [read, write]

Hide style sheet from designer as style calculation by the styleManager and not directly setable per se. This also stops transient styles being saved to the ui file.

9.90.4.7 UserLevels QEScript::userLevelEnabled [read, write]

Lowest user level at which the widget is enabled. Default is 'User'. Used when designing GUIs that allow access to more and more detail according to the user mode. The user mode is set application wide through the [QELogin](#) widget, or programmatically through `setUserLevel()`. Widgets that are always accessible should be visible at 'User'. Widgets that are only accessible to scientists managing the facility should be visible at 'Scientist'. Widgets that are only accessible to engineers maintaining the facility should be visible at 'Engineer'.

9.90.4.8 QString QEScript::userLevelEngineerStyle [read, write]

Style Sheet string to be applied when the widget is displayed in 'Engineer' mode. Default is an empty string. The syntax is the standard Qt Style Sheet syntax. For example, 'background-color: red' This Style Sheet string will be applied by the styleManager class. Refer to the styleManager class for details about how this Style Sheet string will be merged with any pre-existing Style Sheet string and any Style Sheet strings generated during the display of data.

9.90.4.9 QString QEScript::userLevelScientistStyle [read, write]

Style Sheet string to be applied when the widget is displayed in 'Scientist' mode. Default is an empty string. The syntax is the standard Qt Style Sheet syntax. For example,

'background-color: red' This Style Sheet string will be applied by the styleManager class. Refer to the styleManager class for details about how this Style Sheet string will be merged with any pre-existing Style Sheet string and any Style Sheet strings generated during the display of data.

9.90.4.10 `QString QEScript::userLevelUserStyle [read, write]`

Style Sheet string to be applied when the widget is displayed in 'User' mode. Default is an empty string. The syntax is the standard Qt Style Sheet syntax. For example, 'background-color: red' This Style Sheet string will be applied by the styleManager class. Refer to the styleManager class for details about how this Style Sheet string will be merged with any pre-existing Style Sheet string and any Style Sheet strings generated during the display of data.

9.90.4.11 `UserLevels QEScript::userLevelVisibility [read, write]`

Lowest user level at which the widget is visible. Default is 'User'. Used when designing GUIs that display more and more detail according to the user mode. The user mode is set application wide through the [QELogin](#) widget, or programmatically through setUserLevel() Widgets that are always visible should be visible at 'User'. Widgets that are only used by scientists managing the facility should be visible at 'Scientist'. Widgets that are only used by engineers maintaining the facility should be visible at 'Engineer'.

9.90.4.12 `bool QEScript::variableAsToolTip [read, write]`

Use the variable as the tool tip. Default is true. Tool tip property will be overwritten by the variable name.

9.90.4.13 `bool QEScript::visible [read, write]`

Display the widget. Default is true. Setting this property false is useful if widget is only used to provide a signal - for example, when supplying data to a [QELink](#) widget. Note, when false the widget will still be visible in Qt Designer.

The documentation for this class was generated from the following files:

- /tmp/epicsqt/trunk/framework/widgets/QEScript/QEScript.h
- /tmp/epicsqt/trunk/framework/widgets/QEScript/QEScript.cpp

9.91 QEShape Class Reference

```
#include <QEShape.h>
```

Public Types

- enum `shapeOptions` {
 Line, Points, Polyline, Polygon,
Rect, RoundedRect, Ellipse, Arc,
Chord, Pie, Path }
- enum `animationOptions` {
 Width, Height, X, Y,
Transparency, Rotation, ColourHue, ColourSaturation,
ColourValue, ColourIndex, Penwidth }
- enum `UserLevels` { **User** = userLevelTypes::USERLEVEL_USER, **Scientist** = userLevelTypes::USERLEVEL_SCIENTIST, **Engineer** = userLevelTypes::USERLEVEL_ENGINEER }
- enum `DisplayAlarmStateOptions` { **Never** = standardProperties::DISPLAY_ALARM_STATE_NEVER, **Always** = standardProperties::DISPLAY_ALARM_STATE_ALWAYS, **WhenInAlarm** = standardProperties::DISPLAY_ALARM_STATE_WHEN_IN_ALARM }

Public Slots

- void `setManagedVisible` (bool v)

Signals

- void `dbValueChanged1` (const qulonglong &out)
- void `dbValueChanged2` (const qulonglong &out)
- void `dbValueChanged3` (const qulonglong &out)
- void `dbValueChanged4` (const qulonglong &out)
- void `dbValueChanged5` (const qulonglong &out)
- void `dbValueChanged6` (const qulonglong &out)

Public Member Functions

- `QEShape` (QWidget *parent=0)
- `QEShape` (const QString &variableName, QWidget *parent=0)
- void `scaleBy` (const int m, const int d)

Scale the widgets my m/d.
- void `setAnimation` (`animationOptions` animation, const int index)

Access function for #animation' properties - refer to animation' properties for details.
- `animationOptions getAnimation` (const int index)

Access function for #animation' properties - refer to animation' properties for details.
- void `setScale` (const double scale, const int index)

Access function for #scale' properties - refer to scale' properties for details.
- double `getScale` (const int index)

- Access function for #scale' properties - refer to scale' properties for details.
 - void `setOffset` (const double offset, const int index)
Access function for #offset' properties - refer to offset' properties for details.
 - double `getOffset` (const int index)
Access function for #offset' properties - refer to offset' properties for details.
 - void `setBorder` (const bool border)
Access function for #border' properties - refer to border' properties for details.
 - bool `getBorder` ()
Access function for #border' properties - refer to border' properties for details.
 - void `setFill` (const bool fill)
Access function for #fill' properties - refer to fill' properties for details.
 - bool `getFill` ()
Access function for #fill' properties - refer to fill' properties for details.
 - void `setShape` (`shapeOptions` shape)
Access function for #shape' properties - refer to shape' properties for details.
 - `shapeOptions` `getShape` ()
Access function for #shape' properties - refer to shape' properties for details.
 - void `setNumPoints` (const unsigned int numPoints)
Access function for #number of points' properties - refer to number of points' properties for details.
 - unsigned int `getNumPoints` ()
Access function for #number of points' properties - refer to number of points' properties for details.
 - void `setOriginTranslation` (const QPoint originTranslation)
Access function for #origin translation' properties - refer to origin translation' properties for details.
 - QPoint `getOriginTranslation` ()
Access function for #origin translation' properties - refer to origin translation' properties for details.
 - void `setPoint` (const QPoint point, const int index)
Access function for #point' properties - refer to point' properties for details.
 - QPoint `getPoint` (const int index)
Access function for #point' properties - refer to point' properties for details.
 - void `setColor` (const QColor color, const int index)
Access function for #colour' properties - refer to colour' properties for details.
 - QColor `getColor` (const int index)
Access function for #colour' properties - refer to colour' properties for details.
 - void `setDrawBorder` (const bool drawBorder)
Access function for #draw border' properties - refer to draw border' properties for details.
 - bool `getDrawBorder` ()
Access function for #draw border' properties - refer to draw border' properties for details.
 - void `setLineWidth` (const unsigned int lineWidth)

- `unsigned int getLineWidth ()`

Access function for #line width' properties - refer to line width' properties for details.
- `void setStartAngle (const double startAngle)`

Access function for #start angle' properties - refer to start angle' properties for details.
- `double getStartAngle ()`

Access function for #start angle' properties - refer to start angle' properties for details.
- `void setRotation (const double rotation)`

Access function for #rotation' properties - refer to rotation' properties for details.
- `double getRotation ()`

Access function for #rotation' properties - refer to rotation' properties for details.
- `void setArcLength (const double arcLength)`

Access function for #arc length' properties - refer to arc length' properties for details.
- `double getArcLength ()`

Access function for #arc length' properties - refer to arc length' properties for details.
- `void setVariableNameSubstitutionsProperty (QString variableNameSubstitutions)`

Property access function for `variableSubstitutions` property. This has special behaviour to work well within designer.
- `QString getVariableNameSubstitutionsProperty ()`

Property access function for `variableSubstitutions` property. This has special behaviour to work well within designer.
- `UserLevels getUserLevelVisibilityProperty ()`

Access function for `userLevelVisibility` property - refer to `userLevelVisibility` property for details.
- `void setUserLevelVisibilityProperty (UserLevels level)`

Access function for `userLevelVisibility` property - refer to `userLevelVisibility` property for details.
- `UserLevels getUserLevelEnabledProperty ()`

Access function for `userLevelEnabled` property - refer to `userLevelEnabled` property for details.
- `void setUserLevelEnabledProperty (UserLevels level)`

Access function for `userLevelEnabled` property - refer to `userLevelEnabled` property for details.
- `DisplayAlarmStateOptions getDisplayAlarmStateOptionProperty ()`

Access function for `displayAlarmStateOption` property - refer to `displayAlarmStateOption` property for details.
- `void setDisplayAlarmStateOptionProperty (DisplayAlarmStateOptions option)`

Access function for `displayAlarmStateOption` property - refer to `displayAlarmStateOption` property for details.

Properties

- `QString variable1`
- `QString variable2`
- `QString variable3`
- `QString variable4`
- `QString variable5`
- `QString variable6`
- `QString variableSubstitutions`
- `bool variableAsToolTip`
- `bool allowDrop`
- `bool visible`
- `unsigned int`
- `QString styleSheet`
- `QString defaultStyle`
- `QString userLevelUserStyle`
- `QString userLevelScientistStyle`
- `QString userLevelEngineerStyle`
- `UserLevels userLevelVisibility`
- `UserLevels userLevelEnabled`
- `bool displayAlarmState`
- `DisplayAlarmStateOptions displayAlarmStateOption`
- `animationOptions animation1`
- `animationOptions animation2`
- `animationOptions animation3`
- `animationOptions animation4`
- `animationOptions animation5`
- `animationOptions animation6`
- `double scale1`

Scale factor applied to data from the 1st variable before it is used to animate the shape.

- `double scale2`
- `double scale3`
- `double scale4`
- `double scale5`
- `double scale6`
- `double offset1`
- `double offset2`
- `double offset3`
- `double offset4`
- `double offset5`
- `double offset6`
- `QPoint point1`
- `QPoint point2`
- `QPoint point3`
- `QPoint point4`
- `QPoint point5`

- QPoint `point6`
- QPoint `point7`
- QPoint `point8`
- QPoint `point9`
- QPoint `point10`
- QColor `color1`
- QColor `color2`
- QColor `color3`
- QColor `color4`
- QColor `color5`
- QColor `color6`
- QColor `color7`
- QColor `color8`
- QColor `color9`
- QColor `color10`

9.91.1 Detailed Description

This class is a EPICS aware shape widget based on the Qt widget. One of several shapes can be drawn within the widget, and up to 6 variables can be used to animate various attributes of the shape. For example to represent beam positino and size, an ellipse can be drawn with four variables animating its vertical and horizontal size and position. It is tighly integrated with the base class QEWidget which provides generic support such as macro substitutions, drag/drop, and standard properties.

9.91.2 Member Enumeration Documentation

9.91.2.1 enum QEShape::animationOptions

Options for how a variable will animate the shape.

9.91.2.2 enum QEShape::DisplayAlarmStateOptions

User friendly enumerations for `displayAlarmStateOption` property - refer to `displayAlarmStateOption` property and `displayAlarmStateOptions` enumeration for details.

Enumerator:

- Never** Refer to `DISPLAY_ALARM_STATE_NEVER` for details.
- Always** Refer to `DISPLAY_ALARM_STATE_ALWAYS` for details.
- WhenInAlarm** Refer to `DISPLAY_ALARM_STATE_WHEN_IN_ALARM` for details.

9.91.2.3 enum QEShape::shapeOptions

Options for the type of shape.

9.91.2.4 enum QEShape::UserLevels

User friendly enumerations for `userLevelVisibility` and `userLevelEnabled` properties - refer to `userLevelVisibility` and `userLevelEnabled` properties and `userLevel` enumeration for details.

Enumerator:

User Refer to `USERLEVEL_USER` for details.

Scientist Refer to `USERLEVEL_SCIENTIST` for details.

Engineer Refer to `USERLEVEL_ENGINEER` for details.

9.91.3 Constructor & Destructor Documentation

9.91.3.1 QEShape::QEShape (`QWidget * parent = 0`)

Create without a variable. Use `setVariableNameProperty()` and `setSubstitutionsProperty()` to define a variable and, optionally, macro substitutions later.

9.91.3.2 QEShape::QEShape (`const QString & variableName, QWidget * parent = 0`)

Create with a single variable. (Note, the `QEShape` widget can use up to 6 variables) A connection is automatically established. If macro substitutions are required, create without a variable and set the variable and macro substitutions after creation.

9.91.4 Member Function Documentation

9.91.4.1 void QEShape::dbValueChanged1 (`const qulonglong & out`) [signal]

Sent when the widget is updated following a data change for the first variable Can be used to pass on EPICS data (as presented in this widget) to other widgets. For example a QList widget could log updates from this widget.

9.91.4.2 void QEShape::dbValueChanged2 (`const qulonglong & out`) [signal]

Sent when the widget is updated following a data change for the second variable Can be used to pass on EPICS data (as presented in this widget) to other widgets. For example a QList widget could log updates from this widget.

9.91.4.3 void QEShape::dbValueChanged3 (`const qulonglong & out`) [signal]

Sent when the widget is updated following a data change for the third variable Can be used to pass on EPICS data (as presented in this widget) to other widgets. For example a QList widget could log updates from this widget.

9.91.4.4 void QEShape::dbValueChanged4 (const qulonglong & out) [signal]

Sent when the widget is updated following a data change for the fourth variable Can be used to pass on EPICS data (as presented in this widget) to other widgets. For example a QList widget could log updates from this widget.

9.91.4.5 void QEShape::dbValueChanged5 (const qulonglong & out) [signal]

Sent when the widget is updated following a data change for the fifth variable Can be used to pass on EPICS data (as presented in this widget) to other widgets. For example a QList widget could log updates from this widget.

9.91.4.6 void QEShape::dbValueChanged6 (const qulonglong & out) [signal]

Sent when the widget is updated following a data change for the sixth variable Can be used to pass on EPICS data (as presented in this widget) to other widgets. For example a QList widget could log updates from this widget.

9.91.4.7 void QEShape::setManagedVisible (bool v) [inline, slot]

Slot to set the visibility of a QE widget, taking into account the user level. Widget will be hidden if hidden by a call this slot, by will only be made visible by a call to this slot if the user level allows.

9.91.5 Property Documentation

9.91.5.1 bool QEShape::allowDrop [read, write]

Allow drag/drops operations to this widget. Default is false. Any dropped text will be used as a new variable name.

9.91.5.2 animationOptions QEShape::animation1 [read, write]

Animation to be effected by the 1st variable. This is used to select what the effect changing data for the 1st variable will have on the shape.

9.91.5.3 animationOptions QEShape::animation2 [read, write]

Animation to be effected by the 2nd variable. This is used to select what the effect changing data for the 2nd variable will have on the shape.

9.91.5.4 animationOptions QEShape::animation3 [read, write]

Animation to be effected by the 3rd variable. This is used to select what the effect changing data for the 3rd variable will have on the shape.

9.91.5.5 animationOptions QEShape::animation4 [read, write]

Animation to be effected by the 4th variable. This is used to select what the effect changing data for the 4th variable will have on the shape.

9.91.5.6 animationOptions QEShape::animation5 [read, write]

Animation to be effected by the 5th variable. This is used to select what the effect changing data for the 5th variable will have on the shape.

9.91.5.7 animationOptions QEShape::animation6 [read, write]

Animation to be effected by the 6th variable. This is used to select what the effect changing data for the 6th variable will have on the shape.

9.91.5.8 QColor QEShape::color1 [read, write]

Used by the color animation to determine the color based on a data value. The scaled and offset data is used as an index to select color properties 'color1' to 'color10'.

9.91.5.9 QColor QEShape::color10 [read, write]

Used by the color animation to determine the color based on a data value. The scaled and offset data is used as an index to select color properties 'color1' to 'color10'.

9.91.5.10 QColor QEShape::color2 [read, write]

Used by the color animation to determine the color based on a data value. The scaled and offset data is used as an index to select color properties 'color1' to 'color10'.

9.91.5.11 QColor QEShape::color3 [read, write]

Used by the color animation to determine the color based on a data value. The scaled and offset data is used as an index to select color properties 'color1' to 'color10'.

9.91.5.12 QColor QEShape::color4 [read, write]

Used by the color animation to determine the color based on a data value. The scaled and offset data is used as an index to select color properties 'color1' to 'color10'.

9.91.5.13 QColor QEShape::color5 [read, write]

Used by the color animation to determine the color based on a data value. The scaled and offset data is used as an index to select color properties 'color1' to 'color10'.

9.91.5.14 QColor QEShape::color6 [read, write]

Used by the color animation to determine the color based on a data value. The scaled and offset data is used as an index to select color properties 'color1' to 'color10'.

9.91.5.15 QColor QEShape::color7 [read, write]

Used by the color animation to determine the color based on a data value. The scaled and offset data is used as an index to select color properties 'color1' to 'color10'.

9.91.5.16 QColor QEShape::color8 [read, write]

Used by the color animation to determine the color based on a data value. The scaled and offset data is used as an index to select color properties 'color1' to 'color10'.

9.91.5.17 QColor QEShape::color9 [read, write]

Used by the color animation to determine the color based on a data value. The scaled and offset data is used as an index to select color properties 'color1' to 'color10'.

9.91.5.18 QString QEShape::defaultStyle [read, write]

Style Sheet string to be applied before, i.e. lower priority than, any other style, e.g. alarm style and/or user level style. Default is an empty string.

9.91.5.19 bool QEShape::displayAlarmState [read, write]

DEPRECATED. USE displayAlarmStateOption INSTEAD. If set (default) widget will indicate the alarm state of any variable data it is displaying. If clear widget will never indicate the alarm state of any variable data it is displaying. Typically the background colour is set to indicate the alarm state. Note, this property is included in the set of standard properties as it applies to most widgets. It will do nothing for widgets that don't display data.

9.91.5.20 DisplayAlarmStateOptions QEShape::displayAlarmStateOption [read, write]

If 'Always' (default) widget will indicate the alarm state of any variable data it is displaying, including 'No Alarm'. If 'Never' widget will never indicate the alarm state of any

variable data it is displaying. If 'WhenInAlarm' widget only indicate the alarm state of any variable data it is displaying if it is 'in alarm'. Typically the background colour is set to indicate the alarm state. Note, this property is included in the set of standard properties as it applies to most widgets. It will do nothing for widgets that don't display data.

9.91.5.21 `unsigned QEShape::int` [read, write]

Set the ID used by the message filtering system. Default is zero. Widgets or applications that use messages from the framework have the option of filtering on this ID. For example, by using a unique message source ID a [QELog](#) widget may be set up to only log messages from a select set of widgets.

The number of points to use when drawing shapes that are defined by a variable number of points, such as polyline, polygon, path, and series of points.

Sets the width of the pen. Used for the following shapes: Line, Points, Polyline, Polygon, Rect, RoundedRectangle, Ellipse, Arc, Chord, Pie, Path

9.91.5.22 `double QEShape::offset1` [read, write]

Offset applied to data from the 1st variable before it is used to animate the shape

9.91.5.23 `double QEShape::offset2` [read, write]

Offset applied to data from the 2nd variable before it is used to animate the shape

9.91.5.24 `double QEShape::offset3` [read, write]

Offset applied to data from the 3rd variable before it is used to animate the shape

9.91.5.25 `double QEShape::offset4` [read, write]

Offset applied to data from the 4th variable before it is used to animate the shape

9.91.5.26 `double QEShape::offset5` [read, write]

Offset applied to data from the 5th variable before it is used to animate the shape

9.91.5.27 `double QEShape::offset6` [read, write]

Offset applied to data from the 6th variable before it is used to animate the shape

9.91.5.28 QPoint QEShape::point1 [read, write]

1st coordinate used when drawing the shape. Used for the following shapes: Line, Points, Polyline, Polygon, Rect, RoundedRectangle, Ellipse, Arc, Chord, Pie, Path, Text,Pixmap

9.91.5.29 QPoint QEShape::point10 [read, write]

10th coordinate used when drawing the shape. Used for the following shapes: Points, Polyline, Polygon, Path

9.91.5.30 QPoint QEShape::point2 [read, write]

2nd coordinate used when drawing the shape. Used for the following shapes: Line, Points, Polyline, Polygon, Rect, RoundedRectangle, Ellipse, Arc, Chord, Pie, Path,Pixmap

9.91.5.31 QPoint QEShape::point3 [read, write]

3rd coordinate used when drawing the shape. Used for the following shapes: Points, Polyline, Polygon, Path

9.91.5.32 QPoint QEShape::point4 [read, write]

4th coordinate used when drawing the shape. Used for the following shapes: Points, Polyline, Polygon, Path

9.91.5.33 QPoint QEShape::point5 [read, write]

5th coordinate used when drawing the shape. Used for the following shapes: Points, Polyline, Polygon, Path

9.91.5.34 QPoint QEShape::point6 [read, write]

6th coordinate used when drawing the shape. Used for the following shapes: Points, Polyline, Polygon, Path

9.91.5.35 QPoint QEShape::point7 [read, write]

7th coordinate used when drawing the shape. Used for the following shapes: Points, Polyline, Polygon, Path

9.91.5.36 QPoint QEShape::point8 [read, write]

8th coordinate used when drawing the shape. Used for the following shapes: Points, Polyline, Polygon, Path

9.91.5.37 QPoint QEShape::point9 [read, write]

9th coordinate used when drawing the shape. Used for the following shapes: Points, Polyline, Polygon, Path

9.91.5.38 double QEShape::scale2 [read, write]

Scale factor applied to data from the 2nd variable before it is used to animate the shape

9.91.5.39 double QEShape::scale3 [read, write]

Scale factor applied to data from the 3rd variable before it is used to animate the shape

9.91.5.40 double QEShape::scale4 [read, write]

Scale factor applied to data from the 4th variable before it is used to animate the shape

9.91.5.41 double QEShape::scale5 [read, write]

Scale factor applied to data from the 5th variable before it is used to animate the shape

9.91.5.42 double QEShape::scale6 [read, write]

Scale factor applied to data from the 6th variable before it is used to animate the shape

9.91.5.43 QString QEShape::styleSheet [read, write]

Hide style sheet from designer as style calculation by the styleManager and not directly setable per se. This also stops transient styles being saved to the ui file.

9.91.5.44 UserLevels QEShape::userLevelEnabled [read, write]

Lowest user level at which the widget is enabled. Default is 'User'. Used when designing GUIs that allow access to more and more detail according to the user mode. The user

mode is set application wide through the [QELogin](#) widget, or programmatically through `setUserLevel()`. Widgets that are always accessible should be visible at 'User'. Widgets that are only accessible to scientists managing the facility should be visible at 'Scientist'. Widgets that are only accessible to engineers maintaining the facility should be visible at 'Engineer'.

9.91.5.45 `QString QEShape::userLevelEngineerStyle [read, write]`

Style Sheet string to be applied when the widget is displayed in 'Engineer' mode. Default is an empty string. The syntax is the standard Qt Style Sheet syntax. For example, 'background-color: red' This Style Sheet string will be applied by the `styleManager` class. Refer to the `styleManager` class for details about how this Style Sheet string will be merged with any pre-existing Style Sheet string and any Style Sheet strings generated during the display of data.

9.91.5.46 `QString QEShape::userLevelScientistStyle [read, write]`

Style Sheet string to be applied when the widget is displayed in 'Scientist' mode. Default is an empty string. The syntax is the standard Qt Style Sheet syntax. For example, 'background-color: red' This Style Sheet string will be applied by the `styleManager` class. Refer to the `styleManager` class for details about how this Style Sheet string will be merged with any pre-existing Style Sheet string and any Style Sheet strings generated during the display of data.

9.91.5.47 `QString QEShape::userLevelUserStyle [read, write]`

Style Sheet string to be applied when the widget is displayed in 'User' mode. Default is an empty string. The syntax is the standard Qt Style Sheet syntax. For example, 'background-color: red' This Style Sheet string will be applied by the `styleManager` class. Refer to the `styleManager` class for details about how this Style Sheet string will be merged with any pre-existing Style Sheet string and any Style Sheet strings generated during the display of data.

9.91.5.48 `UserLevels QEShape::userLevelVisibility [read, write]`

Lowest user level at which the widget is visible. Default is 'User'. Used when designing GUIs that display more and more detail according to the user mode. The user mode is set application wide through the [QELogin](#) widget, or programmatically through `setUserLevel()`. Widgets that are always visible should be visible at 'User'. Widgets that are only used by scientists managing the facility should be visible at 'Scientist'. Widgets that are only used by engineers maintaining the facility should be visible at 'Engineer'.

9.91.5.49 `QString QEShape::variable1 [read, write]`

EPICS variable name (CA PV). This variable is read and used to animate an attribute of the shape. The value read is first scaled and offset by properties `scale1` and `offset1`

then the attribute selected for animation is selected by the property animation1.

9.91.5.50 QString QEShape::variable2 [read, write]

EPICS variable name (CA PV). This variable is read and used to animate an attribute of the shape. The value read is first scaled and offset by properties scale2 and offset2 then the attribute selected for animation is selected by the property animation2.

9.91.5.51 QString QEShape::variable3 [read, write]

EPICS variable name (CA PV). This variable is read and used to animate an attribute of the shape. The value read is first scaled and offset by properties scale3 and offset3 then the attribute selected for animation is selected by the property animation3.

9.91.5.52 QString QEShape::variable4 [read, write]

EPICS variable name (CA PV). This variable is read and used to animate an attribute of the shape. The value read is first scaled and offset by properties scale4 and offset4 then the attribute selected for animation is selected by the property animation4.

9.91.5.53 QString QEShape::variable5 [read, write]

EPICS variable name (CA PV). This variable is read and used to animate an attribute of the shape. The value read is first scaled and offset by properties scale5 and offset5 then the attribute selected for animation is selected by the property animation5.

9.91.5.54 QString QEShape::variable6 [read, write]

EPICS variable name (CA PV). This variable is read and used to animate an attribute of the shape. The value read is first scaled and offset by properties scale6 and offset6 then the attribute selected for animation is selected by the property animation6.

9.91.5.55 bool QEShape::variableAsToolTip [read, write]

Use the variable as the tool tip. Default is true. Tool tip property will be overwritten by the variable name.

9.91.5.56 QString QEShape::variableSubstitutions [read, write]

Macro substitutions. The default is no substitutions. The format is NAME1=VALUE1[,] NAME2=VALUE2... Values may be quoted strings. For example, 'SAMPLE=SAM1, NAME = "Ref foil"' These substitutions are applied to all the variable names.

9.91.5.57 bool QEShape::visible [read, write]

Display the widget. Default is true. Setting this property false is usefull if widget is only used to provide a signal - for example, when supplying data to a [QELink](#) widget. Note, when false the widget will still be visible in Qt Designer.

The documentation for this class was generated from the following files:

- /tmp/epicsqt/trunk/framework/widgets/QEShape/QEShape.h
- /tmp/epicsqt/trunk/framework/widgets/QEShape/QEShape.cpp

9.92 QESlider Class Reference

Public Types

- enum **UserLevels** { **User** = userLevelTypes::USERLEVEL_USER, **Scientist** = userLevelTypes::USERLEVEL_SCIENTIST, **Engineer** = userLevelTypes::USERLEVEL_ENGINEER }
- enum **DisplayAlarmStateOptions** { **Never** = standardProperties::DISPLAY_ALARM_STATE_NEVER, **Always** = standardProperties::DISPLAY_ALARM_STATE_ALWAYS, **WhenInAlarm** = standardProperties::DISPLAY_ALARM_STATE_WHEN_IN_ALARM }

Public Slots

- void **setDefaultStyle** (const QString &style)
Update the default style applied to this widget.
- void **setManagedVisible** (bool v)

Signals

- void **dbValueChanged** (const qulonglong &out)

Public Member Functions

- **QESlider** (QWidget *parent=0)
- **QESlider** (const QString &variableName, QWidget *parent=0)
- void **setWriteOnChange** (bool **writeOnChange**)
- bool **getWriteOnChange** ()
- void **setSubscribe** (bool subscribe)
- bool **getSubscribe** ()
- void **setScale** (double scaleIn)
- double **getScale** ()
- void **setOffset** (double offsetIn)
- double **getOffset** ()
- void **setVariableNameProperty** (QString variableName)

Property access function for `variable` property. This has special behaviour to work well within designer.

- `QString getVariableNameProperty ()`

Property access function for `variable` property. This has special behaviour to work well within designer.

- `void setVariableNameSubstitutionsProperty (QString variableNameSubstitutions)`

Property access function for `variableSubstitutions` property. This has special behaviour to work well within designer.

- `QString getVariableNameSubstitutionsProperty ()`

Property access function for `variableSubstitutions` property. This has special behaviour to work well within designer.

- `UserLevels getUserLevelVisibilityProperty ()`

Access function for `userLevelVisibility` property - refer to `userLevelVisibility` property for details.

- `void setUserLevelVisibilityProperty (UserLevels level)`

Access function for `userLevelVisibility` property - refer to `userLevelVisibility` property for details.

- `UserLevels getUserLevelEnabledProperty ()`

Access function for `userLevelEnabled` property - refer to `userLevelEnabled` property for details.

- `void setUserLevelEnabledProperty (UserLevels level)`

Access function for `userLevelEnabled` property - refer to `userLevelEnabled` property for details.

- `DisplayAlarmStateOptions getDisplayAlarmStateOptionProperty ()`

Access function for `displayAlarmStateOption` property - refer to `displayAlarmStateOption` property for details.

- `void setDisplayAlarmStateOptionProperty (DisplayAlarmStateOptions option)`

Access function for `displayAlarmStateOption` property - refer to `displayAlarmStateOption` property for details.

Protected Member Functions

- `void establishConnection (unsigned int variableIndex)`
- `void dragEnterEvent (QDragEnterEvent *event)`
- `void dropEvent (QDropEvent *event)`
- `void setDrop (QVariant drop)`
- `QVariant getDrop ()`
- `QString copyVariable ()`
- `QVariant copyData ()`
- `void paste (QVariant s)`

Protected Attributes

- `QEFloatingFormatting floatingFormatting`
- `bool writeOnChange`

Properties

- QString [variable](#)
- QString [variableSubstitutions](#)
- bool [subscribe](#)
- bool [variableAsToolTip](#)
- bool [allowDrop](#)
- bool [visible](#)
- unsigned [int](#)
- QString [styleSheet](#)
- QString [defaultStyle](#)
- QString [userLevelUserStyle](#)
- QString [userLevelScientistStyle](#)
- QString [userLevelEngineerStyle](#)
- [UserLevels userLevelVisibility](#)
- [UserLevels userLevelEnabled](#)
- bool [displayAlarmState](#)
- [DisplayAlarmStateOptions displayAlarmStateOption](#)
- double [value](#)
- int [sliderPosition](#)

9.92.1 Member Enumeration Documentation

9.92.1.1 enum QESlider::DisplayAlarmStateOptions

User friendly enumerations for [displayAlarmStateOption](#) property - refer to [displayAlarmStateOption](#) property and [displayAlarmStateOptions](#) enumeration for details.

Enumerator:

- Never** Refer to DISPLAY_ALARM_STATE_NEVER for details.
Always Refer to DISPLAY_ALARM_STATE_ALWAYS for details.
WhenInAlarm Refer to DISPLAY_ALARM_STATE_WHEN_IN_ALARM for details.

9.92.1.2 enum QESlider::UserLevels

User friendly enumerations for [userLevelVisibility](#) and [userLevelEnabled](#) properties - refer to [userLevelVisibility](#) and [userLevelEnabled](#) properties and [userLevel](#) enumeration for details.

Enumerator:

- User** Refer to USERLEVEL_USER for details.
Scientist Refer to USERLEVEL_SCIENTIST for details.
Engineer Refer to USERLEVEL_ENGINEER for details.

9.92.2 Member Function Documentation

9.92.2.1 `void QESlider::dbValueChanged (const qulonglong & out) [signal]`

Sent when the widget is updated following a data change. Can be used to pass on EPICS data (as presented in this widget) to other widgets. For example a QList widget could log updates from this widget.

9.92.2.2 `void QESlider::setManagedVisible (bool v) [inline, slot]`

Slot to set the visibility of a QE widget, taking into account the user level. Widget will be hidden if hidden by a call to this slot, by will only be made visible by a call to this slot if the user level allows.

9.92.3 Member Data Documentation

9.92.3.1 `bool QESlider::writeOnChange [read, write, protected]`

Sets if this widget writes any changes as the user moves the slider (the QSlider 'valueChanged' signal is emitted). Default is 'true' (writes any changes when the QSlider 'valueChanged' signal is emitted).

9.92.4 Property Documentation

9.92.4.1 `bool QESlider::allowDrop [read, write]`

Allow drag/drops operations to this widget. Default is false. Any dropped text will be used as a new variable name.

9.92.4.2 `QString QESlider::defaultStyle [read, write]`

Style Sheet string to be applied before, i.e. lower priority than, any other style, e.g. alarm style and/or user level style. Default is an empty string.

9.92.4.3 `bool QESlider::displayAlarmState [read, write]`

DEPRECATED. USE `displayAlarmStateOption` INSTEAD. If set (default) widget will indicate the alarm state of any variable data it is displaying. If clear widget will never indicate the alarm state of any variable data it is displaying. Typically the background colour is set to indicate the alarm state. Note, this property is included in the set of standard properties as it applies to most widgets. It will do nothing for widgets that don't display data.

9.92.4.4 DisplayAlarmStateOptions `QESlider::displayAlarmStateOption` [read, write]

If 'Always' (default) widget will indicate the alarm state of any variable data it is displaying, including 'No Alarm'. If 'Never' widget will never indicate the alarm state of any variable data it is displaying. If 'WhenInAlarm' widget only indicate the alarm state of any variable data it is displaying if it is 'in alarm'. Typically the background colour is set to indicate the alarm state. Note, this property is included in the set of standard properties as it applies to most widgets. It will do nothing for widgets that don't display data.

9.92.4.5 unsigned `QESlider::int` [read, write]

Set the ID used by the message filtering system. Default is zero. Widgets or applications that use messages from the framework have the option of filtering on this ID. For example, by using a unique message source ID a [QELog](#) widget may be set up to only log messages from a select set of widgets.

9.92.4.6 QString `QESlider::styleSheet` [read, write]

Hide style sheet from designer as style calculation by the styleManager and not directly setable per se. This also stops transient styles being saved to the ui file.

9.92.4.7 bool `QESlider::subscribe` [read, write]

Sets if this widget subscribes for data updates and displays current data. Default is 'true' (subscribes for and displays data updates)

9.92.4.8 UserLevels `QESlider::userLevelEnabled` [read, write]

Lowest user level at which the widget is enabled. Default is 'User'. Used when designing GUIs that allow access to more and more detail according to the user mode. The user mode is set application wide through the [QELogin](#) widget, or programmatically through `setUserLevel()`. Widgets that are always accessible should be visible at 'User'. Widgets that are only accessible to scientists managing the facility should be visible at 'Scientist'. Widgets that are only accessible to engineers maintaining the facility should be visible at 'Engineer'.

9.92.4.9 QString `QESlider::userLevelEngineerStyle` [read, write]

Style Sheet string to be applied when the widget is displayed in 'Engineer' mode. Default is an empty string. The syntax is the standard Qt Style Sheet syntax. For example, 'background-color: red' This Style Sheet string will be applied by the styleManager class. Refer to the styleManager class for details about how this Style Sheet string will be merged with any pre-existing Style Sheet string and any Style Sheet strings generated during the display of data.

9.92.4.10 QString QESlider::userLevelScientistStyle [read, write]

Style Sheet string to be applied when the widget is displayed in 'Scientist' mode. Default is an empty string. The syntax is the standard Qt Style Sheet syntax. For example, 'background-color: red' This Style Sheet string will be applied by the styleManager class. Refer to the styleManager class for details about how this Style Sheet string will be merged with any pre-existing Style Sheet string and any Style Sheet strings generated during the display of data.

9.92.4.11 QString QESlider::userLevelUserStyle [read, write]

Style Sheet string to be applied when the widget is displayed in 'User' mode. Default is an empty string. The syntax is the standard Qt Style Sheet syntax. For example, 'background-color: red' This Style Sheet string will be applied by the styleManager class. Refer to the styleManager class for details about how this Style Sheet string will be merged with any pre-existing Style Sheet string and any Style Sheet strings generated during the display of data.

9.92.4.12 UserLevels QESlider::userLevelVisibility [read, write]

Lowest user level at which the widget is visible. Default is 'User'. Used when designing GUIs that display more and more detail according to the user mode. The user mode is set application wide through the [QELogin](#) widget, or 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'.

9.92.4.13 QString QESlider::variable [read, write]

EPICS variable name (CA PV)

9.92.4.14 bool QESlider::variableAsToolTip [read, write]

Use the variable as the tool tip. Default is true. Tool tip property will be overwritten by the variable name.

9.92.4.15 QString QESlider::variableSubstitutions [read, write]

Macro substitutions. The default is no substitutions. The format is NAME1=VALUE1[,] NAME2=VALUE2... Values may be quoted strings. For example, 'PUMP=PMP3, NAME = "My Pump"' These substitutions are applied to variable names for all QE widgets. In some widgets are also used for other purposes.

9.92.4.16 bool QESlider::visible [read, write]

Display the widget. Default is true. Setting this property false is usefull if widget is only used to provide a signal - for example, when supplying data to a [QELink](#) widget. Note, when false the widget will still be visible in Qt Designer.

The documentation for this class was generated from the following files:

- /tmp/epicsqt/trunk/framework/widgets/QESlider/QESlider.h
- /tmp/epicsqt/trunk/framework/widgets/QESlider/QESlider.cpp

9.93 QESpinBox Class Reference

Public Types

- enum [UserLevels](#) { [User](#) = userLevelTypes::USERLEVEL_USER, [Scientist](#) = userLevelTypes::USERLEVEL_SCIENTIST, [Engineer](#) = userLevelTypes::USERLEVEL_ENGINEER }
- enum [DisplayAlarmStateOptions](#) { [Never](#) = standardProperties::DISPLAY_ALARM_STATE_NEVER, [Always](#) = standardProperties::DISPLAY_ALARM_STATE_ALWAYS, [WhenInAlarm](#) = standardProperties::DISPLAY_ALARM_STATE_WHEN_IN_ALARM }

Public Slots

- void [setDefaultCellStyle](#) (const QString &style)
Update the default style applied to this widget.
- void [setManagedVisible](#) (bool v)

Signals

- void [dbValueChanged](#) (const double &out)
- void [userChange](#) (const QString &oldValue, const QString &newValue, const QString &lastValue)
Internal use only. Used by [QEConfiguredLayout](#) to be notified when one of its widgets has written something.

Public Member Functions

- **QESpinBox** (QWidget *parent=0)
- **QESpinBox** (const QString &variableName, QWidget *parent=0)
- void **setWriteOnChange** (bool writeOnChangeIn)
- bool **getWriteOnChange** ()
- 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

- QEFormatting **floatingFormatting**
- bool **writeOnChange**
- bool **addUnitsAsSuffix**
- bool **useDbPrecisionForDecimal**

Properties

- QString **variable**
- QString **variableSubstitutions**
- bool **variableAsToolTip**
- bool **allowDrop**
- bool **visible**
- unsigned **int**
- QString **styleSheet**
- QString **defaultStyle**
- QString **userLevelUserStyle**
- QString **userLevelScientistStyle**
- QString **userLevelEngineerStyle**
- UserLevels **userLevelVisibility**
- UserLevels **userLevelEnabled**
- bool **displayAlarmState**
- DisplayAlarmStateOptions **displayAlarmStateOption**
- bool **subscribe**
- bool **useDbPrecision**
- bool **addUnits**
- double **value**

9.93.1 Member Enumeration Documentation

9.93.1.1 enum QESpinBox::DisplayAlarmStateOptions

User friendly enumerations for **displayAlarmStateOption** property - refer to **displayAlarmStateOption** property and **displayAlarmStateOptions** enumeration for details.

Enumerator:

Never Refer to DISPLAY_ALARM_STATE_NEVER for details.

Always Refer to DISPLAY_ALARM_STATE_ALWAYS for details.

WhenInAlarm Refer to DISPLAY_ALARM_STATE_WHEN_IN_ALARM for details.

9.93.1.2 enum QESpinBox::UserLevels

User friendly enumerations for `userLevelVisibility` and `userLevelEnabled` properties - refer to `userLevelVisibility` and `userLevelEnabled` properties and userLevel enumeration for details.

Enumerator:

- User** Refer to USERLEVEL_USER for details.
- Scientist** Refer to USERLEVEL_SCIENTIST for details.
- Engineer** Refer to USERLEVEL_ENGINEER for details.

9.93.2 Member Function Documentation

9.93.2.1 void QESpinBox::dbValueChanged (const double & out) [signal]

Sent when the widget is updated following a data change Can be used to pass on EPICS data (as presented in this widget) to other widgets. For example a QList widget could log updates from this widget.

9.93.2.2 void QESpinBox::setManagedVisible (bool v) [inline, slot]

Slot to set the visibility of a QE widget, taking into account the user level. Widget will be hidden if hidden by a call this slot, by will only be made visible by a call to this slot if the user level allows.

9.93.3 Property Documentation

9.93.3.1 bool QESpinBox::allowDrop [read, write]

Allow drag/drops operations to this widget. Default is false. Any dropped text will be used as a new variable name.

9.93.3.2 QString QESpinBox::defaultStyle [read, write]

Style Sheet string to be applied before, i.e. lower priority than, any other style, e.g. alarm style and/or user level style. Default is an empty string.

9.93.3.3 bool QESpinBox::displayAlarmState [read, write]

DEPRECATED. USE `displayAlarmStateOption` INSTEAD. If set (default) widget will indicate the alarm state of any variable data it is displaying. If clear widget will never indicate the alarm state of any variable data it is displaying. Typically the background colour is set to indicate the alarm state. Note, this property is included in the set of standard properties as it applies to most widgets. It will do nothing for widgets that don't display data.

9.93.3.4 DisplayAlarmStateOptions `QESpinBox::displayAlarmStateOption` [read, write]

If 'Always' (default) widget will indicate the alarm state of any variable data it is displaying, including 'No Alarm'. If 'Never' widget will never indicate the alarm state of any variable data it is displaying. If 'WhenInAlarm' widget only indicate the alarm state of any variable data it is displaying if it is 'in alarm'. Typically the background colour is set to indicate the alarm state. Note, this property is included in the set of standard properties as it applies to most widgets. It will do nothing for widgets that don't display data.

9.93.3.5 unsigned `QESpinBox::int` [read, write]

Set the ID used by the message filtering system. Default is zero. Widgets or applications that use messages from the framework have the option of filtering on this ID. For example, by using a unique message source ID a [QELog](#) widget may be set up to only log messages from a select set of widgets.

9.93.3.6 QString `QESpinBox::styleSheet` [read, write]

Hide style sheet from designer as style calculation by the styleManager and not directly setable per se. This also stops transient styles being saved to the ui file.

9.93.3.7 bool `QESpinBox::subscribe` [read, write]

Sets if this widget subscribes for data updates and displays current data. Default is 'true' (subscribes for and displays data updates)

9.93.3.8 UserLevels `QESpinBox::userLevelEnabled` [read, write]

Lowest user level at which the widget is enabled. Default is 'User'. Used when designing GUIs that allow access to more and more detail according to the user mode. The user mode is set application wide through the [QELogin](#) widget, or programmatically through `setUserLevel()`. Widgets that are always accessible should be visible at 'User'. Widgets that are only accessible to scientists managing the facility should be visible at 'Scientist'. Widgets that are only accessible to engineers maintaining the facility should be visible at 'Engineer'.

9.93.3.9 QString `QESpinBox::userLevelEngineerStyle` [read, write]

Style Sheet string to be applied when the widget is displayed in 'Engineer' mode. Default is an empty string. The syntax is the standard Qt Style Sheet syntax. For example, 'background-color: red' This Style Sheet string will be applied by the styleManager class. Refer to the styleManager class for details about how this Style Sheet string will be merged with any pre-existing Style Sheet string and any Style Sheet strings generated during the display of data.

9.93.3.10 QString QESpinBox::userLevelScientistStyle [read, write]

Style Sheet string to be applied when the widget is displayed in 'Scientist' mode. Default is an empty string. The syntax is the standard Qt Style Sheet syntax. For example, 'background-color: red' This Style Sheet string will be applied by the styleManager class. Refer to the styleManager class for details about how this Style Sheet string will be merged with any pre-existing Style Sheet string and any Style Sheet strings generated during the display of data.

9.93.3.11 QString QESpinBox::userLevelUserStyle [read, write]

Style Sheet string to be applied when the widget is displayed in 'User' mode. Default is an empty string. The syntax is the standard Qt Style Sheet syntax. For example, 'background-color: red' This Style Sheet string will be applied by the styleManager class. Refer to the styleManager class for details about how this Style Sheet string will be merged with any pre-existing Style Sheet string and any Style Sheet strings generated during the display of data.

9.93.3.12 UserLevels QESpinBox::userLevelVisibility [read, write]

Lowest user level at which the widget is visible. Default is 'User'. Used when designing GUIs that display more and more detail according to the user mode. The user mode is set application wide through the [QELogin](#) widget, or 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'.

9.93.3.13 QString QESpinBox::variable [read, write]

EPICS variable name (CA PV)

9.93.3.14 bool QESpinBox::variableAsToolTip [read, write]

Use the variable as the tool tip. Default is true. Tool tip property will be overwritten by the variable name.

9.93.3.15 QString QESpinBox::variableSubstitutions [read, write]

Macro substitutions. The default is no substitutions. The format is NAME1=VALUE1[,] NAME2=VALUE2... Values may be quoted strings. For example, 'PUMP=PMP3, NAME = "My Pump"' These substitutions are applied to variable names for all QE widgets. In some widgets are also used for other purposes.

9.93.3.16 bool QESpinBox::visible [read, write]

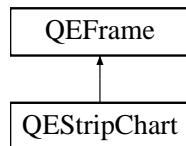
Display the widget. Default is true. Setting this property false is usefull if widget is only used to provide a signal - for example, when supplying data to a [QELink](#) widget. Note, when false the widget will still be visible in Qt Designer.

The documentation for this class was generated from the following files:

- /tmp/epicsqt/trunk/framework/widgets/QESpinBox/QESpinBox.h
- /tmp/epicsqt/trunk/framework/widgets/QESpinBox/QESpinBox.cpp

9.94 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 endDateTimeIn)
- int **duration** () const
- void **setDuration** (int durationIn)
- double **YMinimum** () const
- void **setYMinimum** (const double yMinimumIn)
- double **YMaximum** () 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](#)

9.94.1 Property Documentation

9.94.1.1 QString QEStripChart::variableSubstitutions [read, write]

Macro substitutions. The default is no substitutions. The format is NAME1=VALUE1[,]
NAME2=VALUE2... Values may be quoted strings. For example, 'SAMPLE=SAM1,
NAME = "Ref foil"' These substitutions are applied to all the variable names.

The documentation for this class was generated from the following files:

- /tmp/epicsqt/trunk/framework/widgets/QEStripChart/QEStripChart.h
- /tmp/epicsqt/trunk/framework/widgets/QEStripChart/QEStripChart.cpp

9.95 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

9.96 QEStripChartContextMenu Class Reference

Signals

- void **contextMenuSelected** (const QEStripChartNames::ContextMenuOptions)

Public Member Functions

- [QEStripChartContextMenu](#) (bool inUse, QWidget *parent=0)
- void **setPredefinedNames** (const QStringList &pvlList)
- void **setUseReceiveTime** (const bool useReceiveTime)
- void **setArchiveReadHow** (const QEArchiveInterface::How how)
- void **setLineDrawMode** (const QEStripChartNames::LineDrawModes mode)

9.96.1 Constructor & Destructor Documentation

9.96.1.1 QEStripChartContextMenu::QEStripChartContextMenu (bool *inUse*, QWidget * *parent* = 0) [explicit]

Construct strip chart item context menu. This menu item creates all required sub menu items. inUse set true for an inuse slot, i.e. already has a PV allocated. inUse set false for an empty slot.

The documentation for this class was generated from the following files:

- /tmp/epicsqt/trunk/framework/widgets/QEStripChart/QEStripChartContextMenu.h
- /tmp/epicsqt/trunk/framework/widgets/QEStripChart/QEStripChartContextMenu.cpp

9.97 QEStripChartDurationDialog Class Reference

Public Member Functions

- [QEStripChartDurationDialog](#) (QWidget *parent=0)
- void **setDuration** (int secs)
- int **getDuration** () const

The documentation for this class was generated from the following files:

- /tmp/epicsqt/trunk/framework/widgets/QEStripChart/QEStripChartDurationDialog.h
- /tmp/epicsqt/trunk/framework/widgets/QEStripChart/QEStripChartDurationDialog.cpp

9.98 QEStripChartItem Class Reference

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

9.99 QEStripChartNames Class Reference

Public Types

- enum **ChartTimeModes** { tmRealTime, tmPaused, tmHistorical }
- enum **ChartYRanges** {
 manual, **operatingRange**, **plotted**, **buffered**,
 dynamic, **normalised** }

```

• enum PlayModes {
    play, pause, forward, backward,
    selectTimes }
• enum StateModes { previous, next }
• enum VideoModes { normal, reverse }
• enum YScaleModes { linear, log }
• enum LineDrawModes { ldmHide, ldmRegular, ldmBold }
• enum ContextMenuOptions {
    SCCM_NONE = contextMenu::CM_SPECIFIC_WIDGETS_START_HERE, SCCM_READ_ARCHIVE, SCCM_SCALE_CHART_AUTO, SCCM_SCALE_CHART_PLOTTED,
    SCCM_SCALE_CHART_BUFFERED, SCCM_SCALE_PV_RESET, SCCM_SCALE_PV_GENERAL, SCCM_SCALE_PV_AUTO,
    SCCM_SCALE_PV_PLOTTED, SCCM_SCALE_PV_BUFFERED, SCCM_SCALE_PV_CENTRE, SCCM_PLOT_RECTANGULAR,
    SCCM_PLOT_SMOOTH, SCCM_PLOT_SERVER_TIME, SCCM_PLOT_CLIENT_TIME, SCCM_ARCH_LINEAR,
    SCCM_ARCH_PLOTBIN, SCCM_ARCH_RAW, SCCM_ARCH_SHEET, SCCM_ARCH_AVERAGED,
    SCCM_LINE_HIDE, SCCM_LINE_REGULAR, SCCM_LINE_BOLD, SCCM_LINE_COLOUR,
    SCCM_PV_EDIT_NAME, SCCM_ADD_TO_PREDEFINED, SCCM_PV_WRITE_TRACE, SCCM_PV_STATS,
    SCCM_PV_CLEAR, SCCM_PV_ADD_NAME, SCCM_PV_PASTE_NAME, SCCM_PREDEFINED_01,
    SCCM_PREDEFINED_02, SCCM_PREDEFINED_03, SCCM_PREDEFINED_04, SCCM_PREDEFINED_05,
    SCCM_PREDEFINED_06, SCCM_PREDEFINED_07, SCCM_PREDEFINED_08, SCCM_PREDEFINED_09,
    SCCM_PREDEFINED_10 }

```

Static Public Attributes

- static const ContextMenuOptions **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

9.100 QEStripChartPushButtonSpecifications Struct Reference

Public Attributes

- int **gap**
- int **width**
- int **value**
- bool **isIcon**
- const QString **captionOrIcon**
- const QString **toolTip**
- const char * **member**

The documentation for this struct was generated from the following file:

- /tmp/epicsqt/trunk/framework/widgets/QEStripChart/QEStripChartToolBar.cpp

9.101 QEStripChartRangeDialog Class Reference

Public Member Functions

- **QEStripChartRangeDialog** (QWidget *parent=0)
- void **setRange** (const double min, const double max)
- double **getMinimum** ()
- double **getMaximum** ()

The documentation for this class was generated from the following files:

- /tmp/epicsqt/trunk/framework/widgets/QEStripChart/QEStripChartRangeDialog.h
- /tmp/epicsqt/trunk/framework/widgets/QEStripChart/QEStripChartRangeDialog.cpp

9.102 QEStripChartState Class Reference

Public Member Functions

- void **saveConfiguration** (PMElement &parentElement)
- void **restoreConfiguration** (PMElement &parentElement)

Public Attributes

- bool **isNormalVideo**
- QEStripChartNames::ChartTimeModes **chartTimeMode**
- QEStripChartNames::YScaleModes **yScaleMode**
- QEStripChartNames::ChartYRanges **chartYScale**

- double **yMinimum**
- double **yMaximum**
- int **duration**
- Qt::TimeSpec **timeZoneSpec**
- QDateTime **endDateTime**

The documentation for this class was generated from the following files:

- /tmp/epicsqt/trunk/framework/widgets/QEStripChart/QEStripChartState.h
- /tmp/epicsqt/trunk/framework/widgets/QEStripChart/QEStripChartState.cpp

9.103 QEStripChartStateList Class Reference

Public Member Functions

- void **clear** ()
- void **push** (const QEStripChartState &state)
- bool **prev** (QEStripChartState &state)
- bool **next** (QEStripChartState &state)
- bool **prevAvailable** ()
- bool **nextAvailable** ()

The documentation for this class was generated from the following files:

- /tmp/epicsqt/trunk/framework/widgets/QEStripChart/QEStripChartState.h
- /tmp/epicsqt/trunk/framework/widgets/QEStripChart/QEStripChartState.cpp

9.104 QEStripChartStatistics Class Reference

Public Member Functions

- **QEStripChartStatistics** (const QString &pvName, const QString &egu, const QCaDataPointList &dataList, QEStripChartItem *owner, QWidget *parent=0)

The documentation for this class was generated from the following files:

- /tmp/epicsqt/trunk/framework/widgets/QEStripChart/QEStripChartStatistics.h
- /tmp/epicsqt/trunk/framework/widgets/QEStripChart/QEStripChartStatistics.cpp

9.105 QEStripChartTimeDialog Class Reference

Public Member Functions

- **QEStripChartTimeDialog** (QWidget *parent=0)
- void **setMaximumDateTime** (QDateTime datetime)
- void **setStartTime** (QDateTime datetime)
- QDateTime **getStartTime** ()
- void **setEndTime** (QDateTime datetime)
- QDateTime **getEndTime** ()

The documentation for this class was generated from the following files:

- /tmp/epicsqt/trunk/framework/widgets/QEStripChart/QEStripChartTimeDialog.h
- /tmp/epicsqt/trunk/framework/widgets/QEStripChart/QEStripChartTimeDialog.cpp

9.106 QEStripChartToolBar Class Reference

This class holds all the StripChart tool bar widgets.

```
#include <QEStripChartToolBar.h>
```

Classes

- class [OwnWidgets](#)

Signals

- void **stateSelected** (const QEStripChartNames::StateModes mode)
- void **videoModeSelected** (const QEStripChartNames::VideoModes mode)
- void **yScaleModeSelected** (const QEStripChartNames::YScaleModes mode)
- void **yRangeSelected** (const QEStripChartNames::ChartYRanges scale)
- void **durationSelected** (const int seconds)
- void **selectDuration** ()
- void **timeZoneSelected** (const Qt::TimeSpec timeSpec)
- void **playModeSelected** (const QEStripChartNames::PlayModes mode)
- void **readArchiveSelected** ()

Public Member Functions

- **QEStripChartToolBar** (QWidget *parent=0)
- void **setYRangeStatus** (const QString &status)
- void **setTimeStatus** (const QString &timeStatus)
- void **setDurationStatus** (const QString &durationStatus)
- void **setStateSelectionEnabled** (const QEStripChartNames::StateModes mode, const bool enabled)

Static Public Attributes

- static const int **designHeight** = 44

Protected Member Functions

- void **resizeEvent** (QResizeEvent *event)

9.106.1 Detailed Description

This class holds all the StripChart tool bar widgets.

The documentation for this class was generated from the following files:

- /tmp/epicsqt/trunk/framework/widgets/QEStripChart/QEStripChartToolBar.h
- /tmp/epicsqt/trunk/framework/widgets/QEStripChart/QEStripChartToolBar.cpp

9.107 QESubstitutedLabel Class Reference

Public Member Functions

- **QESubstitutedLabel** (QWidget *parent=0)
- void **setLabelTextProperty** (QString labelTextIn)
- QString **getLabelTextProperty** ()
- void **setSubstitutionsProperty** (QString macroSubstitutionsIn)
- QString **getSubstitutionsProperty** ()
- QString **getLabelTextPropertyFormat** ()
- void **setLabelTextPropertyFormat** (QString labelTextIn)

Protected Attributes

- QString **labelText**

Properties

- QString **textSubstitutions**

9.107.1 Member Data Documentation

9.107.1.1 QString QESubstitutedLabel::labelText [read, write, protected]

Label text to be substituted. This text will be copied to the label text after applying any macro substitutions from the textSubstitutions property

9.107.2 Property Documentation

9.107.2.1 QString QESubstitutedLabel::textSubstitutions [read, write]

Text substitutions. These substitutions are applied to the 'labelText' property prior to copying it to the label text.

The documentation for this class was generated from the following files:

- /tmp/epicsqt/trunk/framework/widgets/QESubstitutedLabel/QESubstitutedLabel.h
- /tmp/epicsqt/trunk/framework/widgets/QESubstitutedLabel/QESubstitutedLabel.cpp

9.108 recording Class Reference

Signals

- void **byteArrayChanged** (const QByteArray &value, unsigned long dataSize, QCaAlarmInfo &alarmInfo, QCaDateTime &timeStamp, const unsigned int &variableIndex)
- void **playingBack** (bool playing)

Public Member Functions

- **recording** (QWidget *parent=0)
- bool **isRecording** ()
- void **recordImage** (QByteArray image, unsigned long dataSize, QCaAlarmInfo &alarmInfo, QCaDateTime &time)
- void **nextFrameDue** ()

The documentation for this class was generated from the following files:

- /tmp/epicsqt/trunk/framework/widgets/QEImage/recording.h
- /tmp/epicsqt/trunk/framework/widgets/QEImage/recording.cpp

9.109 imageDisplayProperties::rgbPixel Struct Reference

Public Attributes

- unsigned char **p** [4]

The documentation for this struct was generated from the following file:

- /tmp/epicsqt/trunk/framework/widgets/QEImage/brightnessContrast.h

9.110 screenSelectDialog Class Reference

Public Types

- enum **screens** { **PRIMARY_SCREEN** = -3, **THIS_SCREEN** = -2, **ALL_SCREENS** = -1 }

Public Member Functions

- **screenSelectDialog** (int numScreens, QWidget *parent=0)
- int **getScreenNum** ()

Static Public Member Functions

- static bool **getFullscreenGeometry** (QWidget *target, QRect &geom)

The documentation for this class was generated from the following files:

- /tmp/epicsqt/trunk/framework/widgets/QElImage/screenSelectDialog.h
- /tmp/epicsqt/trunk/framework/widgets/QElImage/screenSelectDialog.cpp

9.111 selectMenu Class Reference

Public Member Functions

- **selectMenu** (QWidget *parent=0)
- imageContextMenu::imageContextMenuOptions **getSelectOption** (const QPoint &pos)
- void **enable** (imageContextMenu::imageContextMenuOptions option, bool state)
- bool **isEnabled** (imageContextMenu::imageContextMenuOptions option)
- void **setChecked** (const int mode)
- void **setItemText** (imageContextMenu::imageContextMenuOptions option, QString title)

The documentation for this class was generated from the following files:

- /tmp/epicsqt/trunk/framework/widgets/QElImage/selectMenu.h
- /tmp/epicsqt/trunk/framework/widgets/QElImage/selectMenu.cpp

9.112 trace Class Reference

Public Attributes

- QVector< QCaDateTime > **timeStamps**

- QVector< double > **xdata**
- QVector< double > **ydata**
- QwtPlotCurve * **curve**
- QColor **color**
- QString **legend**
- bool **waveform**
- QwtPlotCurve::CurveStyle **style**

The documentation for this class was generated from the following file:

- /tmp/epicsqt/trunk/framework/widgets/QEPlot/QEPlot.h

9.113 userInfoStruct Class Reference

Public Attributes

- bool **enable**
- double **value1**
- double **value2**
- QString **elementText**

The documentation for this class was generated from the following file:

- /tmp/epicsqt/trunk/framework/widgets/QEPeriodic/QEPeriodic.h

9.114 QEPeriodic::userInfoStructArray Struct Reference

Public Attributes

- **userInfoStruct array [NUM_ELEMENTS]**

The documentation for this struct was generated from the following file:

- /tmp/epicsqt/trunk/framework/widgets/QEPeriodic/QEPeriodic.h

9.115 ValueScaling Class Reference

Public Member Functions

- void **reset** ()
- void **assign** (const [ValueScaling](#) &s)
- void **set** (const double dIn, const double mIn, const double cIn)

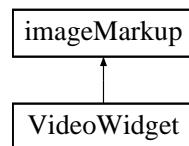
- void **get** (double &dOut, double &mOut, double &cOut) const
- void **map** (const double fromLower, const double fromUpper, const double toLower, const double toUpper)
- bool **isScaled** () const
- double **value** (const double x) const
- QEDisplayRanges **value** (const QEDisplayRanges &x) const
- void **saveConfiguration** (PMElement &parentElement) const
- void **restoreConfiguration** (PMElement &parentElement)

The documentation for this class was generated from the following files:

- /tmp/epicsqt/trunk/framework/widgets/QEStripChart/QEStripChartUtilities.h
- /tmp/epicsqt/trunk/framework/widgets/QEStripChart/QEStripChartUtilities.cpp

9.116 VideoWidget Class Reference

Inheritance diagram for VideoWidget:



Signals

- void **userSelection** (imageMarkup::markupIds mode, bool complete, bool clearing, QPoint point1, QPoint point2, unsigned int thickness)
- void **zoomInOut** (int zoomAmount)
- void **currentPixelInfo** (QPoint pos)
- void **pan** (QPoint pos)
- void **redraw** ()

Public Member Functions

- **VideoWidget** (QWidget *parent=0)
- void **setNewImage** (QImage image, QCaDateTime &time)
- void **setPanning** (bool panningIn)
- bool **getPanning** ()
- QPoint **scalePoint** (QPoint pnt)
- int **scaleOrdinate** (int ord)
- QPoint **scaleImagePoint** (QPoint pnt)
- QRect **scaleImageRectangle** (QRect r)
- int **scaleImageOrdinate** (int ord)

- `QImage getImage ()`
- `QSize getImageSize ()`
- `bool hasCurrentImage ()`
- `void markupChange ()`

Protected Member Functions

- `void paintEvent (QPaintEvent *)`
- `void mousePressEvent (QMouseEvent *event)`
- `void mouseReleaseEvent (QMouseEvent *event)`
- `void mouseMoveEvent (QMouseEvent *event)`
- `void wheelEvent (QWheelEvent *event)`
- `void keyPressEvent (QKeyEvent *event)`
- `void markupChange (QVector< QRect > &changedAreas)`
- `void resizeEvent (QResizeEvent *event)`
- `void markupSetCursor (QCursor cursor)`
- `void markupAction (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`

9.117 zoomMenu Class Reference

Public Member Functions

- `zoomMenu (QWidget *parent=0)`
- `void enableAreaSelected (bool enable)`
- `imageContextMenu::imageContextMenuOptions getZoom (const QPoint &pos)`

The documentation for this class was generated from the following files:

- `/tmp/epicsqt/trunk/framework/widgets/QElImage/zoomMenu.h`
- `/tmp/epicsqt/trunk/framework/widgets/QElImage/zoomMenu.cpp`

Index

_CopyPaste, 27
_Field, 27
_Item, 28
_QDialogItem, 29
_QPushButtonGroup, 29
_QTableWidgetFileBrowser, 29
_QTableWidgetLog, 30
_QTableWidgetScript, 30

addUnits
 QEAnalogProgressBar, 74
 QECheckBox, 92
 QELabel, 196
 QELineEdit, 204
 QENumericEdit, 218
 QEPushButton, 242
 QERadioButton, 261
alarmSeverityDisplayMode
 QEAnalogProgressBar, 74
alignment
 QECheckBox, 92
 QEPushButton, 242
 QERadioButton, 261
allowDrop
 QEAnalogProgressBar, 74
 QEBitStatus, 82
 QECheckBox, 92
 QEComboBox, 105
 QEConfiguredLayout, 110
 QEFileBrowser, 117
 QEForm, 122
 QEFrame, 126
 QEGenericEdit, 135
 QEGroupBox, 140
 QEImage, 171
 QELabel, 196
 QELog, 212
 QEPeriodic, 223
 QEPlot, 232
 QEPushButton, 242
 QERadioButton, 261

QEScript, 278
QEShape, 287
QESlider, 298
QESpinBox, 304
altReadbackVariable
 QEPushButton, 242
Always
 QEAnalogProgressBar, 72
 QEBitStatus, 81
 QECheckBox, 89
 QEComboBox, 104
 QEConfiguredLayout, 110
 QEFileBrowser, 116
 QEFrame, 125
 QEGenericEdit, 133
 QEGroupBox, 140
 QEImage, 167
 QELabel, 194
 QELog, 211
 QEPeriodic, 222
 QEPlot, 231
 QEPushButton, 239
 QERadioButton, 258
 QEScript, 277
 QEShape, 285
 QESlider, 297
 QESpinBox, 303
animation1
 QEShape, 287
animation2
 QEShape, 287
animation3
 QEShape, 287
animation4
 QEShape, 288
animation5
 QEShape, 288
animation6
 QEShape, 288
animationOptions
 QEShape, 285

Append
QEAnalogProgressBar, 72
QECheckBox, 88
QELabel, 194
QELineEdit, 203
QEPushButton, 239
QERadioButton, 257

areaColor
QEImage, 171

arealInfo, 30

arguments
QECheckBox, 92
QEPushButton, 243
QERadioButton, 261

arguments1
QEImage, 171

arguments2
QEImage, 171

arrayAction
QEAnalogProgressBar, 74
QECheckBox, 92
QELabel, 196
QELineEdit, 204
QEPushButton, 243
QERadioButton, 261

ArrayActions
QEAnalogProgressBar, 72
QECheckBox, 88
QELabel, 194
QELineEdit, 203
QEPushButton, 239
QERadioButton, 257

Ascii
QEAnalogProgressBar, 72
QECheckBox, 88
QELabel, 194
QELineEdit, 203
QEPushButton, 239
QERadioButton, 257

autoBrightnessContrast
QEImage, 171

Automatic
QEAnalogProgressBar, 73
QECheckBox, 90
QELabel, 194
QELineEdit, 204
QEPushButton, 240
QERadioButton, 259

autoScale
QENumericEdit, 218

backgroundColour
QEAnalogIndicator, 66

Bar
QEAnalogIndicator, 66

Bayer
QEImage, 167

BayerBG
QEImage, 167

BayerGB
QEImage, 167

BayerGR
QEImage, 167

BayerRG
QEImage, 167

beamColor
QEImage, 171

beamXVariable
QEImage, 171

beamYVariable
QEImage, 171

bitDepthVariable
QEImage, 171

borderColour
QEAnalogIndicator, 66

Bottom_To_Top
QEAnalogIndicator, 66

BOUNDING_RECTANGLE
QEImage, 167

BoundingRectangle
QEImage, 167

briefInfoArea
QEImage, 172

buildImageCore
imagePropertiesCore, 47

CenterAndSize
QEImage, 167

centreAngle
QEAnalogIndicator, 66

clickCheckedText
QECheckBox, 92
QEPushButton, 243
QERadioButton, 262

clicked
QECheckBox, 91
QEPushButton, 241
QERadioButton, 260

clickText
QECheckBox, 93
QEPushButton, 243

QERadioButton, 262
clippingHighVariable
 QEImage, 172
clippingLowVariable
 QEImage, 172
clippingOnOffVariable
 QEImage, 172
color1
 QEShape, 288
color10
 QEShape, 288
color2
 QEShape, 288
color3
 QEShape, 288
color4
 QEShape, 288
color5
 QEShape, 288
color6
 QEShape, 289
color7
 QEShape, 289
color8
 QEShape, 289
color9
 QEShape, 289
confirmAction
 QECheckBox, 93
 QEPushButton, 243
 QERadioButton, 262
confirmText
 QECheckBox, 93
 QEPushButton, 244
 QERadioButton, 262
confirmWrite
 QEGenericEdit, 135
contrastReversal
 QEImage, 172
creationOption
 QECheckBox, 93
 QEPushButton, 244
 QERadioButton, 262
CreationOptionNames
 QECheckBox, 88
 QEPushButton, 239
 QERadioButton, 257
customisationName
 QECheckBox, 93
 QEPushButton, 244
 QERadioButton, 262
 QEImage, 172
 QELabel, 196
 QEadioButton, 263
 dbElementChanged
 QEPeriodic, 223
 dbValueChanged
 QEAnalogProgressBar, 73
 QEBitStatus, 81
 QECheckBox, 91
 QEComboBox, 104
 QEImage, 170
 QELabel, 196
 QELineEdit, 204
 QENumericEdit, 218
 QEPeriodic, 223
 QEPlot, 231
 QEPushButton, 241
 QERadioButton, 260
 QESlider, 298
 QESpinBox, 304
 dbValueChanged1
 QEShape, 286
 dbValueChanged2
 QEShape, 286
 dbValueChanged3
 QEShape, 286
 dbValueChanged4
 QEShape, 286
 dbValueChanged5
 QEShape, 287
 dbValueChanged6
 QEShape, 287
 Default
 QEAnalogProgressBar, 72
 QECheckBox, 89
 QELabel, 194
 QELineEdit, 203
 QEPushButton, 240
 QERadioButton, 258
 defaultStyle
 QEAnalogProgressBar, 74
 QEBitStatus, 82
 QECheckBox, 94
 QEComboBox, 105
 QEConfiguredLayout, 110
 QEFileBrowser, 117
 QEFrame, 126
 QEGenericEdit, 135
 QEGroupBox, 140
 QEImage, 172
 QELabel, 196

QELog, 212
QEPlot, 232
QEPushButton, 244
QERadioButton, 263
QEScript, 278
QEShape, 289
QESlider, 298
QESpinBox, 304
dimension1Variable
 QEImage, 172
dimension2Variable
 QEImage, 172
dimension3Variable
 QEImage, 172
dimensionsVariable
 QEImage, 173
displayAlarmState
 QEAnalogProgressBar, 74
 QEBitStatus, 82
 QECheckBox, 94
 QEComboBox, 105
 QEConfiguredLayout, 110
 QEFileBrowser, 117
 QEFrame, 126
 QEGenericEdit, 135
 QEGroupBox, 141
 QEImage, 173
 QELabel, 196
 QELog, 212
 QEPeriodic, 223
 QEPlot, 232
 QEPushButton, 244
 QERadioButton, 263
 QEScript, 278
 QEShape, 289
 QESlider, 298
 QESpinBox, 304
displayAlarmStateOption
 QEAnalogProgressBar, 75
 QEBitStatus, 82
 QECheckBox, 94
 QEComboBox, 105
 QEConfiguredLayout, 111
 QEFileBrowser, 117
 QEForm, 122
 QEFrame, 126
 QEGenericEdit, 136
 QEGroupBox, 141
 QEImage, 173
 QELabel, 197
QELog, 212
QEPeriodic, 223
QEPlot, 232
QEPushButton, 244
QERadioButton, 263
QEScript, 278
QEShape, 289
QESlider, 298
QESpinBox, 304
DisplayAlarmStateOptions
 QEAnalogProgressBar, 72
 QEBitStatus, 81
 QECheckBox, 89
 QEComboBox, 103
 QEConfiguredLayout, 110
 QEFileBrowser, 116
 QEFrame, 125
 QEGenericEdit, 133
 QEGroupBox, 140
 QEImage, 167
 QELabel, 194
 QELog, 211
 QEPeriodic, 222
 QEPlot, 231
 QEPushButton, 239
 QERadioButton, 258
 QEScript, 277
 QEShape, 285
 QESlider, 297
 QESpinBox, 303
displayArea1Selection
 QEImage, 173
displayArea2Selection
 QEImage, 173
displayArea3Selection
 QEImage, 173
displayArea4Selection
 QEImage, 173
displayBeamSelection
 QEImage, 174
displayButtonBar
 QEImage, 170
displayCursorPixelInfo
 QEImage, 174
displayEllipse
 QEImage, 174
displayHozSlice1Selection
 QEImage, 174
displayHozSlice2Selection
 QEImage, 174

displayHozSlice3Selection
QEImage, 174
displayHozSlice4Selection
QEImage, 174
displayHozSlice5Selection
QEImage, 174
displayProfileSelection
QEImage, 174
displayTargetSelection
QEImage, 175
displayVertSliceSelection
QEImage, 175
DockBottom
QECheckBox, 89
QEPushButton, 239
QERadioButton, 258
DockBottomTabbed
QECheckBox, 89
QEPushButton, 239
QERadioButton, 258
DockFloating
QECheckBox, 89
QEPushButton, 239
QERadioButton, 258
DockLeft
QECheckBox, 89
QEPushButton, 239
QERadioButton, 258
DockLeftTabbed
QECheckBox, 89
QEPushButton, 239
QERadioButton, 258
DockRight
QECheckBox, 89
QEPushButton, 239
QERadioButton, 258
DockRightTabbed
QECheckBox, 89
QEPushButton, 239
QERadioButton, 258
DockTop
QECheckBox, 89
QEPushButton, 239
QERadioButton, 258
DockTopTabbed
QECheckBox, 89
QEPushButton, 239
QERadioButton, 258
DottedFullCrosshair
QEImage, 169
drawMarkup
markupHLine, 51
markupVLine, 56
ellipseColor
QEImage, 175
ellipseHVariable
QEImage, 175
EllipseVariableDefinitions
QEImage, 167
ellipseVariableDefinitions
QEImage, 167
ellipseWVariable
QEImage, 175
ellipseXVariable
QEImage, 175
ellipseYVariable
QEImage, 175
enableArea1Selection
QEImage, 176
enableArea2Selection
QEImage, 176
enableArea3Selection
QEImage, 176
enableArea4Selection
QEImage, 176
enableBeamSelection
QEImage, 176
enableHozSlice1Selection
QEImage, 176
enableHozSlice2Selection
QEImage, 176
enableHozSlice3Selection
QEImage, 176
enableHozSlice4Selection
QEImage, 177
enableHozSlice5Selection
QEImage, 177
enableProfileSelection
QEImage, 177
enableTargetSelection
QEImage, 177
enableVertSlice1Selection
QEImage, 177
enableVertSlice2Selection
QEImage, 177
enableVertSlice3Selection
QEImage, 177
enableVertSlice4Selection
QEImage, 178

enableVertSlice5Selection
QEImage, 178

Engineer
QEAnalogProgressBar, 73
QEBitStatus, 81
QECheckBox, 90
QEComboBox, 104
QEConfiguredLayout, 110
QEFileBrowser, 117
QEFrame, 125
QEGenericEdit, 133
QEGroupBox, 140
QEImage, 169
QELabel, 195
QELog, 212
QEPeriodic, 223
QEPlot, 231
QEPushButton, 241
QERadioButton, 260
QEScript, 278
QEShape, 286
QESlider, 297
QESpinBox, 304

externalControls
QEImage, 178

FFBuffer, 32
FFThread, 32

Fit
QEImage, 168

Fixed
QEAnalogProgressBar, 73
QECheckBox, 90
QELabel, 194
QELineEdit, 204
QEPushButton, 240
QERadioButton, 259

flipRotateMenu, 33

Floating
QEAnalogProgressBar, 72
QECheckBox, 89
QELabel, 194
QELineEdit, 203
QEPushButton, 240
QERadioButton, 258

fontColour
QEAnalogIndicator, 66

foregroundColour
QEAnalogIndicator, 67

format
QEAnalogProgressBar, 75
QECheckBox, 94
QELabel, 197
QELineEdit, 205
QEPushButton, 245
QERadioButton, 263

formatOption
QEImage, 178

FormatOptions
QEImage, 167

Formats
QEAnalogProgressBar, 72
QECheckBox, 89
QELabel, 194
QELineEdit, 203
QEPushButton, 239
QERadioButton, 258

formatVariable
QEImage, 178

fullScreenWindow, 33

getConfirmWrite
QEGenericEdit, 134

getPixelValueFromData
imageProcessor, 44

getSubscribe
QEGenericEdit, 134

getWriteOnEnter
QEGenericEdit, 134

getWriteOnFinish
QEGenericEdit, 134

getWriteOnLoseFocus
QEGenericEdit, 134

guiFile
QECheckBox, 94
QEPushButton, 245
QERadioButton, 264

handleGuiLaunchRequests
QEForm, 122

heightVariable
QEImage, 178

histogram, 34

histogramScroll, 34

historicImage, 34

horizontalFlip
QEImage, 178

hozSlice1Color
QEImage, 178

hozSlice2Color

QEImage, 179
 hozSlice3Color
 QEImage, 179
 hozSlice4Color
 QEImage, 179
 hozSlice5Color
 QEImage, 179

 Icon
 QECheckBox, 90
 QEPushButton, 241
 QRadioButton, 259

 imageContextMenu, 35
 imageDisplayProperties, 36
 imageDisplayProperties::rgbPixel, 317
 imageInfo, 37
 imageMarkup, 38
 imageMarkupLegendSetText, 40
 imageProcessor, 41
 getPixelValueFromData, 44
 imageProperties, 44
 imageProperties, 46
 ROTATION_0, 46
 ROTATION_180, 46
 ROTATION_90_LEFT, 46
 ROTATION_90_RIGHT, 46
 rotationOptions, 46
 imagePropertiesCore, 46
 buildImageCore, 47
 imageUpdateIndicator, 47
 imageVariable
 QEImage, 179
 Index
 QEAnalogProgressBar, 72
 QECheckBox, 88
 QELabel, 194
 QELineEdit, 203
 QEPushButton, 239
 QRadioButton, 257
 initialHosScrollPos
 QEImage, 179
 initialVertScrollPos
 QEImage, 170
 int
 QEAnalogProgressBar, 75
 QEBitStatus, 82
 QECheckBox, 94
 QEComboBox, 105
 QEConfiguredLayout, 111
 QEFileBrowser, 118

 QEForm, 122
 QEFrame, 126
 QEGenericEdit, 136
 QEGroupBox, 141
 QEImage, 179
 QELabel, 197
 QELineEdit, 205
 QELog, 213
 QEPeriodic, 224
 QEPlot, 232
 QEPushButton, 245
 QRadioButton, 264
 QEScript, 279
 QEShape, 290
 QESlider, 299
 QESpinBox, 305

 Integer
 QEAnalogProgressBar, 72
 QECheckBox, 89
 QELabel, 194
 QELineEdit, 203
 QEPushButton, 240
 QRadioButton, 258

 labelText
 QECheckBox, 95
 QEPushButton, 245
 QRadioButton, 264
 QESubstitutedLabel, 316

 leadingZero
 QEAnalogProgressBar, 75
 QECheckBox, 95
 QELabel, 197
 QELineEdit, 205
 QEPushButton, 246
 QRadioButton, 264

 leadingZeros
 QENumericEdit, 218

 Left_To_Right
 QEAnalogIndicator, 66

 lineProfileArrayVariable
 QEImage, 179

 lineProfileThicknessVariable
 QEImage, 180

 lineProfileX1Variable
 QEImage, 180

 lineProfileX2Variable
 QEImage, 180

 lineProfileY1Variable
 QEImage, 180

lineProfileY2Variable
QEImage, 180

LocalEnumeration
QEAnalogProgressBar, 72
QECheckBox, 89
QELabel, 194
QELineEdit, 203
QEPushButton, 240
QERadioButton, 259

localEnumeration
QEAnalogProgressBar, 75
QECheckBox, 95
QEComboBox, 105
QELabel, 197
QELineEdit, 205
QEPushButton, 246
QERadioButton, 264

logBrightness
QEImage, 180

loginWidget, 47

LogOutput
QECheckBox, 90
QEImage, 168
QEPushButton, 240
QERadioButton, 259

logScale
QEAnalogIndicator, 67

logScaleInterval
QEAnalogIndicator, 67

majorInterval
QEAnalogIndicator, 67

markupCrosshair1, 48

markupCrosshair2, 48

markupDisplayMenu, 49

markupEllipse, 49

markupHLine, 50
drawMarkup, 51

markupItem, 51

markupLine, 54

markupRegion, 54

markupText, 55

markupVLine, 56
drawMarkup, 56

maximum
QEAnalogIndicator, 67
QENumericEdit, 218

messageFormFilter
QEForm, 123

messageSourceFilter

QEForm, 123

Meter
QEAnalogIndicator, 66

minimum
QEAnalogIndicator, 67
QENumericEdit, 218

minorInterval
QEAnalogIndicator, 67

mode
QEAnalogIndicator, 67

Modes
QEAnalogIndicator, 66

Mono
QEImage, 167

mpegSource, 57
updateImage, 57

mpegSourceObject, 57

Never
QEAnalogProgressBar, 72
QEBitStatus, 81
QECheckBox, 89
QEComboBox, 104
QEConfiguredLayout, 110
QEFileBrowser, 116
QEFrame, 125
QEGenericEdit, 133
QEGroupBox, 140
QEImage, 167
QELabel, 194
QELog, 211
QEPeriodic, 222
QEPlot, 231
QEPushButton, 239
QERadioButton, 258
QEScript, 277
QEShape, 285
QESlider, 297
QESpinBox, 303

NewTab
QECheckBox, 88
QEPushButton, 239
QERadioButton, 258

NewWindow
QECheckBox, 89
QEPushButton, 239
QERadioButton, 258

None
QECheckBox, 90
QEImage, 168

QEPushButton, 240
 QERadioButton, 259
 NoRotation
 QEImage, 168
 notation
 QEAnalogProgressBar, 76
 QECheckBox, 96
 QELabel, 198
 QELineEdit, 206
 QEPushButton, 246
 QERadioButton, 265
 Notations
 QEAnalogProgressBar, 72
 QECheckBox, 89
 QELabel, 194
 QELineEdit, 203
 QEPushButton, 240
 QERadioButton, 259
 offset1
 QEShape, 290
 offset2
 QEShape, 290
 offset3
 QEShape, 290
 offset4
 QEShape, 290
 offset5
 QEShape, 290
 offset6
 QEShape, 290
 Open
 QECheckBox, 88
 QEPushButton, 239
 QERadioButton, 258
 orientation
 QEAnalogIndicator, 67
 Orientations
 QEAnalogIndicator, 66
 password
 QECheckBox, 96
 QEPushButton, 246
 QERadioButton, 265
 PeriodicDialog, 58
 PeriodicElementSetupForm, 59
 PeriodicSetupDialog, 59
 Picture
 QELabel, 195
 pixmap0
 pixmap1
 QECheckBox, 96
 QELabel, 198
 QEPushButton, 246
 QERadioButton, 265
 pixmap2
 QECheckBox, 96
 QELabel, 198
 QEPushButton, 247
 QERadioButton, 265
 pixmap3
 QECheckBox, 96
 QELabel, 198
 QEPushButton, 247
 QERadioButton, 266
 pixmap4
 QECheckBox, 96
 QELabel, 199
 QEPushButton, 247
 QERadioButton, 266
 pixmap5
 QECheckBox, 97
 QELabel, 199
 QEPushButton, 247
 QERadioButton, 266
 pixmap6
 QECheckBox, 97
 QELabel, 199
 QEPushButton, 247
 QERadioButton, 266
 pixmap7
 QECheckBox, 97
 QELabel, 199
 QEPushButton, 247
 QERadioButton, 266
 playbackTimer, 59
 point1
 QEShape, 290
 point10
 QEShape, 291
 point2
 QEShape, 291
 point3
 QEShape, 291
 point4
 QEShape, 291

point5
 QEShape, 291
point6
 QEShape, 291
point7
 QEShape, 291
point8
 QEShape, 291
point9
 QEShape, 292
pointInfo, 60
precision
 QEAnalogProgressBar, 76
 QECheckBox, 97
 QELabel, 199
 QELineEdit, 206
 QNumericUpDown, 218
 QEPushButton, 247
 QRadioButton, 266
pressed
 QECheckBox, 91
 QEPushButton, 241
 QRadioButton, 260
pressText
 QECheckBox, 97
 QEPushButton, 247
 QRadioButton, 266
prioritySubstitutions
 QECheckBox, 97
 QEPushButton, 248
 QRadioButton, 266
profileColor
 QEImage, 180
profileHoz1ThicknessVariable
 QEImage, 180
profileHoz1Variable
 QEImage, 180
profileHoz2ThicknessVariable
 QEImage, 181
profileHoz2Variable
 QEImage, 181
profileHoz3ThicknessVariable
 QEImage, 181
profileHoz3Variable
 QEImage, 181
profileHoz4ThicknessVariable
 QEImage, 181
profileHoz4Variable
 QEImage, 181
profileHoz5ThicknessVariable
 QEImage, 181
profileHoz5Variable
 QEImage, 181
profileVert1ThicknessVariable
 QEImage, 182
profileVert1Variable
 QEImage, 182
profileVert2ThicknessVariable
 QEImage, 182
profileVert2Variable
 QEImage, 182
profileVert3ThicknessVariable
 QEImage, 182
profileVert3Variable
 QEImage, 182
profileVert4ThicknessVariable
 QEImage, 182
profileVert4Variable
 QEImage, 182
profileVert5ThicknessVariable
 QEImage, 182
profileVert5Variable
 QEImage, 183
profileVertArrayList
 QEImage, 183
program
 QECheckBox, 97
 QEPushButton, 248
 QRadioButton, 267
program1
 QEImage, 183
program2
 QEImage, 183
programStartupOption
 QECheckBox, 98
 QEPushButton, 248
 QRadioButton, 267
programStartupOption1
 QEImage, 183
programStartupOption2
 QEImage, 183
ProgramStartupOptionNames
 QECheckBox, 90
 QEImage, 168
 QEPushButton, 240
 QRadioButton, 259

QBitStatus, 61
 QEAnalogIndicator, 63
 backgroundColour, 66
 Bar, 66
 borderColour, 66
 Bottom_To_Top, 66
 centreAngle, 66
 fontColour, 66
 foregroundColour, 67
 Left_To_Right, 66
 logScale, 67
 logScaleInterval, 67
 majorInterval, 67
 maximum, 67
 Meter, 66
 minimum, 67
 minorInterval, 67
 mode, 67
 Modes, 66
 orientation, 67
 Orientations, 66
 Right_To_Left, 66
 Scale, 66
 showScale, 67
 showText, 68
 spanAngle, 68
 Top_To_Bottom, 66
 value, 68
 QEAnalogIndicator::Band, 31
 QEAnalogIndicator::BandList, 31
 QEAnalogProgressBar, 68
 addUnits, 74
 alarmSeverityDisplayStyle, 74
 allowDrop, 74
 Always, 72
 Append, 72
 arrayAction, 74
 ArrayActions, 72
 Ascii, 72
 Automatic, 73
 dbValueChanged, 73
 Default, 72
 defaultStyle, 74
 displayAlarmState, 74
 displayAlarmStateOption, 75
 DisplayAlarmStateOptions, 72
 Engineer, 73
 Fixed, 73
 Floating, 72
 format, 75
 Formats, 72
 Index, 72
 int, 75
 Integer, 72
 leadingZero, 75
 LocalEnumeration, 72
 localEnumeration, 75
 Never, 72
 notation, 76
 Notations, 72
 precision, 76
 QEAnalogProgressBar, 73
 Scientific, 73
 Scientist, 73
 setManagedVisible, 73
 styleSheet, 76
 Time, 72
 trailingZeros, 76
 UnsignedInteger, 72
 useDbDisplayLimits, 76
 useDbPrecision, 77
 User, 73
 userLevelEnabled, 77
 userLevelEngineerStyle, 77
 UserLevels, 73
 userLevelScientistStyle, 77
 userLevelUserStyle, 77
 userLevelVisibility, 77
 value, 78
 variable, 78
 variableAsToolTip, 78
 variableSubstitutions, 78
 visible, 78
 WhenInAlarm, 72
 QEBitStatus, 78
 allowDrop, 82
 Always, 81
 dbValueChanged, 81
 defaultStyle, 82
 displayAlarmState, 82
 displayAlarmStateOption, 82
 DisplayAlarmStateOptions, 81
 Engineer, 81
 int, 82
 Never, 81
 Scientist, 81
 setManagedVisible, 81
 styleSheet, 82
 User, 81
 userLevelEnabled, 82

userLevelEngineerStyle, 83
UserLevels, 81
userLevelScientistStyle, 83
userLevelUserStyle, 83
userLevelVisibility, 83
variable, 83
variableAsToolTip, 84
variableSubstitutions, 84
visible, 84
WhenInAlarm, 81
QECheckBox, 84
 addUnits, 92
 alignment, 92
 allowDrop, 92
 Always, 89
 Append, 88
 arguments, 92
 arrayAction, 92
 ArrayActions, 88
 Ascii, 88
 Automatic, 90
 clickCheckedText, 92
 clicked, 91
 clickText, 93
 confirmAction, 93
 confirmText, 93
 creationOption, 93
 CreationOptionNames, 88
 customisationName, 93
 dbValueChanged, 91
 Default, 89
 defaultStyle, 94
 displayAlarmState, 94
 displayAlarmStateOption, 94
 DisplayAlarmStateOptions, 89
 DockBottom, 89
 DockBottomTabbed, 89
 DockFloating, 89
 DockLeft, 89
 DockLeftTabbed, 89
 DockRight, 89
 DockRightTabbed, 89
 DockTop, 89
 DockTopTabbed, 89
 Engineer, 90
 Fixed, 90
 Floating, 89
 format, 94
 Formats, 89
 guiFile, 94
 Icon, 90
 Index, 88
 int, 94
 Integer, 89
 labelText, 95
 leadingZero, 95
 LocalEnumeration, 89
 localEnumeration, 95
 LogOutput, 90
 Never, 89
 NewTab, 88
 NewWindow, 89
 None, 90
 notation, 96
 Notations, 89
 Open, 88
 password, 96
 pixmap0, 96
 pixmap1, 96
 pixmap2, 96
 pixmap3, 96
 pixmap4, 96
 pixmap5, 97
 pixmap6, 97
 pixmap7, 97
 precision, 97
 pressed, 91
 pressText, 97
 prioritySubstitutions, 97
 program, 97
 programStartupOption, 98
 ProgramStartupOptionNames, 90
 QECheckBox, 91
 released, 91
 releaseText, 98
 requestAction, 91
 Scientific, 90
 Scientist, 90
 setManagedVisible, 92
 State, 90
 StdOutput, 90
 styleSheet, 98
 subscribe, 98
 Terminal, 90
 Text, 90
 TextAndIcon, 90
 Time, 89
 trailingZeros, 98
 UnsignedInteger, 89
 updateOption, 98

UpdateOptions, 90
useDbPrecision, 98
User, 90
userLevelEnabled, 98
userLevelEngineerStyle, 99
UserLevels, 90
userLevelScientistStyle, 99
userLevelUserStyle, 99
userLevelVisibility, 99
variable, 99
variableAsToolTip, 100
variableSubstitutions, 100
visible, 100
WhenInAlarm, 89
writeOnClick, 100
writeOnPress, 100
writeOnRelease, 100
QECheckBoxManager, 101
QEComboBox, 101
allowDrop, 105
Always, 104
dbValueChanged, 104
defaultStyle, 105
displayAlarmState, 105
displayAlarmStateOption, 105
DisplayAlarmStateOptions, 103
Engineer, 104
int, 105
localEnumeration, 105
Never, 104
Scientist, 104
setManagedVisible, 104
styleSheet, 105
subscribe, 106
useDbEnumerations, 104
User, 104
userLevelEnabled, 106
userLevelEngineerStyle, 106
UserLevels, 104
userLevelScientistStyle, 106
userLevelUserStyle, 106
userLevelVisibility, 106
variable, 107
variableAsToolTip, 107
variableSubstitutions, 107
visible, 107
WhenInAlarm, 104
writeOnChange, 104
QEConfiguredLayout, 107
allowDrop, 110
Always, 110
defaultStyle, 110
displayAlarmState, 110
displayAlarmStateOption, 111
DisplayAlarmStateOptions, 110
Engineer, 110
int, 111
Never, 110
Scientist, 110
setManagedVisible, 110
styleSheet, 111
User, 110
userLevelEnabled, 111
userLevelEngineerStyle, 111
UserLevels, 110
userLevelScientistStyle, 112
userLevelUserStyle, 112
userLevelVisibility, 112
variableAsToolTip, 112
visible, 112
WhenInAlarm, 110
QEConfiguredLayoutManager, 113
QEFileBrowser, 113
allowDrop, 117
Always, 116
defaultStyle, 117
displayAlarmState, 117
displayAlarmStateOption, 117
DisplayAlarmStateOptions, 116
Engineer, 117
int, 118
Never, 116
Scientist, 117
selected, 117
setManagedVisible, 117
styleSheet, 118
User, 117
userLevelEnabled, 118
userLevelEngineerStyle, 118
UserLevels, 116
userLevelScientistStyle, 118
userLevelUserStyle, 119
userLevelVisibility, 119
variable, 119
variableAsToolTip, 119
variableSubstitutions, 119
visible, 119
WhenInAlarm, 116
QEForm, 120
allowDrop, 122

displayAlarmStateOption, 122
handleGuiLaunchRequests, 122
int, 122
messageFormFilter, 123
messageSourceFilter, 123
resizeContents, 122
uiFile, 123
variableAsToolTip, 123
variableSubstitutions, 123
QEFrame, 124
 allowDrop, 126
 Always, 125
 defaultStyle, 126
 displayAlarmState, 126
 displayAlarmStateOption, 126
 DisplayAlarmStateOptions, 125
 Engineer, 125
 int, 126
 Never, 125
 Scientist, 125
 setManagedVisible, 126
 styleSheet, 127
 User, 125
 userLevelEnabled, 127
 userLevelEngineerStyle, 127
 UserLevels, 125
 userLevelScientistStyle, 127
 userLevelUserStyle, 127
 userLevelVisibility, 127
 variableAsToolTip, 128
 visible, 128
 WhenInAlarm, 125
QEGenericButton, 128
QEGenericEdit, 130
 allowDrop, 135
 Always, 133
 confirmWrite, 135
 defaultStyle, 135
 displayAlarmState, 135
 displayAlarmStateOption, 136
 DisplayAlarmStateOptions, 133
 Engineer, 133
 getConfirmWrite, 134
 getSubscribe, 134
 getWriteOnEnter, 134
 getWriteOnFinish, 134
 getWriteOnLoseFocus, 134
 int, 136
 Never, 133
 QEGenericEdit, 133
 Scientist, 133
 setConfirmWrite, 134
 setManagedVisible, 134
 setSubscribe, 134
 setWriteOnEnter, 135
 setWriteOnFinish, 135
 setWriteOnLoseFocus, 135
 styleSheet, 136
 subscribe, 136
 User, 133
 userLevelEnabled, 136
 userLevelEngineerStyle, 136
 UserLevels, 133
 userLevelScientistStyle, 137
 userLevelUserStyle, 137
 userLevelVisibility, 137
 variable, 137
 variableAsToolTip, 137
 variableSubstitutions, 137
 visible, 138
 WhenInAlarm, 133
 writeOnEnter, 138
 writeOnFinish, 138
 writeOnLoseFocus, 138
QEGroupBox, 138
 allowDrop, 140
 Always, 140
 defaultStyle, 140
 displayAlarmState, 141
 displayAlarmStateOption, 141
 DisplayAlarmStateOptions, 140
 Engineer, 140
 int, 141
 Never, 140
 Scientist, 140
 setManagedVisible, 140
 styleSheet, 141
 substitutedTitle, 141
 textSubstitutions, 141
 User, 140
 userLevelEnabled, 142
 userLevelEngineerStyle, 142
 UserLevels, 140
 userLevelScientistStyle, 142
 userLevelUserStyle, 142
 userLevelVisibility, 142
 variableAsToolTip, 143
 visible, 143
 WhenInAlarm, 140
QEImage, 143

allowDrop, 171
Always, 167
areaColor, 171
arguments1, 171
arguments2, 171
autoBrightnessContrast, 171
Bayer, 167
BayerBG, 167
BayerGB, 167
BayerGR, 167
BayerRG, 167
beamColor, 171
beamXVariable, 171
beamYVariable, 171
bitDepthVariable, 171
BOUNDING_RECTANGLE, 167
BoundingRectangle, 167
briefInfoArea, 172
CenterAndSize, 167
clippingHighVariable, 172
clippingLowVariable, 172
clippingOnOffVariable, 172
contrastReversal, 172
dbValueChanged, 170
defaultStyle, 172
dimension1Variable, 172
dimension2Variable, 172
dimension3Variable, 172
dimensionsVariable, 173
displayAlarmState, 173
displayAlarmStateOption, 173
DisplayAlarmStateOptions, 167
displayArea1Selection, 173
displayArea2Selection, 173
displayArea3Selection, 173
displayArea4Selection, 173
displayBeamSelection, 174
displayButtonBar, 170
displayCursorPixelInfo, 174
displayEllipse, 174
displayHozSlice1Selection, 174
displayHozSlice2Selection, 174
displayHozSlice3Selection, 174
displayHozSlice4Selection, 174
displayHozSlice5Selection, 174
displayProfileSelection, 174
displayTargetSelection, 175
displayVertSliceSelection, 175
DottedFullCrosshair, 169
ellipseColor, 175
ellipseHVariable, 175
EllipseVariableDefinitions, 167
ellipseVariableDefinitions, 167
ellipseWVariable, 175
ellipseXVariable, 175
ellipseYVariable, 175
enableArea1Selection, 176
enableArea2Selection, 176
enableArea3Selection, 176
enableArea4Selection, 176
enableBeamSelection, 176
enableHozSlice1Selection, 176
enableHozSlice2Selection, 176
enableHozSlice3Selection, 176
enableHozSlice4Selection, 177
enableHozSlice5Selection, 177
enableProfileSelection, 177
enableTargetSelection, 177
enableVertSlice1Selection, 177
enableVertSlice2Selection, 177
enableVertSlice3Selection, 177
enableVertSlice4Selection, 178
enableVertSlice5Selection, 178
Engineer, 169
externalControls, 178
Fit, 168
formatOption, 178
FormatOptions, 167
formatVariable, 178
heightVariable, 178
horizontalFlip, 178
hozSlice1Color, 178
hozSlice2Color, 179
hozSlice3Color, 179
hozSlice4Color, 179
hozSlice5Color, 179
imageVariable, 179
initialHosScrollPos, 179
initialVertScrollPos, 170
int, 179
lineProfileArrayVariable, 179
lineProfileThicknessVariable, 180
lineProfileX1Variable, 180
lineProfileX2Variable, 180
lineProfileY1Variable, 180
lineProfileY2Variable, 180
logBrightness, 180
LogOutput, 168
Mono, 167
Never, 167

None, 168
NoRotation, 168
profileColor, 180
profileHoz1ThicknessVariable, 180
profileHoz1Variable, 180
profileHoz2ThicknessVariable, 181
profileHoz2Variable, 181
profileHoz3ThicknessVariable, 181
profileHoz3Variable, 181
profileHoz4ThicknessVariable, 181
profileHoz4Variable, 181
profileHoz5ThicknessVariable, 181
profileHoz5Variable, 181
profileHozArrayVariable, 181
profileVert1ThicknessVariable, 182
profileVert1Variable, 182
profileVert2ThicknessVariable, 182
profileVert2Variable, 182
profileVert3ThicknessVariable, 182
profileVert3Variable, 182
profileVert4ThicknessVariable, 182
profileVert4Variable, 182
profileVert5ThicknessVariable, 182
profileVert5Variable, 183
profileVertArrayVariable, 183
program1, 183
program2, 183
programStartupOption1, 183
programStartupOption2, 183
ProgramStartupOptionNames, 168
QEImage, 170
regionOfInterest1HVariable, 183
regionOfInterest1WVariable, 184
regionOfInterest1XVariable, 184
regionOfInterest1YVariable, 184
regionOfInterest2HVariable, 184
regionOfInterest2WVariable, 184
regionOfInterest2XVariable, 184
regionOfInterest2YVariable, 184
regionOfInterest3HVariable, 184
regionOfInterest3WVariable, 184
regionOfInterest3XVariable, 185
regionOfInterest3YVariable, 185
regionOfInterest4HVariable, 185
regionOfInterest4WVariable, 185
regionOfInterest4XVariable, 185
regionOfInterest4YVariable, 185
RESIZE_OPTION_FIT, 168
RESIZE_OPTION_ZOOM, 168
resizeOption, 185
ResizeOptions, 168
resizeOptions, 168
rgb1, 167
rgb2, 167
rgb3, 168
Rotate180, 168
Rotate90Left, 168
Rotate90Right, 168
rotation, 185
RotationOptions, 168
Scientist, 169
selectOptions, 168
setImageFile, 170
setManagedVisible, 170
showTime, 185
SO_AREA4, 169
SO_BEAM, 169
SO_HSLICE1, 169
SO_HSLICE2, 169
SO_HSLICE3, 169
SO_HSLICE4, 169
SO_HSLICE5, 169
SO_NONE, 169
SO_PANNING, 169
SO_PROFILE, 169
SO_TARGET, 169
SO_VSLICE1, 169
SO_VSLICE2, 169
SO_VSLICE3, 169
SO_VSLICE4, 169
SO_VSLICE5, 169
SolidSmallCrosshair, 169
StdOutput, 168
styleSheet, 186
targetColor, 186
TargetOptions, 169
targetTriggerVariable, 186
targetXVariable, 186
targetYVariable, 186
Terminal, 168
timeColor, 186
URL, 186
useFalseColour, 186
User, 169
userLevelEnabled, 186
userLevelEngineerStyle, 187
UserLevels, 169
userLevelScientistStyle, 187
userLevelUserStyle, 187
userLevelVisibility, 187

variableAsToolTip, 187
variableSubstitutions, 188
verticalFlip, 188
vertSlice1Color, 188
vertSlice2Color, 188
vertSlice3Color, 188
vertSlice4Color, 188
vertSlice5Color, 188
visible, 188
WhenInAlarm, 167
widthVariable, 188
yuv422, 168
yuv444, 168
Zoom, 168
QEImageMarkupThickness, 189
QEImageOptionsDialog, 189
QELabel, 190
 addUnits, 196
 allowDrop, 196
 Always, 194
 Append, 194
 arrayAction, 196
 ArrayActions, 194
 Ascii, 194
 Automatic, 194
 dbValueChanged, 196
 Default, 194
 defaultStyle, 196
 displayAlarmState, 196
 displayAlarmStateOption, 197
 DisplayAlarmStateOptions, 194
 Engineer, 195
 Fixed, 194
 Floating, 194
 format, 197
 Formats, 194
 Index, 194
 int, 197
 Integer, 194
 leadingZero, 197
 LocalEnumeration, 194
 localEnumeration, 197
 Never, 194
 notation, 198
 Notations, 194
 Picture, 195
 pixmap0, 198
 pixmap1, 198
 pixmap2, 198
 pixmap3, 198
pixmap4, 199
pixmap5, 199
pixmap6, 199
pixmap7, 199
precision, 199
QELabel, 195
Scientific, 194
Scientist, 195
setManagedVisible, 196
styleSheet, 199
Text, 195
Time, 194
trailingZeros, 199
UnsignedInteger, 194
UPDATE_PIXMAP, 195
UPDATE_TEXT, 195
updateOption, 199
UpdateOptions, 195
updateOptions, 194
useDbPrecision, 200
User, 195
userLevelEnabled, 200
userLevelEngineerStyle, 200
UserLevels, 195
userLevelScientistStyle, 200
userLevelUserStyle, 200
userLevelVisibility, 200
variable, 201
variableAsToolTip, 201
variableSubstitutions, 201
visible, 201
WhenInAlarm, 194
QELineEdit, 201
 addUnits, 204
 Append, 203
 arrayAction, 204
 ArrayActions, 203
 Ascii, 203
 Automatic, 204
 dbValueChanged, 204
 Default, 203
 Fixed, 204
 Floating, 203
 format, 205
 Formats, 203
 Index, 203
 int, 205
 Integer, 203
 leadingZero, 205
 LocalEnumeration, 203

localEnumeration, 205
notation, 206
Notations, 203
precision, 206
QELineEdit, 204
Scientific, 204
Time, 203
trailingZeros, 206
UnsignedInteger, 203
useDbPrecision, 206
QELineEditManager, 206
QELink, 207
QELog, 209
 allowDrop, 212
 Always, 211
 defaultStyle, 212
 displayAlarmState, 212
 displayAlarmStateOption, 212
 DisplayAlarmStateOptions, 211
 Engineer, 212
 int, 213
 Never, 211
 Scientist, 212
 setManagedVisible, 212
 styleSheet, 213
 User, 212
 userLevelEnabled, 213
 userLevelEngineerStyle, 213
 UserLevels, 211
 userLevelScientistStyle, 213
 userLevelUserStyle, 213
 userLevelVisibility, 214
 variableAsToolTip, 214
 visible, 214
 WhenInAlarm, 211
QELogin, 214
QELoginDialog, 215
QENumericEdit, 215
 addUnits, 218
 autoScale, 218
 dbValueChanged, 218
 leadingZeros, 218
 maximum, 218
 minimum, 218
 precision, 218
 QENumericEdit, 217
QENumericEditManager, 219
QEPeriodic, 219
 allowDrop, 223
 Always, 222
dbElementChanged, 223
dbValueChanged, 223
displayAlarmState, 223
displayAlarmStateOption, 223
DisplayAlarmStateOptions, 222
Engineer, 223
int, 224
Never, 222
readbackLabelVariable1, 224
readbackLabelVariable2, 224
Scientist, 223
subscribe, 224
User, 223
userLevelEnabled, 224
userLevelEngineerStyle, 224
UserLevels, 222
userLevelScientistStyle, 225
userLevelUserStyle, 225
userLevelVisibility, 225
variableAsToolTip, 225
variableSubstitutions, 225
visible, 225
WhenInAlarm, 222
writeButtonVariable1, 226
writeButtonVariable2, 226
QEPeriodic::elementInfoStruct, 31
QEPeriodic::userInfoStructArray, 319
QEPeriodicComponentData, 226
QEPeriodicTaskMenu, 226
QEPeriodicTaskMenuFactory, 227
QEPlot, 227
 allowDrop, 232
 Always, 231
 dbValueChanged, 231
 defaultStyle, 232
 displayAlarmState, 232
 displayAlarmStateOption, 232
 DisplayAlarmStateOptions, 231
 Engineer, 231
 int, 232
 Never, 231
 Scientist, 231
 setManagedVisible, 231
 styleSheet, 233
 User, 231
 userLevelEnabled, 233
 userLevelEngineerStyle, 233
 UserLevels, 231
 userLevelScientistStyle, 233
 userLevelUserStyle, 233

userLevelVisibility, 233
variable1, 234
variable2, 234
variable3, 234
variable4, 234
variableAsToolTip, 234
variableSubstitutions, 234
visible, 234
WhenInAlarm, 231
QEPushButton, 235
addUnits, 242
alignment, 242
allowDrop, 242
altReadbackVariable, 242
Always, 239
Append, 239
arguments, 243
arrayAction, 243
ArrayActions, 239
Ascii, 239
Automatic, 240
clickCheckedText, 243
clicked, 241
clickText, 243
confirmAction, 243
confirmText, 244
creationOption, 244
CreationOptionNames, 239
customisationName, 244
dbValueChanged, 241
Default, 240
defaultStyle, 244
displayAlarmState, 244
displayAlarmStateOption, 244
DisplayAlarmStateOptions, 239
DockBottom, 239
DockBottomTabbed, 239
DockFloating, 239
DockLeft, 239
DockLeftTabbed, 239
DockRight, 239
DockRightTabbed, 239
DockTop, 239
DockTopTabbed, 239
Engineer, 241
Fixed, 240
Floating, 240
format, 245
Formats, 239
guiFile, 245
Icon, 241
Index, 239
int, 245
Integer, 240
labelText, 245
leadingZero, 246
LocalEnumeration, 240
localEnumeration, 246
LogOutput, 240
Never, 239
NewTab, 239
NewWindow, 239
None, 240
notation, 246
Notations, 240
Open, 239
password, 246
 pixmap0, 246
 pixmap1, 247
 pixmap2, 247
 pixmap3, 247
 pixmap4, 247
 pixmap5, 247
 pixmap6, 247
 pixmap7, 247
precision, 247
pressed, 241
pressText, 247
prioritySubstitutions, 248
program, 248
programStartupOption, 248
ProgramStartupOptionNames, 240
QEPushButton, 241
released, 242
releaseText, 248
requestAction, 242
Scientific, 240
Scientist, 241
setManagedVisible, 242
State, 241
StdOutput, 240
styleSheet, 248
subscribe, 248
Terminal, 240
Text, 241
TextAndIcon, 241
Time, 240
trailingZeros, 248
UnsignedInteger, 240
updateOption, 249

UpdateOptions, 240
useDbPrecision, 249
User, 241
userLevelEnabled, 249
userLevelEngineerStyle, 249
UserLevels, 241
userLevelScientistStyle, 249
userLevelUserStyle, 249
userLevelVisibility, 250
variable, 250
variableAsToolTip, 250
variableSubstitutions, 250
visible, 250
WhenInAlarm, 239
writeOnClick, 250
writeOnPress, 250
writeOnRelease, 251
QEPVNameLists, 251
QEPvProperties, 251
 variable, 252
 variableSubstitutions, 252
QEPvPropertiesManager, 253
QERadioButton, 253
 addUnits, 261
 alignment, 261
 allowDrop, 261
 Always, 258
 Append, 257
 arguments, 261
 arrayAction, 261
 ArrayActions, 257
 Ascii, 257
 Automatic, 259
 clickCheckedText, 262
 clicked, 260
 clickText, 262
 confirmAction, 262
 confirmText, 262
 creationOption, 262
 CreationOptionNames, 257
 customisationName, 263
 dbValueChanged, 260
 Default, 258
 defaultStyle, 263
 displayAlarmState, 263
 displayAlarmStateOption, 263
 DisplayAlarmStateOptions, 258
 DockBottom, 258
 DockBottomTabbed, 258
 DockFloating, 258
 DockLeft, 258
 DockLeftTabbed, 258
 DockRight, 258
 DockRightTabbed, 258
 DockTop, 258
 DockTopTabbed, 258
 Engineer, 260
 Fixed, 259
 Floating, 258
 format, 263
 Formats, 258
 guiFile, 264
 Icon, 259
 Index, 257
 int, 264
 Integer, 258
 labelText, 264
 leadingZero, 264
 LocalEnumeration, 259
 localEnumeration, 264
 LogOutput, 259
 Never, 258
 NewTab, 258
 NewWindow, 258
 None, 259
 notation, 265
 Notations, 259
 Open, 258
 password, 265
 pixmap0, 265
 pixmap1, 265
 pixmap2, 265
 pixmap3, 266
 pixmap4, 266
 pixmap5, 266
 pixmap6, 266
 pixmap7, 266
 precision, 266
 pressed, 260
 pressText, 266
 prioritySubstitutions, 266
 program, 267
 programStartupOption, 267
 ProgramStartupOptionNames, 259
 QERadioButton, 260
 released, 260
 releaseText, 267
 requestAction, 261
 Scientific, 259
 Scientist, 260

setManagedVisible, 261
State, 259
StdOutput, 259
styleSheet, 267
subscribe, 267
Terminal, 259
Text, 259
TextAndIcon, 259
Time, 259
trailingZeros, 267
UnsignedInteger, 259
updateOption, 267
UpdateOptions, 259
useDbPrecision, 268
User, 260
userLevelEnabled, 268
userLevelEngineerStyle, 268
UserLevels, 259
userLevelScientistStyle, 268
userLevelUserStyle, 268
userLevelVisibility, 269
variable, 269
variableAsToolTip, 269
variableSubstitutions, 269
visible, 269
WhenInAlarm, 258
writeOnClick, 269
writeOnPress, 269
writeOnRelease, 269
QERecipe, 270
QERecordFieldName, 272
QERecordSpec, 272
QERecordSpecList, 273
QEScript, 273
 allowDrop, 278
 Always, 277
 defaultStyle, 278
 displayAlarmState, 278
 displayAlarmStateOption, 278
 DisplayAlarmStateOptions, 277
 Engineer, 278
 int, 279
 Never, 277
 Scientist, 278
 setManagedVisible, 278
 styleSheet, 279
 User, 278
 userLevelEnabled, 279
 userLevelEngineerStyle, 279
 UserLevels, 278
userLevelScientistStyle, 279
userLevelUserStyle, 280
userLevelVisibility, 280
variableAsToolTip, 280
visible, 280
WhenInAlarm, 278
QEShape, 280
 allowDrop, 287
 Always, 285
 animation1, 287
 animation2, 287
 animation3, 287
 animation4, 288
 animation5, 288
 animation6, 288
 animationOptions, 285
 color1, 288
 color10, 288
 color2, 288
 color3, 288
 color4, 288
 color5, 288
 color6, 289
 color7, 289
 color8, 289
 color9, 289
 dbValueChanged1, 286
 dbValueChanged2, 286
 dbValueChanged3, 286
 dbValueChanged4, 286
 dbValueChanged5, 287
 dbValueChanged6, 287
 defaultStyle, 289
 displayAlarmState, 289
 displayAlarmStateOption, 289
 DisplayAlarmStateOptions, 285
 Engineer, 286
 int, 290
 Never, 285
 offset1, 290
 offset2, 290
 offset3, 290
 offset4, 290
 offset5, 290
 offset6, 290
 point1, 290
 point10, 291
 point2, 291
 point3, 291
 point4, 291

point5, 291
point6, 291
point7, 291
point8, 291
point9, 292
QEShape, 286
scale2, 292
scale3, 292
scale4, 292
scale5, 292
scale6, 292
Scientist, 286
setManagedVisible, 287
shapeOptions, 285
styleSheet, 292
User, 286
userLevelEnabled, 292
userLevelEngineerStyle, 293
UserLevels, 285
userLevelScientistStyle, 293
userLevelUserStyle, 293
userLevelVisibility, 293
variable1, 293
variable2, 294
variable3, 294
variable4, 294
variable5, 294
variable6, 294
variableAsToolTip, 294
variableSubstitutions, 294
visible, 294
WhenInAlarm, 285
QESlider, 295
allowDrop, 298
Always, 297
dbValueChanged, 298
defaultStyle, 298
displayAlarmState, 298
displayAlarmStateOption, 298
DisplayAlarmStateOptions, 297
Engineer, 297
int, 299
Never, 297
Scientist, 297
setManagedVisible, 298
styleSheet, 299
subscribe, 299
User, 297
userLevelEnabled, 299
userLevelEngineerStyle, 299
UserLevels, 297
userLevelScientistStyle, 299
userLevelUserStyle, 300
userLevelVisibility, 300
variable, 300
variableAsToolTip, 300
variableSubstitutions, 300
visible, 300
WhenInAlarm, 297
writeOnChange, 298
QESpinBox, 301
allowDrop, 304
Always, 303
dbValueChanged, 304
defaultStyle, 304
displayAlarmState, 304
displayAlarmStateOption, 304
DisplayAlarmStateOptions, 303
Engineer, 304
int, 305
Never, 303
Scientist, 304
setManagedVisible, 304
styleSheet, 305
subscribe, 305
User, 304
userLevelEnabled, 305
userLevelEngineerStyle, 305
UserLevels, 303
userLevelScientistStyle, 305
userLevelUserStyle, 306
userLevelVisibility, 306
variable, 306
variableAsToolTip, 306
variableSubstitutions, 306
visible, 306
WhenInAlarm, 303
QEStripChart, 307
variableSubstitutions, 309
QEStripChartAdjustPVDialog, 309
QEStripChartContextMenu, 309
QEStripChartContextMenu, 310
QEStripChartDurationDialog, 310
QEStripChartItem, 310
QEStripChartNames, 311
QEStripChartPushButtonSpecifications, 313
QEStripChartRangeDialog, 313
QEStripChartState, 313
QEStripChartStateList, 314
QEStripChartStatistics, 314

QEStripChartTimeDialog, 315
QEStripToolBar, 315
QEStripToolBar::OwnWidgets, 58
QESubstitutedLabel, 316
 labelText, 316
 textSubstitutions, 317

readbackLabelVariable1
 QEPeriodic, 224
readbackLabelVariable2
 QEPeriodic, 224
recording, 317
regionOfInterest1HVariable
 QEImage, 183
regionOfInterest1WVariable
 QEImage, 184
regionOfInterest1XVariable
 QEImage, 184
regionOfInterest1YVariable
 QEImage, 184
regionOfInterest2HVariable
 QEImage, 184
regionOfInterest2WVariable
 QEImage, 184
regionOfInterest2XVariable
 QEImage, 184
regionOfInterest2YVariable
 QEImage, 184
regionOfInterest3HVariable
 QEImage, 184
regionOfInterest3WVariable
 QEImage, 184
regionOfInterest3XVariable
 QEImage, 185
regionOfInterest3YVariable
 QEImage, 185
regionOfInterest4HVariable
 QEImage, 185
regionOfInterest4WVariable
 QEImage, 185
regionOfInterest4XVariable
 QEImage, 185
regionOfInterest4YVariable
 QEImage, 185
released
 QECheckBox, 91
 QEPushButton, 242
 QERadioButton, 260
releaseText
 QECheckBox, 98

QEPushButton, 248
QERadioButton, 267
requestAction
 QECheckBox, 91
 QEPushButton, 242
 QERadioButton, 261
RESIZE_OPTION_FIT
 QEImage, 168
RESIZE_OPTION_ZOOM
 QEImage, 168
resizeContents
 QEForm, 122
resizeOption
 QEImage, 185
ResizeOptions
 QEImage, 168
resizeOptions
 QEImage, 168
rgb1
 QEImage, 167
rgb2
 QEImage, 167
rgb3
 QEImage, 168
Right_To_Left
 QEAnalogIndicator, 66
Rotate180
 QEImage, 168
Rotate90Left
 QEImage, 168
Rotate90Right
 QEImage, 168
rotation
 QEImage, 185
ROTATION_0
 imageProperties, 46
ROTATION_180
 imageProperties, 46
ROTATION_90_LEFT
 imageProperties, 46
ROTATION_90_RIGHT
 imageProperties, 46
RotationOptions
 QEImage, 168
rotationOptions
 imageProperties, 46

Scale
 QEAnalogIndicator, 66
scale2

QEShape, 292
scale3
 QEShape, 292
scale4
 QEShape, 292
scale5
 QEShape, 292
scale6
 QEShape, 292
Scientific
 QEAnalogProgressBar, 73
 QECheckBox, 90
 QELabel, 194
 QELineEdit, 204
 QEPushButton, 240
 QERadioButton, 259
Scientist
 QEAnalogProgressBar, 73
 QEBitStatus, 81
 QECheckBox, 90
 QEComboBox, 104
 QEConfiguredLayout, 110
 QEFileBrowser, 117
 QEFrame, 125
 QEGenericEdit, 133
 QEGroupBox, 140
 QEImage, 169
 QELabel, 195
 QELog, 212
 QEPeriodic, 223
 QEPlot, 231
 QEPushButton, 241
 QERadioButton, 260
 QEScript, 278
 QEShape, 286
 QESlider, 297
 QESpinBox, 304
screenSelectDialog, 318
selected
 QEFileBrowser, 117
selectMenu, 318
selectOptions
 QEImage, 168
setConfirmWrite
 QEGenericEdit, 134
setImageFile
 QEImage, 170
setManagedVisible
 QEAnalogProgressBar, 73
 QEBitStatus, 81
 QECheckBox, 92
 QEComboBox, 104
 QEConfiguredLayout, 110
 QEFileBrowser, 117
 QEFrame, 126
 QEGenericEdit, 134
 QEGroupBox, 140
 QEImage, 170
 QELabel, 196
 QELog, 212
 QEPlot, 231
 QEPushButton, 242
 QERadioButton, 261
 QEScript, 278
 QEShape, 287
 QESlider, 298
 QESpinBox, 304
setSubscribe
 QEGenericEdit, 134
setWriteOnEnter
 QEGenericEdit, 135
setWriteOnFinish
 QEGenericEdit, 135
setWriteOnLoseFocus
 QEGenericEdit, 135
shapeOptions
 QEShape, 285
showScale
 QEAnalogIndicator, 67
showText
 QEAnalogIndicator, 68
showTime
 QEImage, 185
SO_AREA4
 QEImage, 169
SO_BEAM
 QEImage, 169
SO_HSLICE1
 QEImage, 169
SO_HSLICE2
 QEImage, 169
SO_HSLICE3
 QEImage, 169
SO_HSLICE4
 QEImage, 169
SO_HSLICE5
 QEImage, 169
SO_NONE
 QEImage, 169
SO_PANNING

QEImage, 169
 SO_PROFILE
 QEImage, 169
 SO_TARGET
 QEImage, 169
 SO_VSLICE1
 QEImage, 169
 SO_VSLICE2
 QEImage, 169
 SO_VSLICE3
 QEImage, 169
 SO_VSLICE4
 QEImage, 169
 SO_VSLICE5
 QEImage, 169
 SolidSmallCrosshair
 QEImage, 169
 spanAngle
 QEAnalogIndicator, 68
 State
 QECheckBox, 90
 QEPushButton, 241
 QERadioButton, 259
 StdOutput
 QECheckBox, 90
 QEImage, 168
 QEPushButton, 240
 QERadioButton, 259
 styleSheet
 QEAnalogProgressBar, 76
 QEBitStatus, 82
 QECheckBox, 98
 QEComboBox, 105
 QEConfiguredLayout, 111
 QEFileBrowser, 118
 QEFrame, 127
 QEGenericEdit, 136
 QEGroupBox, 141
 QEImage, 186
 QELabel, 199
 QELog, 213
 QEPlot, 233
 QEPushButton, 248
 QERadioButton, 267
 QEScript, 279
 QEShape, 292
 QESlider, 299
 QESpinBox, 305
 subscribe
 QECheckBox, 98
 QEComboBox, 106
 QEGenericEdit, 136
 QEPeriodic, 224
 QEPushButton, 248
 QERadioButton, 267
 QESlider, 299
 QESpinBox, 305
 substitutedTitle
 QEGroupBox, 141
 targetColor
 QEImage, 186
 TargetOptions
 QEImage, 169
 targetTriggerVariable
 QEImage, 186
 targetXVariable
 QEImage, 186
 targetYVariable
 QEImage, 186
 Terminal
 QECheckBox, 90
 QEImage, 168
 QEPushButton, 240
 QERadioButton, 259
 Text
 QECheckBox, 90
 QELabel, 195
 QEPushButton, 241
 QERadioButton, 259
 TextAndIcon
 QECheckBox, 90
 QEPushButton, 241
 QERadioButton, 259
 textSubstitutions
 QEGroupBox, 141
 QESubstitutedLabel, 317
 Time
 QEAnalogProgressBar, 72
 QECheckBox, 89
 QELabel, 194
 QELineEdit, 203
 QEPushButton, 240
 QERadioButton, 259
 timeColor
 QEImage, 186
 Top_To_Bottom
 QEAnalogIndicator, 66
 trace, 318
 trailingZeros

QEAnalogProgressBar, 76
QECheckBox, 98
QELabel, 199
QELineEdit, 206
QEPushButton, 248
QERadioButton, 267

uiFile
QEForm, 123

UnsignedInteger
QEAnalogProgressBar, 72
QECheckBox, 89
QELabel, 194
QELineEdit, 203
QEPushButton, 240
QERadioButton, 259

UPDATE_PIXMAP
QELabel, 195

UPDATE_TEXT
QELabel, 195

updateImage
mpegSource, 57

updateOption
QECheckBox, 98
QELabel, 199
QEPushButton, 249
QERadioButton, 267

UpdateOptions
QECheckBox, 90
QELabel, 195
QEPushButton, 240
QERadioButton, 259

updateOptions
QELabel, 194

URL
QEImage, 186

useDbDisplayLimits
QEAnalogProgressBar, 76

useDbEnumerations
QEComboBox, 104

useDbPrecision
QEAnalogProgressBar, 77
QECheckBox, 98
QELabel, 200
QELineEdit, 206
QEPushButton, 249
QERadioButton, 268

useFalseColour
QEImage, 186

User

QEAnalogProgressBar, 73
QEBitStatus, 81
QECheckBox, 90
QEComboBox, 104
QEConfiguredLayout, 110
QEFileBrowser, 117
QEFrame, 125

QEGenericEdit, 133
QEGroupBox, 140

QEImage, 169
QELabel, 195
QELog, 212
QEPeriodic, 223
QEPlot, 231
QEPushButton, 241
QERadioButton, 260

QEScript, 278
QEShape, 286
QESlider, 297
QESpinBox, 304

userInfoStruct, 319

userLevelEnabled
QEAnalogProgressBar, 77
QEBitStatus, 82
QECheckBox, 98
QEComboBox, 106
QEConfiguredLayout, 111
QEFileBrowser, 118
QEFrame, 127

QEGenericEdit, 136
QEGroupBox, 142

QEImage, 186
QELabel, 200
QELog, 213
QEPeriodic, 224
QEPlot, 233
QEPushButton, 249
QERadioButton, 268

QEScript, 279
QEShape, 292
QESlider, 299
QESpinBox, 305

userLevelEngineerStyle
QEAnalogProgressBar, 77
QEBitStatus, 83
QECheckBox, 99
QEComboBox, 106
QEConfiguredLayout, 111
QEFileBrowser, 118
QEFrame, 127

QEGenericEdit, 136
 QEGroupBox, 142
 QEImage, 187
 QELabel, 200
 QELog, 213
 QEPeriodic, 224
 QEPlot, 233
 QEPushButton, 249
 QERadioButton, 268
 QEScript, 279
 QEShape, 293
 QESlider, 299
 QESpinBox, 305
UserLevels
 QEAnalogProgressBar, 73
 QEBitStatus, 81
 QECheckBox, 90
 QEComboBox, 104
 QEConfiguredLayout, 110
 QEFileBrowser, 116
 QEFrame, 125
 QEGenericEdit, 133
 QEGroupBox, 140
 QEImage, 169
 QELabel, 195
 QELog, 211
 QEPeriodic, 222
 QEPlot, 231
 QEPushButton, 241
 QERadioButton, 259
 QEScript, 278
 QEShape, 285
 QESlider, 297
 QESpinBox, 303
userLevelScientistStyle
 QEAnalogProgressBar, 77
 QEBitStatus, 83
 QECheckBox, 99
 QEComboBox, 106
 QEConfiguredLayout, 112
 QEFileBrowser, 118
 QEFrame, 127
 QEGenericEdit, 137
 QEGroupBox, 142
 QEImage, 187
 QELabel, 200
 QELog, 213
 QEPeriodic, 225
 QEPlot, 233
 QEPushButton, 249
value
 QEAnalogIndicator, 68

QEAnalogProgressBar, 78
ValueScaling, 319
variable
 QEAnalogProgressBar, 78
 QEBitStatus, 83
 QECheckBox, 99
 QEComboBox, 107
 QEFileBrowser, 119
 QEGenericEdit, 137
 QELabel, 201
 QEPushButton, 250
 QE_pvProperties, 252
 QERadioButton, 269
 QESlider, 300
 QESpinBox, 306
variable1
 QEPlot, 234
 QEShape, 293
variable2
 QEPlot, 234
 QEShape, 294
variable3
 QEPlot, 234
 QEShape, 294
variable4
 QEPlot, 234
 QEShape, 294
variable5
 QEShape, 294
variable6
 QEShape, 294
variableAsToolTip
 QEAnalogProgressBar, 78
 QEBitStatus, 84
 QECheckBox, 100
 QEComboBox, 107
 QEConfiguredLayout, 112
 QEFileBrowser, 119
 QEForm, 123
 QEFrame, 128
 QEGenericEdit, 137
 QEGroupBox, 143
 QEImage, 187
 QELabel, 201
 QELog, 214
 QEPeriodic, 225
 QEPlot, 234
 QEPushButton, 250
 QERadioButton, 269
 QEScript, 280
QEShape, 294
QESlider, 300
QESpinBox, 306
variableSubstitutions
 QEAnalogProgressBar, 78
 QEBitStatus, 84
 QECheckBox, 100
 QEComboBox, 107
 QEFileBrowser, 119
 QEForm, 123
 QEGenericEdit, 137
 QEImage, 188
 QELabel, 201
 QEPeriodic, 225
 QEPlot, 234
 QEPushButton, 250
 QE_pvProperties, 252
 QERadioButton, 269
 QEShape, 294
 QESlider, 300
 QESpinBox, 306
 QEStripChart, 309
verticalFlip
 QEImage, 188
vertSlice1Color
 QEImage, 188
vertSlice2Color
 QEImage, 188
vertSlice3Color
 QEImage, 188
vertSlice4Color
 QEImage, 188
vertSlice5Color
 QEImage, 188
VideoWidget, 320
visible
 QEAnalogProgressBar, 78
 QEBitStatus, 84
 QECheckBox, 100
 QEComboBox, 107
 QEConfiguredLayout, 112
 QEFileBrowser, 119
 QEFrame, 128
 QEGenericEdit, 138
 QEGroupBox, 143
 QEImage, 188
 QELabel, 201
 QELog, 214
 QEPeriodic, 225
 QEPlot, 234

QEPushButton, 250
QERadioButton, 269
QEScript, 280
QEShape, 294
QESlider, 300
QESpinBox, 306

WhenInAlarm
QEAnalogProgressBar, 72
QEBitStatus, 81
QECheckBox, 89
QEComboBox, 104
QEConfiguredLayout, 110
QEFileBrowser, 116
QEFrame, 125
QEGenericEdit, 133
QEGroupBox, 140
QEImage, 167
QELabel, 194
QELog, 211
QEPeriodic, 222
QEPlot, 231
QEPushButton, 239
QERadioButton, 258
QEScript, 278
QEShape, 285
QESlider, 297
QESpinBox, 303

widthVariable
QEImage, 188
writeButtonVariable1
QEPeriodic, 226
writeButtonVariable2
QEPeriodic, 226
writeOnChange
QEComboBox, 104
QESlider, 298
writeOnClick
QECheckBox, 100
QEPushButton, 250
QERadioButton, 269
writeOnEnter
QEGenericEdit, 138
writeOnFinish
QEGenericEdit, 138
writeOnLoseFocus
QEGenericEdit, 138
writeOnPress
QECheckBox, 100
QEPushButton, 250

yuv422
QEImage, 168
yuv444
QEImage, 168

Zoom
QEImage, 168
zoomMenu, 321