

EPICS QT Framework

2.1

Generated by Doxygen 1.7.4

Mon Jan 7 2013 12:25:46

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	_Field Class Reference	27

9.2	_Item Class Reference	28
9.3	_QDialogItem Class Reference	28
9.4	_QDialogLogin Class Reference	28
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	QEAnalogIndicator::Band Struct Reference	30
9.10	QEAnalogIndicator::BandList Class Reference	31
9.11	qcastatemachine::ConnectionQCaStateMachine Class Reference	31
9.12	ContainerProfile Class Reference	31
9.13	contextMenu Class Reference	33
9.14	contextMenuObject Class Reference	35
9.15	QEPeriodic::elementInfoStruct Struct Reference	35
9.16	flipRotateMenu Class Reference	36
9.17	imageContextMenu Class Reference	36
9.18	imageMarkup Class Reference	37
9.19	localEnumerationItem Class Reference	38
9.20	managePixmap Class Reference	39
9.21	markupBeam Class Reference	39
9.22	markupHLine Class Reference	40
9.23	markupItem Class Reference	41
9.24	markupLine Class Reference	42
9.25	markupRegion Class Reference	43
9.26	markupTarget Class Reference	44
9.27	markupText Class Reference	44
9.28	markupVLine Class Reference	45
9.29	PeriodicDialog Class Reference	46
9.30	PeriodicElementSetupForm Class Reference	46
9.31	PeriodicSetupDialog Class Reference	46
9.32	QEStripChart::PrivateData Class Reference	47
9.33	QEStripChartItem::PrivateData Class Reference	47
9.34	profilePlot Class Reference	47
9.35	PushButtonSpecifications Struct Reference	48

9.36 QBitStatus Class Reference	48
9.37 QCaAlarmInfo Class Reference	50
9.38 QCaConnectionInfo Class Reference	50
9.39 QCaDataPoint Struct Reference	51
9.40 QCaDataPointList Class Reference	51
9.41 QCaDateTime Class Reference	51
9.42 QCaEventFilter Class Reference	52
9.43 QCaEventItem Class Reference	52
9.44 QCaEventUpdate Class Reference	52
9.45 QCaInstalledFiltersListItem Class Reference	53
9.46 qcaobject::QCaObject Class Reference	53
9.47 qcastatemachine::QCaStateMachine Class Reference	55
9.48 QCaVariableNamePropertyManager Class Reference	55
9.49 QEAnalogIndicator Class Reference	56
9.49.1 Detailed Description	59
9.49.2 Member Enumeration Documentation	59
9.49.2.1 Modes	59
9.49.2.2 Orientations	59
9.49.3 Property Documentation	60
9.49.3.1 backgroundColour	60
9.49.3.2 borderColour	60
9.49.3.3 centreAngle	60
9.49.3.4 fontColour	60
9.49.3.5 foregroundColour	60
9.49.3.6 logScale	60
9.49.3.7 logScaleInterval	60
9.49.3.8 majorInterval	60
9.49.3.9 maximum	60
9.49.3.10 minimum	60
9.49.3.11 minorInterval	61
9.49.3.12 mode	61
9.49.3.13 orientation	61
9.49.3.14 showScale	61
9.49.3.15 showText	61

9.49.3.16	spanAngle	61
9.49.3.17	value	61
9.50	QEAAnalogProgressBar Class Reference	61
9.50.1	Member Enumeration Documentation	64
9.50.1.1	ArrayActions	64
9.50.1.2	Formats	65
9.50.1.3	Notations	65
9.50.1.4	UserLevels	65
9.50.2	Constructor & Destructor Documentation	65
9.50.2.1	QEAAnalogProgressBar	65
9.50.2.2	QEAAnalogProgressBar	66
9.50.3	Member Function Documentation	66
9.50.3.1	dbValueChanged	66
9.50.3.2	requestEnabled	66
9.50.3.3	setVariableNameAndSubstitutions	66
9.50.4	Property Documentation	66
9.50.4.1	addUnits	66
9.50.4.2	alarmSeverityDisplayMode	66
9.50.4.3	allowDrop	67
9.50.4.4	arrayAction	67
9.50.4.5	enabled	67
9.50.4.6	format	67
9.50.4.7	int	67
9.50.4.8	leadingZero	68
9.50.4.9	localEnumeration	68
9.50.4.10	notation	68
9.50.4.11	precision	68
9.50.4.12	trailingZeros	68
9.50.4.13	useDbDisplayLimits	68
9.50.4.14	useDbPrecision	68
9.50.4.15	userLevelEnabled	68
9.50.4.16	userLevelEngineerStyle	69
9.50.4.17	userLevelScientistStyle	69
9.50.4.18	userLevelUserStyle	69

9.50.4.19	userLevelVisibility	69
9.50.4.20	variable	69
9.50.4.21	variableAsToolTip	69
9.50.4.22	variableSubstitutions	70
9.50.4.23	visible	70
9.51	QEBitStatus Class Reference	70
9.51.1	Member Enumeration Documentation	72
9.51.1.1	UserLevels	72
9.51.2	Member Function Documentation	72
9.51.2.1	dbValueChanged	72
9.51.2.2	requestEnabled	72
9.51.2.3	setVariableNameAndSubstitutions	72
9.51.3	Property Documentation	73
9.51.3.1	allowDrop	73
9.51.3.2	enabled	73
9.51.3.3	int	73
9.51.3.4	userLevelEnabled	73
9.51.3.5	userLevelEngineerStyle	73
9.51.3.6	userLevelScientistStyle	74
9.51.3.7	userLevelUserStyle	74
9.51.3.8	userLevelVisibility	74
9.51.3.9	variable	74
9.51.3.10	variableAsToolTip	74
9.51.3.11	variableSubstitutions	74
9.51.3.12	visible	75
9.52	QEByteArray Class Reference	75
9.53	QEComboBox Class Reference	76
9.53.1	Member Enumeration Documentation	78
9.53.1.1	UserLevels	78
9.53.2	Member Function Documentation	78
9.53.2.1	dbValueChanged	78
9.53.2.2	requestEnabled	78
9.53.3	Member Data Documentation	78
9.53.3.1	writeOnChange	78

9.53.4	Property Documentation	78
9.53.4.1	allowDrop	78
9.53.4.2	enabled	79
9.53.4.3	int	79
9.53.4.4	subscribe	79
9.53.4.5	userLevelEnabled	79
9.53.4.6	userLevelEngineerStyle	79
9.53.4.7	userLevelScientistStyle	79
9.53.4.8	userLevelUserStyle	80
9.53.4.9	userLevelVisibility	80
9.53.4.10	variable	80
9.53.4.11	variableAsToolTip	80
9.53.4.12	variableSubstitutions	80
9.53.4.13	visible	80
9.54	QEConfiguredLayout Class Reference	81
9.55	QEConfiguredLayoutManager Class Reference	82
9.56	QEDragDrop Class Reference	83
9.57	QEFileBrowser Class Reference	84
9.58	QEFloating Class Reference	86
9.59	QEFloatingFormatting Class Reference	86
9.60	QEForm Class Reference	87
9.60.1	Member Function Documentation	88
9.60.1.1	setVariableNameAndSubstitutions	88
9.61	QEFrame Class Reference	89
9.61.1	Member Enumeration Documentation	90
9.61.1.1	UserLevels	90
9.61.2	Member Function Documentation	90
9.61.2.1	requestEnabled	90
9.61.3	Property Documentation	90
9.61.3.1	allowDrop	90
9.61.3.2	enabled	91
9.61.3.3	int	91
9.61.3.4	userLevelEnabled	91
9.61.3.5	userLevelEngineerStyle	91

9.61.3.6	userLevelScientistStyle	91
9.61.3.7	userLevelUserStyle	92
9.61.3.8	userLevelVisibility	92
9.61.3.9	variableAsToolTip	92
9.61.3.10	visible	92
9.62	QEGenericButton Class Reference	92
9.63	QEGroupBox Class Reference	94
9.63.1	Member Enumeration Documentation	95
9.63.1.1	UserLevels	95
9.63.2	Member Function Documentation	96
9.63.2.1	requestEnabled	96
9.63.3	Property Documentation	96
9.63.3.1	allowDrop	96
9.63.3.2	enabled	96
9.63.3.3	int	96
9.63.3.4	userLevelEnabled	96
9.63.3.5	userLevelEngineerStyle	96
9.63.3.6	userLevelScientistStyle	97
9.63.3.7	userLevelUserStyle	97
9.63.3.8	userLevelVisibility	97
9.63.3.9	variableAsToolTip	97
9.63.3.10	visible	97
9.64	QEImage Class Reference	98
9.64.1	Member Enumeration Documentation	102
9.64.1.1	UserLevels	102
9.64.2	Member Function Documentation	103
9.64.2.1	dbValueChanged	103
9.64.2.2	requestEnabled	103
9.64.3	Property Documentation	103
9.64.3.1	allowDrop	103
9.64.3.2	beamXVariable	103
9.64.3.3	beamYVariable	103
9.64.3.4	clippingHighVariable	103
9.64.3.5	clippingLowVariable	103

9.64.3.6	clippingOnOffVariable	104
9.64.3.7	enabled	104
9.64.3.8	heightVariable	104
9.64.3.9	imageVariable	104
9.64.3.10	int	104
9.64.3.11	regionOfInterestHVariable	104
9.64.3.12	regionOfInterestWVariable	104
9.64.3.13	regionOfInterestXVariable	104
9.64.3.14	regionOfInterestYVariable	105
9.64.3.15	targetTriggerVariable	105
9.64.3.16	targetXVariable	105
9.64.3.17	targetYVariable	105
9.64.3.18	userLevelEnabled	105
9.64.3.19	userLevelEngineerStyle	105
9.64.3.20	userLevelScientistStyle	105
9.64.3.21	userLevelUserStyle	106
9.64.3.22	userLevelVisibility	106
9.64.3.23	variableAsToolTip	106
9.64.3.24	variableSubstitutions	106
9.64.3.25	visible	106
9.64.3.26	widthVariable	106
9.65	QEInteger Class Reference	107
9.66	QEIntegerFormatting Class Reference	107
9.66.1	Detailed Description	108
9.66.2	Member Function Documentation	108
9.66.2.1	formatInteger	108
9.66.2.2	formatIntegerArray	108
9.66.2.3	formatValue	109
9.67	QELabel Class Reference	109
9.67.1	Detailed Description	112
9.67.2	Member Enumeration Documentation	113
9.67.2.1	ArrayActions	113
9.67.2.2	Formats	113
9.67.2.3	Notations	113

9.67.2.4	updateOptions	113
9.67.2.5	UserLevels	114
9.67.3	Constructor & Destructor Documentation	114
9.67.3.1	QELabel	114
9.67.3.2	QELabel	114
9.67.4	Member Function Documentation	114
9.67.4.1	dbValueChanged	114
9.67.4.2	requestEnabled	114
9.67.5	Property Documentation	114
9.67.5.1	addUnits	114
9.67.5.2	allowDrop	115
9.67.5.3	arrayAction	115
9.67.5.4	enabled	115
9.67.5.5	format	115
9.67.5.6	int	115
9.67.5.7	leadingZero	116
9.67.5.8	localEnumeration	116
9.67.5.9	notation	116
9.67.5.10	pixmap0	116
9.67.5.11	pixmap1	116
9.67.5.12	pixmap2	116
9.67.5.13	pixmap3	116
9.67.5.14	pixmap4	116
9.67.5.15	pixmap5	116
9.67.5.16	pixmap6	117
9.67.5.17	pixmap7	117
9.67.5.18	precision	117
9.67.5.19	trailingZeros	117
9.67.5.20	updateOption	117
9.67.5.21	useDbPrecision	117
9.67.5.22	userLevelEnabled	117
9.67.5.23	userLevelEngineerStyle	118
9.67.5.24	userLevelScientistStyle	118
9.67.5.25	userLevelUserStyle	118

9.67.5.26	userLevelVisibility	118
9.67.5.27	variable	118
9.67.5.28	variableAsToolTip	118
9.67.5.29	variableSubstitutions	119
9.67.5.30	visible	119
9.68	QLineEdit Class Reference	119
9.68.1	Member Enumeration Documentation	122
9.68.1.1	ArrayActions	122
9.68.1.2	Formats	122
9.68.1.3	Notations	122
9.68.1.4	UserLevels	122
9.68.2	Constructor & Destructor Documentation	123
9.68.2.1	QLineEdit	123
9.68.2.2	QLineEdit	123
9.68.3	Member Function Documentation	123
9.68.3.1	dbValueChanged	123
9.68.3.2	getConfirmWrite	123
9.68.3.3	getSubscribe	123
9.68.3.4	getWriteOnEnter	123
9.68.3.5	getWriteOnFinish	123
9.68.3.6	getWriteOnLoseFocus	124
9.68.3.7	requestEnabled	124
9.68.3.8	setConfirmWrite	124
9.68.3.9	setSubscribe	124
9.68.3.10	setWriteOnEnter	124
9.68.3.11	setWriteOnFinish	124
9.68.3.12	setWriteOnLoseFocus	124
9.68.4	Property Documentation	124
9.68.4.1	addUnits	124
9.68.4.2	allowDrop	125
9.68.4.3	arrayAction	125
9.68.4.4	confirmWrite	125
9.68.4.5	enabled	125
9.68.4.6	format	125

9.68.4.7	int	125
9.68.4.8	leadingZero	126
9.68.4.9	localEnumeration	126
9.68.4.10	notation	126
9.68.4.11	precision	126
9.68.4.12	subscribe	126
9.68.4.13	trailingZeros	126
9.68.4.14	useDbPrecision	126
9.68.4.15	userLevelEnabled	127
9.68.4.16	userLevelEngineerStyle	127
9.68.4.17	userLevelScientistStyle	127
9.68.4.18	userLevelUserStyle	127
9.68.4.19	userLevelVisibility	127
9.68.4.20	variable	128
9.68.4.21	variableAsToolTip	128
9.68.4.22	variableSubstitutions	128
9.68.4.23	visible	128
9.68.4.24	writeOnEnter	128
9.68.4.25	writeOnFinish	128
9.68.4.26	writeOnLoseFocus	128
9.69	QELineEditManager Class Reference	129
9.70	QELink Class Reference	129
9.71	QELog Class Reference	131
9.72	QELogin Class Reference	133
9.73	QEPeriodic Class Reference	134
9.73.1	Member Enumeration Documentation	138
9.73.1.1	UserLevels	138
9.73.2	Member Function Documentation	138
9.73.2.1	dbElementChanged	138
9.73.2.2	dbValueChanged	138
9.73.2.3	requestEnabled	138
9.73.3	Member Data Documentation	138
9.73.3.1	allowDrop	138
9.73.4	Property Documentation	138

9.73.4.1	enabled	139
9.73.4.2	int	139
9.73.4.3	readbackLabelVariable1	139
9.73.4.4	readbackLabelVariable2	139
9.73.4.5	subscribe	139
9.73.4.6	userLevelEnabled	139
9.73.4.7	userLevelEngineerStyle	139
9.73.4.8	userLevelScientistStyle	140
9.73.4.9	userLevelUserStyle	140
9.73.4.10	userLevelVisibility	140
9.73.4.11	variableAsToolTip	140
9.73.4.12	variableSubstitutions	140
9.73.4.13	visible	141
9.73.4.14	writeButtonVariable1	141
9.73.4.15	writeButtonVariable2	141
9.74	QEPeiodicComponentData Class Reference	141
9.75	QEPeiodicTaskMenu Class Reference	141
9.76	QEPeiodicTaskMenuFactory Class Reference	142
9.77	QEpicsPV Class Reference	142
9.78	QEPlot Class Reference	143
9.78.1	Member Enumeration Documentation	146
9.78.1.1	UserLevels	146
9.78.2	Member Function Documentation	147
9.78.2.1	dbValueChanged	147
9.78.2.2	dbValueChanged	147
9.78.2.3	requestEnabled	147
9.78.3	Member Data Documentation	147
9.78.3.1	allowDrop	147
9.78.4	Property Documentation	147
9.78.4.1	enabled	147
9.78.4.2	int	148
9.78.4.3	userLevelEnabled	148
9.78.4.4	userLevelEngineerStyle	148
9.78.4.5	userLevelScientistStyle	148

9.78.4.6	userLevelUserStyle	148
9.78.4.7	userLevelVisibility	149
9.78.4.8	variable1	149
9.78.4.9	variable2	149
9.78.4.10	variable3	149
9.78.4.11	variable4	149
9.78.4.12	variableAsToolTip	149
9.78.4.13	variableSubstitutions	149
9.78.4.14	visible	150
9.79	QEPushButton Class Reference	150
9.79.1	Member Enumeration Documentation	153
9.79.1.1	UserLevels	153
9.79.2	Member Function Documentation	153
9.79.2.1	dbValueChanged	153
9.79.2.2	requestEnabled	153
9.79.3	Property Documentation	153
9.79.3.1	allowDrop	153
9.79.3.2	altReadbackVariable	153
9.79.3.3	enabled	153
9.79.3.4	int	154
9.79.3.5	subscribe	154
9.79.3.6	userLevelEnabled	154
9.79.3.7	userLevelEngineerStyle	154
9.79.3.8	userLevelScientistStyle	154
9.79.3.9	userLevelUserStyle	155
9.79.3.10	userLevelVisibility	155
9.79.3.11	variable	155
9.79.3.12	variableAsToolTip	155
9.79.3.13	visible	155
9.80	QEPvProperties Class Reference	155
9.80.1	Member Enumeration Documentation	157
9.80.1.1	UserLevels	157
9.80.2	Member Function Documentation	158
9.80.2.1	requestEnabled	158

9.80.3	Property Documentation	158
9.80.3.1	allowDrop	158
9.80.3.2	enabled	158
9.80.3.3	int	158
9.80.3.4	userLevelEnabled	158
9.80.3.5	userLevelEngineerStyle	158
9.80.3.6	userLevelScientistStyle	159
9.80.3.7	userLevelUserStyle	159
9.80.3.8	userLevelVisibility	159
9.80.3.9	variable	159
9.80.3.10	variableAsToolTip	159
9.80.3.11	variableSubstitutions	160
9.80.3.12	visible	160
9.81	QEPvPropertiesManager Class Reference	160
9.82	QERadioButton Class Reference	160
9.82.1	Member Enumeration Documentation	163
9.82.1.1	UserLevels	163
9.82.2	Member Function Documentation	164
9.82.2.1	dbValueChanged	164
9.82.2.2	requestEnabled	164
9.82.3	Property Documentation	164
9.82.3.1	allowDrop	164
9.82.3.2	enabled	164
9.82.3.3	int	164
9.82.3.4	subscribe	164
9.82.3.5	userLevelEnabled	165
9.82.3.6	userLevelEngineerStyle	165
9.82.3.7	userLevelScientistStyle	165
9.82.3.8	userLevelUserStyle	165
9.82.3.9	userLevelVisibility	165
9.82.3.10	variable	166
9.82.3.11	variableAsToolTip	166
9.82.3.12	variableSubstitutions	166
9.82.3.13	visible	166

9.83	QERecipe Class Reference	166
9.84	QEScript Class Reference	168
9.85	QEShape Class Reference	170
9.85.1	Detailed Description	174
9.85.2	Member Enumeration Documentation	174
9.85.2.1	animationOptions	174
9.85.2.2	shapeOptions	175
9.85.2.3	UserLevels	175
9.85.3	Constructor & Destructor Documentation	175
9.85.3.1	QEShape	175
9.85.3.2	QEShape	175
9.85.4	Member Function Documentation	175
9.85.4.1	dbValueChanged1	175
9.85.4.2	dbValueChanged2	175
9.85.4.3	dbValueChanged3	176
9.85.4.4	dbValueChanged4	176
9.85.4.5	dbValueChanged5	176
9.85.4.6	dbValueChanged6	176
9.85.4.7	requestEnabled	176
9.85.5	Property Documentation	176
9.85.5.1	allowDrop	176
9.85.5.2	animation1	176
9.85.5.3	animation2	177
9.85.5.4	animation3	177
9.85.5.5	animation4	177
9.85.5.6	animation5	177
9.85.5.7	animation6	177
9.85.5.8	color1	177
9.85.5.9	color10	177
9.85.5.10	color2	177
9.85.5.11	color3	177
9.85.5.12	color4	178
9.85.5.13	color5	178
9.85.5.14	color6	178

9.85.5.15 color7	178
9.85.5.16 color8	178
9.85.5.17 color9	178
9.85.5.18 enabled	178
9.85.5.19 int	178
9.85.5.20 offset1	179
9.85.5.21 offset2	179
9.85.5.22 offset3	179
9.85.5.23 offset4	179
9.85.5.24 offset5	179
9.85.5.25 offset6	179
9.85.5.26 point1	179
9.85.5.27 point10	179
9.85.5.28 point2	180
9.85.5.29 point3	180
9.85.5.30 point4	180
9.85.5.31 point5	180
9.85.5.32 point6	180
9.85.5.33 point7	180
9.85.5.34 point8	180
9.85.5.35 point9	180
9.85.5.36 scale2	180
9.85.5.37 scale3	181
9.85.5.38 scale4	181
9.85.5.39 scale5	181
9.85.5.40 scale6	181
9.85.5.41 userLevelEnabled	181
9.85.5.42 userLevelEngineerStyle	181
9.85.5.43 userLevelScientistStyle	181
9.85.5.44 userLevelUserStyle	182
9.85.5.45 userLevelVisibility	182
9.85.5.46 variable1	182
9.85.5.47 variable2	182
9.85.5.48 variable3	182

9.85.5.49 variable4	182
9.85.5.50 variable5	183
9.85.5.51 variable6	183
9.85.5.52 variableAsToolTip	183
9.85.5.53 variableSubstitutions	183
9.85.5.54 visible	183
9.86 QESlider Class Reference	183
9.86.1 Member Enumeration Documentation	185
9.86.1.1 UserLevels	185
9.86.2 Member Function Documentation	186
9.86.2.1 dbValueChanged	186
9.86.2.2 requestEnabled	186
9.86.3 Member Data Documentation	186
9.86.3.1 writeOnChange	186
9.86.4 Property Documentation	186
9.86.4.1 allowDrop	186
9.86.4.2 enabled	186
9.86.4.3 int	186
9.86.4.4 subscribe	187
9.86.4.5 userLevelEnabled	187
9.86.4.6 userLevelEngineerStyle	187
9.86.4.7 userLevelScientistStyle	187
9.86.4.8 userLevelUserStyle	187
9.86.4.9 userLevelVisibility	188
9.86.4.10 variable	188
9.86.4.11 variableAsToolTip	188
9.86.4.12 variableSubstitutions	188
9.86.4.13 visible	188
9.87 QESpinBox Class Reference	188
9.87.1 Member Enumeration Documentation	191
9.87.1.1 UserLevels	191
9.87.2 Member Function Documentation	191
9.87.2.1 dbValueChanged	191
9.87.2.2 requestEnabled	191

9.87.3	Property Documentation	191
9.87.3.1	allowDrop	191
9.87.3.2	enabled	191
9.87.3.3	int	192
9.87.3.4	subscribe	192
9.87.3.5	userLevelEnabled	192
9.87.3.6	userLevelEngineerStyle	192
9.87.3.7	userLevelScientistStyle	192
9.87.3.8	userLevelUserStyle	192
9.87.3.9	userLevelVisibility	193
9.87.3.10	variable	193
9.87.3.11	variableAsToolTip	193
9.87.3.12	variableSubstitutions	193
9.87.3.13	visible	193
9.88	QString Class Reference	194
9.89	QStringFormatting Class Reference	194
9.89.1	Member Enumeration Documentation	195
9.89.1.1	arrayActions	195
9.89.1.2	formats	196
9.89.1.3	notations	196
9.90	QStringFormattingMethods Class Reference	196
9.91	QStripChart Class Reference	197
9.92	QStripChartItem Class Reference	199
9.93	QStripChartItemDialog Class Reference	200
9.94	QStripChartTimeDialog Class Reference	200
9.95	QSubstitutedLabel Class Reference	200
9.95.1	Member Data Documentation	201
9.95.1.1	labelText	201
9.95.2	Property Documentation	201
9.95.2.1	textSubstitutions	201
9.96	QEToolTip Class Reference	201
9.97	QEWidget Class Reference	203
9.97.1	Detailed Description	204
9.97.2	Member Function Documentation	205

9.97.2.1	activate	205
9.97.2.2	defaultFileLocation	205
9.97.2.3	getColor	205
9.97.2.4	getMessageSourceId	206
9.97.2.5	getQcaltem	206
9.97.2.6	openQEFile	206
9.97.2.7	readNow	206
9.97.2.8	setMessageSourceId	206
9.97.2.9	setupContextMenu	206
9.97.2.10	setVariableNameAndSubstitutions	206
9.97.2.11	writeNow	206
9.98	QEWidgets Class Reference	207
9.99	qcastatemachine::ReadQCaStateMachine Class Reference	207
9.100	RecordSpec Class Reference	207
9.101	RecordSpecList Class Reference	208
9.102	selectMenu Class Reference	208
9.103	standardProperties Class Reference	208
9.104	StateMachineTemplate Class Reference	210
9.105	qcastatemachine::SubscriptionQCaStateMachine Class Reference	210
9.106	trace Class Reference	211
9.107	TrackRange Class Reference	211
9.108	userInfoStruct Class Reference	212
9.109	QEPeriodic::userInfoStructArray Struct Reference	212
9.110	userLevelSignal Class Reference	212
9.111	userLevelSlot Class Reference	213
9.112	UserMessage Class Reference	213
9.112.1	Detailed Description	215
9.113	UserMessageSignal Class Reference	216
9.113.1	Detailed Description	217
9.114	UserMessageSlot Class Reference	217
9.114.1	Detailed Description	217
9.115	VideoWidget Class Reference	218
9.116	WidgetRef Class Reference	219
9.117	qcastatemachine::WriteQCaStateMachine Class Reference	219

9.118zoomMenu Class Reference	219
---	-----

Chapter 1

QE framework - EPICS aware Qt Widgets and data access classes

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

1.1 Documentation

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

1.2 License

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

1.3 Platforms

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

1.4 Screenshots

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

Screenshots are only available in the HTML docs.

1.5 Downloads

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

For getting a development snapshot from the SVN repository:

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

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

1.6 Installation

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

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

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

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

1.7 Support

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

1.8 Related Projects

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

1.9 Credits:

Authors:

Andrew Rhyder, Anthony Owen, Glenn Jackson

Project admin:

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

Chapter 2

GNU General Public License

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

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

See the GNU General Public License for more details.

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

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

Chapter 3

ASgui screen shots

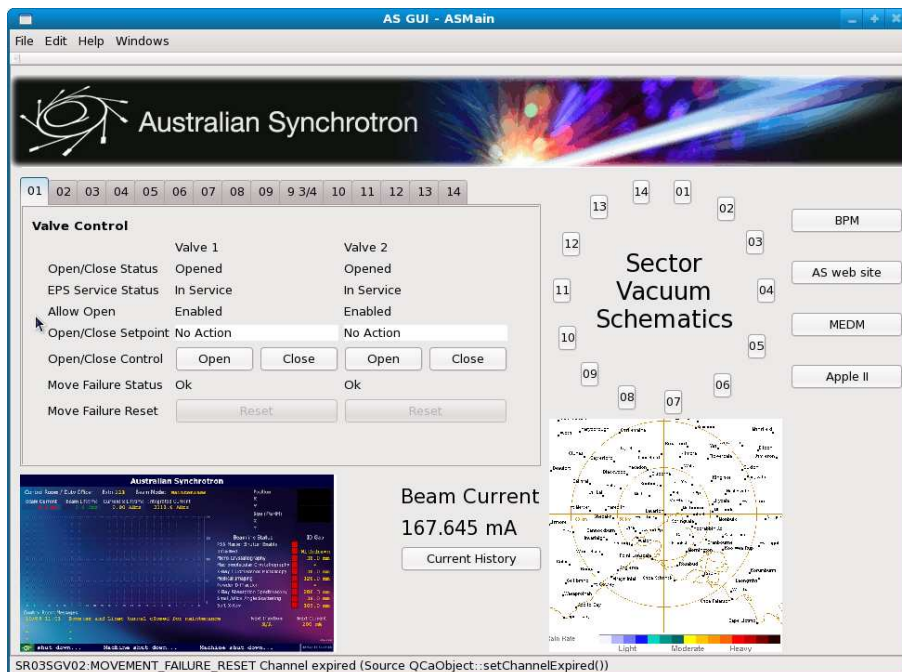


Figure 3.1: Australian Synchrotron mock up

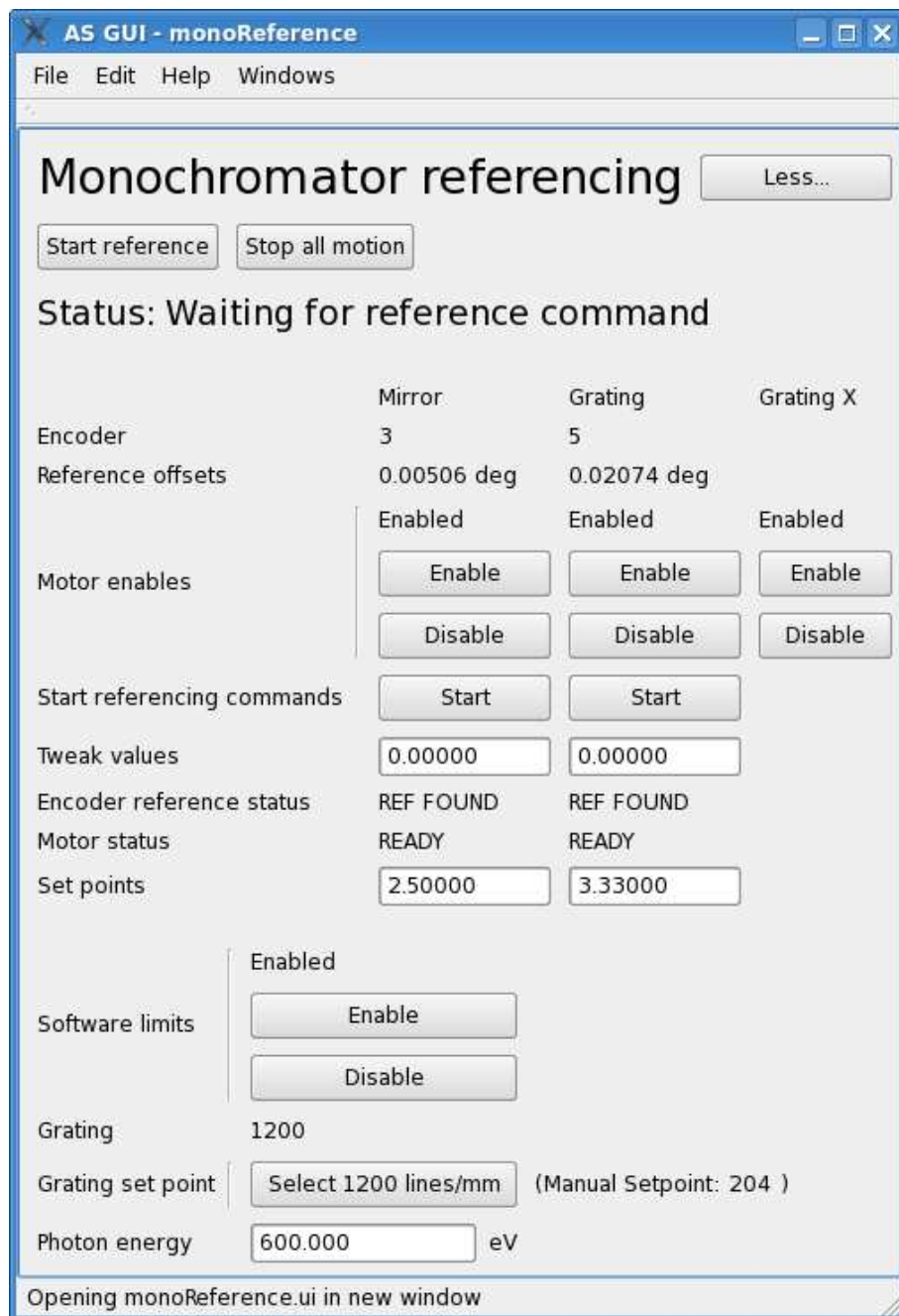


Figure 3.2: Monochromator referencing

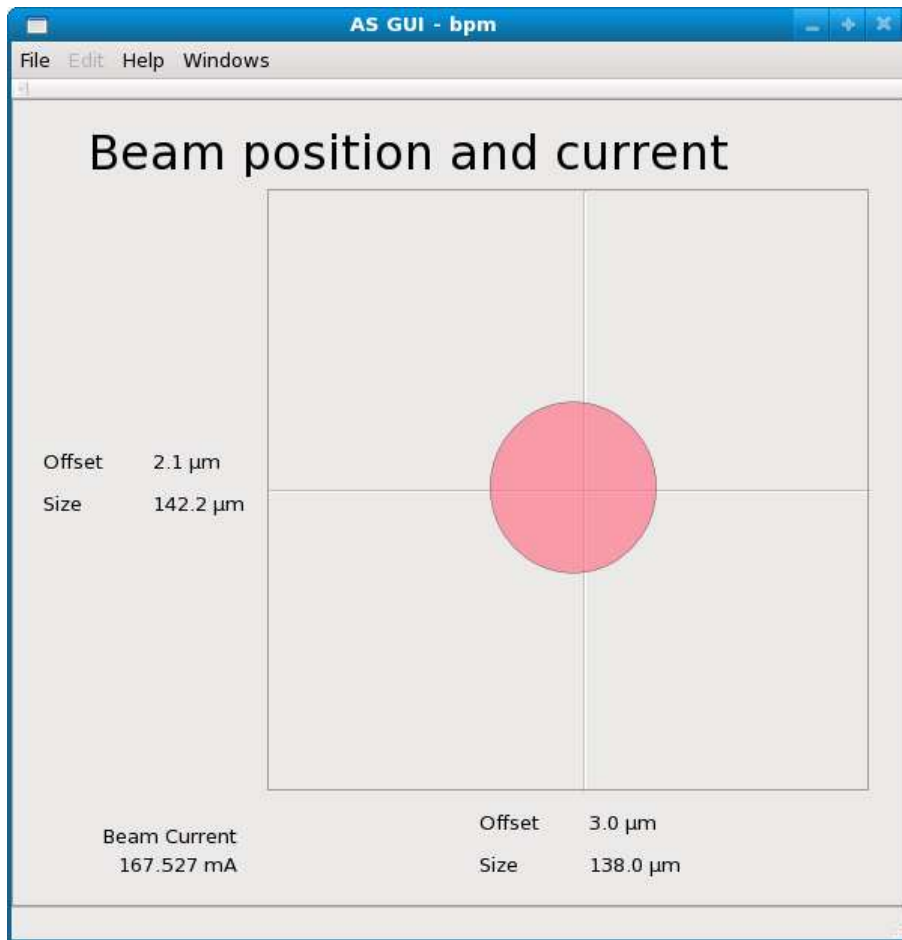


Figure 3.3: Beam position monitor

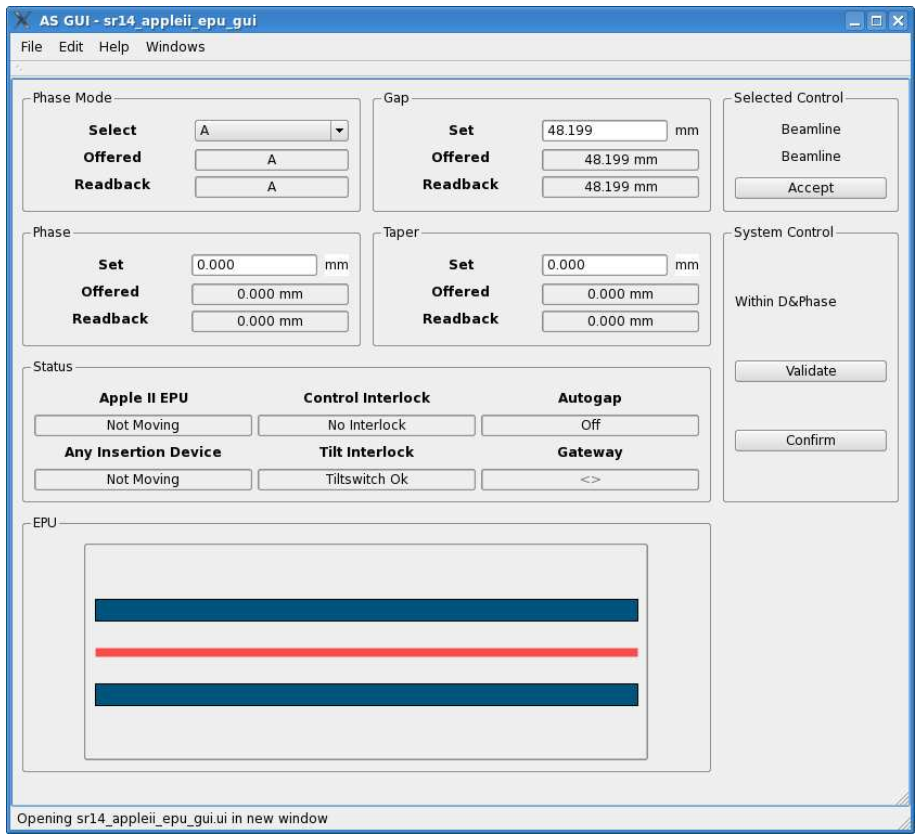


Figure 3.4: Insertion device



Figure 3.5: Injection efficiency monitor

Chapter 4

other applications using epicsqt widgets

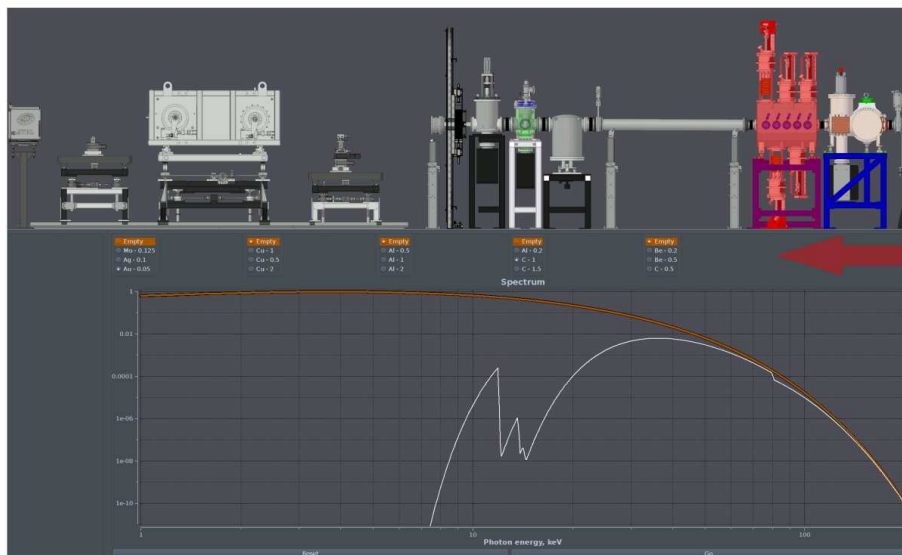


Figure 4.1: Medical Imaging beamline

2B SampleTable Z <@SR08ID01OPI01>

PV name: SR08ID01:MTR32B View mode: Macro

Description: 2B SampleTable Z

Precision: 5 Units: mm

Message: Connection established. clean

User: 6mm Move absolutely Raw: -80932

JOG< UNDO >JOG

LIMIT< Move relatively >LIMIT

< 1mm >

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

User: 6mm = Hi limit

Resolution: 0.0001mm/step * 42271.2068mm

Raw: -80932 + Lo limit

Offset: -2.0932mm -42275.3932mm

Speed Acceleration

Maximum: 1.2mm/s

Normal: 0.8mm/s 1s

Backlash: 0mm/s 1s

log: 1mm/s 10s

Backlash: 0mm

Figure 4.2: Motor controller

MotorMx <@SR08ID01OPI01>

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

Add motor

Figure 4.3: Motor controller

Chapter 5

Qt Designer

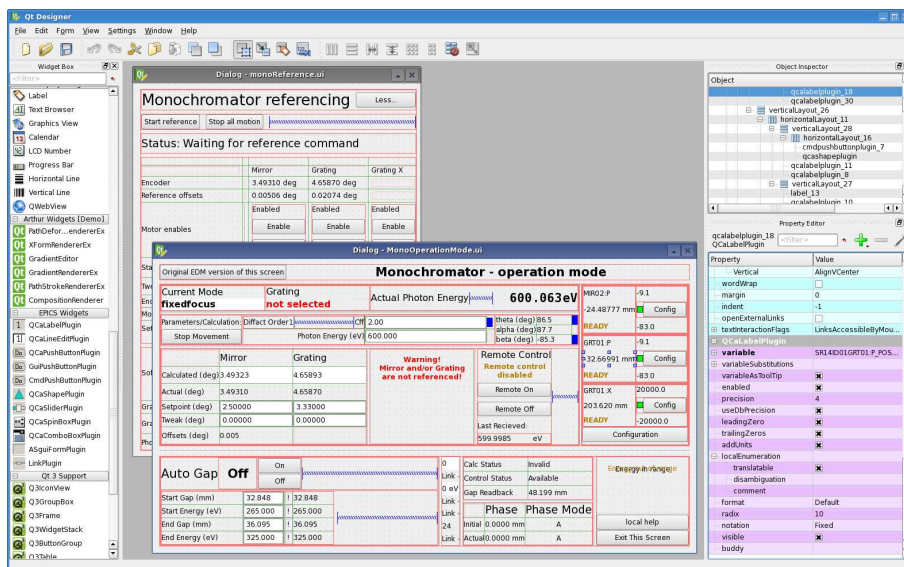


Figure 5.1: Editing multiple GUIs

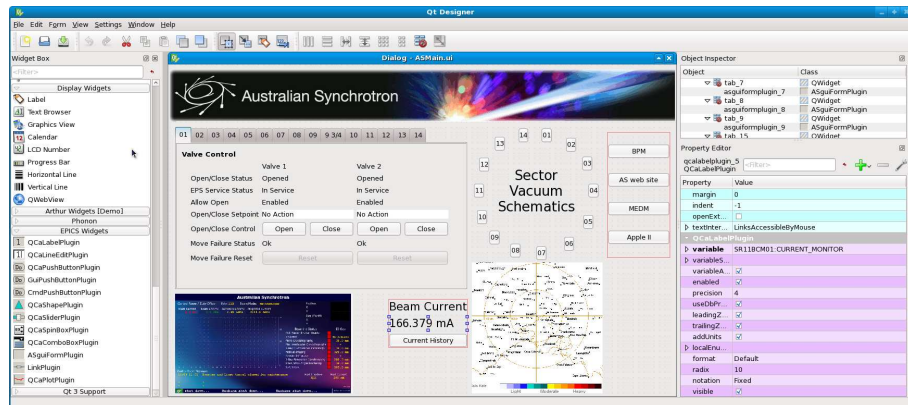


Figure 5.2: Editing a GUI

Chapter 6

Qt Creator

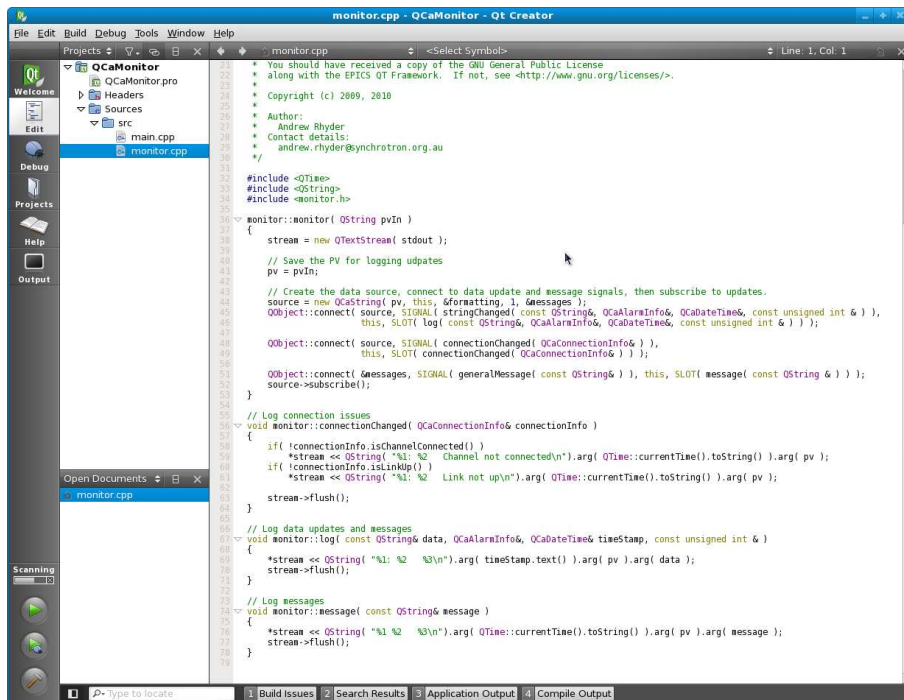


Figure 6.1: Application using epicsqt data source classes

Chapter 7

Class Index

7.1 Class Hierarchy

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

_Field	27
_Item	28
_QDialogItem	28
_QDialogLogin	28
_QPushButtonGroup	29
_QTableWidgetFileBrowser	29
_QTableWidgetLog	30
_QTableWidgetScript	30
QEEAnalogIndicator::Band	30
QEEAnalogIndicator::BandList	31
ContainerProfile	31
QEEWidget	203
QEEAnalogProgressBar	61
QEEBitStatus	70
QEEComboBox	76
QEEConfiguredLayout	81
QEEFileBrowser	84
QEEForm	87
QEEFrame	89
QEEGenericButton	92
QEEPushButton	150
QEERadioButton	160
QEEGroupBox	94
QEEImage	98
QEELabel	109
QEELineEdit	119
QEELink	129
QEELog	131
QEELogin	133

QEPeriodic	134
QEPlot	143
QEPvProperties	155
QERecipe	166
QEScript	168
QEShape	170
QESlider	183
QESpinBox	188
QEStripChart	197
QESubstitutedLabel	200
contextMenu	33
QEWidget	203
contextMenuObject	35
QEPeriodic::elementInfoStruct	35
flipRotateMenu	36
imageContextMenu	36
imageMarkup	37
VideoWidget	218
localEnumerationItem	38
managePixmap	39
QEGenericButton	92
QELabel	109
markupItem	41
markupBeam	39
markupHLine	40
markupLine	42
markupRegion	43
markupTarget	44
markupText	44
markupVLine	45
PeriodicDialog	46
PeriodicElementSetupForm	46
PeriodicSetupDialog	46
QEStripChart::PrivateData	47
QEStripChartItem::PrivateData	47
profilePlot	47
PushButtonSpecifications	48
QBitStatus	48
QEBitStatus	70
QCaAlarmInfo	50
QCaConnectionInfo	50
QCaDataPoint	51
QCaDataPointList	51
QCaDateTime	51
QCaEventFilter	52
QCaEventItem	52
QCaEventUpdate	52
QCaInstalledFiltersListItem	53

qcaobject::QCaObject	53
QEByteArray	75
QEFloating	86
QEInteger	107
QEString	194
QCaVariableNamePropertyManager	55
QEAnalogIndicator	56
QEAnalogProgressBar	61
QEConfiguredLayoutManager	82
QEDragDrop	83
QEWidget	203
QEFloatingFormatting	86
QEIntegerFormatting	107
QELineEditManager	129
QEPeriodicComponentData	141
QEPeriodicTaskMenu	141
QEPeriodicTaskMenuFactory	142
QEpicsPV	142
QEPvPropertiesManager	160
QEStringFormatting	194
QEStringFormattingMethods	196
QEAnalogProgressBar	61
QEGenericButton	92
QELabel	109
QELineEdit	119
QEStripChartItem	199
QEStripChartItemDialog	200
QEStripChartTimeDialog	200
QEToolTip	201
QEWidget	203
QEWidgets	207
RecordSpec	207
RecordSpecList	208
selectMenu	208
standardProperties	208
QEWidget	203
StateMachineTemplate	210
qcastatemachine::QCaStateMachine	55
qcastatemachine::ConnectionQCaStateMachine	31
qcastatemachine::ReadQCaStateMachine	207
qcastatemachine::SubscriptionQCaStateMachine	210
qcastatemachine::WriteQCaStateMachine	219
trace	211
TrackRange	211
userInfoStruct	212
QEPeriodic::userInfoStructArray	212
userLevelSignal	212

userLevelSlot	213
UserMessage	213
QEWidget	203
UserMessageSignal	216
UserMessageSlot	217
WidgetRef	219
zoomMenu	219

Chapter 8

Class Index

8.1 Class List

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

_Field	27
_Item	28
_QDialogItem	28
_QDialogLogin	28
_QPushButtonGroup	29
_QTableWidgetFileBrowser	29
_QTableWidgetLog	30
_QTableWidgetScript	30
QEAnalogIndicator::Band	30
QEAnalogIndicator::BandList	31
qcastatemachine::ConnectionQCaStateMachine	31
ContainerProfile	31
contextMenu	33
contextMenuObject	35
QEPeriodic::elementInfoStruct	35
flipRotateMenu	36
imageContextMenu	36
imageMarkup	37
localEnumerationItem	38
managePixmaps	39
markupBeam	39
markupHLine	40
markupItem	41
markupLine	42
markupRegion	43
markupTarget	44
markupText	44
markupVLine	45
PeriodicDialog	46

PeriodicElementSetupForm	46
PeriodicSetupDialog	46
QEStripChart::PrivateData	47
QEStripChartItem::PrivateData	47
profilePlot	47
PushButtonSpecifications	48
QBitStatus	48
QCaAlarmInfo	50
QCaConnectionInfo	50
QCaDataPoint	51
QCaDataPointList	51
QCaDateTime	51
QCaEventFilter	52
QCaEventItem	52
QCaEventUpdate	52
QCaInstalledFiltersListItem	53
qcaobject::QCaObject	53
qcastatemachine::QCaStateMachine	55
QCaVariableNamePropertyManager	55
QERAnalogIndicator	56
QERAnalogProgressBar	61
QERBitStatus	70
QERByteArray	75
QERComboBox	76
QERConfiguredLayout	81
QERConfiguredLayoutManager	82
QERDragDrop	83
QERFileBrowser	84
QERFloating	86
QERFloatingFormatting	86
QERForm	87
QERFrame	89
QERGenericButton	92
QERGroupBox	94
QERImage	98
QERInteger	107
QERIntegerFormatting	107
QERLabel	109
QERLineEdit	119
QERLineEditManager	129
QERLink	129
QERLog	131
QERLogin	133
QERPeriodic	134
QERPeriodicComponentData	141
QERPeriodicTaskMenu	141
QERPeriodicTaskMenuFactory	142
QERpicsPV	142
QERPlot	143
QERPushButton	150

QEPvProperties	155
QEPvPropertiesManager	160
QERadioButton	160
QERecipe	166
QEScript	168
QEShape	170
QESlider	183
QESpinBox	188
QEStrng	194
QEStrngFormatting	194
QEStrngFormattingMethods	196
QEStrngChart	197
QEStrngChartItem	199
QEStrngChartItemDialog	200
QEStrngChartTimeDialog	200
QESubstitutedLabel	200
QEToolTip	201
QEWidgert	203
QEWidgerts	207
qcastatemachine::ReadQCaStateMachine	207
RecordSpec	207
RecordSpecList	208
selectMenu	208
standardProperties	208
StateMachineTemplate	210
qcastatemachine::SubscriptionQCaStateMachine	210
trace	211
TrackRange	211
userInfoStruct	212
QEPeriodic::userInfoStructArray	212
userLevelSignal	212
userLevelSlot	213
UserMessage	213
UserMessageSignal	216
UserMessageSlot	217
VideoWidget	218
WidgetRef	219
qcastatemachine::WriteQCaStateMachine	219
zoomMenu	219

Chapter 9

Class Documentation

9.1 _Field Class Reference

Public Member Functions

- [QEObject](#) * **getObject** ()
- void **setObject** (QObject *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

- [QEObject](#) * **qCaWidget**

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

- /home/rhydera/epicsqt/framework/widgets/QEObjectifiedLayout/QEObjectifiedLayout.h
- /home/rhydera/epicsqt/framework/widgets/QEObjectifiedLayout/QEObjectifiedLayout.cpp

9.2 `_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:

- /home/rhydera/epicsqt/framework/widgets/QEConfiguredLayout/QEConfiguredLayout.h
- /home/rhydera/epicsqt/framework/widgets/QEConfiguredLayout/QEConfiguredLayout.cpp

9.3 `_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:

- /home/rhydera/epicsqt/framework/widgets/QEConfiguredLayout/QEConfiguredLayout.h
- /home/rhydera/epicsqt/framework/widgets/QEConfiguredLayout/QEConfiguredLayout.cpp

9.4 `_QDialogLogin` Class Reference

Public Member Functions

- **_QDialogLogin** (QWidget *pParent=0, int pUserType=-1, Qt::WindowFlags pF=0)
- void **setCurrentUserType** (int pValue)
- void **setPassword** (QString pValue)

Protected Attributes

- `QGridLayout` * **`qGridLayout`**
- `QVBoxLayout` * **`qVBoxLayout`**
- `QGroupBox` * **`qGroupBox`**
- `QRadioButton` * **`qRadioButtonUser`**
- `QRadioButton` * **`qRadioButtonScientist`**
- `QRadioButton` * **`qRadioButtonEngineer`**
- `QLabel` * **`qLabelType`**
- `QLineEdit` * **`qLineEditPassword`**
- `QPushButton` * **`qPushButtonOk`**
- `QPushButton` * **`qPushButtonCancel`**
- `int` **`userType`**

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

- `/home/rhydera/epicsqt/framework/widgets/QELogin/QELogin.h`
- `/home/rhydera/epicsqt/framework/widgets/QELogin/QELogin.cpp`

9.5 `_QPushButtonGroup` Class Reference

Public Slots

- `void` **`buttonGroupClicked`** ()

Public Member Functions

- **`_QPushButtonGroup`** (`QWidget` *`pParent`=0, `QString` `pltemName`="", `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:

- `/home/rhydera/epicsqt/framework/widgets/QEConfiguredLayout/QEConfiguredLayout.h`
- `/home/rhydera/epicsqt/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:

- /home/rhydera/epicsqt/framework/widgets/QFileBrowser/QFileBrowser.h
- /home/rhydera/epicsqt/framework/widgets/QFileBrowser/QFileBrowser.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:

- /home/rhydera/epicsqt/framework/widgets/QELog/QELog.h
- /home/rhydera/epicsqt/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:

- /home/rhydera/epicsqt/framework/widgets/QEScript/QEScript.h
- /home/rhydera/epicsqt/framework/widgets/QEScript/QEScript.cpp

9.9 QEAnalogIndicator::Band Struct Reference

Public Attributes

- double **lower**

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

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

- /home/rhydera/epicsqt/framework/widgets/QEAnalogIndicator/QEAnalogIndicator.h

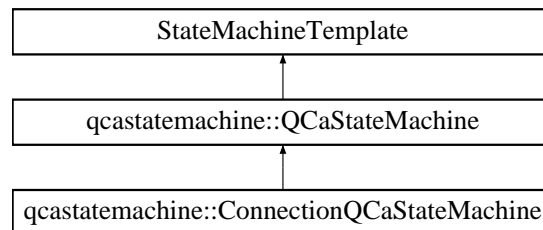
9.10 QEAnalogIndicator::BandList Class Reference

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

- /home/rhydera/epicsqt/framework/widgets/QEAnalogIndicator/QEAnalogIndicator.h

9.11 qcastatemachine::ConnectionQCaStateMachine Class Reference

Inheritance diagram for qcastatemachine::ConnectionQCaStateMachine:



Public Member Functions

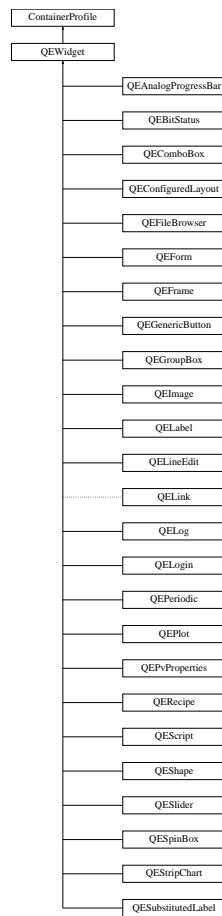
- **ConnectionQCaStateMachine** (void *parent)
- bool **process** (int requestedState)

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

- /home/rhydera/epicsqt/framework/data/include/QCaStateMachine.h
- /home/rhydera/epicsqt/framework/data/src/QCaStateMachine.cpp

9.12 ContainerProfile Class Reference

Inheritance diagram for ContainerProfile:



Public Member Functions

- void **takeLocalCopy** ()
- void **setupProfile** (QObject *guiLaunchConsumerIn, QStringList pathListIn, QString parentPathIn, QString macroSubstitutionsIn)
- void **setupLocalProfile** (QObject *guiLaunchConsumerIn, QStringList pathListIn, QString parentPathIn, QString macroSubstitutionsIn)
- void **updateConsumers** (QObject *guiLaunchConsumerIn)
- QObject * **replaceGuiLaunchConsumer** (QObject *newGuiLaunchConsumerIn)

- void **addMacroSubstitutions** (QString macroSubstitutionsIn)
- void **removeMacroSubstitutions** ()
- QObject * **getGuiLaunchConsumer** ()
- QString **getPath** ()
- QStringList **getPathList** ()
- QString **getParentPath** ()
- void **setPublishedParentPath** (QString publishedParentPathIn)
- QString **getMacroSubstitutions** ()

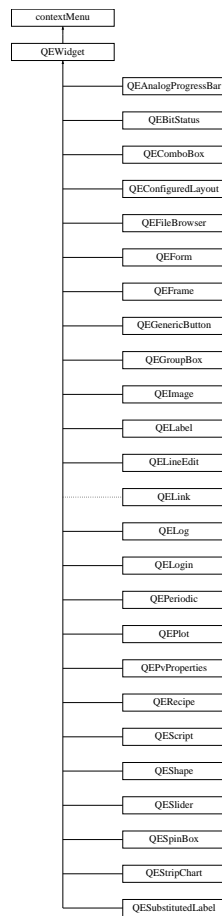
- bool **isProfileDefined** ()
- void **addContainedWidget** (QEWWidget *containedWidget)
- QEWWidget * **getNextContainedWidget** ()
- void **removeContainedWidget** (QEWWidget *containedWidget)
- unsigned int **getMessageFormId** ()
- unsigned int **getPublishedMessageFormId** ()
- void **setPublishedMessageFormId** (unsigned int publishedMessageFormIdIn)
- void **releaseProfile** ()
- void **publishOwnProfile** ()
- void **setUserLevel** (userLevels level)
- userLevels **getUserLevel** ()
- virtual void **userLevelChanged** (userLevels)

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

- /home/rhydera/epicsqt/framework/widgets/include/ContainerProfile.h
- /home/rhydera/epicsqt/framework/widgets/src/ContainerProfile.cpp

9.13 contextMenu Class Reference

Inheritance diagram for contextMenu:



Public Types

- enum **contextMenuOptions** {
CM_NONE, **CM_COPY_VARIABLE**, **CM_COPY_DATA**, **CM_PASTE**,
CM_DRAG_VARIABLE, **CM_DRAG_DATA**, **CM_SPECIFIC_WIDGETS_START_ - HERE** }

Public Member Functions

- void **addContextMenuToWidget** (QWidget *w)
- bool **isDraggingVariable** ()
- QMenu * **getContextMenu** ()
- virtual QString **copyVariable** ()
- virtual QVariant **copyData** ()
- virtual void **paste** (QVariant)

Friends

- class [contextMenuObject](#)

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

- /home/rhydera/epicsqt/framework/widgets/include/contextMenu.h
- /home/rhydera/epicsqt/framework/widgets/src/contextMenu.cpp

9.14 contextMenuObject Class Reference

Public Slots

- void **contextMenuTriggered** (QAction *selectedItem)
- void **showContextMenu** (const QPoint &pos)
- void **setChecked** ()

Public Member Functions

- void **addContextMenuToWidget** (QWidget *w)
- void **manageChecked** (bool draggingVariable)
- void **setMenu** ([contextMenu](#) *menuIn)
- bool **isDraggingVariable** ()

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

- /home/rhydera/epicsqt/framework/widgets/include/contextMenu.h
- /home/rhydera/epicsqt/framework/widgets/src/contextMenu.cpp

9.15 QEP periodic::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:

- /home/rhydera/epicsqt/framework/widgets/QEPeriodic/QEPeriodic.h

9.16 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:

- /home/rhydera/epicsqt/framework/widgets/QEImage/flipRotateMenu.h
- /home/rhydera/epicsqt/framework/widgets/QEImage/flipRotateMenu.cpp

9.17 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_ENABLE_CONTRAST_REVERSAL**, **ICM_-**
ENABLE_VERT, **ICM_ENABLE_HOZ**,
ICM_ENABLE_AREA, **ICM_ENABLE_LINE**, **ICM_ENABLE_TARGET**, **ICM_DISPLAY_-**
BUTTON_BAR,
ICM_ZOOM_SELECTED, **ICM_ZOOM_FIT**, **ICM_ZOOM_10**, **ICM_ZOOM_25**,
ICM_ZOOM_50, **ICM_ZOOM_75**, **ICM_ZOOM_100**, **ICM_ZOOM_150**,
ICM_ZOOM_200, **ICM_ZOOM_300**, **ICM_ZOOM_400**, **ICM_ROTATE_NONE**,
ICM_ROTATE_RIGHT, **ICM_ROTATE_LEFT**, **ICM_ROTATE_180**, **ICM_FLIP_HORIZONTAL**,
ICM_FLIP_VERTICAL, **ICM_SELECT_PAN**, **ICM_SELECT_HSLICE**, **ICM_SELECT_-**
VSLICE,
ICM_SELECT_AREA, **ICM_SELECT_PROFILE**, **ICM_SELECT_TARGET**, **ICM_-**
SELECT_BEAM,
ICM_CLEAR_MARKUP }

Public Member Functions

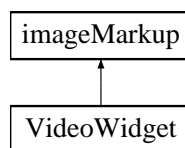
- **imageContextMenu** (QWidget *parent=0)
- void **getContextMenuOption** (const QPoint &, imageContextMenuOptions *option, bool *checked)
- void **addMenuItem** (const QString &title, const bool checkable, const bool checked, const imageContextMenuOptions option)
- void **addOptionMenuItem** (const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

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

- /home/rhydera/epicsqt/framework/widgets/QEImage/imageContextMenu.h
- /home/rhydera/epicsqt/framework/widgets/QEImage/imageContextMenu.cpp

9.18 imageMarkup Class Reference

Inheritance diagram for imageMarkup:



Public Types

- enum **markupIds** {
MARKUP_ID_REGION, **MARKUP_ID_H_SLICE**, **MARKUP_ID_V_SLICE**, **MARKUP_ID_LINE**,
MARKUP_ID_TARGET, **MARKUP_ID_BEAM**, **MARKUP_ID_TIMESTAMP**, **MARKUP_ID_COUNT**,
MARKUP_ID_NONE }

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)

- QCursor **getCircleCursor** ()
- QCursor **getTargetCursor** ()
- virtual void **markupSetCursor** (QCursor cursor)=0

Public Attributes

- QImage * **markupImage**
- QVector< [markupItem](#) * > **items**
- QPoint **grabOffset**
- bool **markupAreasStale**

Protected Member Functions

- bool **anyVisibleMarkups** ()
- QVector< QRect > & **getMarkupAreas** ()
- 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** (QSize newSize)
- virtual void **markupChange** (QImage &markups, QVector< QRect > &changedAreas)=0
- virtual void **markupAction** (markupIds mode, bool clearing, QPoint point1, QPoint point2)=0

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

- /home/rhydera/epicsqt/framework/widgets/QEImage/imageMarkup.h
- /home/rhydera/epicsqt/framework/widgets/QEImage/imageMarkup.cpp

9.19 localEnumerationItem Class Reference

Public Types

- enum **operations** {
LESS, LESS_EQUAL, EQUAL, NOT_EQUAL,
GREATER_EQUAL, GREATER, ALWAYS, UNKNOWN }

Public Attributes

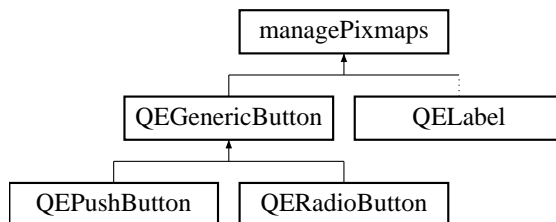
- double **dValue**
- QString **sValue**
- operations **op**
- QString **text**

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

- /home/rhydera/epicsqt/framework/data/include/QCStringFormatting.h

9.20 managePixmaps Class Reference

Inheritance diagram for managePixmaps:



Public Member Functions

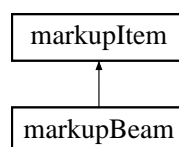
- void **setDataPixmap** (const QPixmap &Pixmap, const unsigned int index)
- QPixmap **getDataPixmap** (const unsigned int index)
- QPixmap **getDataPixmap** (const QString value)

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

- /home/rhydera/epicsqt/framework/widgets/include/managePixmaps.h
- /home/rhydera/epicsqt/framework/widgets/src/managePixmaps.cpp

9.21 markupBeam Class Reference

Inheritance diagram for markupBeam:



Public Member Functions

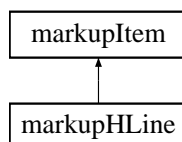
- **markupBeam** ([imageMarkup](#) *ownerIn, bool interactiveIn, bool reportOnMoveIn)
- void **startDrawing** (QPoint pos)
- void **setArea** ()
- void **drawMarkup** (QPainter &p)
- void **moveTo** (QPoint pos)
- bool **isOver** (QPoint point, QCursor *cursor)
- QPoint **origin** ()
- QPoint **getPoint1** ()
- QPoint **getPoint2** ()
- QCursor **defaultCursor** ()
- void **scaleSpecific** (double xScale, double yScale)

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

- /home/rhydera/epicsqt/framework/widgets/QEImage/imageMarkup.h
- /home/rhydera/epicsqt/framework/widgets/QEImage/imageMarkup.cpp

9.22 markupHLine Class Reference

Inheritance diagram for markupHLine:



Public Member Functions

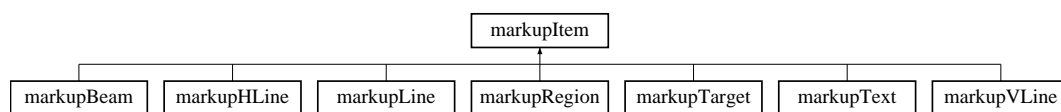
- **markupHLine** ([imageMarkup](#) *ownerIn, bool interactiveIn, bool reportOnMoveIn)
- void **startDrawing** (QPoint pos)
- void **setArea** ()
- void **drawMarkup** (QPainter &p)
- void **moveTo** (QPoint pos)
- bool **isOver** (QPoint point, QCursor *cursor)
- QPoint **origin** ()
- QPoint **getPoint1** ()
- QPoint **getPoint2** ()
- QCursor **defaultCursor** ()
- void **scaleSpecific** (double xScale, double yScale)

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

- /home/rhydera/epicsqt/framework/widgets/QEImage/imageMarkup.h
- /home/rhydera/epicsqt/framework/widgets/QEImage/imageMarkup.cpp

9.23 markupItem Class Reference

Inheritance diagram for markupItem:



Public Member Functions

- void **erase** ()
- void **drawMarkupIn** ()
- void **drawMarkupOut** ()
- void **setColor** (QColor colorIn)
- void **scale** (double xScale, double yScale)
- virtual QPoint **origin** ()=0
- virtual void **moveTo** (QPoint pos)=0
- virtual void **startDrawing** (QPoint pos)=0
- virtual bool **isOver** (const QPoint point, QCursor *cursor)=0
- virtual QPoint **getPoint1** ()=0
- virtual QPoint **getPoint2** ()=0
- virtual QCursor **defaultCursor** ()=0

Public Attributes

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

Protected Types

- enum **isOverOptions** { **OVER_LINE**, **OVER_BORDER**, **OVER_AREA** }

- enum **markupHandles** {
MARKUP_HANDLE_NONE, **MARKUP_HANDLE_START**, **MARKUP_HANDLE_END**, **MARKUP_HANDLE_TL**,
MARKUP_HANDLE_TR, **MARKUP_HANDLE_BL**, **MARKUP_HANDLE_BR**, **MARKUP_HANDLE_T**,
MARKUP_HANDLE_B, **MARKUP_HANDLE_L**, **MARKUP_HANDLE_R** }

Protected Member Functions

- **markupItem** ([imageMarkup](#) *ownerIn, isOverOptions over, bool interactiveIn, bool reportOnMoveIn)
- virtual void **setArea** ()=0
- virtual void **drawMarkup** (QPainter &p)=0
- bool **pointIsNear** (QPoint p1, QPoint p)
- QColor **getColor** ()

Protected Attributes

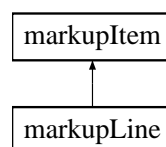
- markupHandles **activeHandle**
- isOverOptions **isOverType**
- bool **highlighted**
- int **highlightMargin**
- [imageMarkup](#) * **owner**

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

- /home/rhydera/epicsqt/framework/widgets/QEImage/imageMarkup.h
- /home/rhydera/epicsqt/framework/widgets/QEImage/imageMarkup.cpp

9.24 markupLine Class Reference

Inheritance diagram for markupLine:



Public Member Functions

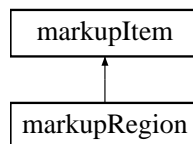
- **markupLine** ([imageMarkup](#) *ownerIn, bool interactiveIn, bool reportOnMoveIn)
- void **startDrawing** (QPoint pos)
- void **setArea** ()
- void **drawMarkup** (QPainter &p)
- void **moveTo** (QPoint pos)
- bool **isOver** (QPoint point, QCursor *cursor)
- QPoint **origin** ()
- QPoint **getPoint1** ()
- QPoint **getPoint2** ()
- QCursor **defaultCursor** ()
- void **scaleSpecific** (double xScale, double yScale)

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

- /home/rhydera/epicsqt/framework/widgets/QEImage/imageMarkup.h
- /home/rhydera/epicsqt/framework/widgets/QEImage/imageMarkup.cpp

9.25 markupRegion Class Reference

Inheritance diagram for markupRegion:



Public Member Functions

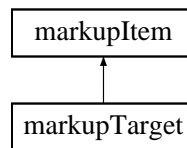
- **markupRegion** ([imageMarkup](#) *ownerIn, bool interactiveIn, bool reportOnMoveIn)
- void **startDrawing** (QPoint pos)
- void **setArea** ()
- void **drawMarkup** (QPainter &p)
- void **moveTo** (QPoint pos)
- bool **isOver** (QPoint point, QCursor *cursor)
- QPoint **origin** ()
- QPoint **getPoint1** ()
- QPoint **getPoint2** ()
- QCursor **defaultCursor** ()
- void **scaleSpecific** (double xScale, double yScale)

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

- /home/rhydera/epicsqt/framework/widgets/QEImage/imageMarkup.h
- /home/rhydera/epicsqt/framework/widgets/QEImage/imageMarkup.cpp

9.26 markupTarget Class Reference

Inheritance diagram for markupTarget:



Public Member Functions

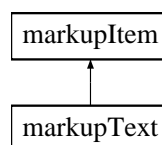
- **markupTarget** ([imageMarkup](#) *ownerIn, bool interactiveIn, bool reportOnMoveIn)
- void **startDrawing** (QPoint pos)
- void **setArea** ()
- void **drawMarkup** (QPainter &p)
- void **moveTo** (QPoint pos)
- bool **isOver** (QPoint point, QCursor *cursor)
- QPoint **origin** ()
- QPoint **getPoint1** ()
- QPoint **getPoint2** ()
- QCursor **defaultCursor** ()
- void **scaleSpecific** (double xScale, double yScale)

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

- /home/rhydera/epicsqt/framework/widgets/QEImage/imageMarkup.h
- /home/rhydera/epicsqt/framework/widgets/QEImage/imageMarkup.cpp

9.27 markupText Class Reference

Inheritance diagram for markupText:



Public Member Functions

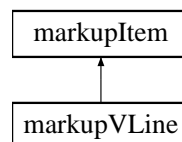
- **markupText** ([imageMarkup](#) *ownerIn, bool interactiveIn, bool reportOnMoveIn)
- void **setText** (QString textIn, bool draw)
- void **startDrawing** (QPoint pos)
- void **setArea** ()
- void **drawMarkup** (QPainter &p)
- void **moveTo** (QPoint pos)
- bool **isOver** (QPoint point, QCursor *cursor)
- QPoint **origin** ()
- QPoint **getPoint1** ()
- QPoint **getPoint2** ()
- QCursor **defaultCursor** ()
- void **scaleSpecific** (double xScale, double yScale)

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

- /home/rhydera/epicsqt/framework/widgets/QEImage/imageMarkup.h
- /home/rhydera/epicsqt/framework/widgets/QEImage/imageMarkup.cpp

9.28 markupVLine Class Reference

Inheritance diagram for markupVLine:



Public Member Functions

- **markupVLine** ([imageMarkup](#) *ownerIn, bool interactiveIn, bool reportOnMoveIn)
- void **startDrawing** (QPoint pos)
- void **setArea** ()
- void **drawMarkup** (QPainter &p)
- void **moveTo** (QPoint pos)
- bool **isOver** (QPoint point, QCursor *cursor)
- QPoint **origin** ()
- QPoint **getPoint1** ()
- QPoint **getPoint2** ()
- QCursor **defaultCursor** ()
- void **scaleSpecific** (double xScale, double yScale)

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

- /home/rhydera/epicsqt/framework/widgets/QEImage/imageMarkup.h
- /home/rhydera/epicsqt/framework/widgets/QEImage/imageMarkup.cpp

9.29 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:

- /home/rhydera/epicsqt/framework/widgets/QEPeriodic/PeriodicDialog.h
- /home/rhydera/epicsqt/framework/widgets/QEPeriodic/PeriodicDialog.cpp

9.30 PeriodicElementSetupForm Class Reference

Public Member Functions

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

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

- /home/rhydera/epicsqt/framework/widgets/QEPeriodic/PeriodicElementSetupForm.h
- /home/rhydera/epicsqt/framework/widgets/QEPeriodic/PeriodicElementSetupForm.cpp

9.31 PeriodicSetupDialog Class Reference

Public Member Functions

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

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

- /home/rhydera/epicsqt/framework/widgets/QEPeriodic/PeriodicSetupDialog.h
- /home/rhydera/epicsqt/framework/widgets/QEPeriodic/PeriodicSetupDialog.cpp

9.32 QEStripChart::PrivateData Class Reference

Public Member Functions

- **PrivateData** ([QEStripChart](#) *chartIn)
- [QEStripChartItem](#) * **getItem** (unsigned int slot)
- [QwtPlotCurve](#) * **allocateCurve** ()
- void **calcDisplayMinMax** ()
- void **plotData** ()
- void **setReadOut** (QString text)

Public Attributes

- enum ChartYScale **chartYScale**
- enum ChartTimeMode **chartTimeMode**

Protected Member Functions

- bool **eventFilter** (QObject *obj, QEvent *event)

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

- /home/rhydera/epicsqt/framework/widgets/QEStripChart/QEStripChart.cpp

9.33 QEStripChartItem::PrivateData Class Reference

Public Attributes

- [QEStripChart](#) * **chart**
- [QLabel](#) * **pvName**
- [QELabel](#) * **caLabel**

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

- /home/rhydera/epicsqt/framework/widgets/QEStripChart/QEStripChartItem.cpp

9.34 profilePlot Class Reference

Public Member Functions

- **profilePlot** (QWidget *parent=0)
- void **setProfile** (QVector< QPointF > &profile, double minX, double maxX, double minY, double maxY)

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

- /home/rhydera/epicsqt/framework/widgets/QEImage/profilePlot.h
- /home/rhydera/epicsqt/framework/widgets/QEImage/profilePlot.cpp

9.35 QPushButtonSpecifications Struct Reference

Public Attributes

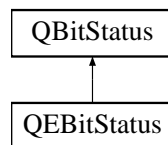
- int **width**
- const QString **caption**
- const QString **iconName**
- const QString **toolTip**
- const char * **member**

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

- /home/rhydera/epicsqt/framework/widgets/QEStripChart/QEStripChart.cpp

9.36 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** ()

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**

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

- /home/rhydera/epicsqt/framework/widgets/QEBitStatus/QEBitStatus.h
- /home/rhydera/epicsqt/framework/widgets/QEBitStatus/QEBitStatus.cpp

9.37 QCaAlarmInfo Class Reference

Public Member Functions

- **QCaAlarmInfo** (unsigned short statusIn, unsigned short severityIn)
- QString **statusName** ()
- QString **severityName** ()
- bool **isInAlarm** ()
- bool **isMinor** ()
- bool **isMajor** ()
- bool **isInvalid** ()
- QString **style** ()
- QString **getColorName** ()
- QCAALARMINFO_SEVERITY **getSeverity** ()

Static Public Member Functions

- static QCAALARMINFO_SEVERITY **getInvalidSeverity** ()

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

- /home/rhydera/epicsqt/framework/data/include/QCaAlarmInfo.h
- /home/rhydera/epicsqt/framework/data/src/QCaAlarmInfo.cpp

9.38 QCaConnectionInfo Class Reference

Public Member Functions

- **QCaConnectionInfo** (unsigned short channelStateIn, unsigned short linkStateIn)

- bool **isChannelConnected** ()
- bool **isLinkUp** ()

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

- /home/rhydera/epicsqt/framework/data/include/QCaConnectionInfo.h
- /home/rhydera/epicsqt/framework/data/src/QCaConnectionInfo.cpp

9.39 QCaDataPoint Struct Reference

Public Attributes

- double **value**
- [QCaDateTime](#) **datetime**
- [QCaAlarmInfo](#) **alarm**

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

- /home/rhydera/epicsqt/framework/data/include/QCaDataPoint.h

9.40 QCaDataPointList Class Reference

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

- /home/rhydera/epicsqt/framework/data/include/QCaDataPoint.h

9.41 QCaDateTime Class Reference

Public Member Functions

- **QCaDateTime** (QDateTime dt)
- void **operator=** (const [QCaDateTime](#) &other)
- **QCaDateTime** (unsigned long seconds, unsigned long nanoseconds)
- QString **text** ()
- double **floating** (QDateTime base)

Public Attributes

- unsigned long **nSec**

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

- /home/rhydera/epicsqt/framework/data/include/QCaDateTime.h
- /home/rhydera/epicsqt/framework/data/src/QCaDateTime.cpp

9.42 QCaEventFilter Class Reference

Public Member Functions

- void **addFilter** (QObject *objectIn)
- void **deleteFilter** (QObject *objectIn)
- bool **eventFilter** (QObject *watched, QEvent *e)

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

- /home/rhydera/epicsqt/framework/data/include/QCaEventFilter.h
- /home/rhydera/epicsqt/framework/data/src/QCaEventFilter.cpp

9.43 QCaEventItem Class Reference

Public Member Functions

- **QCaEventItem** ([QCaEventUpdate](#) *newEvent)

Public Attributes

- [QCaEventUpdate](#) * **event**

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

- /home/rhydera/epicsqt/framework/data/include/QCaEventUpdate.h

9.44 QCaEventUpdate Class Reference

Public Member Functions

- **QCaEventUpdate** ([qcaobject::QCaObject](#) *emitterObjectIn, long newReason, void *newDataPtr)

Public Attributes

- bool **acceptThisEvent**
- [qcaobject::QCaObject](#) * **emitterObject**
- long **reason**
- void * **dataPtr**

Static Public Attributes

- static QEvent::Type **EVENT_UPDATE_TYPE** = QEvent::User

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

- /home/rhydera/epicsqt/framework/data/include/QCaEventUpdate.h
- /home/rhydera/epicsqt/framework/data/src/QCaEventUpdate.cpp

9.45 QCalInstalledFiltersListItem Class Reference

Public Member Functions

- **QCalInstalledFiltersListItem** (QObject *eventObjectIn)

Public Attributes

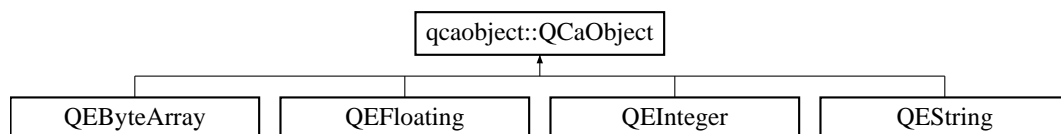
- QObject * **eventObject**
- long **referenceCount**

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

- /home/rhydera/epicsqt/framework/data/include/QCaEventFilter.h

9.46 qcaobject::QCaObject Class Reference

Inheritance diagram for qcaobject::QCaObject:



Public Slots

- bool **writeData** (const QVariant &value)
- void **resendLastData** ()

Signals

- void **dataChanged** (const QVariant &value, [QCaAlarmInfo](#) &alarmInfo, [QCaDateTime](#) &timeStamp)
- void **dataChanged** (const QByteArray &value, unsigned long dataSize, [QCaAlarmInfo](#) &alarmInfo, [QCaDateTime](#) &timeStamp)
- void **connectionChanged** ([QCaConnectionInfo](#) &connectionInfo)

Public Member Functions

- **QCaObject** (const QString &recordName, QObject *eventObject, unsigned char signalsToSendIn=SIG_VARIANT)
- **QCaObject** (const QString &recordName, QObject *eventObject, [UserMessage](#) *userMessageIn, unsigned char signalsToSendIn=SIG_VARIANT)
- bool **subscribe** ()
- bool **singleShotRead** ()
- bool **dataTypeKnown** ()
- bool **createChannel** ()
- void **deleteChannel** ()
- bool **createSubscription** ()
- bool **getChannel** ()
- bool **putChannel** ()
- bool **isChannelConnected** ()
- void **startConnectionTimer** ()
- void **stopConnectionTimer** ()
- void **setUserMessage** ([UserMessage](#) *userMessageIn)
- void **enableWriteCallbacks** (bool enable)
- bool **isWriteCallbacksEnabled** ()
- QString **getEgu** ()
- QStringList **getEnumerations** ()
- unsigned int **getPrecision** ()
- double **getDisplayLimitUpper** ()
- double **getDisplayLimitLower** ()
- double **getAlarmLimitUpper** ()
- double **getAlarmLimitLower** ()
- double **getWarningLimitUpper** ()
- double **getWarningLimitLower** ()
- double **getControlLimitUpper** ()
- double **getControlLimitLower** ()
- generic::generic_types **getDataType** ()

Static Public Member Functions

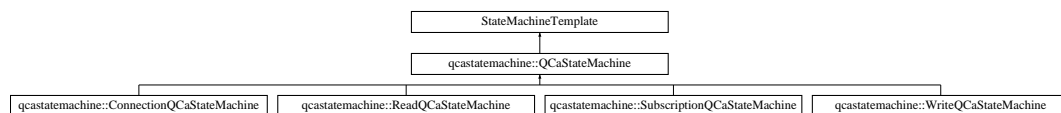
- static void **processEventStatic** (QCaEventUpdate *dataUpdateEvent)

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

- /home/rhydera/epicsqt/framework/data/include/QCaObject.h
- /home/rhydera/epicsqt/framework/data/src/QCaObject.cpp

9.47 qcastatemachine::QCaStateMachine Class Reference

Inheritance diagram for qcastatemachine::QCaStateMachine:



Public Member Functions

- **QCaStateMachine** (void *parent)
- virtual bool **process** (int requestedState)=0

Public Attributes

- QMutex **lock**
- bool **pending**
- bool **active**
- bool **expired**
- void * **myWorker**

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

- /home/rhydera/epicsqt/framework/data/include/QCaStateMachine.h
- /home/rhydera/epicsqt/framework/data/src/QCaStateMachine.cpp

9.48 QCaVariableNamePropertyManager Class Reference

Signals

- void **newVariableNameProperty** (QString variable, QString Substitutions, unsigned int variableIndex)

Public Member Functions

- QString **getVariableNameProperty** ()
- void **setVariableNameProperty** (QString variableNamePropertyIn)
- QString **getSubstitutionsProperty** ()
- void **setSubstitutionsProperty** (QString substitutionsPropertyIn)
- void **setVariableIndex** (unsigned int variableIndexIn)

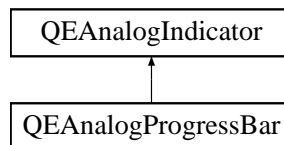
The documentation for this class was generated from the following files:

- /home/rhydera/epicsqt/framework/data/include/QCaVariableNamePropertyManager.h
- /home/rhydera/epicsqt/framework/data/src/QCaVariableNamePropertyManager.cpp

9.49 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](#) ()
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](#) ()
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](#) ()
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](#) ()
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](#) ()
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](#) ()
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](#) ()
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](#) ()
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](#) ()
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](#) ()
- 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](#) ()
- 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](#) ()
- 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](#) ()
- 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](#) ()
- 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](#) ()
- 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](#) ()
- 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](#) ()
- Access function for [logScale](#) - refer to [logScale](#) property for details.

Protected Member Functions

- virtual QString [getTextImage](#) ()
- virtual [BandList](#) [getBandList](#) ()

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](#)

9.49.1 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.

9.49.2 Member Enumeration Documentation

9.49.2.1 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)

9.49.2.2 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.

9.49.3 Property Documentation

9.49.3.1 QColor QEAnalogIndicator::backgroundColour [read, write]

Background colour

9.49.3.2 QColor QEAnalogIndicator::borderColour [read, write]

Border colour

9.49.3.3 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.

9.49.3.4 QColor QEAnalogIndicator::fontColour [read, write]

Font colour

9.49.3.5 QColor QEAnalogIndicator::foregroundColour [read, write]

Foreground colour

9.49.3.6 bool QEAnalogIndicator::logScale [read, write]

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

9.49.3.7 int QEAnalogIndicator::logScaleInterval [read, write]

Log scale interval.

9.49.3.8 double QEAnalogIndicator::majorInterval [read, write]

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

9.49.3.9 double QEAnalogIndicator::maximum [read, write]

Maximum indicated value.

9.49.3.10 double QEAnalogIndicator::minimum [read, write]

Minimum indicated value.

9.49.3.11 `double QEAnalogIndicator::minorInterval` [read, write]

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

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

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

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

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

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

If set, show the scale

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

If set, show textual representation of value on the indicator

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

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

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

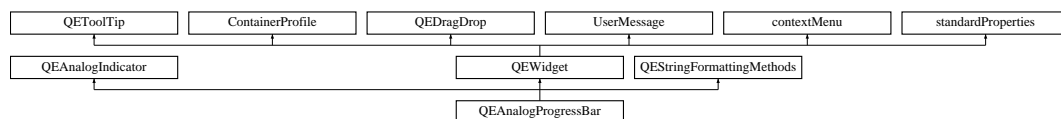
Current indicated value.

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

- /home/rhydera/epicsqt/framework/widgets/QEAnalogIndicator/QEAnalogIndicator.h
- /home/rhydera/epicsqt/framework/widgets/QEAnalogIndicator/QEAnalogIndicator.cpp

9.50 QEAnalogProgressBar Class Reference

Inheritance diagram for QEAnalogProgressBar:



Public Types

- enum [UserLevels](#) { [User](#) = USERLEVEL_USER, [Scientist](#) = USERLEVEL_SCIENTIST, [Engineer](#) = USERLEVEL_ENGINEER }
- enum [Formats](#) {
[Default](#) = QStringFormatting::FORMAT_DEFAULT, [Floating](#) = QStringFormatting::FORMAT_FLOATING, [Integer](#) = QStringFormatting::FORMAT_INTEGER, [UnsignedInteger](#) = QStringFormatting::FORMAT_UNSIGNEDINTEGER,
[Time](#) = QStringFormatting::FORMAT_TIME, [LocalEnumeration](#) = QStringFormatting::FORMAT_LOCAL_ENUMERATE }
- enum [Notations](#) { [Fixed](#) = QStringFormatting::NOTATION_FIXED, [Scientific](#) = QStringFormatting::NOTATION_SCIENTIFIC, [Automatic](#) = QStringFormatting::NOTATION_AUTOMATIC }
- enum [ArrayActions](#) { [Append](#) = QStringFormatting::APPEND, [Ascii](#) = QStringFormatting::ASCII, [Index](#) = QStringFormatting::INDEX }
- enum [AlarmSeverityDisplayModes](#) { [none](#), [foreground](#), [background](#) }

Public Slots

- void [requestEnabled](#) (const bool &state)

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

- bool [isEnabled](#) () const
Access function for [enabled](#) property - refer to [enabled](#) property for details.
- void [setEnabled](#) (bool state)
Access function for [enabled](#) property - refer to [enabled](#) property for details.
- [UserLevels](#) [getUserLevelVisibilityProperty](#) ()
Access function for [userLevel/Visibility](#) property - refer to [userLevel/Visibility](#) property for details.
- void [setUserLevelVisibilityProperty](#) ([UserLevels](#) level)
Access function for [userLevel/Visibility](#) property - refer to [userLevel/Visibility](#) property for details.
- [UserLevels](#) [getUserLevelEnabledProperty](#) ()
Access function for [userLevel/Enabled](#) property - refer to [userLevel/Enabled](#) property for details.
- void [setUserLevelEnabledProperty](#) ([UserLevels](#) level)

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

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

Protected Member Functions

- QString [getTextImage](#) ()
- [BandList](#) [getBandList](#) ()
- void [establishConnection](#) (unsigned int variableIndex)
- void [stringFormattingChange](#) ()
- void [dragEnterEvent](#) (QDragEnterEvent *event)
- void [dropEvent](#) (QDropEvent *event)
- void [mousePressEvent](#) (QMouseEvent *event)
- void [setDrop](#) (QVariant drop)
- QVariant [getDrop](#) ()
- QString [copyVariable](#) ()
- QVariant [copyData](#) ()

Protected Attributes

- [QEFloatingFormatting](#) **floatingFormatting**

Properties

- QString [variable](#)
- QString [variableSubstitutions](#)
- bool [variableAsToolTip](#)
- bool [enabled](#)
- bool [allowDrop](#)
- bool [visible](#)
- unsigned int
- QString [userLevelUserStyle](#)
- QString [userLevelScientistStyle](#)
- QString [userLevelEngineerStyle](#)
- [UserLevels](#) [userLevelVisibility](#)
- [UserLevels](#) [userLevelEnabled](#)
- int [precision](#)
- bool [useDbPrecision](#)
- bool [leadingZero](#)
- bool [trailingZeros](#)
- bool [addUnits](#)
- QString [localEnumeration](#)
- [Formats](#) [format](#)
- [Notations](#) [notation](#)
- [ArrayActions](#) [arrayAction](#)
- bool [useDbDisplayLimits](#)
- [AlarmSeverityDisplayModes](#) [alarmSeverityDisplayMode](#)

9.50.1 Member Enumeration Documentation

9.50.1.1 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.

9.50.1.2 enum QEAnalogProgressBar::Formats

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

Enumerator:

Default Format according to the EPICS database record 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.

9.50.1.3 enum QEAnalogProgressBar::Notations

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

Enumerator:

Fixed Refer to [QCStringFormatting::NOTATION_FIXED](#) for details.

Scientific Refer to [QCStringFormatting::NOTATION_SCIENTIFIC](#) for details.

Automatic Refer to [QCStringFormatting::NOTATION_AUTOMATIC](#) for details.

9.50.1.4 enum QEAnalogProgressBar::UserLevels

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

Enumerator:

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

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

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

9.50.2 Constructor & Destructor Documentation

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

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

9.50.2.2 `QEAAnalogProgressBar::QEAAnalogProgressBar (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.

9.50.3 Member Function Documentation

9.50.3.1 `void QEAAnalogProgressBar::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.

9.50.3.2 `void QEAAnalogProgressBar::requestEnabled (const bool & state) [inline, slot]`

Similar to standard `setEnabled` slot, but allows QE widget to determine if the widget remains disabled due to invalid data. If disabled due to invalid data, a request to enable the widget will be honoured when the data is no longer invalid.

9.50.3.3 `void QEAAnalogProgressBar::setVariableNameAndSubstitutions (QString variableNameIn, QString variableNameSubstitutionsIn, unsigned int variableIndex) [virtual]`

Virtual function that may be implimented by users of [QEWidget](#) to update variable names and macro substitutions. A default is provided that is suitable in most cases.

Reimplemented from [QEWidget](#).

9.50.4 Property Documentation

9.50.4.1 `bool QEAAnalogProgressBar::addUnits [read, write]`

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

9.50.4.2 `AlarmSeverityDisplayModes QEAAnalogProgressBar::alarmSeverityDisplayMode [read, write]`

Visualise the EPICS alarm severity

9.50.4.3 `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.

Reimplemented from [QEDragDrop](#).

9.50.4.4 `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'.

9.50.4.5 `bool QEAnalogProgressBar::enabled` [read, write]

Set the preferred 'enabled' state. Default is true. This property is copied to the standard Qt 'enabled' property if the data being displayed is valid. If the data being displayed is invalid the standard Qt 'enabled' property will always be set to false to indicate invalid data. The value of this property will only be copied to the standard Qt 'enabled' property once data is valid.

9.50.4.6 `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).

9.50.4.7 `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.

9.50.4.8 `bool QEAnalogProgressBar::leadingZero` [read, write]

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

9.50.4.9 `QString QEAnalogProgressBar::localEnumeration` [read, write]

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

9.50.4.10 `Notations QEAnalogProgressBar::notation` [read, write]

Notation used for numerical formatting. Default is fixed.

9.50.4.11 `int QEAnalogProgressBar::precision` [read, write]

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

9.50.4.12 `bool QEAnalogProgressBar::trailingZeros` [read, write]

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

9.50.4.13 `bool QEAnalogProgressBar::useDbDisplayLimits` [read, write]

Use the EPICS database display limits

9.50.4.14 `bool QEAnalogProgressBar::useDbPrecision` [read, write]

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

9.50.4.15 `UserLevels QEAnalogProgressBar::userLevelEnabled` [read, write]

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

9.50.4.16 `QString QEAnalogProgressBar::userLevelEngineerStyle` `[read, write]`

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

9.50.4.17 `QString QEAnalogProgressBar::userLevelScientistStyle` `[read, write]`

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

9.50.4.18 `QString QEAnalogProgressBar::userLevelUserStyle` `[read, write]`

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

9.50.4.19 `UserLevels QEAnalogProgressBar::userLevelVisibility` `[read, write]`

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

9.50.4.20 `QString QEAnalogProgressBar::variable` `[read, write]`

EPICS variable name (CA PV)

9.50.4.21 `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.

Reimplemented from [QEToolTip](#).

9.50.4.22 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.

9.50.4.23 bool QEAnalogProgressBar::visible [read, write]

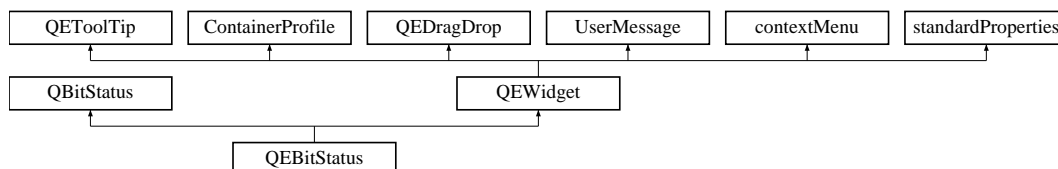
Display the widget. Default is true. Setting this property false is usefull if widget is only used to provide a signal - for example, when supplying data to a [QELink](#) widget. Note, when false the widget will still be visible in Qt Designer.

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

- /home/rhydera/epicsqt/framework/widgets/QEAnalogProgressBar/QEAnalogProgressBar.h
- /home/rhydera/epicsqt/framework/widgets/QEAnalogProgressBar/QEAnalogProgressBar.cpp

9.51 QEBitStatus Class Reference

Inheritance diagram for QEBitStatus:



Public Types

- enum [UserLevels](#) { [User](#) = USERLEVEL_USER, [Scientist](#) = USERLEVEL_SCIENTIST, [Engineer](#) = USERLEVEL_ENGINEER }

Public Slots

- void [requestEnabled](#) (const bool &state)

Signals

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

Public Member Functions

- bool [isEnabled](#) () const
Access function for [enabled](#) property - refer to [enabled](#) property for details.
- void [setEnabled](#) (bool state)
Access function for [enabled](#) property - refer to [enabled](#) property for details.
- [UserLevels](#) [getUserLevelVisibilityProperty](#) ()
Access function for [userLevelVisibility](#) property - refer to [userLevelVisibility](#) property for details.
- void [setUserLevelVisibilityProperty](#) ([UserLevels](#) level)
Access function for [userLevelVisibility](#) property - refer to [userLevelVisibility](#) property for details.
- [UserLevels](#) [getUserLevelEnabledProperty](#) ()
Access function for [userLevelEnabled](#) property - refer to [userLevelEnabled](#) property for details.
- void [setUserLevelEnabledProperty](#) ([UserLevels](#) level)
Access function for [userLevelEnabled](#) property - refer to [userLevelEnabled](#) property for details.
- **QEBitStatus** (QWidget *parent=0)
- **QEBitStatus** (const QString &variableName, QWidget *parent=0)
- void [setVariableNameAndSubstitutions](#) (QString variableNameIn, QString variableNameSubstitutionsIn, unsigned int variableIndex)

Protected Member Functions

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

Protected Attributes

- [QEIntegerFormatting](#) **integerFormatting**

Properties

- QString [variable](#)
- QString [variableSubstitutions](#)
- bool [variableAsToolTip](#)
- bool [enabled](#)
- bool [allowDrop](#)

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

9.51.1 Member Enumeration Documentation

9.51.1.1 enum [QEBitStatus::UserLevels](#)

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

Enumerator:

User Refer to `USERLEVEL_USER` for details.

Scientist Refer to `USERLEVEL_SCIENTIST` for details.

Engineer Refer to `USERLEVEL_ENGINEER` for details.

9.51.2 Member Function Documentation

9.51.2.1 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.

9.51.2.2 void [QEBitStatus::requestEnabled](#) (const bool & *state*) [[inline](#), [slot](#)]

Similar to standard `setEnabled` slot, but allows QE widget to determine if the widget remains disabled due to invalid data. If disabled due to invalid data, a request to enable the widget will be honoured when the data is no longer invalid.

9.51.2.3 void [QEBitStatus::setVariableNameAndSubstitutions](#) (QString *variableNameIn*, QString *variableNameSubstitutionsIn*, unsigned int *variableIndex*) [[virtual](#)]

Virtual function that may be implimented by users of [QEWidget](#) to update variable names and macro substitutions. A default is provided that is suitable in most cases.

Reimplemented from [QEWidget](#).

9.51.3 Property Documentation

9.51.3.1 `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.

Reimplemented from [QEDragDrop](#).

9.51.3.2 `bool QEBitStatus::enabled` [read, write]

Set the preferred 'enabled' state. Default is true. This property is copied to the standard Qt 'enabled' property if the data being displayed is valid. If the data being displayed is invalid the standard Qt 'enabled' property will always be set to false to indicate invalid data. The value of this property will only be copied to the standard Qt 'enabled' property once data is valid.

9.51.3.3 `unsigned QEBitStatus::int` [read, write]

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

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

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

9.51.3.5 `QString QEBitStatus::userLevelEngineerStyle` [read, write]

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

9.51.3.6 QString QEBitStatus::userLevelScientistStyle [read, write]

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

9.51.3.7 QString QEBitStatus::userLevelUserStyle [read, write]

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

9.51.3.8 UserLevels QEBitStatus::userLevelVisibility [read, write]

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

9.51.3.9 QString QEBitStatus::variable [read, write]

EPICS variable name (CA PV)

9.51.3.10 bool QEBitStatus::variableAsToolTip [read, write]

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

Reimplemented from [QEToolTip](#).

9.51.3.11 QString QEBitStatus::variableSubstitutions [read, write]

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

9.51.3.12 bool QEBitStatus::visible [read, write]

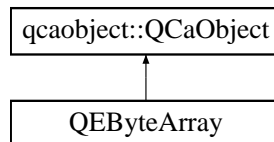
Display the widget. Default is true. Setting this property false is usefull if widget is only used to provide a signal - for example, when supplying data to a [QELink](#) widget. Note, when false the widget will still be visible in Qt Designer.

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

- /home/rhydera/epicsqt/framework/widgets/QEBitStatus/QEBitStatus.h
- /home/rhydera/epicsqt/framework/widgets/QEBitStatus/QEBitStatus.cpp

9.52 QByteArray Class Reference

Inheritance diagram for QByteArray:



Public Slots

- void **writeByteArray** (const QByteArray &data)

Signals

- void **byteArrayConnectionChanged** ([QCaConnectionInfo](#) &connectionInfo, const unsigned int &variableIndex)
- void **byteArrayChanged** (const QByteArray &value, unsigned long dataSize, [QCaAlarmInfo](#) &alarmInfo, [QCaDateTime](#) &timeStamp, const unsigned int &variableIndex)

Public Member Functions

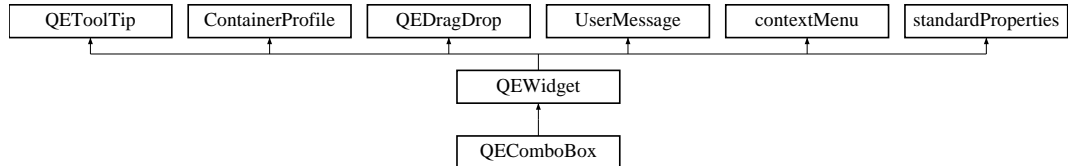
- **QByteArray** (QString recordName, QObject *eventObject, unsigned int variableIndexIn)
- **QByteArray** (QString recordName, QObject *eventObject, unsigned int variableIndexIn, [UserMessage](#) *userMessageIn)

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

- /home/rhydera/epicsqt/framework/data/include/QByteArray.h
- /home/rhydera/epicsqt/framework/data/src/QByteArray.cpp

9.53 QComboBox Class Reference

Inheritance diagram for QComboBox:



Public Types

- enum [UserLevels](#) { [User](#) = USERLEVEL_USER, [Scientist](#) = USERLEVEL_SCIENTIST, [Engineer](#) = USERLEVEL_ENGINEER }

Public Slots

- void [requestEnabled](#) (const bool &state)

Signals

- void [dbValueChanged](#) (const qlonglong &out)
- void [userChange](#) (const QString &oldValue, const QString &newValue, const QString &lastValue)

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

Public Member Functions

- QComboBox** (QWidget *parent=0)
- QComboBox** (const QString &variableName, QWidget *parent=0)
- void **setWriteOnChange** (bool writeOnChangeIn)
- bool **getWriteOnChange** ()
- void **setSubscribe** (bool subscribe)
- bool **getSubscribe** ()
- void **setUseDbEnumerations** (bool useDbEnumerations)
- bool **getUseDbEnumerations** ()
- bool [isEnabled](#) () const
- Access function for [enabled](#) property - refer to [enabled](#) property for details.*
- void [setEnabled](#) (bool state)
- Access function for [enabled](#) property - refer to [enabled](#) property for details.*
- [UserLevels](#) [getUserLevelVisibilityProperty](#) ()

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

- void [setUserLevelVisibilityProperty](#) (UserLevels level)

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

- UserLevels [getUserLevelEnabledProperty](#) ()

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

- void [setUserLevelEnabledProperty](#) (UserLevels level)

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

Protected Member Functions

- void **establishConnection** (unsigned int variableIndex)
- void **dragEnterEvent** (QDragEnterEvent *event)
- void **dropEvent** (QDropEvent *event)
- void **setDrop** (QVariant drop)
- QVariant **getDrop** ()

Protected Attributes

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

Properties

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

9.53.1 Member Enumeration Documentation

9.53.1.1 enum QComboBox::UserLevels

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

Enumerator:

User Refer to USERLEVEL_USER for details.

Scientist Refer to USERLEVEL_SCIENTIST for details.

Engineer Refer to USERLEVEL_ENGINEER for details.

9.53.2 Member Function Documentation

9.53.2.1 void QComboBox::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.

9.53.2.2 void QComboBox::requestEnabled (const bool & state) [inline, slot]

Similar to standard setEnabled slot, but allows QE widget to determine if the widget remains disabled due to invalid data. If disabled due to invalid data, a request to enable the widget will be honoured when the data is no longer invalid.

9.53.3 Member Data Documentation

9.53.3.1 bool QComboBox::writeOnChange [read, write, protected]

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

9.53.4 Property Documentation

9.53.4.1 bool QComboBox::allowDrop [read, write]

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

Reimplemented from [QEDragDrop](#).

9.53.4.2 `bool QComboBox::enabled` [read, write]

Set the preferred 'enabled' state. Default is true. This property is copied to the standard Qt 'enabled' property if the data being displayed is valid. If the data being displayed is invalid the standard Qt 'enabled' property will always be set to false to indicate invalid data. The value of this property will only be copied to the standard Qt 'enabled' property once data is valid.

9.53.4.3 `unsigned QComboBox::id` [read, write]

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

9.53.4.4 `bool QComboBox::subscribe` [read, write]

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

Reimplemented from [QEWidget](#).

9.53.4.5 `UserLevels QComboBox::userLevelEnabled` [read, write]

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

9.53.4.6 `QString QComboBox::userLevelEngineerStyle` [read, write]

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

9.53.4.7 `QString QComboBox::userLevelScientistStyle` [read, write]

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

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

9.53.4.8 `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.

9.53.4.9 `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 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'.

9.53.4.10 `QString QEComboBox::variable` [read, write]

EPICS variable name (CA PV)

9.53.4.11 `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.

Reimplemented from `QEToolTip`.

9.53.4.12 `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.

9.53.4.13 `bool QEComboBox::visible` [read, write]

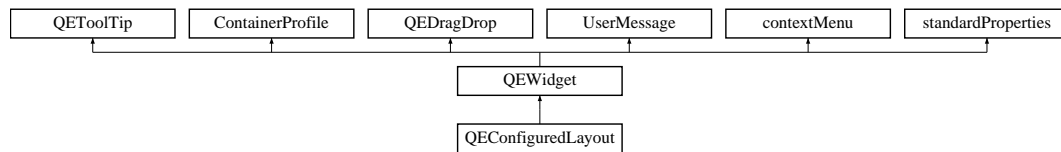
Display the widget. Default is true. Setting this property false is usefull if widget is only used to provide a signal - for example, when supplying data to a `QELink` widget. Note, when false the widget will still be visible in Qt Designer.

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

- /home/rhydera/epicsqt/framework/widgets/QEComboBox/QEComboBox.h
- /home/rhydera/epicsqt/framework/widgets/QEComboBox/QEComboBox.cpp

9.54 QEConfiguredLayout Class Reference

Inheritance diagram for QEConfiguredLayout:



Public Types

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

Public Member Functions

- **QEConfiguredLayout** (QWidget *pParent=0, bool pSubscription=true)
- void **setItemDescription** (QString pValue)
- QString **getItemDescription** ()
- void **setShowItemList** (bool pValue)
- bool **getShowItemList** ()
- void **setConfigurationType** (int pValue)
- int **getConfigurationType** ()
- void **setConfigurationFile** (QString pValue)
- QString **getConfigurationFile** ()
- void **setConfigurationText** (QString pValue)
- QString **getConfigurationText** ()
- void **setDetailsLayout** (int pValue)
- int **getDetailsLayout** ()
- void **setCurrentUserType** (int pValue)
- int **getCurrentUserType** ()
- void **refreshFields** ()
- void **userLevelChanged** (userLevels pValue)
- void **setConfigurationTypeProperty** (configurationTypesProperty pConfigurationType)

- configurationTypesProperty **getConfigurationTypeProperty** ()
- void **setDetailsLayoutProperty** (detailsLayoutProperty pDetailsLayout)
- detailsLayoutProperty **getDetailsLayoutProperty** ()
- void **setCurrentUserTypeProperty** (userTypesProperty pUserType)
- userTypesProperty **getCurrentUserTypeProperty** ()

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 **detailsLayout**
- int **currentUserType**
- bool **subscription**

Properties

- QString **itemDescription**
- bool **showItemList**
- configurationTypesProperty **configurationType**
- detailsLayoutProperty **detailsLayout**
- userTypesProperty **currentUserType**

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

- /home/rhydera/epicsqt/framework/widgets/QEConfiguredLayout/QEConfiguredLayout.h
- /home/rhydera/epicsqt/framework/widgets/QEConfiguredLayout/QEConfiguredLayout.cpp

9.55 QEConfiguredLayoutManager Class Reference

Public Member Functions

- **QEConfiguredLayoutManager** (QObject *pParent=0)
- bool **isContainer** () const
- bool **isInitialized** () const

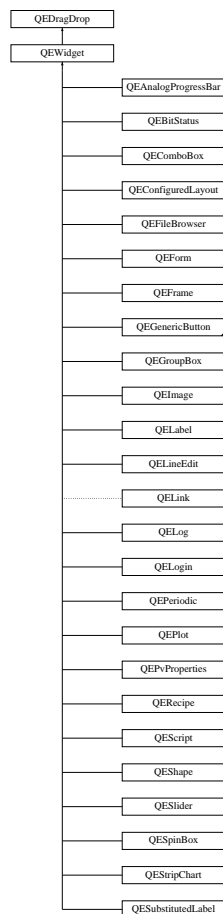
- QIcon **icon** () const
- QString **group** () const
- QString **includeFile** () const
- QString **name** () const
- QString **toolTip** () const
- QString **whatsThis** () const
- QWidget * **createWidget** (QWidget *pParent)
- void **initialize** (QDesignerFormEditorInterface *pCore)

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

- /home/rhydera/epicsqt/framework/widgets/QEConfiguredLayout/QEConfiguredLayoutManager.h
- /home/rhydera/epicsqt/framework/widgets/QEConfiguredLayout/QEConfiguredLayoutManager.cpp

9.56 QEDragDrop Class Reference

Inheritance diagram for QEDragDrop:



Public Member Functions

- **QEDragDrop** (QWidget *ownerIn)

Protected Member Functions

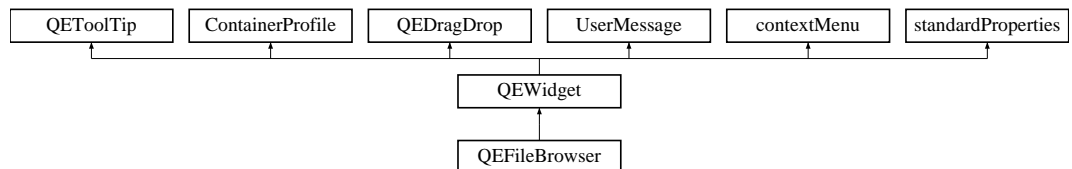
- void **qcaDragEnterEvent** (QDragEnterEvent *event)
- void **qcaDropEvent** (QDropEvent *event)
- void **qcaMousePressEvent** (QMouseEvent *event)
- virtual void **setDrop** (QVariant)
- virtual QVariant **getDrop** ()
- void **setAllowDrop** (bool allowDropIn)
- bool **getAllowDrop** ()

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

- /home/rhydera/epicsqt/framework/widgets/include/QEDragDrop.h
- /home/rhydera/epicsqt/framework/widgets/src/QEDragDrop.cpp

9.57 QEFileBrowser Class Reference

Inheritance diagram for QEFileBrowser:



Public Types

- enum **detailsLayoutProperty** { **Top** = TOP, **Bottom** = BOTTOM, **Left** = LEFT, **Right** = RIGHT }

Signals

- void **selected** (QString pFilename)

Public Member Functions

- **QEFileBrowser** (QWidget *pParent=0)
- 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 **setShowColumnTime** (bool pValue)
- bool **getShowColumnTime** ()
- void **setShowColumnSize** (bool pValue)
- bool **getShowColumnSize** ()
- void **setShowColumnFilename** (bool pValue)
- bool **getShowColumnFilename** ()
- void **setShowFileExtension** (bool pValue)
- bool **getShowFileExtension** ()
- void **setFileFilter** (QString pValue)
- QString **getFileFilter** ()
- void **setDetailsLayout** (int pValue)
- int **getDetailsLayout** ()
- void **updateTable** ()
- void **setDetailsLayoutProperty** (detailsLayoutProperty pDetailsLayout)
- detailsLayoutProperty **getDetailsLayoutProperty** ()

Protected Attributes

- QLineEdit * **qlineEditDirectoryPath**
- QPushButton * **qPushButtonDirectoryBrowser**
- QPushButton * **qPushButtonRefresh**
- [_QTableWidgetFileBrowser](#) * **qTableWidgetFileBrowser**
- QString **fileFilter**
- bool **showFileExtension**
- int **detailsLayout**

Properties

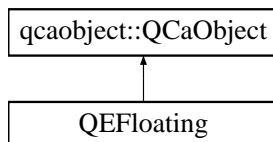
- QString **directoryPath**
- bool **showDirectoryPath**
- bool **showDirectoryBrowser**
- bool **showRefresh**
- bool **showColumnTime**
- bool **showColumnSize**
- bool **showColumnFilename**
- detailsLayoutProperty **detailsLayout**

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

- /home/rhydera/epicsqt/framework/widgets/QFileDialog/QFileDialog.h
- /home/rhydera/epicsqt/framework/widgets/QFileDialog/QFileDialog.cpp

9.58 QEFloating Class Reference

Inheritance diagram for QEFloating:



Public Slots

- void **writeFloating** (const double &data)

Signals

- void **floatingConnectionChanged** ([QCaConnectionInfo](#) &connectionInfo, const unsigned int &variableIndex)
- void **floatingChanged** (const double &value, [QCaAlarmInfo](#) &alarmInfo, [QCaDateTime](#) &timeStamp, const unsigned int &variableIndex)
- void **floatingArrayChanged** (const QVector< double > &values, [QCaAlarmInfo](#) &alarmInfo, [QCaDateTime](#) &timeStamp, const unsigned int &variableIndex)

Public Member Functions

- **QEFloating** (QString recordName, QObject *eventObject, [QEFloatingFormatting](#) *floatingFormattingIn, unsigned int variableIndexIn)
- **QEFloating** (QString recordName, QObject *eventObject, [QEFloatingFormatting](#) *floatingFormattingIn, unsigned int variableIndexIn, [UserMessage](#) *userMessageIn)

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

- /home/rhydera/epicsqt/framework/data/include/QEFloating.h
- /home/rhydera/epicsqt/framework/data/src/QEFloating.cpp

9.59 QEFloatingFormatting Class Reference

Public Types

- enum **formats** {
FORMAT_e = 'e', **FORMAT_E** = 'E', **FORMAT_f** = 'f', **FORMAT_g** = 'g',
FORMAT_G = 'G' }

Public Member Functions

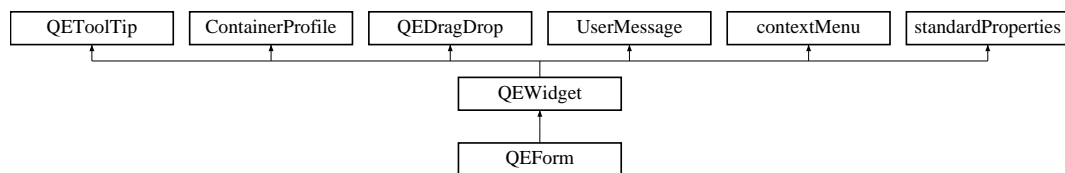
- double **formatFloating** (const QVariant &value)
- QVector< double > **formatFloatingArray** (const QVariant &value)
- QVariant **formatValue** (const double &floatingValue, generic::generic_types valueType)
- void **setPrecision** (unsigned int precision)
- void **setFormat** (formats format)
- unsigned int **getPrecision** ()
- int **getFormat** ()

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

- /home/rhydera/epicsqt/framework/data/include/QEFloatingFormatting.h
- /home/rhydera/epicsqt/framework/data/src/QEFloatingFormatting.cpp

9.60 QEForm Class Reference

Inheritance diagram for QEForm:



Public Types

- enum **creationOptions** { **CREATION_OPTION_OPEN**, **CREATION_OPTION_NEW_TAB**, **CREATION_OPTION_NEW_WINDOW** }
- enum **MessageFilterOptions** { **Match** = UserMessage::MESSAGE_FILTER_MATCH, **None** = UserMessage::MESSAGE_FILTER_NONE }

Public Slots

- bool **readUiFile** ()
- void **launchGui** (QString guiName, QEForm::creationOptions createOption)

Public Member Functions

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

- QString **getQEGuiTitle** ()
- QString **getFullFileName** ()
- void **setVariableNameAndSubstitutions** (QString variableNameIn, QString variableNameSubstitutionsIn, unsigned int variableIndex)
- void **setUiFileName** (QString uiFile)
- QString **getUiFileName** ()
- void **setHandleGuiLaunchRequests** (bool handleGuiLaunchRequests)
- bool **getHandleGuiLaunchRequests** ()
- void **setResizeContents** (bool resizeContentsIn)
- bool **getResizeContents** ()
- void **setVariableNameSubstitutionsProperty** (QString variableNameSubstitutions)
- QString **getVariableNameSubstitutionsProperty** ()
- MessageFilterOptions **getMessageFormFilter** ()
- void **setMessageFormFilter** (MessageFilterOptions messageFormFilter)
- MessageFilterOptions **getMessageSourceFilter** ()
- void **setMessageSourceFilter** (MessageFilterOptions messageSourceFilter)

Protected Member Functions

- void **setVariableNameSubstitutions** (QString variableNameSubstitutionsIn)

Protected Attributes

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

Properties

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

9.60.1 Member Function Documentation

9.60.1.1 void **QEForm::setVariableNameAndSubstitutions** (QString *variableNameIn*, QString *variableNameSubstitutionsIn*, unsigned int *variableIndex*) [virtual]

Virtual function that may be implimented by users of [QEWidget](#) to update variable names and macro substitutions. A default is provided that is suitable in most cases.

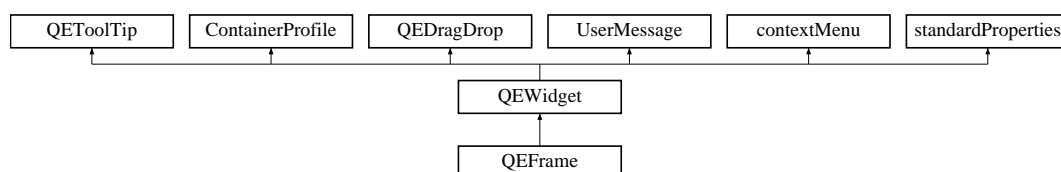
Reimplemented from [QEWidget](#).

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

- `/home/rhydera/epicsqt/framework/widgets/QEForm/QEForm.h`
- `/home/rhydera/epicsqt/framework/widgets/QEForm/QEForm.cpp`

9.61 QEFrame Class Reference

Inheritance diagram for QEFrame:



Public Types

- enum [UserLevels](#) { [User](#) = USERLEVEL_USER, [Scientist](#) = USERLEVEL_SCIENTIST, [Engineer](#) = USERLEVEL_ENGINEER }

Public Slots

- void [requestEnabled](#) (const bool &state)

Public Member Functions

- bool [isEnabled](#) () const
Access function for [enabled](#) property - refer to [enabled](#) property for details.
- void [setEnabled](#) (bool state)
Access function for [enabled](#) property - refer to [enabled](#) property for details.
- [UserLevels](#) [getUserLevelVisibilityProperty](#) ()
Access function for [userLevelVisibility](#) property - refer to [userLevelVisibility](#) property for details.
- void [setUserLevelVisibilityProperty](#) ([UserLevels](#) level)
Access function for [userLevelVisibility](#) property - refer to [userLevelVisibility](#) property for details.
- [UserLevels](#) [getUserLevelEnabledProperty](#) ()
Access function for [userLevelEnabled](#) property - refer to [userLevelEnabled](#) property for details.
- void [setUserLevelEnabledProperty](#) ([UserLevels](#) level)

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

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

Properties

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

9.61.1 Member Enumeration Documentation

9.61.1.1 enum QEFrame::UserLevels

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

Enumerator:

User Refer to USERLEVEL_USER for details.

Scientist Refer to USERLEVEL_SCIENTIST for details.

Engineer Refer to USERLEVEL_ENGINEER for details.

9.61.2 Member Function Documentation

9.61.2.1 void QEFrame::requestEnabled (const bool & state) [inline, slot]

Similar to standard setEnabled slot, but allows QE widget to determine if the widget remains disabled due to invalid data. If disabled due to invalid data, a request to enable the widget will be honoured when the data is no longer invalid.

9.61.3 Property Documentation

9.61.3.1 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.

Reimplemented from [QEDragDrop](#).

9.61.3.2 `bool QEFrame::enabled` [read, write]

Set the preferred 'enabled' state. Default is true. This property is copied to the standard Qt 'enabled' property if the data being displayed is valid. If the data being displayed is invalid the standard Qt 'enabled' property will always be set to false to indicate invalid data. The value of this property will only be copied to the standard Qt 'enabled' property once data is valid.

9.61.3.3 `unsigned QEFrame::int` [read, write]

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

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

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

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

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

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

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

9.61.3.7 QString QFrame::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.

9.61.3.8 UserLevels QFrame::userLevelVisibility [read, write]

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

9.61.3.9 bool QFrame::variableAsToolTip [read, write]

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

Reimplemented from [QEToolTip](#).

9.61.3.10 bool QFrame::visible [read, write]

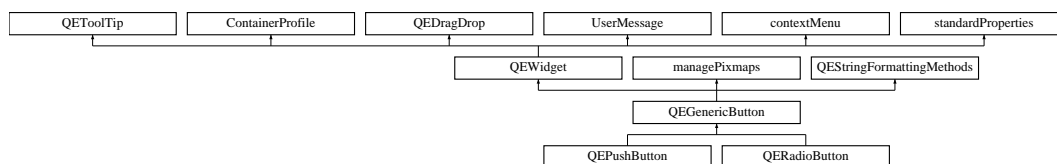
Display the widget. Default is true. Setting this property false is usefull if widget is only used to provide a signal - for example, when supplying data to a [QELink](#) widget. Note, when false the widget will still be visible in Qt Designer.

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

- /home/rhydera/epicsqt/framework/widgets/QFrame/QFrame.h
- /home/rhydera/epicsqt/framework/widgets/QFrame/QFrame.cpp

9.62 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 **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 **setGuiName** (QString guiName)
- QString **getGuiName** ()
- void **setCreationOption** (QForm::creationOptions creationOption)
- QForm::creationOptions **getCreationOption** ()
- void **setLabelTextProperty** (QString labelTextIn)
- QString **getLabelTextProperty** ()
- void **onGeneralMessage** (QString message)

Protected Member Functions

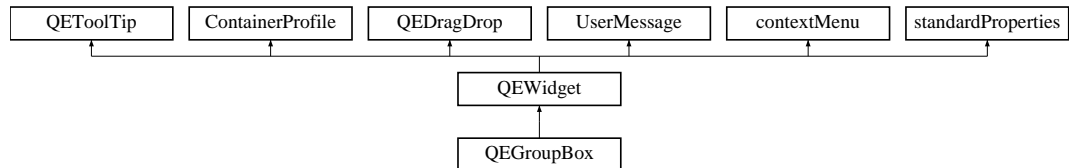
- void **connectionChanged** ([QCaConnectionInfo](#) &connectionInfo)
- void **setGenericButtonText** (const QString &text, [QCaAlarmInfo](#) &alarmInfo, [QCaDateTime](#) &, const unsigned int &variableIndex)
- void **userPressed** ()
- void **userReleased** ()
- void **userClicked** (bool checked)
- void **launchGui** (QString guiName, QForm::creationOptions creationOption)
- virtual updateOptions **getDefaultUpdateOption** ()=0
- void **setup** ()
- void **establishConnection** (unsigned int variableIndex)

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

- /home/rhydera/epicsqt/framework/widgets/QEButton/QEGenericButton.h
- /home/rhydera/epicsqt/framework/widgets/QEButton/QEGenericButton.cpp

9.63 QEGroupBox Class Reference

Inheritance diagram for QEGroupBox:



Public Types

- enum [UserLevels](#) { [User](#) = USERLEVEL_USER, [Scientist](#) = USERLEVEL_SCIENTIST, [Engineer](#) = USERLEVEL_ENGINEER }

Public Slots

- void [requestEnabled](#) (const bool &state)

Public Member Functions

- bool [isEnabled](#) () const
Access function for [enabled](#) property - refer to [enabled](#) property for details.
- void [setEnabled](#) (bool state)
Access function for [enabled](#) property - refer to [enabled](#) property for details.

- [UserLevels getUserLevelVisibilityProperty \(\)](#)
Access function for [userLevelVisibility](#) property - refer to [userLevelVisibility](#) property for details.
- void [setUserLevelVisibilityProperty \(UserLevels level\)](#)
Access function for [userLevelVisibility](#) property - refer to [userLevelVisibility](#) property for details.
- [UserLevels getUserLevelEnabledProperty \(\)](#)
Access function for [userLevelEnabled](#) property - refer to [userLevelEnabled](#) property for details.
- void [setUserLevelEnabledProperty \(UserLevels level\)](#)
Access function for [userLevelEnabled](#) property - refer to [userLevelEnabled](#) property for details.
- **QEGroupBox** (QWidget *parent=0)
- QSize **sizeHint** () const

Properties

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

9.63.1 Member Enumeration Documentation

9.63.1.1 enum QEGroupBox::UserLevels

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

Enumerator:

User Refer to USERLEVEL_USER for details.

Scientist Refer to USERLEVEL_SCIENTIST for details.

Engineer Refer to USERLEVEL_ENGINEER for details.

9.63.2 Member Function Documentation

9.63.2.1 void QEGroupBox::requestEnabled (const bool & *state*) [inline, slot]

Similar to standard setEnabled slot, but allows QE widget to determine if the widget remains disabled due to invalid data. If disabled due to invalid data, a request to enable the widget will be honoured when the data is no longer invalid.

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.

Reimplemented from [QEDragDrop](#).

9.63.3.2 bool QEGroupBox::enabled [read, write]

Set the preferred 'enabled' state. Default is true. This property is copied to the standard Qt 'enabled' property if the data being displayed is valid. If the data being displayed is invalid the standard Qt 'enabled' property will always be set to false to indicate invalid data. The value of this property will only be copied to the standard Qt 'enabled' property once data is valid.

9.63.3.3 unsigned QEGroupBox::int [read, write]

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

9.63.3.4 UserLevels QEGroupBox::userLevelEnabled [read, write]

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

9.63.3.5 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.

9.63.3.6 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.

9.63.3.7 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.

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

9.63.3.9 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.

Reimplemented from [QEToolTip](#).

9.63.3.10 bool QEGroupBox::visible [read, write]

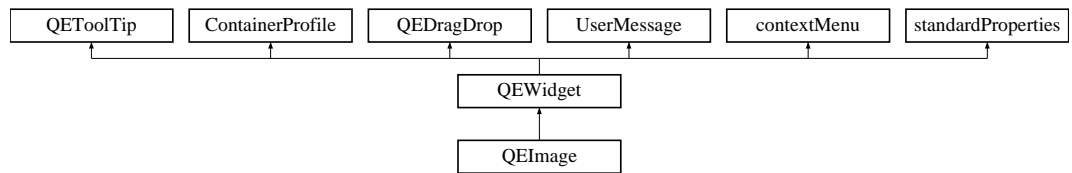
Display the widget. Default is true. Setting this property false is usefull if widget is only used to provide a signal - for example, when supplying data to a [QELink](#) widget. Note, when false the widget will still be visible in Qt Designer.

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

- /home/rhydera/epicsqt/framework/widgets/QEGroupBox/QEGroupBox.h
- /home/rhydera/epicsqt/framework/widgets/QEGroupBox/QEGroupBox.cpp

9.64 QEImage Class Reference

Inheritance diagram for QEImage:



Public Types

- enum **selectOptions** {
SO_NONE, **SO_PANNING**, **SO_VSLICE**, **SO_HSLICE**,
SO_AREA, **SO_PROFILE**, **SO_TARGET**, **SO_BEAM** }
- enum **formatOptions** { **GREY8**, **GREY12**, **GREY16**, **RGB_888** }
- enum **resizeOptions** { **RESIZE_OPTION_ZOOM**, **RESIZE_OPTION_FIT** }
- enum **rotationOptions** { **ROTATION_0**, **ROTATION_90_RIGHT**, **ROTATION_90_LEFT**, **ROTATION_180** }
- enum **UserLevels** { **User** = USERLEVEL_USER, **Scientist** = USERLEVEL_SCIENTIST, **Engineer** = USERLEVEL_ENGINEER }
- enum **FormatOptions** { **Grey_8** = QEImage::GREY8, **Grey_12** = QEImage::GREY12, **Grey_16** = QEImage::GREY16, **RGB** = QEImage::RGB_888 }
- enum **ResizeOptions** { **Zoom** = QEImage::RESIZE_OPTION_ZOOM, **Fit** = QEImage::RESIZE_OPTION_FIT }
- enum **RotationOptions** { **NoRotation** = QEImage::ROTATION_0, **Rotate90Right** = QEImage::ROTATION_90_RIGHT, **Rotate90Left** = QEImage::ROTATION_90_LEFT, **Rotate180** = QEImage::ROTATION_180 }

Public Slots

- void **setSelectPanMode** ()
- void **setSelectVSliceMode** ()
- void **setSelectHSliceMode** ()
- void **setSelectAreaMode** ()
- void **setSelectProfileMode** ()
- void **setSelectTargetMode** ()
- void **setSelectBeamMode** ()
- void **pauseClicked** ()
- void **saveClicked** ()

- void **roiClicked** ()
- void **resetRoiClicked** ()
- void **targetClicked** ()
- void **requestEnabled** (const bool &state)

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

- **QEImage** (QWidget *parent=0)
- **QEImage** (const QString &variableName, QWidget *parent=0)
- selectOptions **getSelectionOption** ()
- void **setFormatOption** (formatOptions formatOption)
- formatOptions **getFormatOption** ()
- void **setResizeOption** (resizeOptions resizeOptionIn)
- resizeOptions **getResizeOption** ()
- void **setZoom** (int zoomIn)
- int **getZoom** ()
- void **setRotation** (rotationOptions rotationIn)
- rotationOptions **getRotation** ()
- void **setHorizontalFlip** (bool flipHozIn)
- bool **getHorizontalFlip** ()
- void **setVerticalFlip** (bool flipVertIn)
- bool **getVerticalFlip** ()
- void **setInitialHozScrollPos** (int initialHosScrollPosIn)
- int **getInitialHozScrollPos** ()
- void **setInitialVertScrollPos** (int initialVertScrollPosIn)
- int **getInitialVertScrollPos** ()
- void **setDisplayAcquirePeriod** (bool displayAcquirePeriodIn)
- bool **getDisplayAcquirePeriod** ()
- void **setDisplayExposureTime** (bool displayExposureTimeIn)
- bool **getDisplayExposureTime** ()
- void **setDisplayButtonBar** (bool displayButtonBarIn)
- bool **getDisplayButtonBar** ()
- void **setShowTime** (bool pValue)
- bool **getShowTime** ()
- void **setVertSliceMarkupColor** (QColor pValue)
- QColor **getVertSliceMarkupColor** ()
- void **setHozSliceMarkupColor** (QColor pValue)
- QColor **getHozSliceMarkupColor** ()

- void **setProfileMarkupColor** (QColor pValue)
- QColor **getProfileMarkupColor** ()
- void **setAreaMarkupColor** (QColor pValue)
- QColor **getAreaMarkupColor** ()
- void **setTargetMarkupColor** (QColor pValue)
- QColor **getTargetMarkupColor** ()
- void **setBeamMarkupColor** (QColor pValue)
- QColor **getBeamMarkupColor** ()
- void **setTimeMarkupColor** (QColor pValue)
- QColor **getTimeMarkupColor** ()
- void **setDisplayCursorPixelInfo** (bool displayCursorPixelInfoIn)
- bool **getDisplayCursorPixelInfo** ()
- void **setContrastReversal** (bool contrastReversalIn)
- bool **getContrastReversal** ()
- void **setEnabledPan** (bool enablePanIn)
- bool **getEnabledPan** ()
- void **setEnabledVertSliceSelection** (bool enableVSliceSelectionIn)
- bool **getEnabledVertSliceSelection** ()
- void **setEnabledHozSliceSelection** (bool enableHSliceSelectionIn)
- bool **getEnabledHozSliceSelection** ()
- void **setEnabledAreaSelection** (bool enableAreaSelectionIn)
- bool **getEnabledAreaSelection** ()
- void **setEnabledProfileSelection** (bool enableProfileSelectionIn)
- bool **getEnabledProfileSelection** ()
- void **setEnabledTargetSelection** (bool enableTargetSelectionIn)
- bool **getEnabledTargetSelection** ()
- bool **isEnabled** () const
- *Access function for [enabled](#) property - refer to [enabled](#) property for details.*
- void **setEnabled** (bool state)
- *Access function for [enabled](#) property - refer to [enabled](#) property for details.*
- [UserLevels](#) **getUserLevelVisibilityProperty** ()
- *Access function for [userLevelVisibility](#) property - refer to [userLevelVisibility](#) property for details.*
- void **setUserLevelVisibilityProperty** ([UserLevels](#) level)
- *Access function for [userLevelVisibility](#) property - refer to [userLevelVisibility](#) property for details.*
- [UserLevels](#) **getUserLevelEnabledProperty** ()
- *Access function for [userLevelEnabled](#) property - refer to [userLevelEnabled](#) property for details.*
- void **setUserLevelEnabledProperty** ([UserLevels](#) level)
- *Access function for [userLevelEnabled](#) property - refer to [userLevelEnabled](#) property for details.*
- void **setFormatOptionProperty** (FormatOptions formatOption)
- FormatOptions **getFormatOptionProperty** ()
- void **setResizeOptionProperty** (ResizeOptions resizeOption)
- ResizeOptions **getResizeOptionProperty** ()
- void **setRotationProperty** (RotationOptions rotation)
- RotationOptions **getRotationProperty** ()

Protected Types

- enum **variableIndexes** {
 IMAGE_VARIABLE, **WIDTH_VARIABLE**, **HEIGHT_VARIABLE**, **ROI_X_VARIABLE**,
 ROI_Y_VARIABLE, **ROI_W_VARIABLE**, **ROI_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, **QEIMAGE_NUM_VARIABLES** }

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 v)
- void **resizeEvent** (QResizeEvent *)

Protected Attributes

- [QEIntegerFormatting](#) **integerFormatting**
- resizeModeOptions **resizeOption**
- int **zoom**
- rotationOptions **rotation**
- bool **flipVert**
- bool **flipHoz**
- int **initialHozScrollPos**
- int **initialVertScrollPos**
- bool **displayButtonBar**

Properties

- QString [imageVariable](#)
- QString [widthVariable](#)
- QString [heightVariable](#)
- QString [regionOfInterestXVariable](#)
- QString [regionOfInterestYVariable](#)
- QString [regionOfInterestWVariable](#)
- QString [regionOfInterestHVariable](#)
- QString [targetXVariable](#)

- QString [targetYVariable](#)
- QString [beamXVariable](#)
- QString [beamYVariable](#)
- QString [targetTriggerVariable](#)
- QString [clippingOnOffVariable](#)
- QString [clippingLowVariable](#)
- QString [clippingHighVariable](#)
- QString [variableSubstitutions](#)
- bool [variableAsToolTip](#)
- bool [enabled](#)
- bool [allowDrop](#)
- bool [visible](#)
- unsigned int
- QString [userLevelUserStyle](#)
- QString [userLevelScientistStyle](#)
- QString [userLevelEngineerStyle](#)
- [UserLevels](#) [userLevelVisibility](#)
- [UserLevels](#) [userLevelEnabled](#)
- FormatOptions **formatOption**
- bool **enableVertSliceSelection**
- bool **enableHozSliceSelection**
- bool **showTime**
- QColor **vertSliceColor**
- QColor **hozSliceColor**
- QColor **profileColor**
- QColor **areaColor**
- QColor **beamColor**
- QColor **targetColor**
- QColor **timeColor**
- ResizeOptions **resizeOption**
- RotationOptions **rotation**
- bool **verticalFlip**
- bool **horizontalFlip**
- int **initialHosScrollPos**

9.64.1 Member Enumeration Documentation

9.64.1.1 enum QImage::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.

9.64.2 Member Function Documentation

9.64.2.1 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.

9.64.2.2 void QEImage::requestEnabled (const bool & *state*) [inline, slot]

Similar to standard setEnabled slot, but allows QE widget to determine if the widget remains disabled due to invalid data. If disabled due to invalid data, a request to enable the widget will be honoured when the data is no longer invalid.

9.64.3 Property Documentation

9.64.3.1 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.

Reimplemented from [QEDragDrop](#).

9.64.3.2 QString QEImage::beamXVariable [read, write]

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

9.64.3.3 QString QEImage::beamYVariable [read, write]

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

9.64.3.4 QString QEImage::clippingHighVariable [read, write]

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

9.64.3.5 QString QEImage::clippingLowVariable [read, write]

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

9.64.3.6 QString QEImage::clippingOnOffVariable [read, write]

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

9.64.3.7 bool QEImage::enabled [read, write]

Set the preferred 'enabled' state. Default is true. This property is copied to the standard Qt 'enabled' property if the data being displayed is valid. If the data being displayed is invalid the standard Qt 'enabled' property will always be set to false to indicate invalid data. The value of this property will only be copied to the standard Qt 'enabled' property once data is valid.

9.64.3.8 QString QEImage::heightVariable [read, write]

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

9.64.3.9 QString QEImage::imageVariable [read, write]

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

9.64.3.10 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.

9.64.3.11 QString QEImage::regionOfInterestHVariable [read, write]

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

9.64.3.12 QString QEImage::regionOfInterestWVariable [read, write]

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

9.64.3.13 QString QEImage::regionOfInterestXVariable [read, write]

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

9.64.3.14 QString QImage::regionOfInterestYVariable [read, write]

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

9.64.3.15 QString QImage::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.

9.64.3.16 QString QImage::targetXVariable [read, write]

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

9.64.3.17 QString QImage::targetYVariable [read, write]

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

9.64.3.18 UserLevels QImage::userLevelEnabled [read, write]

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

9.64.3.19 QString QImage::userLevelEngineerStyle [read, write]

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

9.64.3.20 QString QImage::userLevelScientistStyle [read, write]

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

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

9.64.3.21 `QString QImage::userLevelUserStyle` [read, write]

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

9.64.3.22 `UserLevels QImage::userLevelVisibility` [read, write]

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

9.64.3.23 `bool QImage::variableAsToolTip` [read, write]

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

Reimplemented from [QEToolTip](#).

9.64.3.24 `QString QImage::variableSubstitutions` [read, write]

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

9.64.3.25 `bool QImage::visible` [read, write]

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

9.64.3.26 `QString QImage::widthVariable` [read, write]

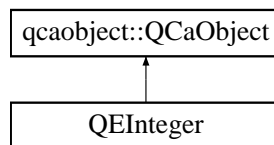
EPICS variable name (CA PV). This variable is used to read the width of the image.

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

- /home/rhydera/epicsqt/framework/widgets/QEImage/QEImage.h
- /home/rhydera/epicsqt/framework/widgets/QEImage/QEImage.cpp

9.65 QEInteger Class Reference

Inheritance diagram for QEInteger:



Public Slots

- void **writelInteger** (const long &data)

Signals

- void **integerConnectionChanged** ([QCaConnectionInfo](#) &connectionInfo, const unsigned int &variableIndex)
- void **integerChanged** (const long &value, [QCaAlarmInfo](#) &alarmInfo, [QCaDateTime](#) &timeStamp, const unsigned int &variableIndex)
- void **integerArrayChanged** (const QVector< long > &values, [QCaAlarmInfo](#) &alarmInfo, [QCaDateTime](#) &timeStamp, const unsigned int &variableIndex)

Public Member Functions

- **QEInteger** (QString recordName, QObject *eventObject, [QEIntegerFormatting](#) *integerFormattingIn, unsigned int variableIndexIn)
- **QEInteger** (QString recordName, QObject *eventObject, [QEIntegerFormatting](#) *integerFormattingIn, unsigned int variableIndexIn, [UserMessage](#) *userMessageIn)

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

- /home/rhydera/epicsqt/framework/data/include/QEInteger.h
- /home/rhydera/epicsqt/framework/data/src/QEInteger.cpp

9.66 QEIntegerFormatting Class Reference

```
#include <QEIntegerFormatting.h>
```

Public Member Functions

- [QEIntegerFormatting](#) ()

Constructor.

- long [formatInteger](#) (const QVariant &value)
- QVector< long > [formatIntegerArray](#) (const QVariant &value)
- QVariant [formatValue](#) (const long &integerValue, generic::generic_types value-Type)
- void [setRadix](#) (unsigned int radix)

Set the radix used for all conversions. Default is 10.

- unsigned int [getPrecision](#) ()

Get the precision used for all conversions.

- unsigned int [getRadix](#) ()

Get the radix used for all conversions.

9.66.1 Detailed Description

This class holds formatting instructions and uses them to convert between an integer and a QVariant of any type. It is generally set up with it's formatting instructions and then passed to a [QEInteger](#) class that will sink and source integer data to widgets or other code. It is used to convert data to and from a QCaObject (which sources and sinks data in the form of a QVariant where the QVariant reflects the underlying variable data type) and the [QEInteger](#) class. An example of a requirement for integer data is a combo box which must determine an integer index to select a menu option.

9.66.2 Member Function Documentation

9.66.2.1 long QEIntegerFormatting::formatInteger (const QVariant & value)

Given a data value of any type, format it as an integer according to the formatting instructions held by the class. This is used to convert the QVariant value received from a QCaObject, which is still based on the data variable type, to an integer.

9.66.2.2 QVector< long > QEIntegerFormatting::formatIntegerArray (const QVariant & value)

Given a data value of any type, format it as an array of integers according to the formatting instructions held by the class. This is used to convert the QVariant value received from a QCaObject, which is still based on the data variable type, to an integer array. Typically used where the input QVariant value is an array of data values, but will work for any QVariant type.

9.66.2.3 QVariant QEIntegerFormatting::formatValue (const long & *integerValue*, generic::generic_types *valueType*)

Given an integer value, format it as a data value of the specified type, according to the formatting instructions held by the class. This is used when writing integer data to a QCaObject.

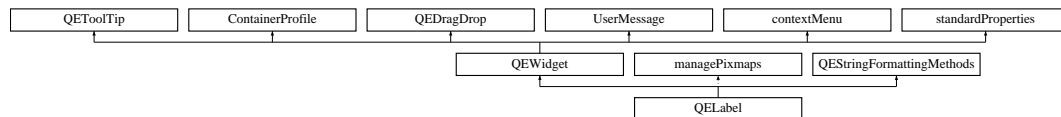
The documentation for this class was generated from the following files:

- /home/rhydera/epicsqt/framework/data/include/QEIntegerFormatting.h
- /home/rhydera/epicsqt/framework/data/src/QEIntegerFormatting.cpp

9.67 QELabel Class Reference

```
#include <QELabel.h>
```

Inheritance diagram for QELabel:



Public Types

- enum `updateOptions` { `UPDATE_TEXT`, `UPDATE_PIXMAP` }
- enum `UserLevels` { `User` = `USERLEVEL_USER`, `Scientist` = `USERLEVEL_SCIENTIST`, `Engineer` = `USERLEVEL_ENGINEER` }
- enum `Formats` {
`Default` = `QEStringFormatting::FORMAT_DEFAULT`, `Floating` = `QEStringFormatting::FORMAT_FLOATING`, `Integer` = `QEStringFormatting::FORMAT_INTEGER`, `UnsignedInteger` = `QEStringFormatting::FORMAT_UNSIGNEDINTEGER`,
`Time` = `QEStringFormatting::FORMAT_TIME`, `LocalEnumeration` = `QEStringFormatting::FORMAT_LOCAL_ENUMERATE` }
- enum `Notations` { `Fixed` = `QEStringFormatting::NOTATION_FIXED`, `Scientific` = `QEStringFormatting::NOTATION_SCIENTIFIC`, `Automatic` = `QEStringFormatting::NOTATION_AUTOMATIC` }
- enum `ArrayActions` { `Append` = `QEStringFormatting::APPEND`, `Ascii` = `QEStringFormatting::ASCII`, `Index` = `QEStringFormatting::INDEX` }
- enum `UpdateOptions` { `Text` = `QELabel::UPDATE_TEXT`, `Picture` = `QELabel::UPDATE_PIXMAP` }

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

Public Slots

- void [requestEnabled](#) (const bool &state)

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

- [QELLabel](#) (QWidget *parent=0)
- [QELLabel](#) (const QString &variableName, QWidget *parent=0)
- bool [isEnabled](#) () const
Access function for [enabled](#) property - refer to [enabled](#) property for details.
- void [setEnabled](#) (bool state)
Access function for [enabled](#) property - refer to [enabled](#) property for details.
- [UserLevels](#) [getUserLevelVisibilityProperty](#) ()
Access function for [userLevelVisibility](#) property - refer to [userLevelVisibility](#) property for details.
- void [setUserLevelVisibilityProperty](#) ([UserLevels](#) level)
Access function for [userLevelVisibility](#) property - refer to [userLevelVisibility](#) property for details.
- [UserLevels](#) [getUserLevelEnabledProperty](#) ()
Access function for [userLevelEnabled](#) property - refer to [userLevelEnabled](#) property for details.
- void [setUserLevelEnabledProperty](#) ([UserLevels](#) level)
Access function for [userLevelEnabled](#) property - refer to [userLevelEnabled](#) property for details.
- 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)

Access function for `pixmap0` property - refer to `pixmap0` property for details.

- void [setPixmap1Property](#) (QPixmap pixmap)

Access function for `pixmap1` property - refer to `pixmap1` property for details.

- void [setPixmap2Property](#) (QPixmap pixmap)

Access function for `pixmap2` property - refer to `pixmap2` property for details.

- void [setPixmap3Property](#) (QPixmap pixmap)

Access function for `pixmap3` property - refer to `pixmap3` property for details.

- void [setPixmap4Property](#) (QPixmap pixmap)

Access function for `pixmap4` property - refer to `pixmap4` property for details.

- void [setPixmap5Property](#) (QPixmap pixmap)

Access function for `pixmap5` property - refer to `pixmap5` property for details.

- void [setPixmap6Property](#) (QPixmap pixmap)

Access function for `pixmap6` property - refer to `pixmap6` property for details.

- void [setPixmap7Property](#) (QPixmap pixmap)

Access function for `pixmap7` property - refer to `pixmap7` property for details.

- QPixmap [getPixmap0Property](#) ()

Access function for `pixmap0` property - refer to `pixmap0` property for details.

- QPixmap [getPixmap1Property](#) ()

Access function for `pixmap1` property - refer to `pixmap1` property for details.

- QPixmap [getPixmap2Property](#) ()

Access function for `pixmap2` property - refer to `pixmap2` property for details.

- QPixmap [getPixmap3Property](#) ()

Access function for `pixmap3` property - refer to `pixmap3` property for details.

- QPixmap [getPixmap4Property](#) ()

Access function for `pixmap4` property - refer to `pixmap4` property for details.

- QPixmap [getPixmap5Property](#) ()

Access function for `pixmap5` property - refer to `pixmap5` property for details.

- QPixmap [getPixmap6Property](#) ()

Access function for `pixmap6` property - refer to `pixmap6` property for details.

- QPixmap [getPixmap7Property](#) ()

Access function for `pixmap7` property - refer to `pixmap7` property for details.

Properties

- QString [variable](#)
- QString [variableSubstitutions](#)
- bool [variableAsToolTip](#)
- bool [enabled](#)
- bool [allowDrop](#)
- bool [visible](#)
- unsigned int
- QString [userLevelUserStyle](#)
- QString [userLevelScientistStyle](#)
- QString [userLevelEngineerStyle](#)
- [UserLevels](#) [userLevelVisibility](#)
- [UserLevels](#) [userLevelEnabled](#)
- int [precision](#)
- bool [useDbPrecision](#)
- bool [leadingZero](#)
- bool [trailingZeros](#)
- bool [addUnits](#)
- QString [localEnumeration](#)
- [Formats](#) [format](#)
- [Notations](#) [notation](#)
- [ArrayActions](#) [arrayAction](#)
- [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 [QCStringFormatting::arrayActions](#) for details.

Enumerator:

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

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

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

9.67.2.2 enum QELabel::Formats

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

9.67.2.3 enum QELabel::Notations

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

Enumerator:

Fixed Refer to [QCStringFormatting::NOTATION_FIXED](#) for details.

Scientific Refer to [QCStringFormatting::NOTATION_SCIENTIFIC](#) for details.

Automatic Refer to [QCStringFormatting::NOTATION_AUTOMATIC](#) for details.

9.67.2.4 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.

9.67.2.5 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::requestEnabled (const bool & *state*) [inline, slot]

Similar to standard `setEnabled` slot, but allows QE widget to determine if the widget remains disabled due to invalid data. If disabled due to invalid data, a request to enable the widget will be honoured when the data is no longer invalid.

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.

Reimplemented from [QEDragDrop](#).

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 `bool QELabel::enabled` [read, write]

Set the preferred 'enabled' state. Default is true. This property is copied to the standard Qt 'enabled' property if the data being displayed is valid. If the data being displayed is invalid the standard Qt 'enabled' property will always be set to false to indicate invalid data. The value of this property will only be copied to the standard Qt 'enabled' property once data is valid.

9.67.5.5 `Formats QELabel::format` [read, write]

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

9.67.5.6 `unsigned QELabel::int` [read, write]

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

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

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

9.67.5.7 bool QELabel::leadingZero [read, write]

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

9.67.5.8 QString QELabel::localEnumeration [read, write]

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

9.67.5.9 Notations QELabel::notation [read, write]

Notation used for numerical formatting. Default is fixed.

9.67.5.10 QPixmap QELabel::pixmap0 [read, write]

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

9.67.5.11 QPixmap QELabel::pixmap1 [read, write]

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

9.67.5.12 QPixmap QELabel::pixmap2 [read, write]

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

9.67.5.13 QPixmap QELabel::pixmap3 [read, write]

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

9.67.5.14 QPixmap QELabel::pixmap4 [read, write]

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

9.67.5.15 QPixmap QELabel::pixmap5 [read, write]

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

9.67.5.16 QPixmap QELabel::pixmap6 [read, write]

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

9.67.5.17 QPixmap QELabel::pixmap7 [read, write]

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

9.67.5.18 int QELabel::precision [read, write]

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

9.67.5.19 bool QELabel::trailingZeros [read, write]

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

9.67.5.20 UpdateOptions QELabel::updateOption [read, write]

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

9.67.5.21 bool QELabel::useDbPrecision [read, write]

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

9.67.5.22 UserLevels QELabel::userLevelEnabled [read, write]

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

9.67.5.23 QString QELabel::userLevelEngineerStyle [read, write]

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

9.67.5.24 QString QELabel::userLevelScientistStyle [read, write]

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

9.67.5.25 QString QELabel::userLevelUserStyle [read, write]

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

9.67.5.26 UserLevels QELabel::userLevelVisibility [read, write]

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

9.67.5.27 QString QELabel::variable [read, write]

EPICS variable name (CA PV)

9.67.5.28 bool QELabel::variableAsToolTip [read, write]

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

Reimplemented from [QEToolTip](#).

9.67.5.29 QString QELabel::variableSubstitutions [read, write]

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

9.67.5.30 bool QELabel::visible [read, write]

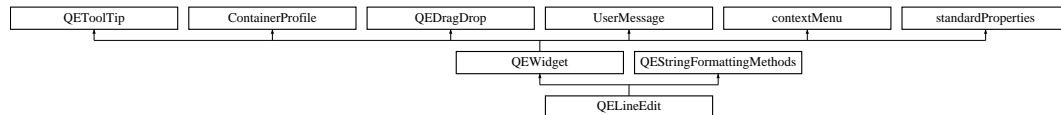
Display the widget. Default is true. Setting this property false is usefull if widget is only used to provide a signal - for example, when supplying data to a [QELink](#) widget. Note, when false the widget will still be visible in Qt Designer.

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

- /home/rhydera/epicsqt/framework/widgets/QELabel/QELabel.h
- /home/rhydera/epicsqt/framework/widgets/QELabel/QELabel.cpp

9.68 QELineEdit Class Reference

Inheritance diagram for QELineEdit:



Public Types

- enum [UserLevels](#) { [User](#) = USERLEVEL_USER, [Scientist](#) = USERLEVEL_SCIENTIST, [Engineer](#) = USERLEVEL_ENGINEER }
- enum [Formats](#) {
[Default](#) = QEStrngFormatting::FORMAT_DEFAULT, [Floating](#) = QEStrngFormatting::FORMAT_FLOATING, [Integer](#) = QEStrngFormatting::FORMAT_INTEGER, [UnsignedInteger](#) = QEStrngFormatting::FORMAT_UNSIGNEDINTEGER,
[Time](#) = QEStrngFormatting::FORMAT_TIME, [LocalEnumeration](#) = QEStrngFormatting::FORMAT_LOCAL_ENUMERATE }
- enum [Notations](#) { [Fixed](#) = QEStrngFormatting::NOTATION_FIXED, [Scientific](#) = QEStrngFormatting::NOTATION_SCIENTIFIC, [Automatic](#) = QEStrngFormatting::NOTATION_AUTOMATIC }
- enum [ArrayActions](#) { [Append](#) = QEStrngFormatting::APPEND, [Ascii](#) = QEStrngFormatting::ASCII, [Index](#) = QEStrngFormatting::INDEX }

Public Slots

- void [requestEnabled](#) (const bool &state)

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

- bool [isEnabled](#) () const
Access function for [enabled](#) property - refer to [enabled](#) property for details.
- void [setEnabled](#) (bool state)
Access function for [enabled](#) property - refer to [enabled](#) property for details.
- [UserLevels](#) [getUserLevelVisibilityProperty](#) ()
Access function for [userLevelVisibility](#) property - refer to [userLevelVisibility](#) property for details.
- void [setUserLevelVisibilityProperty](#) ([UserLevels](#) level)
Access function for [userLevelVisibility](#) property - refer to [userLevelVisibility](#) property for details.
- [UserLevels](#) [getUserLevelEnabledProperty](#) ()
Access function for [userLevelEnabled](#) property - refer to [userLevelEnabled](#) property for details.
- void [setUserLevelEnabledProperty](#) ([UserLevels](#) level)
Access function for [userLevelEnabled](#) property - refer to [userLevelEnabled](#) property for details.
- 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.

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

Properties

- QString [variable](#)
- QString [variableSubstitutions](#)
- bool [subscribe](#)
- bool [writeOnLoseFocus](#)
- bool [writeOnEnter](#)
- bool [writeOnFinish](#)
- bool [confirmWrite](#)
- bool [variableAsToolTip](#)
- bool [enabled](#)
- bool [allowDrop](#)
- bool [visible](#)
- unsigned int
- QString [userLevelUserStyle](#)
- QString [userLevelScientistStyle](#)
- QString [userLevelEngineerStyle](#)
- [UserLevels](#) [userLevelVisibility](#)
- [UserLevels](#) [userLevelEnabled](#)
- int [precision](#)
- bool [useDbPrecision](#)
- bool [leadingZero](#)
- bool [trailingZeros](#)
- bool [addUnits](#)
- QString [localEnumeration](#)
- [Formats](#) [format](#)
- [Notations](#) [notation](#)
- [ArrayActions](#) [arrayAction](#)

9.68.1 Member Enumeration Documentation

9.68.1.1 enum `QLineEdit::ArrayActions`

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

Enumerator:

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

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

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

9.68.1.2 enum `QLineEdit::Formats`

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

Enumerator:

Default Refer to [QStringFormatting::FORMAT_DEFAULT](#) for details.

Floating Refer to [QStringFormatting::FORMAT_FLOATING](#) for details.

Integer Refer to [QStringFormatting::FORMAT_INTEGER](#) for details.

UnsignedInteger Refer to [QStringFormatting::FORMAT_UNSIGNEDINTEGER](#) for details.

Time Refer to [QStringFormatting::FORMAT_TIME](#) for details.

LocalEnumeration Refer to [QStringFormatting::FORMAT_LOCAL_ENUMERATE](#) for details (and the [localEnumeration](#) property)

9.68.1.3 enum `QLineEdit::Notations`

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

Enumerator:

Fixed Refer to [QStringFormatting::NOTATION_FIXED](#) for details.

Scientific Refer to [QStringFormatting::NOTATION_SCIENTIFIC](#) for details.

Automatic Refer to [QStringFormatting::NOTATION_AUTOMATIC](#) for details.

9.68.1.4 enum `QLineEdit::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.

9.68.2 Constructor & Destructor Documentation**9.68.2.1 QLEdit::QLEdit (QWidget * *parent* = 0)**

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

9.68.2.2 QLEdit::QLEdit (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 QLEdit::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.

9.68.3.2 bool QLEdit::getConfirmWrite ()

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

9.68.3.3 bool QLEdit::getSubscribe ()

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

9.68.3.4 bool QLEdit::getWriteOnEnter ()

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

9.68.3.5 bool QLEdit::getWriteOnFinish ()

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

9.68.3.6 `bool QLEdit::getWriteOnLoseFocus ()`

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

9.68.3.7 `void QLEdit::requestEnabled (const bool & state) [inline, slot]`

Similar to standard `setEnabled` slot, but allows QE widget to determine if the widget remains disabled due to invalid data. If disabled due to invalid data, a request to enable the widget will be honoured when the data is no longer invalid.

9.68.3.8 `void QLEdit::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).

9.68.3.9 `void QLEdit::setSubscribe (bool subscribe)`

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

9.68.3.10 `void QLEdit::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').

9.68.3.11 `void QLEdit::setWriteOnFinish (bool writeOnFinish)`

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

9.68.3.12 `void QLEdit::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).

9.68.4 Property Documentation

9.68.4.1 `bool QLEdit::addUnits [read, write]`

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

9.68.4.2 `bool QLEdit::allowDrop` [read, write]

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

Reimplemented from [QEDragDrop](#).

9.68.4.3 `ArrayActions QLEdit::arrayAction` [read, write]

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

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

9.68.4.4 `bool QLEdit::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).

9.68.4.5 `bool QLEdit::enabled` [read, write]

Set the preferred 'enabled' state. Default is true. This property is copied to the standard Qt 'enabled' property if the data being displayed is valid. If the data being displayed is invalid the standard Qt 'enabled' property will always be set to false to indicate invalid data. The value of this property will only be copied to the standard Qt 'enabled' property once data is valid.

9.68.4.6 `Formats QLEdit::format` [read, write]

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

9.68.4.7 `unsigned QLEdit::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.

9.68.4.8 `bool QLEdit::leadingZero` [read, write]

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

9.68.4.9 `QString QLEdit::localEnumeration` [read, write]

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

9.68.4.10 `Notations QLEdit::notation` [read, write]

Notation used for numerical formatting. Default is fixed.

9.68.4.11 `int QLEdit::precision` [read, write]

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

9.68.4.12 `bool QLEdit::subscribe` [read, write]

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

Reimplemented from [QEWidget](#).

9.68.4.13 `bool QLEdit::trailingZeros` [read, write]

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

9.68.4.14 `bool QLEdit::useDbPrecision` [read, write]

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

9.68.4.15 UserLevels QLEdit::userLevelEnabled [read, write]

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

9.68.4.16 QString QLEdit::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.

9.68.4.17 QString QLEdit::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.

9.68.4.18 QString QLEdit::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.

9.68.4.19 UserLevels QLEdit::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'.

9.68.4.20 QString QLEdit::variable [read, write]

EPICS variable name (CA PV)

9.68.4.21 bool QLEdit::variableAsToolTip [read, write]

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

Reimplemented from [QEToolTip](#).

9.68.4.22 QString QLEdit::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.

9.68.4.23 bool QLEdit::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.

9.68.4.24 bool QLEdit::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').

9.68.4.25 bool QLEdit::writeOnFinish [read, write]

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

9.68.4.26 bool QLEdit::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:

- /home/rhydera/epicsqt/framework/widgets/QLEdit/QLEdit.h
- /home/rhydera/epicsqt/framework/widgets/QLEdit/QLEdit.cpp

9.69 QELineEditManager Class Reference

Public Member Functions

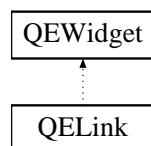
- **QELineEditManager** (QObject *parent=0)
- bool **isContainer** () const
- bool **isInitialized** () const
- QIcon **icon** () const
- QString **group** () const
- QString **includeFile** () const
- QString **name** () const
- QString **toolTip** () const
- QString **whatsThis** () const
- QWidget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

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

- /home/rhydera/epicsqt/framework/widgets/QELineEdit/QELineEditManager.h

9.70 QELink Class Reference

Inheritance diagram for QELink:



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 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 **setRunVisible** (bool visibleIn)
- bool **getRunVisible** ()
- void **setConditionProperty** (ConditionNames condition)
- ConditionNames **getConditionProperty** ()

Protected Attributes

- conditions **condition**
- QVariant **comparisonValue**
- bool **signalTrue**
- bool **signalFalse**
- QVariant **outTrueValue**
- QVariant **outFalseValue**
- bool **visible**

Properties

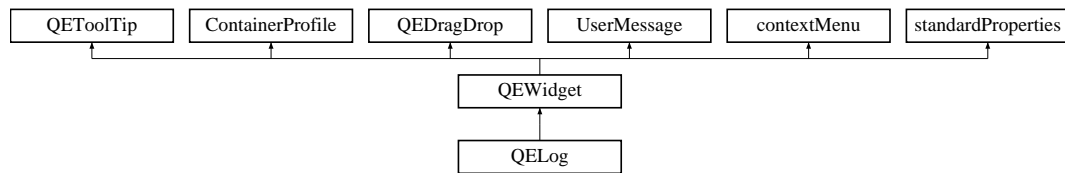
- ConditionNames **condition**
- QString **comparisonValue**
- QString **outTrueValue**
- QString **outFalseValue**
- bool **runVisible**

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

- /home/rhydera/epicsqt/framework/widgets/QELink/QELink.h
- /home/rhydera/epicsqt/framework/widgets/QELink/QELink.cpp

9.71 QELog Class Reference

Inheritance diagram for QELog:



Public Types

- enum **detailsLayoutProperty** { **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 }

Public Member Functions

- **QELog** (QWidget *pParent=0)
- void **setShowColumnTime** (bool pValue)
- bool **getShowColumnTime** ()
- void **setShowColumnType** (bool pValue)
- bool **getShowColumnType** ()
- void **setShowColumnMessage** (bool pValue)
- bool **getShowColumnMessage** ()
- void **setShowMessageFilter** (bool pValue)
- bool **getShowMessageFilter** ()
- void **setShowClear** (bool pValue)

- bool **getShowClear** ()
- void **setShowSave** (bool pValue)
- bool **getShowSave** ()
- void **setDetailsLayout** (int pValue)
- int **getDetailsLayout** ()
- 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 **setDetailsLayoutProperty** (detailsLayoutProperty pDetailsLayout)
- detailsLayoutProperty **getDetailsLayoutProperty** ()
- MessageFilterOptions **getMessageFormFilter** ()
- void **setMessageFormFilter** (MessageFilterOptions messageFormFilter)
- MessageFilterOptions **getMessageSourceFilter** ()
- void **setMessageSourceFilter** (MessageFilterOptions messageSourceFilter)

Protected Attributes

- [_QTableWidgetLog](#) * **qTableWidgetLog**
- QCheckBox * **qCheckBoxInfoMessage**
- QCheckBox * **qCheckBoxWarningMessage**
- QCheckBox * **qCheckBoxErrorMessage**
- QPushButton * **qPushButtonClear**
- QPushButton * **qPushButtonSave**
- QColor **qColorInfo**
- QColor **qColorWarning**
- QColor **qColorError**
- bool **scrollToBottom**
- int **detailsLayout**

Properties

- bool **showColumnTime**
- bool **showColumnType**
- bool **showColumnMessage**
- bool **showMessageFilter**
- bool **showClear**
- bool **showSave**

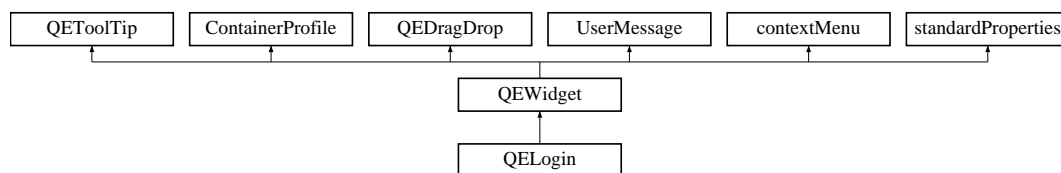
- detailsLayoutProperty **detailsLayout**
- QColor **infoColor**
- QColor **warningColor**
- QColor **errorColor**
- MessageFilterOptions **messageFormFilter**
- MessageFilterOptions **messageSourceFilter**
- unsigned **int**

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

- /home/rhydera/epicsqt/framework/widgets/QELog/QELog.h
- /home/rhydera/epicsqt/framework/widgets/QELog/QELog.cpp

9.72 QELogin Class Reference

Inheritance diagram for QELogin:



Public Types

- enum **userTypesProperty** { **User** = USERLEVEL_USER, **Scientist** = USERLEVEL_SCIENTIST, **Engineer** = USERLEVEL_ENGINEER }
- enum **detailsLayoutProperty** { **Top** = TOP, **Bottom** = BOTTOM, **Left** = LEFT, **Right** = RIGHT }

Public Member Functions

- **QELogin** (QWidget *pParent=0)
- void **setShowUserType** (bool pValue)
- bool **getShowUserType** ()
- void **setShowLogin** (bool pValue)
- bool **getShowButtonLogin** ()
- void **setShowLogout** (bool pValue)
- bool **getShowButtonLogout** ()
- void **setUserPassword** (QString pValue)
- QString **getUserPassword** ()
- void **setScientistPassword** (QString pValue)
- QString **getScientistPassword** ()

- void **setEngineerPassword** (QString pValue)
- QString **getEngineerPassword** ()
- void **setCurrentUserType** (int pValue)
- int **getCurrentUserType** ()
- void **setDetailsLayout** (int pValue)
- int **getDetailsLayout** ()
- QString **getUserTypeName** (userLevels type)
- void **logoutCurrentUserType** ()
- void **setCurrentUserTypeProperty** (userTypesProperty pUserType)
- userTypesProperty **getCurrentUserTypeProperty** ()
- void **setDetailsLayoutProperty** (detailsLayoutProperty pDetailsLayout)
- detailsLayoutProperty **getDetailsLayoutProperty** ()

Protected Attributes

- QStack< int > **loginHistory**
- QPushButton * **qPushButtonLogin**
- QPushButton * **qPushButtonLogout**
- QLabel * **qLabelUserType**
- QString **userPassword**
- QString **scientistPassword**
- QString **engineerPassword**
- int **currentUserType**
- int **detailsLayout**

Properties

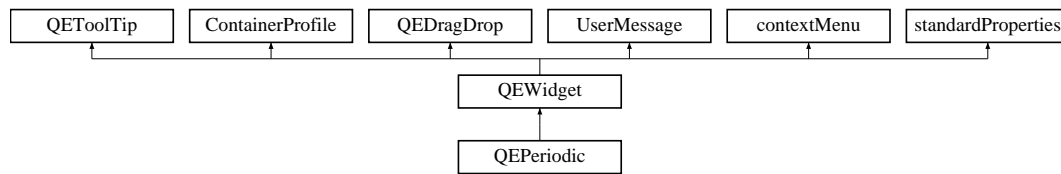
- bool **showUserType**
- bool **showLogin**
- bool **showLogout**
- userTypesProperty **currentUserType**
- detailsLayoutProperty **detailsLayout**

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

- /home/rhydera/epicsqt/framework/widgets/QELogin/QELogin.h
- /home/rhydera/epicsqt/framework/widgets/QELogin/QELogin.cpp

9.73 QEPeiodic Class Reference

Inheritance diagram for QEPeiodic:



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

Public Slots

- void [requestEnabled](#) (const bool &state)

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** ()
- bool **isEnabled** () const
Access function for [enabled](#) property - refer to [enabled](#) property for details.
- void **setEnabled** (bool state)
Access function for [enabled](#) property - refer to [enabled](#) property for details.
- **UserLevels** **getUserLevelVisibilityProperty** ()
Access function for [userLevelVisibility](#) property - refer to [userLevelVisibility](#) property for details.
- void **setUserLevelVisibilityProperty** (UserLevels level)
Access function for [userLevelVisibility](#) property - refer to [userLevelVisibility](#) property for details.
- **UserLevels** **getUserLevelEnabledProperty** ()
Access function for [userLevelEnabled](#) property - refer to [userLevelEnabled](#) property for details.
- void **setUserLevelEnabledProperty** (UserLevels level)
Access function for [userLevelEnabled](#) property - refer to [userLevelEnabled](#) property for details.
- 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 **setDrop** (QVariant drop)
- QVariant **getDrop** ()

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 [enabled](#)
- bool [visible](#)
- unsigned [int](#)
- QString [userLevelUserStyle](#)
- QString [userLevelScientistStyle](#)
- QString [userLevelEngineerStyle](#)
- [UserLevels](#) [userLevelVisibility](#)
- [UserLevels](#) [userLevelEnabled](#)
- PresentationOptions **presentationOption**
- VariableTypes **variableType1**
- VariableTypes **variableType2**
- QString **userInfo**

9.73.1 Member Enumeration Documentation

9.73.1.1 enum QEPeriodic::UserLevels

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

Enumerator:

User Refer to USERLEVEL_USER for details.

Scientist Refer to USERLEVEL_SCIENTIST for details.

Engineer Refer to USERLEVEL_ENGINEER for details.

9.73.2 Member Function Documentation

9.73.2.1 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.

9.73.2.2 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.

9.73.2.3 void QEPeriodic::requestEnabled (const bool & state) [inline, slot]

Similar to standard setEnabled slot, but allows QE widget to determine if the widget remains disabled due to invalid data. If disabled due to invalid data, a request to enable the widget will be honoured when the data is no longer invalid.

9.73.3 Member Data Documentation

9.73.3.1 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.

Reimplemented from [QEDragDrop](#).

9.73.4 Property Documentation

9.73.4.1 `bool QEPeiodic::enabled` [read, write]

Set the preferred 'enabled' state. Default is true. This property is copied to the standard Qt 'enabled' property if the data being displayed is valid. If the data being displayed is invalid the standard Qt 'enabled' property will always be set to false to indicate invalid data. The value of this property will only be copied to the standard Qt 'enabled' property once data is valid.

9.73.4.2 `unsigned QEPeiodic::int` [read, write]

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

9.73.4.3 `QString QEPeiodic::readbackLabelVariable1` [read, write]

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

9.73.4.4 `QString QEPeiodic::readbackLabelVariable2` [read, write]

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

9.73.4.5 `bool QEPeiodic::subscribe` [read, write]

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

Reimplemented from [QEWidget](#).

9.73.4.6 `UserLevels QEPeiodic::userLevelEnabled` [read, write]

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

9.73.4.7 `QString QEPeiodic::userLevelEngineerStyle` [read, write]

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

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

9.73.4.8 QString QEPeiodic::userLevelScientistStyle [read, write]

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

9.73.4.9 QString QEPeiodic::userLevelUserStyle [read, write]

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

9.73.4.10 UserLevels QEPeiodic::userLevelVisibility [read, write]

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

9.73.4.11 bool QEPeiodic::variableAsToolTip [read, write]

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

Reimplemented from [QEToolTip](#).

9.73.4.12 QString QEPeiodic::variableSubstitutions [read, write]

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

9.73.4.13 bool QEPeiodic::visible [read, write]

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

9.73.4.14 QString QEPeiodic::writeButtonVariable1 [read, write]

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

9.73.4.15 QString QEPeiodic::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:

- /home/rhydera/epicsqt/framework/widgets/QEPeiodic/QEPeiodic.h
- /home/rhydera/epicsqt/framework/widgets/QEPeiodic/QEPeiodic.cpp

9.74 QEPeiodicComponentData 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:

- /home/rhydera/epicsqt/framework/widgets/QEPeiodic/QEPeiodic.h

9.75 QEPeiodicTaskMenu Class Reference

Public Member Functions

- **QEPeiodicTaskMenu** ([QEPeiodic](#) *periodic, QObject *parent)
- QAction * **preferredEditAction** () const
- QList< QAction * > **taskActions** () const

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

- /home/rhydera/epicsqt/framework/widgets/QEPeriodic/QEPeriodicTaskMenu.h
- /home/rhydera/epicsqt/framework/widgets/QEPeriodic/QEPeriodicTaskMenuExtension.cpp

9.76 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:

- /home/rhydera/epicsqt/framework/widgets/QEPeriodic/QEPeriodicTaskMenu.h
- /home/rhydera/epicsqt/framework/widgets/QEPeriodic/QEPeriodicTaskMenuExtension.cpp

9.77 QEpicsPV Class Reference

Public Slots

- const QVariant & **set** (QVariant value, int delay=-1)
- void **setPV** (const QString &_pvName="")

Signals

- void **connectionChanged** (bool connected)
- void **connected** ()
- void **disconnected** ()
- void **valueChanged** (const QVariant &value)
- void **valueUpdated** (const QVariant &value)
- void **valueInitd** (const QVariant &value)

Public Member Functions

- **QEpicsPV** (const QString &_pvName, QObject *parent=0)
- **QEpicsPV** (QObject *parent=0)
- const QVariant & **get** () const

- void **needUpdated** () const
- const QVariant & **getUpdated** (int delay=defaultDelay) const
- bool **isConnected** () const
- const QStringList & **getEnum** () const
- const QString & **pv** () const
- const QVariant & **getReady** (int delay=defaultDelay) const

Static Public Member Functions

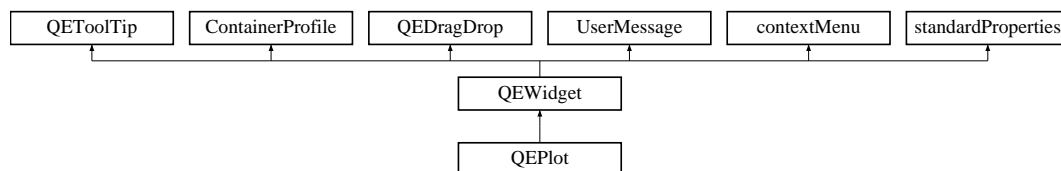
- static void **setDebugLevel** (unsigned level=0)
- static QVariant **get** (const QString &_pvName, int delay=defaultDelay)
- static QVariant **set** (QString &_pvName, const QVariant &value, int delay=-1)

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

- /home/rhydera/epicsqt/framework/data/include/qepicspv.h
- /home/rhydera/epicsqt/framework/data/src/qepicspv.cpp

9.78 QEPlot Class Reference

Inheritance diagram for QEPlot:



Public Types

- enum **UserLevels** { **User** = USERLEVEL_USER, **Scientist** = USERLEVEL_SCIENTIST, **Engineer** = USERLEVEL_ENGINEER }
- enum **TraceStyles** { **Lines** = QwtPlotCurve::Lines, **Sticks** = QwtPlotCurve::Sticks, **Steps** = QwtPlotCurve::Steps, **Dots** = QwtPlotCurve::Dots }

Public Slots

- void **requestEnabled** (const bool &state)

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)
- 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** ()
- bool **isEnabled** () const
Access function for [enabled](#) property - refer to [enabled](#) property for details.
- void **setEnabled** (bool state)
Access function for [enabled](#) property - refer to [enabled](#) property for details.
- [UserLevels](#) **getUserLevelVisibilityProperty** ()
Access function for [userLevelVisibility](#) property - refer to [userLevelVisibility](#) property for details.
- void **setUserLevelVisibilityProperty** ([UserLevels](#) level)
Access function for [userLevelVisibility](#) property - refer to [userLevelVisibility](#) property for details.
- [UserLevels](#) **getUserLevelEnabledProperty** ()
Access function for [userLevelEnabled](#) property - refer to [userLevelEnabled](#) property for details.
- void **setUserLevelEnabledProperty** ([UserLevels](#) level)
Access function for [userLevelEnabled](#) property - refer to [userLevelEnabled](#) property for details.
- 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** ()

Protected Attributes

- [QEFloatingFormatting](#) **floatingFormatting**
- bool **localEnabled**
- bool [allowDrop](#)

Properties

- QString [variable1](#)
- QString [variable2](#)
- QString [variable3](#)
- QString [variable4](#)
- QString [variableSubstitutions](#)
- bool [variableAsToolTip](#)
- bool [enabled](#)
- bool [visible](#)
- unsigned int
- QString [userLevelUserStyle](#)
- QString [userLevelScientistStyle](#)
- QString [userLevelEngineerStyle](#)
- [UserLevels](#) [userLevelVisibility](#)
- [UserLevels](#) [userLevelEnabled](#)
- QColor **traceColor1**
- QColor **traceColor2**
- QColor **traceColor3**
- QColor **traceColor4**
- TraceStyles **traceStyle1**
- TraceStyles **traceStyle2**
- TraceStyles **traceStyle3**
- TraceStyles **traceStyle4**
- QString **traceLegend1**
- QString **traceLegend2**
- QString **traceLegend3**
- QString **traceLegend4**
- QString **title**
- QColor **backgroundColor**
- QString **xUnit**
- QString **yUnit**

9.78.1 Member Enumeration Documentation

9.78.1.1 enum QEPLOT::UserLevels

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

Enumerator:

User Refer to USERLEVEL_USER for details.

Scientist Refer to USERLEVEL_SCIENTIST for details.

Engineer Refer to USERLEVEL_ENGINEER for details.

9.78.2 Member Function Documentation

9.78.2.1 `void QEPlot::dbValueChanged (const double & out) [signal]`

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

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

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

9.78.2.3 `void QEPlot::requestEnabled (const bool & state) [inline, slot]`

Similar to standard setEnabled slot, but allows QE widget to determine if the widget remains disabled due to invalid data. If disabled due to invalid data, a request to enable the widget will be honoured when the data is no longer invalid.

9.78.3 Member Data Documentation

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

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

Reimplemented from [QEDragDrop](#).

9.78.4 Property Documentation

9.78.4.1 `bool QEPlot::enabled [read, write]`

Set the preferred 'enabled' state. Default is true. This property is copied to the standard Qt 'enabled' property if the data being displayed is valid. If the data being displayed is invalid the standard Qt 'enabled' property will always be set to false to indicate invalid data. The value of this property will only be copied to the standard Qt 'enabled' property once data is valid.

9.78.4.2 `unsigned QEPlot::int` [read, write]

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

9.78.4.3 `UserLevels QEPlot::userLevelEnabled` [read, write]

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

9.78.4.4 `QString QEPlot::userLevelEngineerStyle` [read, write]

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

9.78.4.5 `QString QEPlot::userLevelScientistStyle` [read, write]

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

9.78.4.6 `QString QEPlot::userLevelUserStyle` [read, write]

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

9.78.4.7 UserLevels QEPlot::userLevelVisibility [read, write]

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

9.78.4.8 QString QEPlot::variable1 [read, write]

EPICS variable name (CA PV). This variable is used to read updating values or waveforms for plotting in the first trace.

9.78.4.9 QString QEPlot::variable2 [read, write]

EPICS variable name (CA PV). This variable is used to read updating values or waveforms for plotting in the second trace.

9.78.4.10 QString QEPlot::variable3 [read, write]

EPICS variable name (CA PV). This variable is used to read updating values or waveforms for plotting in the third trace.

9.78.4.11 QString QEPlot::variable4 [read, write]

EPICS variable name (CA PV). This variable is used to read updating values or waveforms for plotting in the fourth trace.

9.78.4.12 bool QEPlot::variableAsToolTip [read, write]

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

Reimplemented from [QEToolTip](#).

9.78.4.13 QString QEPlot::variableSubstitutions [read, write]

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

9.78.4.14 bool QEPlot::visible [read, write]

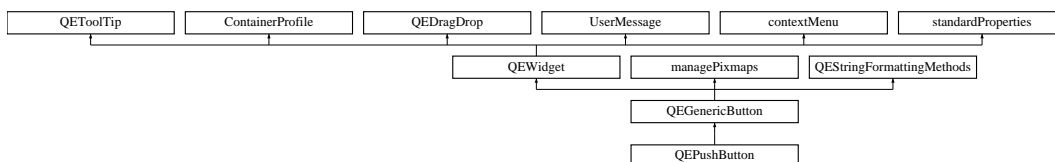
Display the widget. Default is true. Setting this property false is usefull if widget is only used to provide a signal - for example, when supplying data to a [QELink](#) widget. Note, when false the widget will still be visible in Qt Designer.

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

- /home/rhydera/epicsqt/framework/widgets/QEPlot/QEPlot.h
- /home/rhydera/epicsqt/framework/widgets/QEPlot/QEPlot.cpp

9.79 QEPushButton Class Reference

Inheritance diagram for QEPushButton:



Public Types

- enum **UserLevels** { **User** = USERLEVEL_USER, **Scientist** = USERLEVEL_SCIENTIST, **Engineer** = USERLEVEL_ENGINEER }
- enum **UpdateOptions** { **Text** = QEPushButton::UPDATE_TEXT, **Icon** = QEPushButton::UPDATE_ICON, **TextAndIcon** = QEPushButton::UPDATE_TEXT_AND_ICON, **State** = QEPushButton::UPDATE_STATE }
- 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 **CreationOptionNames** { **Open** = QForm::CREATION_OPTION_OPEN, **NewTab** = QForm::CREATION_OPTION_NEW_TAB, **NewWindow** = QForm::CREATION_OPTION_NEW_WINDOW }

Public Slots

- void **launchGui** (QString guiName, QForm::creationOptions creationOption)
- void **onGeneralMessage** (QString message)
- void **requestEnabled** (const bool &state)

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](#) (QString guiName, QForm::creationOptions creationOption)
Internal use only. Request a new GUI is created. Typically, this is caught by the QEGui application.

Public Member Functions

- **QEPushButton** (QWidget *parent=0)
- **QEPushButton** (const QString &variableName, QWidget *parent=0)
- bool [isEnabled](#) () const
Access function for [enabled](#) property - refer to [enabled](#) property for details.
- void [setEnabled](#) (bool state)
Access function for [enabled](#) property - refer to [enabled](#) property for details.
- **UserLevels** [getUserLevelVisibilityProperty](#) ()
Access function for [userLevelVisibility](#) property - refer to [userLevelVisibility](#) property for details.
- void [setUserLevelVisibilityProperty](#) (**UserLevels** level)
Access function for [userLevelVisibility](#) property - refer to [userLevelVisibility](#) property for details.
- **UserLevels** [getUserLevelEnabledProperty](#) ()
Access function for [userLevelEnabled](#) property - refer to [userLevelEnabled](#) property for details.
- void [setUserLevelEnabledProperty](#) (**UserLevels** level)
Access function for [userLevelEnabled](#) property - refer to [userLevelEnabled](#) property for details.

Protected Member Functions

- void **dragEnterEvent** (QDragEnterEvent *event)
- void **dropEvent** (QDropEvent *event)
- void **setDrop** (QVariant drop)
- QVariant **getDrop** ()

Properties

- QString [variable](#)
- QString [altReadbackVariable](#)
- QString **variableSubstitutions**
- bool [subscribe](#)
- bool [variableAsToolTip](#)

- bool [enabled](#)
- bool [allowDrop](#)
- bool [visible](#)
- unsigned [int](#)
- QString [userLevelUserStyle](#)
- QString [userLevelScientistStyle](#)
- QString [userLevelEngineerStyle](#)
- [UserLevels](#) [userLevelVisibility](#)
- [UserLevels](#) [userLevelEnabled](#)
- UpdateOptions [updateOption](#)
- QPixmap [pixmap0](#)
- QPixmap [pixmap1](#)
- QPixmap [pixmap2](#)
- QPixmap [pixmap3](#)
- QPixmap [pixmap4](#)
- QPixmap [pixmap5](#)
- QPixmap [pixmap6](#)
- QPixmap [pixmap7](#)
- bool [useDbPrecision](#)
- bool [leadingZero](#)
- bool [trailingZeros](#)
- bool [addUnits](#)
- QString [localEnumeration](#)
- Qt::Alignment [alignment](#)
- Formats [format](#)
- Notations [notation](#)
- QString [password](#)
- bool [confirmAction](#)
- bool [writeOnPress](#)
- bool [writeOnRelease](#)
- bool [writeOnClick](#)
- QString [pressText](#)
- QString [releaseText](#)
- QString [clickText](#)
- QString [clickCheckedText](#)
- QString [labelText](#)
- QString [program](#)
- QStringList [arguments](#)
- QString [guiFile](#)
- CreationOptionNames [creationOption](#)

9.79.1 Member Enumeration Documentation

9.79.1.1 enum QEPushButton::UserLevels

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

Enumerator:

User Refer to `USERLEVEL_USER` for details.

Scientist Refer to `USERLEVEL_SCIENTIST` for details.

Engineer Refer to `USERLEVEL_ENGINEER` for details.

9.79.2 Member Function Documentation

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

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

9.79.2.2 void QEPushButton::requestEnabled (const bool & state) [inline, slot]

Similar to standard `setEnabled` slot, but allows QE widget to determine if the widget remains disabled due to invalid data. If disabled due to invalid data, a request to enable the widget will be honoured when the data is no longer invalid.

9.79.3 Property Documentation

9.79.3.1 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.

Reimplemented from [QEDragDrop](#).

9.79.3.2 QString QEPushButton::altReadbackVariable [read, write]

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

9.79.3.3 bool QEPushButton::enabled [read, write]

Set the preferred 'enabled' state. Default is true. This property is copied to the standard Qt 'enabled' property if the data being displayed is valid. If the data being displayed is

invalid the standard Qt 'enabled' property will always be set to false to indicate invalid data. The value of this property will only be copied to the standard Qt 'enabled' property once data is valid.

9.79.3.4 `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.

9.79.3.5 `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)

Reimplemented from [QEWidget](#).

9.79.3.6 `UserLevels QEPushButton::userLevelEnabled` [read, write]

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

9.79.3.7 `QString QEPushButton::userLevelEngineerStyle` [read, write]

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

9.79.3.8 `QString QEPushButton::userLevelScientistStyle` [read, write]

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

9.79.3.9 QString QEPushButton::userLevelUserStyle [read, write]

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

9.79.3.10 UserLevels QEPushButton::userLevelVisibility [read, write]

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

9.79.3.11 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.

9.79.3.12 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.

Reimplemented from [QEToolTip](#).

9.79.3.13 bool QEPushButton::visible [read, write]

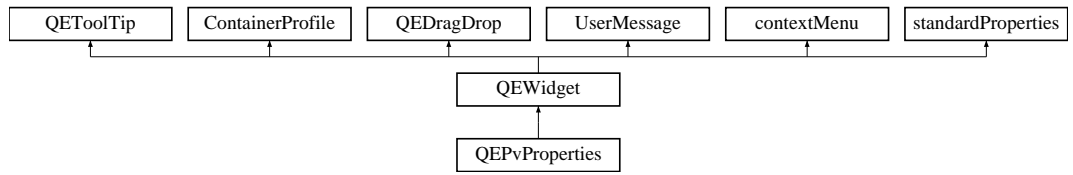
Display the widget. Default is true. Setting this property false is usefull if widget is only used to provide a signal - for example, when supplying data to a [QELink](#) widget. Note, when false the widget will still be visible in Qt Designer.

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

- /home/rhydera/epicsqt/framework/widgets/QEButton/QEPushButton.h
- /home/rhydera/epicsqt/framework/widgets/QEButton/QEPushButton.cpp

9.80 QEPvProperties Class Reference

Inheritance diagram for QEPvProperties:



Classes

- struct **WidgetHolder**

Public Types

- enum **UserLevels** { **User** = USERLEVEL_USER, **Scientist** = USERLEVEL_SCIENTIST, **Engineer** = USERLEVEL_ENGINEER }

Public Slots

- void **requestEnabled** (const bool &state)

Signals

- void **setCurrentBoxIndex** (int index)

Public Member Functions

- bool **isEnabled** () const
*Access function for **enabled** property - refer to **enabled** property for details.*
- void **setEnabled** (bool state)
*Access function for **enabled** property - refer to **enabled** property for details.*
- **UserLevels** **getUserLevelVisibilityProperty** ()
*Access function for **userLevelVisibility** property - refer to **userLevelVisibility** property for details.*
- void **setUserLevelVisibilityProperty** (**UserLevels** level)
*Access function for **userLevelVisibility** property - refer to **userLevelVisibility** property for details.*
- **UserLevels** **getUserLevelEnabledProperty** ()
*Access function for **userLevelEnabled** property - refer to **userLevelEnabled** property for details.*
- void **setUserLevelEnabledProperty** (**UserLevels** level)
*Access function for **userLevelEnabled** property - refer to **userLevelEnabled** property for details.*
- **QEPvProperties** (**QWidget** *parent=0)
- **QEPvProperties** (const **QString** &variableName, **QWidget** *parent=0)

- QSize **sizeHint** () const
- void **establishConnection** (unsigned int variableIndex)

Protected Member Functions

- void **setup** ()
- [qcaobject::QCaObject](#) * **createQcaltem** (unsigned int variableIndex)
- void **dragEnterEvent** (QDragEnterEvent *event)
- void **dropEvent** (QDropEvent *event)
- void **mousePressEvent** (QMouseEvent *event)
- QString **copyVariable** ()
- QVariant **copyData** ()
- void **paste** (QVariant s)
- void **setDrop** (QVariant drop)
- QVariant **getDrop** ()

Properties

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

9.80.1 Member Enumeration Documentation

9.80.1.1 enum QEPvProperties::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 QEPvProperties::requestEnabled (const bool & state) [inline, slot]

Similar to standard setEnabled slot, but allows QE widget to determine if the widget remains disabled due to invalid data. If disabled due to invalid data, a request to enable the widget will be honoured when the data is no longer invalid.

9.80.3 Property Documentation

9.80.3.1 bool QEPvProperties::allowDrop [read, write]

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

Reimplemented from [QEDragDrop](#).

9.80.3.2 bool QEPvProperties::enabled [read, write]

Set the preferred 'enabled' state. Default is true. This property is copied to the standard Qt 'enabled' property if the data being displayed is valid. If the data being displayed is invalid the standard Qt 'enabled' property will always be set to false to indicate invalid data. The value of this property will only be copied to the standard Qt 'enabled' property once data is valid.

9.80.3.3 unsigned QEPvProperties::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.

9.80.3.4 UserLevels QEPvProperties::userLevelEnabled [read, write]

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

9.80.3.5 QString QEPvProperties::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.

9.80.3.6 QString QEPvProperties::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.

9.80.3.7 QString QEPvProperties::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.

9.80.3.8 UserLevels QEPvProperties::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'.

9.80.3.9 QString QEPvProperties::variable [read, write]

EPICS variable name (CA PV)

9.80.3.10 bool QEPvProperties::variableAsToolTip [read, write]

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

Reimplemented from [QEToolTip](#).

9.80.3.11 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 are also used for other purposes.

9.80.3.12 bool QEPvProperties::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:

- /home/rhydera/epicsqt/framework/widgets/QEPvProperties/QEPvProperties.h
- /home/rhydera/epicsqt/framework/widgets/QEPvProperties/QEPvProperties.cpp

9.81 QEPvPropertiesManager Class Reference

Public Member Functions

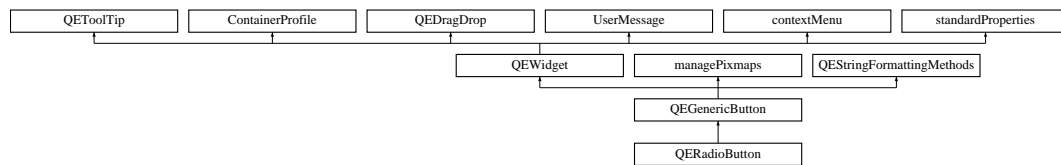
- **QEPvPropertiesManager** (QObject *parent=0)
- bool **isContainer** () const
- bool **isInitialized** () const
- QIcon **icon** () const
- QString **group** () const
- QString **includeFile** () const
- QString **name** () const
- QString **toolTip** () const
- QString **whatsThis** () const
- QWidget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

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

- /home/rhydera/epicsqt/framework/widgets/QEPvProperties/QEPvPropertiesManager.h
- /home/rhydera/epicsqt/framework/widgets/QEPvProperties/QEPvPropertiesManager.cpp

9.82 QERadioButton Class Reference

Inheritance diagram for QERadioButton:



Public Types

- enum **UserLevels** { **User** = USERLEVEL_USER, **Scientist** = USERLEVEL_SCIENTIST, **Engineer** = USERLEVEL_ENGINEER }
- enum **UpdateOptions** { **Text** = QERadioButton::UPDATE_TEXT, **Icon** = QERadioButton::UPDATE_ICON, **TextAndIcon** = QERadioButton::UPDATE_TEXT_AND_ICON, **State** = QERadioButton::UPDATE_STATE }
- enum **Formats** {
Default = QStringFormatting::FORMAT_DEFAULT, **Floating** = QStringFormatting::FORMAT_FLOATING, **Integer** = QStringFormatting::FORMAT_INTEGER, **UnsignedInteger** = QStringFormatting::FORMAT_UNSIGNEDINTEGER,
Time = QStringFormatting::FORMAT_TIME }
- enum **Notations** { **Fixed** = QStringFormatting::NOTATION_FIXED, **Scientific** = QStringFormatting::NOTATION_SCIENTIFIC, **Automatic** = QStringFormatting::NOTATION_AUTOMATIC }
- enum **CreationOptionNames** { **Open** = QForm::CREATION_OPTION_OPEN, **NewTab** = QForm::CREATION_OPTION_NEW_TAB, **NewWindow** = QForm::CREATION_OPTION_NEW_WINDOW }

Public Slots

- void **launchGui** (QString guiName, QForm::creationOptions creationOption)
- void **onGeneralMessage** (QString message)
- void **requestEnabled** (const bool &state)

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** (QString guiName, QForm::creationOptions creationOption)
Internal use only. Request a new GUI is created. Typically, this is caught by the QEGui application.

Public Member Functions

- **QERadioButton** (QWidget *parent=0)
- **QERadioButton** (const QString &variableName, QWidget *parent=0)
- bool **isEnabled** () const
Access function for [enabled](#) property - refer to [enabled](#) property for details.
- void **setEnabled** (bool state)
Access function for [enabled](#) property - refer to [enabled](#) property for details.
- **UserLevels getUserLevelVisibilityProperty** ()
Access function for [userLevelVisibility](#) property - refer to [userLevelVisibility](#) property for details.
- void **setUserLevelVisibilityProperty** (UserLevels level)
Access function for [userLevelVisibility](#) property - refer to [userLevelVisibility](#) property for details.
- **UserLevels getUserLevelEnabledProperty** ()
Access function for [userLevelEnabled](#) property - refer to [userLevelEnabled](#) property for details.
- void **setUserLevelEnabledProperty** (UserLevels level)
Access function for [userLevelEnabled](#) property - refer to [userLevelEnabled](#) property for details.

Protected Member Functions

- void **dragEnterEvent** (QDragEnterEvent *event)
- void **dropEvent** (QDropEvent *event)
- void **setDrop** (QVariant drop)
- QVariant **getDrop** ()

Properties

- QString [variable](#)
- QString [variableSubstitutions](#)
- bool [subscribe](#)
- bool [variableAsToolTip](#)
- bool [enabled](#)
- bool [allowDrop](#)
- bool [visible](#)
- unsigned [int](#)
- QString [userLevelUserStyle](#)
- QString [userLevelScientistStyle](#)
- QString [userLevelEngineerStyle](#)
- [UserLevels userLevelVisibility](#)
- [UserLevels userLevelEnabled](#)
- UpdateOptions **updateOption**
- QPixmap **pixmap0**

- QPixmap **pixmap1**
- QPixmap **pixmap2**
- QPixmap **pixmap3**
- QPixmap **pixmap4**
- QPixmap **pixmap5**
- QPixmap **pixmap6**
- QPixmap **pixmap7**
- bool **useDbPrecision**
- bool **leadingZero**
- bool **trailingZeros**
- bool **addUnits**
- Qt::Alignment **alignment**
- Formats **format**
- Notations **notation**
- QString **password**
- bool **confirmAction**
- bool **writeOnPress**
- bool **writeOnRelease**
- bool **writeOnClick**
- QString **pressText**
- QString **releaseText**
- QString **clickText**
- QString **clickCheckedText**
- QString **labelText**
- QString **program**
- QStringList **arguments**
- QString **guiFile**
- CreationOptionNames **creationOption**

9.82.1 Member Enumeration Documentation

9.82.1.1 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.

9.82.2 Member Function Documentation

9.82.2.1 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.

9.82.2.2 void QERadioButton::requestEnabled (const bool & state) [inline, slot]

Similar to standard setEnabled slot, but allows QE widget to determine if the widget remains disabled due to invalid data. If disabled due to invalid data, a request to enable the widget will be honoured when the data is no longer invalid.

9.82.3 Property Documentation

9.82.3.1 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.

Reimplemented from [QEDragDrop](#).

9.82.3.2 bool QERadioButton::enabled [read, write]

Set the preferred 'enabled' state. Default is true. This property is copied to the standard Qt 'enabled' property if the data being displayed is valid. If the data being displayed is invalid the standard Qt 'enabled' property will always be set to false to indicate invalid data. The value of this property will only be copied to the standard Qt 'enabled' property once data is valid.

9.82.3.3 unsigned QERadioButton::int [read, write]

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

9.82.3.4 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)

Reimplemented from [QEWidget](#).

9.82.3.5 UserLevels QERadioButton::userLevelEnabled [read, write]

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

9.82.3.6 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.

9.82.3.7 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.

9.82.3.8 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.

9.82.3.9 UserLevels QERadioButton::userLevelVisibility [read, write]

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

9.82.3.10 QString QERadioButton::variable [read, write]

EPICS variable name (CA PV)

9.82.3.11 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.

Reimplemented from [QEToolTip](#).

9.82.3.12 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.

9.82.3.13 bool QERadioButton::visible [read, write]

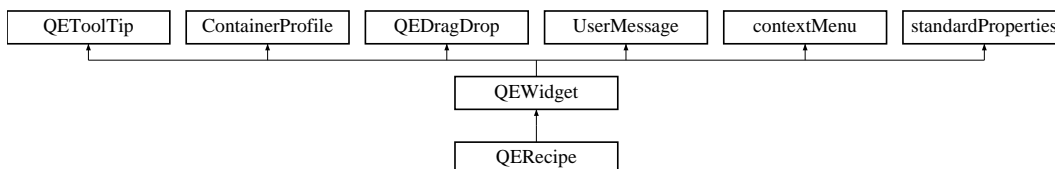
Display the widget. Default is true. Setting this property false is usefull if widget is only used to provide a signal - for example, when supplying data to a [QELink](#) widget. Note, when false the widget will still be visible in Qt Designer.

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

- /home/rhydera/epicsqt/framework/widgets/QEButton/QERadioButton.h
- /home/rhydera/epicsqt/framework/widgets/QEButton/QERadioButton.cpp

9.83 QERecipe Class Reference

Inheritance diagram for QERecipe:



Public Types

- enum **configurationTypesProperty** { **File** = FROM_FILE, **Text** = FROM_TEXT }
- enum **detailsLayoutProperty** { **Top** = TOP, **Bottom** = BOTTOM, **Left** = LEFT, **Right** = RIGHT }

- enum **userTypesProperty** { **User** = USERLEVEL_USER, **Scientist** = USERLEVEL_SCIENTIST, **Engineer** = 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 **setDetailsLayout** (int pValue)
- int **getDetailsLayout** ()
- void **setCurrentUserType** (int pValue)
- int **getCurrentUserType** ()
- bool **saveRecipeList** ()
- void **refreshRecipeList** ()
- void **refreshButton** ()
- void **userLevelChanged** (userLevels pValue)
- void **setConfigurationTypeProperty** (configurationTypesProperty pConfigurationType)
- configurationTypesProperty **getConfigurationTypeProperty** ()
- void **setDetailsLayoutProperty** (detailsLayoutProperty pDetailsLayout)
- detailsLayoutProperty **getDetailsLayoutProperty** ()
- 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 **detailsLayout**
- int **currentUserType**

Properties

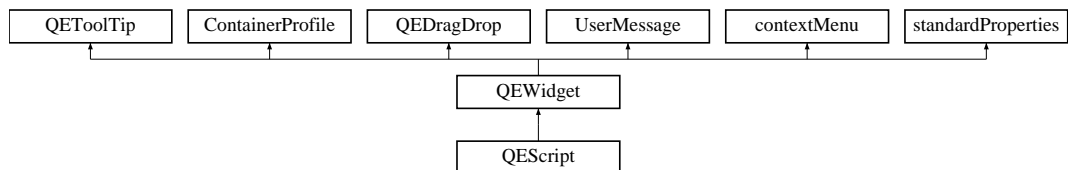
- QString **recipeDescription**
- bool **showRecipeList**
- bool **showNew**
- bool **showSave**
- bool **showDelete**
- bool **showApply**
- bool **showRead**
- bool **showFields**
- configurationTypesProperty **configurationType**
- QString **configurationFile**
- QString **configurationText**
- detailsLayoutProperty **detailsLayout**
- userTypesProperty **currentUserType**

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

- /home/rhydera/epicsqt/framework/widgets/QERecipe/QERecipe.h
- /home/rhydera/epicsqt/framework/widgets/QERecipe/QERecipe.cpp

9.84 QEScript Class Reference

Inheritance diagram for QEScript:



Public Types

- enum **detailsLayoutProperty** { **Top** = TOP, **Bottom** = BOTTOM, **Left** = LEFT, **Right** = RIGHT }

Signals

- void **selected** (QString pFilename)

Public Member Functions

- **QEScript** (QWidget *pParent=0)
- 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 **setShowColumnTime** (bool pValue)
- bool **getShowColumnTime** ()
- void **setShowColumnSize** (bool pValue)
- bool **getShowColumnSize** ()
- void **setShowColumnFilename** (bool pValue)
- bool **getShowColumnFilename** ()
- void **setShowFileExtension** (bool pValue)
- bool **getShowFileExtension** ()
- void **setFileFilter** (QString pValue)
- QString **getFileFilter** ()
- void **setDetailsLayout** (int pValue)
- int **getDetailsLayout** ()
- void **updateTable** ()
- void **setDetailsLayoutProperty** (detailsLayoutProperty pDetailsLayout)
- detailsLayoutProperty **getDetailsLayoutProperty** ()

Protected Attributes

- QLineEdit * **qlineEditDirectoryPath**
- QPushButton * **qPushButtonDirectoryBrowser**
- QPushButton * **qPushButtonRefresh**
- [_QTableWidgetScript](#) * **qTableWidgetScript**
- QString **fileFilter**
- bool **showFileExtension**
- int **detailsLayout**

Properties

- QString **directoryPath**
- bool **showDirectoryPath**
- bool **showDirectoryBrowser**
- bool **showRefresh**
- bool **showColumnTime**
- bool **showColumnSize**
- bool **showColumnFilename**
- detailsLayoutProperty **detailsLayout**

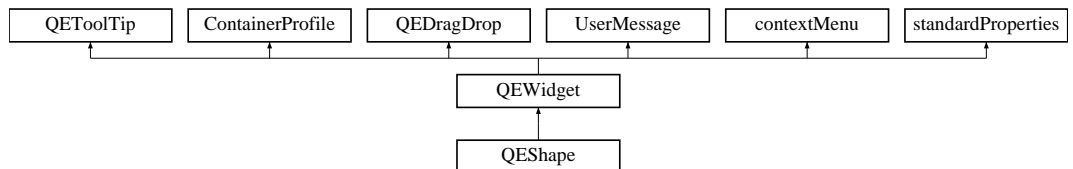
The documentation for this class was generated from the following files:

- /home/rhydera/epicsqt/framework/widgets/QEScript/QEScript.h
- /home/rhydera/epicsqt/framework/widgets/QEScript/QEScript.cpp

9.85 QEShape Class Reference

```
#include <QEShape.h>
```

Inheritance diagram for QEShape:



Public Types

- enum **shapeOptions** {
Line, Points, Polyline, Polygon,
Rect, RoundedRect, Ellipse, Arc,
Chord, Pie, Path, Text,
Pixmap }
- enum **animationOptions** {
Width, Height, X, Y,
Transperency, Rotation, ColourHue, ColourSaturation,
ColourValue, ColourIndex, Penwidth }
- enum **UserLevels** { **User** = USERLEVEL_USER, **Scientist** = USERLEVEL_SCIENTIST,
Engineer = USERLEVEL_ENGINEER }

Public Slots

- void [requestEnabled](#) (const bool &state)

Signals

- void [dbValueChanged1](#) (const qlonglong &out)
- void [dbValueChanged2](#) (const qlonglong &out)
- void [dbValueChanged3](#) (const qlonglong &out)
- void [dbValueChanged4](#) (const qlonglong &out)
- void [dbValueChanged5](#) (const qlonglong &out)
- void [dbValueChanged6](#) (const qlonglong &out)

Public Member Functions

- [QEShape](#) (QWidget *parent=0)
- [QEShape](#) (const QString &variableName, QWidget *parent=0)
- void [setAnimation](#) ([animationOptions](#) animation, const int index)
Access function for #animation' properties - refer to animation' properties for details.
- [animationOptions](#) [getAnimation](#) (const int index)
Access function for #animation' properties - refer to animation' properties for details.
- void [setScale](#) (const double scale, const int index)
Access function for #scale' properties - refer to scale' properties for details.
- double [getScale](#) (const int index)
Access function for #scale' properties - refer to scale' properties for details.
- void [setOffset](#) (const double offset, const int index)
Access function for #offset' properties - refer to offset' properties for details.
- double [getOffset](#) (const int index)
Access function for #offset' properties - refer to offset' properties for details.
- void [setBorder](#) (const bool border)
Access function for #border' properties - refer to border' properties for details.
- bool [getBorder](#) ()
Access function for #border' properties - refer to border' properties for details.
- void [setFill](#) (const bool fill)
Access function for #fill' properties - refer to fill' properties for details.
- bool [getFill](#) ()
Access function for #fill' properties - refer to fill' properties for details.
- void [setShape](#) ([shapeOptions](#) shape)
Access function for #shape' properties - refer to shape' properties for details.
- [shapeOptions](#) [getShape](#) ()
Access function for #shape' properties - refer to shape' properties for details.
- void [setNumPoints](#) (const unsigned int numPoints)
Access function for #number of points' properties - refer to number of points' properties for details.

- unsigned int [getNumPoints](#) ()
Access function for #number of points' properties - refer to number of points' properties for details.
- void [setOriginTranslation](#) (const QPoint originTranslation)
Access function for #origin translation' properties - refer to origin translation' properties for details.
- QPoint [getOriginTranslation](#) ()
Access function for #origin translation' properties - refer to origin translation' properties for details.
- void [setPoint](#) (const QPoint point, const int index)
Access function for #point' properties - refer to point' properties for details.
- QPoint [getPoint](#) (const int index)
Access function for #point' properties - refer to point' properties for details.
- void [setColor](#) (const QColor color, const int index)
Access function for #colour' properties - refer to colour' properties for details.
- QColor [getColor](#) (const int index)
Access function for #colour' properties - refer to colour' properties for details.
- void [setDrawBorder](#) (const bool drawBorder)
Access function for #draw border' properties - refer to draw border' properties for details.
- bool [getDrawBorder](#) ()
Access function for #draw border' properties - refer to draw border' properties for details.
- void [setLineWidth](#) (const unsigned int lineWidth)
Access function for #line width' properties - refer to line width' properties for details.
- unsigned int [getLineWidth](#) ()
Access function for #line width' properties - refer to line width' properties for details.
- void [setStartAngle](#) (const double startAngle)
Access function for #start angle' properties - refer to start angle' properties for details.
- double [getStartAngle](#) ()
Access function for #start angle' properties - refer to start angle' properties for details.
- void [setRotation](#) (const double rotation)
Access function for #rotation' properties - refer to rotation' properties for details.
- double [getRotation](#) ()
Access function for #rotation' properties - refer to rotation' properties for details.
- void [setArcLength](#) (const double arcLength)
Access function for #arc length' properties - refer to arc length' properties for details.
- double [getArcLength](#) ()
Access function for #arc length' properties - refer to arc length' properties for details.
- void [setText](#) (const QString text)
Access function for #text' properties - refer to text' properties for details.
- QString [getText](#) ()
Access function for #text' properties - refer to text' properties for details.
- bool [isEnabled](#) () const

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

- void [setEnabled](#) (bool state)

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

- [UserLevels](#) [getUserLevelVisibilityProperty](#) ()

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

- void [setUserLevelVisibilityProperty](#) ([UserLevels](#) level)

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

- [UserLevels](#) [getUserLevelEnabledProperty](#) ()

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

- void [setUserLevelEnabledProperty](#) ([UserLevels](#) level)

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

Properties

- QString [variable1](#)
- QString [variable2](#)
- QString [variable3](#)
- QString [variable4](#)
- QString [variable5](#)
- QString [variable6](#)
- QString [variableSubstitutions](#)
- bool [variableAsToolTip](#)
- bool [enabled](#)
- bool [allowDrop](#)
- bool [visible](#)
- unsigned int
- QString [userLevelUserStyle](#)
- QString [userLevelScientistStyle](#)
- QString [userLevelEngineerStyle](#)
- [UserLevels](#) [userLevelVisibility](#)
- [UserLevels](#) [userLevelEnabled](#)
- [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](#)

9.85.1 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 elipse can be drawn with four variables animating its vertcal and horizontal size and position. It is tightly integrated with the base class [QEWidget](#) which provides generic support such as macro substitutions, drag/drop, and standard properties.

9.85.2 Member Enumeration Documentation

9.85.2.1 enum [QEShape::animationOptions](#)

Options for how a variable will animate the shape.

9.85.2.2 enum QEShape::shapeOptions

Options for the type of shape.

9.85.2.3 enum QEShape::UserLevels

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

Enumerator:

User Refer to USERLEVEL_USER for details.

Scientist Refer to USERLEVEL_SCIENTIST for details.

Engineer Refer to USERLEVEL_ENGINEER for details.

9.85.3 Constructor & Destructor Documentation

9.85.3.1 QEShape::QEShape (QWidget * *parent* = 0)

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

9.85.3.2 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.

9.85.4 Member Function Documentation

9.85.4.1 void QEShape::dbValueChanged1 (const qlonglong & *out*) [signal]

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

9.85.4.2 void QEShape::dbValueChanged2 (const qlonglong & *out*) [signal]

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

9.85.4.3 void QEShape::dbValueChanged3 (const qlonglong & out) [signal]

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

9.85.4.4 void QEShape::dbValueChanged4 (const qlonglong & out) [signal]

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

9.85.4.5 void QEShape::dbValueChanged5 (const qlonglong & out) [signal]

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

9.85.4.6 void QEShape::dbValueChanged6 (const qlonglong & out) [signal]

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

9.85.4.7 void QEShape::requestEnabled (const bool & state) [inline, slot]

Similar to standard setEnabled slot, but allows QE widget to determine if the widget remains disabled due to invalid data. If disabled due to invalid data, a request to enable the widget will be honoured when the data is no longer invalid.

9.85.5 Property Documentation

9.85.5.1 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.

Reimplemented from [QEDragDrop](#).

9.85.5.2 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.

9.85.5.3 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.

9.85.5.4 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.

9.85.5.5 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.

9.85.5.6 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.

9.85.5.7 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.

9.85.5.8 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'.

9.85.5.9 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'.

9.85.5.10 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'.

9.85.5.11 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'.

9.85.5.12 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'.

9.85.5.13 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'.

9.85.5.14 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'.

9.85.5.15 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'.

9.85.5.16 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'.

9.85.5.17 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'.

9.85.5.18 bool QEShape::enabled [read, write]

Set the preferred 'enabled' state. Default is true. This property is copied to the standard Qt 'enabled' property if the data being displayed is valid. If the data being displayed is invalid the standard Qt 'enabled' property will always be set to false to indicate invalid data. The value of this property will only be copied to the standard Qt 'enabled' property once data is valid.

9.85.5.19 unsigned QEShape::int [read, write]

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

The number of points to use when drawing shapes that are defined by a variable number of points, such as polyline, polygon, path, and series of points.

Sets the width of the pen. Used for the following shapes: Line, Points, Polyline, Polygon, Rect, RoundedRect, Ellipse, Arc, Chord, Pie, Path

9.85.5.20 `double QEShape::offset1` [read, write]

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

9.85.5.21 `double QEShape::offset2` [read, write]

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

9.85.5.22 `double QEShape::offset3` [read, write]

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

9.85.5.23 `double QEShape::offset4` [read, write]

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

9.85.5.24 `double QEShape::offset5` [read, write]

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

9.85.5.25 `double QEShape::offset6` [read, write]

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

9.85.5.26 `QPoint QEShape::point1` [read, write]

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

9.85.5.27 `QPoint QEShape::point10` [read, write]

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

9.85.5.28 QPoint QEShape::point2 [read, write]

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

9.85.5.29 QPoint QEShape::point3 [read, write]

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

9.85.5.30 QPoint QEShape::point4 [read, write]

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

9.85.5.31 QPoint QEShape::point5 [read, write]

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

9.85.5.32 QPoint QEShape::point6 [read, write]

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

9.85.5.33 QPoint QEShape::point7 [read, write]

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

9.85.5.34 QPoint QEShape::point8 [read, write]

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

9.85.5.35 QPoint QEShape::point9 [read, write]

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

9.85.5.36 double QEShape::scale2 [read, write]

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

9.85.5.37 `double QEShape::scale3` [read, write]

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

9.85.5.38 `double QEShape::scale4` [read, write]

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

9.85.5.39 `double QEShape::scale5` [read, write]

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

9.85.5.40 `double QEShape::scale6` [read, write]

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

9.85.5.41 `UserLevels QEShape::userLevelEnabled` [read, write]

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

9.85.5.42 `QString QEShape::userLevelEngineerStyle` [read, write]

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

9.85.5.43 `QString QEShape::userLevelScientistStyle` [read, write]

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

will be merged with any pre-existing Style Sheet string and any Style Sheet strings generated during the display of data.

9.85.5.44 `QString QEShape::userLevelUserStyle` [read, write]

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

9.85.5.45 `UserLevels QEShape::userLevelVisibility` [read, write]

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

9.85.5.46 `QString QEShape::variable1` [read, write]

EPICS variable name (CA PV). This variable is read and used to animate an attribute of the shape. The value read is first scaled and offset by properties scale1 and offset1 then the attribute selected for animation is selected by the property animation1.

9.85.5.47 `QString QEShape::variable2` [read, write]

EPICS variable name (CA PV). This variable is read and used to animate an attribute of the shape. The value read is first scaled and offset by properties scale2 and offset2 then the attribute selected for animation is selected by the property animation2.

9.85.5.48 `QString QEShape::variable3` [read, write]

EPICS variable name (CA PV). This variable is read and used to animate an attribute of the shape. The value read is first scaled and offset by properties scale3 and offset3 then the attribute selected for animation is selected by the property animation3.

9.85.5.49 `QString QEShape::variable4` [read, write]

EPICS variable name (CA PV). This variable is read and used to animate an attribute of the shape. The value read is first scaled and offset by properties scale4 and offset4 then the attribute selected for animation is selected by the property animation4.

9.85.5.50 `QString QEShape::variable5` [read, write]

EPICS variable name (CA PV). This variable is read and used to animate an attribute of the shape. The value read is first scaled and offset by properties scale5 and offset5 then the attribute selected for animation is selected by the property animation5.

9.85.5.51 `QString QEShape::variable6` [read, write]

EPICS variable name (CA PV). This variable is read and used to animate an attribute of the shape. The value read is first scaled and offset by properties scale6 and offset6 then the attribute selected for animation is selected by the property animation6.

9.85.5.52 `bool QEShape::variableAsToolTip` [read, write]

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

Reimplemented from [QEToolTip](#).

9.85.5.53 `QString QEShape::variableSubstitutions` [read, write]

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

9.85.5.54 `bool QEShape::visible` [read, write]

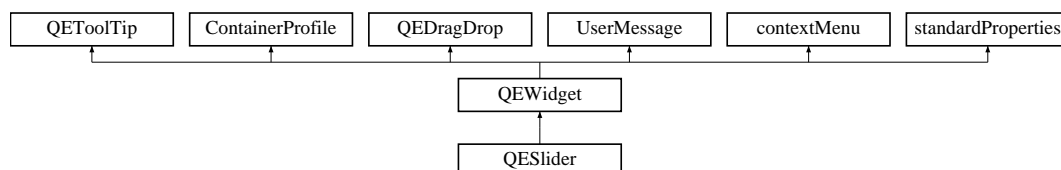
Display the widget. Default is true. Setting this property false is usefull if widget is only used to provide a signal - for example, when supplying data to a [QELink](#) widget. Note, when false the widget will still be visible in Qt Designer.

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

- /home/rhydera/epicsqt/framework/widgets/QEShape/QEShape.h
- /home/rhydera/epicsqt/framework/widgets/QEShape/QEShape.cpp

9.86 QESlider Class Reference

Inheritance diagram for QESlider:



Public Types

- enum [UserLevels](#) { [User](#) = USERLEVEL_USER, [Scientist](#) = USERLEVEL_SCIENTIST, [Engineer](#) = USERLEVEL_ENGINEER }

Public Slots

- void [requestEnabled](#) (const bool &state)

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** ()
- bool **isEnabled** () const
Access function for [enabled](#) property - refer to [enabled](#) property for details.
- void **setEnabled** (bool state)
Access function for [enabled](#) property - refer to [enabled](#) property for details.
- [UserLevels](#) **getUserLevelVisibilityProperty** ()
Access function for [userLevelVisibility](#) property - refer to [userLevelVisibility](#) property for details.
- void **setUserLevelVisibilityProperty** ([UserLevels](#) level)
Access function for [userLevelVisibility](#) property - refer to [userLevelVisibility](#) property for details.
- [UserLevels](#) **getUserLevelEnabledProperty** ()
Access function for [userLevelEnabled](#) property - refer to [userLevelEnabled](#) property for details.
- void **setUserLevelEnabledProperty** ([UserLevels](#) level)
Access function for [userLevelEnabled](#) property - refer to [userLevelEnabled](#) property for details.

Protected Member Functions

- void **establishConnection** (unsigned int variableIndex)
- void **dragEnterEvent** (QDragEnterEvent *event)
- void **dropEvent** (QDropEvent *event)
- void **setDrop** (QVariant drop)
- QVariant **getDrop** ()

Protected Attributes

- [QEFloatingFormatting](#) **floatingFormatting**
- bool [writeOnChange](#)

Properties

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

9.86.1 Member Enumeration Documentation

9.86.1.1 enum QESlider::UserLevels

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

Enumerator:

User Refer to `USERLEVEL_USER` for details.

Scientist Refer to `USERLEVEL_SCIENTIST` for details.

Engineer Refer to `USERLEVEL_ENGINEER` for details.

9.86.2 Member Function Documentation

9.86.2.1 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.

9.86.2.2 void QESlider::requestEnabled (const bool & state) [inline, slot]

Similar to standard setEnabled slot, but allows QE widget to determine if the widget remains disabled due to invalid data. If disabled due to invalid data, a request to enable the widget will be honoured when the data is no longer invalid.

9.86.3 Member Data Documentation

9.86.3.1 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).

9.86.4 Property Documentation

9.86.4.1 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.

Reimplemented from [QEDragDrop](#).

9.86.4.2 bool QESlider::enabled [read, write]

Set the preferred 'enabled' state. Default is true. This property is copied to the standard Qt 'enabled' property if the data being displayed is valid. If the data being displayed is invalid the standard Qt 'enabled' property will always be set to false to indicate invalid data. The value of this property will only be copied to the standard Qt 'enabled' property once data is valid.

9.86.4.3 unsigned QESlider::int [read, write]

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

9.86.4.4 `bool QESlider::subscribe` [read, write]

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

Reimplemented from [QESlider](#).

9.86.4.5 `UserLevels QESlider::userLevelEnabled` [read, write]

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

9.86.4.6 `QString QESlider::userLevelEngineerStyle` [read, write]

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

9.86.4.7 `QString QESlider::userLevelScientistStyle` [read, write]

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

9.86.4.8 `QString QESlider::userLevelUserStyle` [read, write]

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

9.86.4.9 UserLevels QESlider::userLevelVisibility [read, write]

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

9.86.4.10 QString QESlider::variable [read, write]

EPICS variable name (CA PV)

9.86.4.11 bool QESlider::variableAsToolTip [read, write]

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

Reimplemented from [QEToolTip](#).

9.86.4.12 QString QESlider::variableSubstitutions [read, write]

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

9.86.4.13 bool QESlider::visible [read, write]

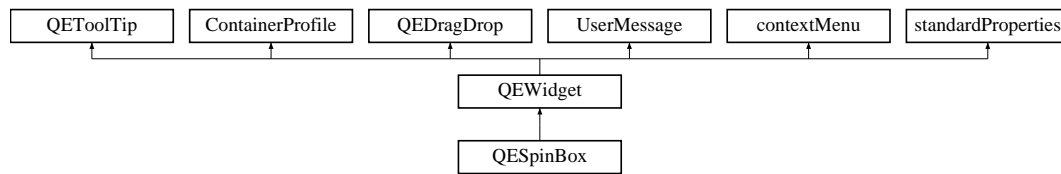
Display the widget. Default is true. Setting this property false is useful if widget is only used to provide a signal - for example, when supplying data to a [QELink](#) widget. Note, when false the widget will still be visible in Qt Designer.

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

- /home/rhydera/epicsqt/framework/widgets/QESlider/QESlider.h
- /home/rhydera/epicsqt/framework/widgets/QESlider/QESlider.cpp

9.87 QESpinBox Class Reference

Inheritance diagram for QESpinBox:



Public Types

- enum [UserLevels](#) { [User](#) = USERLEVEL_USER, [Scientist](#) = USERLEVEL_SCIENTIST, [Engineer](#) = USERLEVEL_ENGINEER }

Public Slots

- void [requestEnabled](#) (const bool &state)

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** ()
- bool **isEnabled** () const
Access function for [enabled](#) property - refer to [enabled](#) property for details.
- void **setEnabled** (bool state)
Access function for [enabled](#) property - refer to [enabled](#) property for details.
- UserLevels** **getUserLevelVisibilityProperty** ()
Access function for [userLevelVisibility](#) property - refer to [userLevelVisibility](#) property for details.
- void **setUserLevelVisibilityProperty** ([UserLevels](#) level)

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

- [UserLevels](#) [getUserLevelEnabledProperty](#) ()

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

- void [setUserLevelEnabledProperty](#) ([UserLevels](#) level)

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

Protected Member Functions

- void **establishConnection** (unsigned int variableIndex)
- void **dragEnterEvent** (QDragEnterEvent *event)
- void **dropEvent** (QDropEvent *event)
- void **setDrop** (QVariant drop)
- QVariant **getDrop** ()

Protected Attributes

- [QEFloatingFormatting](#) **floatingFormatting**
- bool **writeOnChange**
- bool **addUnitsAsSuffix**
- bool **useDbPrecisionForDecimal**

Properties

- QString [variable](#)
- QString [variableSubstitutions](#)
- bool [variableAsToolTip](#)
- bool [enabled](#)
- bool [allowDrop](#)
- bool [visible](#)
- unsigned [int](#)
- QString [userLevelUserStyle](#)
- QString [userLevelScientistStyle](#)
- QString [userLevelEngineerStyle](#)
- [UserLevels](#) [userLevelVisibility](#)
- [UserLevels](#) [userLevelEnabled](#)
- bool [subscribe](#)
- bool **useDbPrecision**
- bool **addUnits**

9.87.1 Member Enumeration Documentation

9.87.1.1 enum QESpinBox::UserLevels

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

Enumerator:

User Refer to `USERLEVEL_USER` for details.

Scientist Refer to `USERLEVEL_SCIENTIST` for details.

Engineer Refer to `USERLEVEL_ENGINEER` for details.

9.87.2 Member Function Documentation

9.87.2.1 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.

9.87.2.2 void QESpinBox::requestEnabled (const bool & state) [inline, slot]

Similar to standard `setEnabled` slot, but allows QE widget to determine if the widget remains disabled due to invalid data. If disabled due to invalid data, a request to enable the widget will be honoured when the data is no longer invalid.

9.87.3 Property Documentation

9.87.3.1 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.

Reimplemented from [QEDragDrop](#).

9.87.3.2 bool QESpinBox::enabled [read, write]

Set the preferred 'enabled' state. Default is true. This property is copied to the standard Qt 'enabled' property if the data being displayed is valid. If the data being displayed is invalid the standard Qt 'enabled' property will always be set to false to indicate invalid data. The value of this property will only be copied to the standard Qt 'enabled' property once data is valid.

9.87.3.3 unsigned QSpinBox::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.

9.87.3.4 bool QSpinBox::subscribe [read, write]

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

Reimplemented from [QEWidget](#).

9.87.3.5 UserLevels QSpinBox::userLevelEnabled [read, write]

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

9.87.3.6 QString QSpinBox::userLevelEngineerStyle [read, write]

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

9.87.3.7 QString QSpinBox::userLevelScientistStyle [read, write]

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

9.87.3.8 QString QSpinBox::userLevelUserStyle [read, write]

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

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

9.87.3.9 UserLevels QESpinBox::userLevelVisibility [read, write]

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

9.87.3.10 QString QESpinBox::variable [read, write]

EPICS variable name (CA PV)

9.87.3.11 bool QESpinBox::variableAsToolTip [read, write]

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

Reimplemented from [QEToolTip](#).

9.87.3.12 QString QESpinBox::variableSubstitutions [read, write]

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

9.87.3.13 bool QESpinBox::visible [read, write]

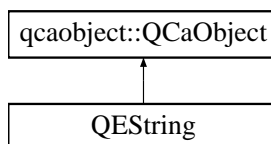
Display the widget. Default is true. Setting this property false is usefull if widget is only used to provide a signal - for example, when supplying data to a [QELink](#) widget. Note, when false the widget will still be visible in Qt Designer.

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

- /home/rhydera/epicsqt/framework/widgets/QESpinBox/QESpinBox.h
- /home/rhydera/epicsqt/framework/widgets/QESpinBox/QESpinBox.cpp

9.88 QEStrng Class Reference

Inheritance diagram for QEStrng:



Public Slots

- void **writeString** (const QString &data)

Signals

- void **stringConnectionChanged** ([QCaConnectionInfo](#) &connectionInfo, const unsigned int &variableIndex)
- void **stringChanged** (const QString &value, [QCaAlarmInfo](#) &alarmInfo, [QCaDateTime](#) &timeStamp, const unsigned int &variableIndex)

Public Member Functions

- **QEStrng** (QString recordName, QObject *eventObject, [QEStrngFormatting](#) *stringFormattingIn, unsigned int variableIndexIn)
- **QEStrng** (QString recordName, QObject *eventObject, [QEStrngFormatting](#) *stringFormattingIn, unsigned int variableIndexIn, [UserMessage](#) *userMessageIn)
- bool **writeString** (const QString &data, QString &message)

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

- /home/rhydera/epicsqt/framework/data/include/QEStrng.h
- /home/rhydera/epicsqt/framework/data/src/QEStrng.cpp

9.89 QEStrngFormatting Class Reference

Public Types

- enum [formats](#) {
[FORMAT_DEFAULT](#), [FORMAT_FLOATING](#), [FORMAT_INTEGER](#), [FORMAT_UNSIGNEDINTEGER](#),
[FORMAT_TIME](#), [FORMAT_LOCAL_ENUMERATE](#), [FORMAT_STRING](#) }

- enum `notations` { `NOTATION_FIXED` = QTextStream::FixedNotation, `NOTATION_SCIENTIFIC` = QTextStream::ScientificNotation, `NOTATION_AUTOMATIC` = QTextStream::SmartNotation }
- enum `arrayActions` { `APPEND`, `ASCII`, `INDEX` }

Public Member Functions

- QString **formatString** (const QVariant &value)
- QVariant **formatValue** (const QString &text, bool &ok)
- void **setDbEgu** (QString egu)
- void **setDbEnumerations** (QStringList enumerations)
- void **setDbPrecision** (unsigned int dbPrecisionIn)
- void **setDbVariableIsStatField** (bool isStatField)
- void **setPrecision** (int precision)
- void **setUseDbPrecision** (bool useDbPrecision)
- void **setLeadingZero** (bool leadingZero)
- void **setTrailingZeros** (bool trailingZeros)
- void **setFormat** (`formats` format)
- void **setRadix** (unsigned int radix)
- void **setNotation** (`notations` notation)
- void **setArrayAction** (`arrayActions` arrayActionIn)
- void **setArrayIndex** (unsigned int arrayIndexIn)
- void **setAddUnits** (bool addUnits)
- void **setLocalEnumeration** (QString localEnumerationIn)
- int **getPrecision** ()
- bool **getUseDbPrecision** ()
- bool **getLeadingZero** ()
- bool **getTrailingZeros** ()
- `formats` **getFormat** ()
- unsigned int **getRadix** ()
- `notations` **getNotation** ()
- `arrayActions` **getArrayAction** ()
- unsigned int **getArrayIndex** ()
- bool **getAddUnits** ()
- QString **getLocalEnumeration** ()

9.89.1 Member Enumeration Documentation

9.89.1.1 enum QStringFormatting::arrayActions

What action to take when formatting array data

Enumerator:

- APPEND** Interpret each element in the array as an unsigned integer and append string representations of each element from the array with a space in between each.

ASCII Interpret each element from the array as a character in a string. Translate all non printing characters to '?' except for trailing zeros (ignore them)

INDEX Interpret the element selected by `setArrayIndex()` as an unsigned integer.

9.89.1.2 enum `QStringFormatting::formats`

Formatting options

Enumerator:

FORMAT_DEFAULT Format according to the EPICS database record type.

FORMAT_FLOATING Format as a floating point number.

FORMAT_INTEGER Format as an integer.

FORMAT_UNSIGNEDINTEGER Format as an unsigned integer.

FORMAT_TIME Format as a time.

FORMAT_LOCAL_ENUMERATE Format as a selection from the local enumerations set by `setLocalEnumeration()`

FORMAT_STRING Format as a string.

9.89.1.3 enum `QStringFormatting::notations`

Notations when formatting a floating point number

Enumerator:

NOTATION_FIXED Standard floating point 123456.789.

NOTATION_SCIENTIFIC Scientific representation 1.23456789e6.

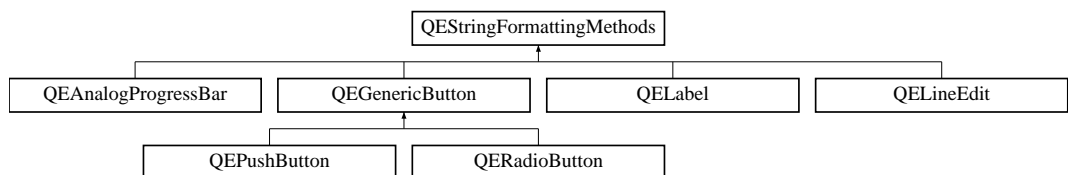
NOTATION_AUTOMATIC Automatic choice of standard or scientific notation.

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

- `/home/rhydera/epicsqt/framework/data/include/QStringFormatting.h`
- `/home/rhydera/epicsqt/framework/data/src/QStringFormatting.cpp`

9.90 `QStringFormattingMethods` Class Reference

Inheritance diagram for `QStringFormattingMethods`:



Public Member Functions

- virtual void **stringFormattingChange** ()=0
- void **setPrecision** (int precision)
- int **getPrecision** ()
- void **setUseDbPrecision** (bool useDbPrecision)
- bool **getUseDbPrecision** ()
- void **setLeadingZero** (bool leadingZero)
- bool **getLeadingZero** ()
- void **setTrailingZeros** (bool trailingZeros)
- bool **getTrailingZeros** ()
- void **setAddUnits** (bool addUnits)
- bool **getAddUnits** ()
- void **setLocalEnumeration** (QString localEnumeration)
- QString **getLocalEnumeration** ()
- void **setFormat** (QESTringFormatting::formats format)
- QESTringFormatting::formats **getFormat** ()
- void **setRadix** (unsigned int radix)
- unsigned int **getRadix** ()
- void **setNotation** (QESTringFormatting::notations notation)
- QESTringFormatting::notations **getNotation** ()
- void **setArrayAction** (QESTringFormatting::arrayActions arrayAction)
- QESTringFormatting::arrayActions **getArrayAction** ()
- void **setArrayIndex** (unsigned int arrayIndex)
- unsigned int **getArrayIndex** ()

Protected Attributes

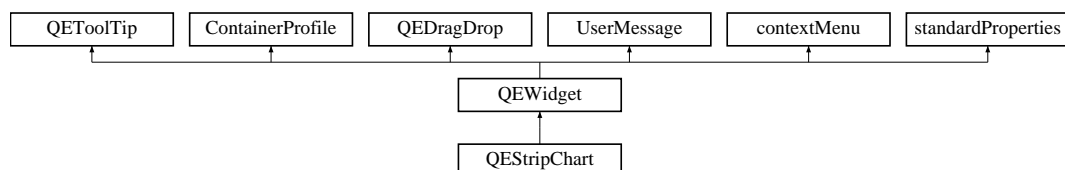
- QESTringFormatting **stringFormatting**

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

- /home/rhydera/epicsqt/framework/widgets/include/QESTringFormattingMethods.h
- /home/rhydera/epicsqt/framework/widgets/src/QESTringFormattingMethods.cpp

9.91 QESTripChart Class Reference

Inheritance diagram for QESTripChart:



Classes

- class [PrivateData](#)

Public Types

- enum **Constants** { **NUMBER_OF_PVS** = 12 }

Public Member Functions

- **QEStripChart** (QWidget *parent=0)
- QSize **sizeHint** () const
- QDateTime **getStartDateTime** ()
- QDateTime **getEndDateTime** ()
- void **setEndDateTime** (QDateTime endTimeIn)
- int **getDuration** ()
- void **setDuration** (int durationIn)
- double **getYMinimum** ()
- void **setYMinimum** (double yMinimumIn)
- double **getYMaximum** ()
- void **setYMaximum** (double yMaximumIn)
- void **plotData** ()

Protected Member Functions

- 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)
- void **setup** ()
- [qcaobject::QCaObject](#) * **createQcaltem** (unsigned int variableIndex)
- void **establishConnection** (unsigned int variableIndex)

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**
- 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 **PrivateData**
- class [QEStripChartItem](#)

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

- /home/rhydera/epicsqt/framework/widgets/QEStripChart/QEStripChart.h
- /home/rhydera/epicsqt/framework/widgets/QEStripChart/QEStripChart.cpp

9.92 QEStripChartItem Class Reference

Classes

- class [PrivateData](#)

Friends

- class [QEStripChart](#)

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

- /home/rhydera/epicsqt/framework/widgets/QEStripChart/QEStripChartItem.h
- /home/rhydera/epicsqt/framework/widgets/QEStripChart/QEStripChartItem.cpp

Public Member Functions

- **QESubstitutedLabel** (QWidget *parent=0)
- void **establishConnection** (unsigned int variableIndex)
- void **setLabelTextProperty** (QString labelTextIn)
- QString **getLabelTextProperty** ()
- QString **getLabelTextPropertyFormat** ()
- void **setLabelTextPropertyFormat** (QString labelTextIn)

Protected Attributes

- QString [labelText](#)

Properties

- QString [textSubstitutions](#)

9.95.1 Member Data Documentation

9.95.1.1 **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

9.95.2 Property Documentation

9.95.2.1 **QString QESubstitutedLabel::textSubstitutions** [read, write]

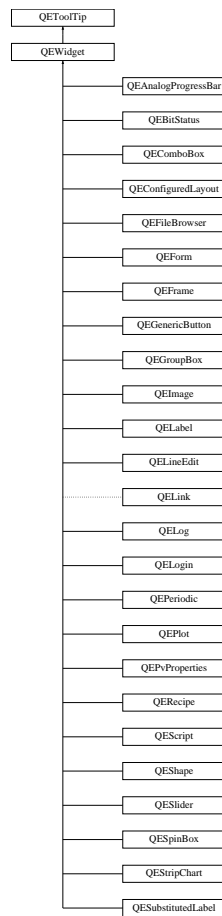
Text substitutions. These substitutions are applied to the 'labelText' property prior to copying it to the label text.

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

- /home/rhydera/epicsqt/framework/widgets/QESubstitutedLabel/QESubstitutedLabel.h
- /home/rhydera/epicsqt/framework/widgets/QESubstitutedLabel/QESubstitutedLabel.cpp

9.96 QEToolTip Class Reference

Inheritance diagram for QEToolTip:



Public Member Functions

- **QEToolTip** (QWidget *ownerIn)
- void **updateToolTipVariable** (const QString &variable)
- void **updateToolTipAlarm** (const QString &alarm)
- void **updateToolTipConnection** (bool connection)

Protected Member Functions

- void **setVariableAsToolTip** (bool variableAsToolTip)
- bool **getVariableAsToolTip** ()

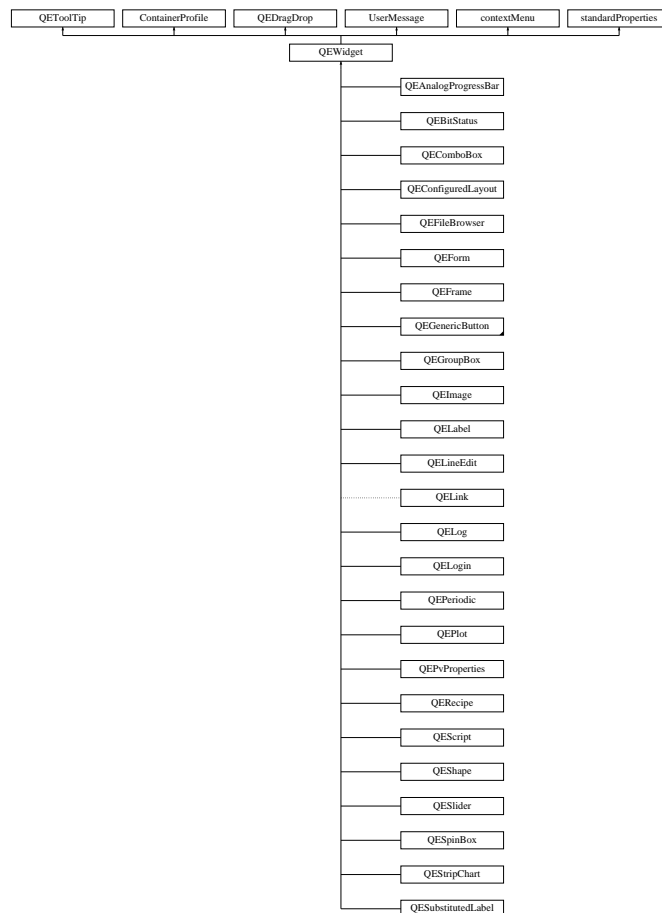
The documentation for this class was generated from the following files:

- /home/rhydera/epicsqt/framework/widgets/include/QEToolTip.h
- /home/rhydera/epicsqt/framework/widgets/src/QEToolTip.cpp

9.97 QEWidget Class Reference

```
#include <QEWidget.h>
```

Inheritance diagram for QEWidget:



Public Member Functions

- [QEWidget](#) (QWidget *ownerIn)
Constructor.
- virtual [~QEWidget](#) ()
Destructor.
- void [activate](#) ()
- unsigned int [getMessageSourceId](#) ()
- void [setMessageSourceId](#) (unsigned int messageSourceId)
- [qcaobject::QCaObject](#) * [getQcaItem](#) (unsigned int variableIndex)
- void [setupContextMenu](#) (QWidget *w)
- QColor [getColor](#) ([QCaAlarmInfo](#) &alarmInfo, const int saturation)

- void [readNow](#) ()
- virtual void [writeNow](#) ()
- virtual void [setVariableNameAndSubstitutions](#) (QString variableNameIn, QString variableNameSubstitutionsIn, unsigned int variableIndex)
- QFile * [openQEFile](#) (QString name, QFile::OpenModeFlag mode)
- QString [defaultFileLocation](#) ()

Static Public Member Functions

- static bool [inDesigner](#) ()

Protected Member Functions

- void [setNumVariables](#) (unsigned int numVariablesIn)
- [qcaobject::QCaObject](#) * [createConnection](#) (unsigned int variableIndex)
- virtual [qcaobject::QCaObject](#) * [createQcaltem](#) (unsigned int variableIndex)
- virtual void [establishConnection](#) (unsigned int variableIndex)

Protected Attributes

- bool [subscribe](#)

9.97.1 Detailed Description

This class is used as a base for all CA aware widgets, such as [QELabel](#), [QESpinBox](#), etc. It manages common issues including creating a source of CA data updates, handling error, warning and status messages, and setting tool tips based on variable names.

Note, there is tight integration between the CA aware widget classes, this class, and its base classes, especially [VariableNameManager](#) and [QEToolTip](#).

In particular, this class manages [QCaObject](#) classes that stream updates to the CA aware widget class. But this class, however, doesn't know how to format the data, or how the updates will be used. To resolve this, this class asks its parent class (such as [QELabel](#)) to create the [QCaObject](#) class in what ever flavour it wants, by calling the virtual function [createQcaltem](#). A [QELabel](#), for example, wants string updates so it creates a [QEString](#) which is based on a [QCaObject](#) class and formats all updates as strings.

The CA aware parent class (such as [QELabel](#)) defines a variable by calling [VariableNameManager::setVariableName\(\)](#). The [VariableNamePropertyManager](#) class calls the [establishConnection](#) function of the CA aware parent class, such as [QELabel](#) when it has a new variable name.

This class uses its base [QEToolTip](#) class to format tool tips. that class in turn calls the CA aware parent class (such as [QELabel](#)) directly to make use of a new tool tip.

After construction, a CA aware widget is activated (starts updating) by calling it's [establishConnection\(\)](#) function in one of two ways:

1) The variable name or variable name substitutions is changed by calling `setVariableName` or `setVariableNameSubstitutions` respectively. These functions are in the `VariableNameManager` class. The `VariableNamePropertyManager` calls a virtual function `establishConnection()` which is implemented by the CA aware widget. This is how a CA aware widget is activated in 'designer'. It occurs when 'designer' updates the variable name property or variable name substitution property.

2) When an `QEForm` widget is created, resulting in a set of CA aware widgets being created by loading a UI file containing plugin definitions. After loading the plugin widgets, code in the `QEForm` class calls the `activate()` function in this class (`QEWidget`). the `activate()` function calls `establishConnection()` in the CA aware widget for each variable. This simulates what the `VariableNamePropertyManager` does as each variable name is entered (see 1, above, for details)

No matter which way a CA aware widget is activated, the `establishConnection()` function in the CA aware widget is called for each variable. The `establishConnection()` function asks this `QEWidget` base class, by calling the `createConnection()` function, to perform the tasks common to all CA aware widgets for establishing a stream of CA data.

The `createConnection()` function sets up the widget 'tool tip', then immediately calls the CA aware widget back asking it to create an object based on `QCaObject`. This object will supply a stream of CA update signals to the CA aware object in a form that it needs. For example a `QELabel` creates a `QEString` object. The `QEString` class is based on the `QCaObject` class and converts all update data to a strings which is required for updating a Qt label widget. This class stores the `QCaObject` based class.

After the `establishConnection()` function in the CA aware widget has called `createConnection()`, the remaining task of the `establishConnection()` function is to connect the signals of the newly created `QCaObject` based classes to its own slots so that data updates can be used. For example, a `QELabel` connects the 'stringChanged' signal from the `QEString` object to its `setLabelText` slot.

9.97.2 Member Function Documentation

9.97.2.1 void QEWidget::activate ()

Initiate updates. Called after all configuration is complete.

9.97.2.2 QString QEWidget::defaultFileLocation ()

Returns the default location to create files. Use this to create files in a consistent location

9.97.2.3 QColor QEWidget::getColor (QCaAlarmInfo & alarmInfo, const int saturation)

Return a colour to update the widget's look to reflect the current alarm state. Note, the color is determined by the `alarmInfo` class, but since that class is used in non gui applications, it can't return a `QColor`

9.97.2.4 `unsigned int QEWidget::getMessageSourceId () [inline]`

Get the message source ID. The message source ID is used as part of the system where QE widgets can emit a message and have the right QE widget in the right form catch the message. Refer to the [UserMessage](#) class for further details.

9.97.2.5 `qcaobject::QCaObject * QEWidget::getQcaltem (unsigned int variableIndex)`

Return a reference to one of the qCaObjects used to stream CA updates

9.97.2.6 `QFile * QEWidget::openQEFile (QString name, QFile::OpenModeFlag mode)`

Looks for a file in a standard set of locations (and opens the file)

9.97.2.7 `void QEWidget::readNow ()`

Perform a single shot read on all variables (Usefull when not subscribing by default)

9.97.2.8 `void QEWidget::setMessageSourceId (unsigned int messageSourceId) [inline]`

Set the message source ID. The message source ID is used as part of the system where QE widgets can emit a message and have the right QE widget in the right form catch the message. Refer to the [UserMessage](#) class for further details.

9.97.2.9 `void QEWidget::setupContextMenu (QWidget * w)`

Take a menu widgt and add it as the context menu for this widget

9.97.2.10 `void QEWidget::setVariableNameAndSubstitutions (QString variableNameIn, QString variableNameSubstitutionsIn, unsigned int variableIndex) [virtual]`

Virtual function that may be implimented by users of [QEWidget](#) to update variable names and macro substitutions. A default is provided that is suitable in most cases.

Reimplemented in [QEAnalogProgressBar](#), [QEBitStatus](#), and [QEForm](#).

9.97.2.11 `virtual void QEWidget::writeNow () [inline, virtual]`

(Control widgets only - such as [QELineEdit](#)) Write the value now. Used when writeOnChange, writeOnEnter, etc are all false

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

- `/home/rhydera/epicsqt/framework/widgets/include/QEWidget.h`

- /home/rhydera/epicsqt/framework/widgets/src/QEWidget.cpp

9.98 QEWidgets Class Reference

Public Member Functions

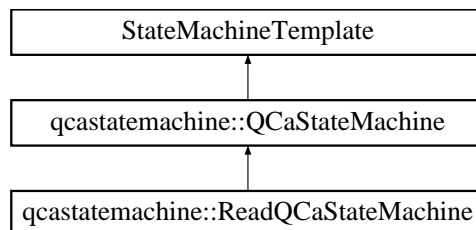
- **QEWidgets** (QObject *parent=0)
- virtual QList< QDesignerCustomWidgetInterface * > **customWidgets** () const

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

- /home/rhydera/epicsqt/framework/widgets/include/QEDesignerPlugin.h
- /home/rhydera/epicsqt/framework/widgets/src/QEDesignerPlugin.cpp

9.99 qcastatemachine::ReadQCaStateMachine Class Reference

Inheritance diagram for qcastatemachine::ReadQCaStateMachine:



Public Member Functions

- **ReadQCaStateMachine** (void *parent)
- bool **process** (int requestedState)

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

- /home/rhydera/epicsqt/framework/data/include/QCaStateMachine.h
- /home/rhydera/epicsqt/framework/data/src/QCaStateMachine.cpp

9.100 RecordSpec Class Reference

Public Member Functions

- **RecordSpec** (const QString theRecordType)

- QString **getRecordType** ()
- QString **getFieldName** (const int index)

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

- /home/rhydera/epicsqt/framework/widgets/QEPvProperties/QEPvProperties.cpp

9.101 RecordSpecList Class Reference

Public Member Functions

- [RecordSpec](#) * **find** (const QString recordType)
- void **appendOrReplace** ([RecordSpec](#) *pRecordSpec)

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

- /home/rhydera/epicsqt/framework/widgets/QEPvProperties/QEPvProperties.cpp

9.102 selectMenu Class Reference

Public Member Functions

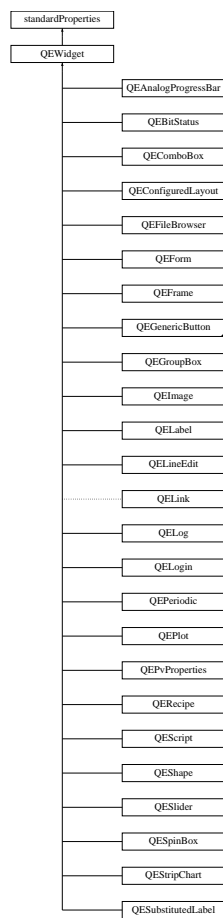
- **selectMenu** (QWidget *parent=0)
- [imageContextMenu::imageContextMenuOptions](#) **getSelectOption** (const QPoint &pos)
- void **setChecked** (const int mode)
- void **setPanEnabled** (bool enablePan)
- void **setVSliceEnabled** (bool enableVSliceSelection)
- void **setHSliceEnabled** (bool enableHSliceSelection)
- void **setAreaEnabled** (bool enableAreaSelection)
- void **setProfileEnabled** (bool enableProfileSelection)
- void **setTargetEnabled** (bool enableTargetSelection)

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

- /home/rhydera/epicsqt/framework/widgets/QEImage/selectMenu.h
- /home/rhydera/epicsqt/framework/widgets/QEImage/selectMenu.cpp

9.103 standardProperties Class Reference

Inheritance diagram for standardProperties:



Public Member Functions

- **standardProperties** (QWidget *ownerIn)

Protected Member Functions

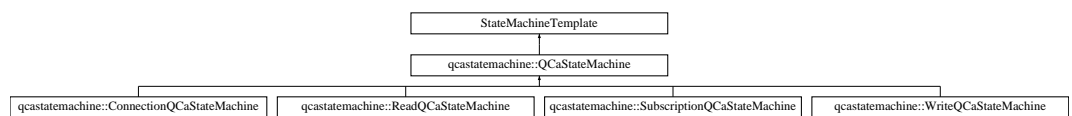
- userLevels **getUserLevelVisibility** ()
- void **setUserLevelVisibility** (userLevels level)
- userLevels **getUserLevelEnabled** ()
- void **setUserLevelEnabled** (userLevels level)
- bool **getApplicationEnabled** () const
- void **setApplicationEnabled** (bool state)
- void **setDataDisabled** (bool disable)
- void **setRunVisible** (bool visibleIn)
- bool **getRunVisible** ()
- void **checkVisibilityEnabledLevel** (userLevels level)

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

- /home/rhydera/epicsqt/framework/widgets/include/standardProperties.h
- /home/rhydera/epicsqt/framework/widgets/src/standardProperties.cpp

9.104 StateMachineTemplate Class Reference

Inheritance diagram for StateMachineTemplate:



Public Member Functions

- virtual bool **process** (int requestedState)=0

Public Attributes

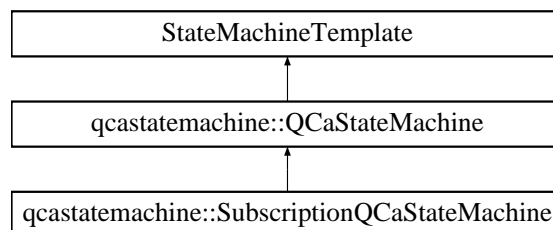
- int **currentState**
- int **requestState**

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

- /home/rhydera/epicsqt/framework/data/include/QCaStateMachine.h

9.105 qcastatemachine::SubscriptionQCaStateMachine Class Reference

Inheritance diagram for qcastatemachine::SubscriptionQCaStateMachine:



Public Member Functions

- **SubscriptionQCaStateMachine** (void *parent)
- bool **process** (int requestedState)

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

- /home/rhydera/epicsqt/framework/data/include/QCaStateMachine.h
- /home/rhydera/epicsqt/framework/data/src/QCaStateMachine.cpp

9.106 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:

- /home/rhydera/epicsqt/framework/widgets/QEPlot/QEPlot.h

9.107 TrackRange Class Reference

Public Member Functions

- void **clear** ()
- void **merge** (const double d)
- void **merge** (const [TrackRange](#) that)
- bool **getMinMax** (double &min, double &max)

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

- /home/rhydera/epicsqt/framework/widgets/QEStripChart/QEStripChartItem.h
- /home/rhydera/epicsqt/framework/widgets/QEStripChart/QEStripChartItem.cpp

9.108 userInfoStruct Class Reference

Public Attributes

- bool **enable**
- double **value1**
- double **value2**
- QString **elementText**

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

- /home/rhydera/epicsqt/framework/widgets/QEPeriodic/QEPeriodic.h

9.109 QEPeriodic::userInfoStructArray Struct Reference

Public Attributes

- [userInfoStruct](#) **array** [NUM_ELEMENTS]

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

- /home/rhydera/epicsqt/framework/widgets/QEPeriodic/QEPeriodic.h

9.110 userLevelSignal Class Reference

Signals

- void [userChanged](#) (userLevels level)
Internal use only. Send when the user level has changed.

Public Member Functions

- void **setLevel** (userLevels levelIn)
- userLevels **getLevel** ()

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

- /home/rhydera/epicsqt/framework/widgets/include/ContainerProfile.h
- /home/rhydera/epicsqt/framework/widgets/src/ContainerProfile.cpp

9.111 `userLevelSlot` Class Reference

Public Slots

- void **userChanged** (`userLevels` level)

Public Member Functions

- void **setOwner** ([ContainerProfile](#) *ownerIn)

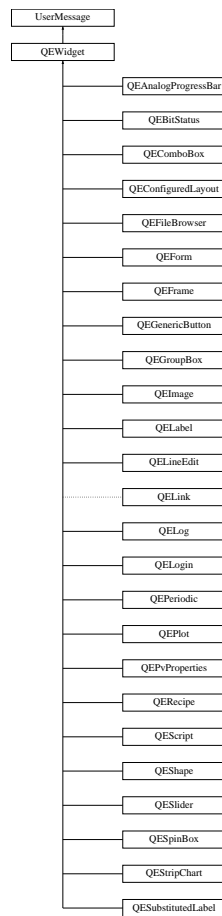
The documentation for this class was generated from the following files:

- `/home/rhydera/epicsqt/framework/widgets/include/ContainerProfile.h`
- `/home/rhydera/epicsqt/framework/widgets/src/ContainerProfile.cpp`

9.112 `UserMessage` Class Reference

```
#include <UserMessage.h>
```

Inheritance diagram for `UserMessage`:



Public Types

- enum **message_filter_options** { MESSAGE_FILTER_ANY, MESSAGE_FILTER_MATCH, MESSAGE_FILTER_NONE }

Public Member Functions

- void **setSourceId** (unsigned int sourceId)
Set the source ID (the ID set up by the GUI designer, usually matched to the source ID of logging widgets)
- void **setFormId** (unsigned int formId)
Set the form ID (the the same ID for all sibling widgets within an [QForm](#) widget)
- void **setFormFilter** (message_filter_options formFilterIn)
Set the message filtering applied to the form ID.
- void **setSourceFilter** (message_filter_options sourceFilterIn)
Set the message filtering applied to the source ID.

- unsigned int [getSourceId](#) ()
Get the source ID (the ID set up by the GUI designer, usually matched to the source ID of logging widgets).
- unsigned int [getFormId](#) ()
Get the form ID (the the same ID for all sibling widgets within an [QForm](#) widget)
- message_filter_options [getFormFilter](#) ()
Get the message filtering applied to the form ID.
- message_filter_options [getSourceFilter](#) ()
Get the message filtering applied to the source ID.
- void [setChildFormId](#) (unsigned int)
Set the for ID of all widgets that are children of this widget.
- unsigned int [getChildFormId](#) ()
Get the for ID of all widgets that are children of this widget.
- unsigned int [getNextMessageFormId](#) ()
Generate a new form ID for all widgets in a new form.
- void [sendMessage](#) (QString message, message_types type=MESSAGE_TYPE_INFO)
Send a message to the user.
- void [sendMessage](#) (QString message, QString source, message_types type=MESSAGE_TYPE_INFO)
Send a message to the user with a source reference.
- QString [getMessageTypeName](#) (message_types type)
Convenience function to provide string names for each message type.
- virtual void [newMessage](#) (QString, message_types)
Virtual function to pass messages to derived classes (typically logging widgets or application windows)

Friends

- class [UserMessageSlot](#)
- class [UserMessageSignal](#)

9.112.1 Detailed Description

A class to manage user messages.

This class passes messages between widgets and application code

This class is used as a base class.

Messages are sent by calling [sendMessage\(\)](#) Messages are received by implementing [newMessage\(\)](#) in the derived class.

Messages can be filtered based on a source ID or a form ID

The derived widget is free to set the source ID to any value

Derived form widgets ([QEForm](#)) get a unique form ID using [getNextMessageFormId\(\)](#) (as well as being able to set a source ID like any other QE widget) and pass this unique form ID to all widgets within the form using the [ContainerProfile](#) class.

Messages sent by a QE widget are received by all QE widgets and can filter the messages required by form ID and source ID. The form ID is under the management of the [QEForm](#) widget, the source ID is under the control of the GUI designer.

The [QEForm](#) widget does not display messages, but re-send them using its own form ID. Read on to see how this can be used.

Widgets that generate messages, and widgets (or application code) that use messages can be set up as follows:

- Application wide logging: An application with a single log window can base a class on the [UserMessage](#) class and set up filtering to receive all messages. An application with log messages for separate windows containing [QEForm](#) widgets (such as QEGui) can base each window class on the [UserMessage](#) class, then set up filtering for the appropriate form ID.
- Logging within a [QEForm](#). A logging widget can be set to filter matching on the current form and so will pick up messages from any sibling widget. This includes messages from a sibling widget which is a nested [QEForm](#). Whatever messages that nested form is set to receive, it will resend to its siblings. For example, if it is set to receive messages from the widgets it contains, these are resent up one level to the main form. If messages are dealt with within the nested [QEForm](#) (for example, it may have its own logging QE widget) then the nested [QEForm](#) could be set up not to filter and resend any messages.

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

- /home/rhydera/epicsqt/framework/widgets/include/UserMessage.h
- /home/rhydera/epicsqt/framework/widgets/src/UserMessage.cpp

9.113 UserMessageSignal Class Reference

```
#include <UserMessage.h>
```

Signals

- void [message](#) (QString msg, message_types type, unsigned int formId, unsigned int sourceId, [UserMessage](#) *originator)

Emit a message signal. Any widget based on the [UserMessage](#) class can receive these messages, filtered on formId and sourceId.

Public Member Functions

- void [sendMessage](#) (QString msg, message_types type, unsigned int formId, unsigned int sourceId, [UserMessage](#) *originator)

Send a message to all widgets based on the [UserMessage](#) class.

9.113.1 Detailed Description

Class used to send message signals. Used only within UserMessage.cpp A single instance of this class is shared by all instances of the [UserMessage](#) class. This allows every [UserMessage](#) class instance to connect to a single source of messages

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

- /home/rhydera/epicsqt/framework/widgets/include/UserMessage.h
- /home/rhydera/epicsqt/framework/widgets/src/UserMessage.cpp

9.114 UserMessageSlot Class Reference

```
#include <UserMessage.h>
```

Public Slots

- void [message](#) (QString msg, message_types type, unsigned int formId, unsigned int sourceId, [UserMessage](#) *originator)

A message has been received.

Public Member Functions

- void [setOwner](#) ([UserMessage](#) *ownerIn)

Set the [UserMessage](#) class this is a part of.

9.114.1 Detailed Description

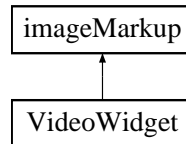
Class used to receive message signals. Used only within UserMessage.cpp An instance of this class is created by all instances of the [UserMessage](#) class. The [UserMessage](#) class uses an instance of this class to receive messages so it does not have to be based on QObject itself. This is required as derived classes generally need to be also based on another object derived from QObject (and QObject can only be the base of a single base class)

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

- /home/rhydera/epicsqt/framework/widgets/include/UserMessage.h
- /home/rhydera/epicsqt/framework/widgets/src/UserMessage.cpp

9.115 VideoWidget Class Reference

Inheritance diagram for VideoWidget:



Signals

- void **userSelection** (imageMarkup::markupIds mode, bool clearing, QPoint point1, QPoint point2)
- void **zoomInOut** (int zoomAmount)
- void **currentPixelInfo** (QPoint pos)
- void **pan** (QPoint pos)

Public Member Functions

- **VideoWidget** (QWidget *parent=0)
- void **setNewImage** (const QImage image, [QCaDateTime](#) &time)
- void **setPanning** (bool panningIn)
- bool **getPanning** ()
- QPoint **scalePoint** (QPoint pnt)
- int **scaleOrdinate** (int ord)
- QImage **getImage** ()

Protected Member Functions

- void **paintEvent** (QPaintEvent *)
- void **mousePressEvent** (QMouseEvent *event)
- void **mouseReleaseEvent** (QMouseEvent *event)
- void **mouseMoveEvent** (QMouseEvent *event)
- void **wheelEvent** (QWheelEvent *event)
- void **markupChange** (QImage &markups, QVector< QRect > &changedAreas)
- void **resizeEvent** (QResizeEvent *event)
- void **markupSetCursor** (QCursor cursor)
- void **markupAction** (markupIds mode, bool clearing, QPoint point1, QPoint point2)

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

- /home/rhydera/epicsqt/framework/widgets/QEImage/videowidget.h
- /home/rhydera/epicsqt/framework/widgets/QEImage/videowidget.cpp

9.116 WidgetRef Class Reference

Public Member Functions

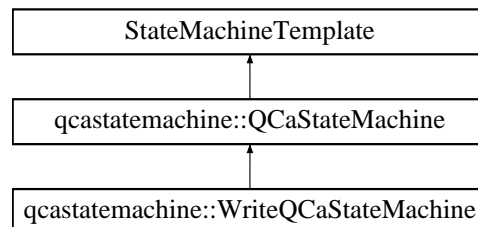
- **WidgetRef** ([QEWidget](#) *refIn)
- [QEWidget](#) * **getRef** ()

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

- /home/rhydera/epicsqt/framework/widgets/include/ContainerProfile.h

9.117 qcastatemachine::WriteQCaStateMachine Class Reference

Inheritance diagram for qcastatemachine::WriteQCaStateMachine:



Public Member Functions

- **WriteQCaStateMachine** (void *parent)
- bool **process** (int requestedState)

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

- /home/rhydera/epicsqt/framework/data/include/QCaStateMachine.h
- /home/rhydera/epicsqt/framework/data/src/QCaStateMachine.cpp

9.118 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:

- /home/rhydera/epicsqt/framework/widgets/QEImage/zoomMenu.h
- /home/rhydera/epicsqt/framework/widgets/QEImage/zoomMenu.cpp

Index

- [_Field, 27](#)
 - [_Item, 28](#)
 - [_QDialogItem, 28](#)
 - [_QDialogLogin, 28](#)
 - [_QPushButtonGroup, 29](#)
 - [_QTableWidgetFileBrowser, 29](#)
 - [_QTableWidgetLog, 30](#)
 - [_QTableWidgetScript, 30](#)
- activate
 - [QEWidget, 205](#)
- addUnits
 - [QEAnalogProgressBar, 66](#)
 - [QELabel, 114](#)
 - [QELineEdit, 124](#)
- alarmSeverityDisplayMode
 - [QEAnalogProgressBar, 66](#)
- allowDrop
 - [QEAnalogProgressBar, 66](#)
 - [QEBitStatus, 73](#)
 - [QEComboBox, 78](#)
 - [QEFrame, 90](#)
 - [QEGroupBox, 96](#)
 - [QEImage, 103](#)
 - [QELabel, 114](#)
 - [QELineEdit, 124](#)
 - [QEPeriodic, 138](#)
 - [QEPlot, 147](#)
 - [QEPushButton, 153](#)
 - [QEPvProperties, 158](#)
 - [QERadioButton, 164](#)
 - [QEShape, 176](#)
 - [QESlider, 186](#)
 - [QESpinBox, 191](#)
- altReadbackVariable
 - [QEPushButton, 153](#)
- animation1
 - [QEShape, 176](#)
- animation2
 - [QEShape, 176](#)
- animation3

- [QEShape, 177](#)
- animation4
 - [QEShape, 177](#)
- animation5
 - [QEShape, 177](#)
- animation6
 - [QEShape, 177](#)
- animationOptions
 - [QEShape, 174](#)
- APPEND
 - [QEStringFormatting, 195](#)
- Append
 - [QEAnalogProgressBar, 64](#)
 - [QELabel, 113](#)
 - [QELineEdit, 122](#)
- arrayAction
 - [QEAnalogProgressBar, 67](#)
 - [QELabel, 115](#)
 - [QELineEdit, 125](#)
- ArrayActions
 - [QEAnalogProgressBar, 64](#)
 - [QELabel, 113](#)
 - [QELineEdit, 122](#)
- arrayActions
 - [QEStringFormatting, 195](#)
- ASCII
 - [QEStringFormatting, 195](#)
- Ascii
 - [QEAnalogProgressBar, 64](#)
 - [QELabel, 113](#)
 - [QELineEdit, 122](#)
- Automatic
 - [QEAnalogProgressBar, 65](#)
 - [QELabel, 113](#)
 - [QELineEdit, 122](#)
- backgroundColour
 - [QEAnalogIndicator, 60](#)
- Bar
 - [QEAnalogIndicator, 59](#)
- beamXVariable

- QEImage, 103
- beamYVariable
 - QEImage, 103
- borderColour
 - QEAnalogIndicator, 60
- Bottom_To_Top
 - QEAnalogIndicator, 59
- centreAngle
 - QEAnalogIndicator, 60
- clippingHighVariable
 - QEImage, 103
- clippingLowVariable
 - QEImage, 103
- clippingOnOffVariable
 - QEImage, 103
- color1
 - QEShape, 177
- color10
 - QEShape, 177
- color2
 - QEShape, 177
- color3
 - QEShape, 177
- color4
 - QEShape, 177
- color5
 - QEShape, 178
- color6
 - QEShape, 178
- color7
 - QEShape, 178
- color8
 - QEShape, 178
- color9
 - QEShape, 178
- confirmWrite
 - QELineEdit, 125
- ContainerProfile, 31
- contextMenu, 33
- contextMenuObject, 35
- dbElementChanged
 - QEPeriodic, 138
- dbValueChanged
 - QEAnalogProgressBar, 66
 - QEBitStatus, 72
 - QEComboBox, 78
 - QEImage, 103
 - QELabel, 114
- QELineEdit, 123
- QEPeriodic, 138
- QEPlot, 147
- QEPushButton, 153
- QERadioButton, 164
- QESlider, 186
- QESpinBox, 191
- dbValueChanged1
 - QEShape, 175
- dbValueChanged2
 - QEShape, 175
- dbValueChanged3
 - QEShape, 175
- dbValueChanged4
 - QEShape, 176
- dbValueChanged5
 - QEShape, 176
- dbValueChanged6
 - QEShape, 176
- Default
 - QEAnalogProgressBar, 65
 - QELabel, 113
 - QELineEdit, 122
- defaultFileLocation
 - QEWWidget, 205
- enabled
 - QEAnalogProgressBar, 67
 - QEBitStatus, 73
 - QEComboBox, 78
 - QEFrame, 91
 - QEGroupBox, 96
 - QEImage, 104
 - QELabel, 115
 - QELineEdit, 125
 - QEPeriodic, 138
 - QEPlot, 147
 - QEPushButton, 153
 - QEPvProperties, 158
 - QERadioButton, 164
 - QEShape, 178
 - QESlider, 186
 - QESpinBox, 191
- Engineer
 - QEAnalogProgressBar, 65
 - QEBitStatus, 72
 - QEComboBox, 78
 - QEFrame, 90
 - QEGroupBox, 95
 - QEImage, 102

- QELabel, 114
- QELineEdit, 123
- QEPeiodic, 138
- QEPlot, 147
- QEPushButton, 153
- QEPvProperties, 157
- QERadioButton, 163
- QEShape, 175
- QESlider, 185
- QESpinBox, 191
- Fixed
 - QEAnalogProgressBar, 65
 - QELabel, 113
 - QELineEdit, 122
- flipRotateMenu, 36
- Floating
 - QEAnalogProgressBar, 65
 - QELabel, 113
 - QELineEdit, 122
- fontColour
 - QEAnalogIndicator, 60
- foregroundColour
 - QEAnalogIndicator, 60
- format
 - QEAnalogProgressBar, 67
 - QELabel, 115
 - QELineEdit, 125
- FORMAT_DEFAULT
 - QStringFormatting, 196
- FORMAT_FLOATING
 - QStringFormatting, 196
- FORMAT_INTEGER
 - QStringFormatting, 196
- FORMAT_LOCAL_ENUMERATE
 - QStringFormatting, 196
- FORMAT_STRING
 - QStringFormatting, 196
- FORMAT_TIME
 - QStringFormatting, 196
- FORMAT_UNSIGNEDINTEGER
 - QStringFormatting, 196
- formatInteger
 - QEIntegerFormatting, 108
- formatIntegerArray
 - QEIntegerFormatting, 108
- Formats
 - QEAnalogProgressBar, 64
 - QELabel, 113
 - QELineEdit, 122
- formats
 - QStringFormatting, 196
- formatValue
 - QEIntegerFormatting, 108
- getColor
 - QEWidget, 205
- getConfirmWrite
 - QELineEdit, 123
- getMessageSourceId
 - QEWidget, 205
- getQcaltm
 - QEWidget, 206
- getSubscribe
 - QELineEdit, 123
- getWriteOnEnter
 - QELineEdit, 123
- getWriteOnFinish
 - QELineEdit, 123
- getWriteOnLoseFocus
 - QELineEdit, 123
- heightVariable
 - QEImage, 104
- imageContextMenu, 36
- imageMarkup, 37
- imageVariable
 - QEImage, 104
- INDEX
 - QStringFormatting, 196
- Index
 - QEAnalogProgressBar, 64
 - QELabel, 113
 - QELineEdit, 122
- int
 - QEAnalogProgressBar, 67
 - QEBitStatus, 73
 - QEComboBox, 79
 - QEFrame, 91
 - QEGroupBox, 96
 - QEImage, 104
 - QELabel, 115
 - QELineEdit, 125
 - QEPeiodic, 139
 - QEPlot, 147
 - QEPushButton, 154
 - QEPvProperties, 158
 - QERadioButton, 164
 - QEShape, 178

- QESlider, 186
- QESpinBox, 191
- Integer
 - QEAnalogProgressBar, 65
 - QELabel, 113
 - QELineEdit, 122
- labelText
 - QESubstitutedLabel, 201
- leadingZero
 - QEAnalogProgressBar, 67
 - QELabel, 115
 - QELineEdit, 126
- Left_To_Right
 - QEAnalogIndicator, 59
- LocalEnumeration
 - QEAnalogProgressBar, 65
 - QELabel, 113
 - QELineEdit, 122
- localEnumeration
 - QEAnalogProgressBar, 68
 - QELabel, 116
 - QELineEdit, 126
- localEnumerationItem, 38
- logScale
 - QEAnalogIndicator, 60
- logScaleInterval
 - QEAnalogIndicator, 60
- majorInterval
 - QEAnalogIndicator, 60
- managePixmap, 39
- markupBeam, 39
- markupHLine, 40
- markupItem, 41
- markupLine, 42
- markupRegion, 43
- markupTarget, 44
- markupText, 44
- markupVLine, 45
- maximum
 - QEAnalogIndicator, 60
- Meter
 - QEAnalogIndicator, 59
- minimum
 - QEAnalogIndicator, 60
- minorInterval
 - QEAnalogIndicator, 60
- mode
 - QEAnalogIndicator, 61
- Modes
 - QEAnalogIndicator, 59
- notation
 - QEAnalogProgressBar, 68
 - QELabel, 116
 - QELineEdit, 126
- NOTATION_AUTOMATIC
 - QEStringFormatting, 196
- NOTATION_FIXED
 - QEStringFormatting, 196
- NOTATION_SCIENTIFIC
 - QEStringFormatting, 196
- Notations
 - QEAnalogProgressBar, 65
 - QELabel, 113
 - QELineEdit, 122
- notations
 - QEStringFormatting, 196
- offset1
 - QEShape, 179
- offset2
 - QEShape, 179
- offset3
 - QEShape, 179
- offset4
 - QEShape, 179
- offset5
 - QEShape, 179
- offset6
 - QEShape, 179
- openQEFile
 - QEWWidget, 206
- orientation
 - QEAnalogIndicator, 61
- Orientations
 - QEAnalogIndicator, 59
- PeriodicDialog, 46
- PeriodicElementSetupForm, 46
- PeriodicSetupDialog, 46
- pixmap0
 - QELabel, 116
- pixmap1
 - QELabel, 116
- pixmap2
 - QELabel, 116
- pixmap3
 - QELabel, 116

- QPixmap4
 - QELabel, [116](#)
 - QPixmap5
 - QELabel, [116](#)
 - QPixmap6
 - QELabel, [116](#)
 - QPixmap7
 - QELabel, [117](#)
 - point1
 - QEShape, [179](#)
 - point10
 - QEShape, [179](#)
 - point2
 - QEShape, [179](#)
 - point3
 - QEShape, [180](#)
 - point4
 - QEShape, [180](#)
 - point5
 - QEShape, [180](#)
 - point6
 - QEShape, [180](#)
 - point7
 - QEShape, [180](#)
 - point8
 - QEShape, [180](#)
 - point9
 - QEShape, [180](#)
 - precision
 - QEAnalogProgressBar, [68](#)
 - QELabel, [117](#)
 - QELineEdit, [126](#)
 - profilePlot, [47](#)
 - QPushButtonSpecifications, [48](#)
 - QBitStatus, [48](#)
 - QCaAlarmInfo, [50](#)
 - QCaConnectionInfo, [50](#)
 - QCaDataPoint, [51](#)
 - QCaDataPointList, [51](#)
 - QCaDateTime, [51](#)
 - QCaEventFilter, [52](#)
 - QCaEventItem, [52](#)
 - QCaEventUpdate, [52](#)
 - QCaInstalledFiltersListItem, [53](#)
 - qcaobject::QCaObject, [53](#)
 - qcastatemachine::ConnectionQCaStateMachine, [31](#)
 - qcastatemachine::QCaStateMachine, [55](#)
 - qcastatemachine::ReadQCaStateMachine, [207](#)
 - qcastatemachine::SubscriptionQCaStateMachine, [210](#)
 - qcastatemachine::WriteQCaStateMachine, [219](#)
 - QCaVariableNamePropertyManager, [55](#)
 - QEAnalogIndicator, [56](#)
 - backgroundColour, [60](#)
 - Bar, [59](#)
 - borderColour, [60](#)
 - Bottom_To_Top, [59](#)
 - centreAngle, [60](#)
 - fontColour, [60](#)
 - foregroundColour, [60](#)
 - Left_To_Right, [59](#)
 - logScale, [60](#)
 - logScaleInterval, [60](#)
 - majorInterval, [60](#)
 - maximum, [60](#)
 - Meter, [59](#)
 - minimum, [60](#)
 - minorInterval, [60](#)
 - mode, [61](#)
 - Modes, [59](#)
 - orientation, [61](#)
 - Orientations, [59](#)
 - Right_To_Left, [59](#)
 - Scale, [59](#)
 - showScale, [61](#)
 - showText, [61](#)
 - spanAngle, [61](#)
 - Top_To_Bottom, [59](#)
 - value, [61](#)
 - QEAnalogIndicator::Band, [30](#)
 - QEAnalogIndicator::BandList, [31](#)
 - QEAnalogProgressBar, [61](#)
 - addUnits, [66](#)
 - alarmSeverityDisplayMode, [66](#)
 - allowDrop, [66](#)
 - Append, [64](#)
 - arrayAction, [67](#)
 - ArrayActions, [64](#)
 - Ascii, [64](#)
 - Automatic, [65](#)
 - dbValueChanged, [66](#)
 - Default, [65](#)
 - Engineer, [65](#)
 - Fixed, [65](#)

- Floating, [65](#)
- format, [67](#)
- Formats, [64](#)
- Index, [64](#)
- int, [67](#)
- Integer, [65](#)
- leadingZero, [67](#)
- LocalEnumeration, [65](#)
- localEnumeration, [68](#)
- notation, [68](#)
- Notations, [65](#)
- precision, [68](#)
- QEAAnalogProgressBar, [65](#)
- requestEnabled, [66](#)
- Scientific, [65](#)
- Scientist, [65](#)
- setVariableNameAndSubstitutions, [66](#)
- Time, [65](#)
- trailingZeros, [68](#)
- UnsignedInteger, [65](#)
- useDbDisplayLimits, [68](#)
- useDbPrecision, [68](#)
- User, [65](#)
- userLevelEnabled, [68](#)
- userLevelEngineerStyle, [68](#)
- UserLevels, [65](#)
- userLevelScientistStyle, [69](#)
- userLevelUserStyle, [69](#)
- userLevelVisibility, [69](#)
- variable, [69](#)
- variableAsToolTip, [69](#)
- variableSubstitutions, [69](#)
- visible, [70](#)
- QEBitStatus, [70](#)
 - allowDrop, [73](#)
 - dbValueChanged, [72](#)
 - enabled, [73](#)
 - Engineer, [72](#)
 - int, [73](#)
 - requestEnabled, [72](#)
 - Scientist, [72](#)
 - setVariableNameAndSubstitutions, [72](#)
 - User, [72](#)
 - userLevelEnabled, [73](#)
 - userLevelEngineerStyle, [73](#)
 - UserLevels, [72](#)
 - userLevelScientistStyle, [73](#)
 - userLevelUserStyle, [74](#)
 - userLevelVisibility, [74](#)
 - variable, [74](#)
 - variableAsToolTip, [74](#)
 - variableSubstitutions, [74](#)
 - visible, [74](#)
- QEByteArray, [75](#)
- QEComboBox, [76](#)
 - allowDrop, [78](#)
 - dbValueChanged, [78](#)
 - enabled, [78](#)
 - Engineer, [78](#)
 - int, [79](#)
 - requestEnabled, [78](#)
 - Scientist, [78](#)
 - subscribe, [79](#)
 - User, [78](#)
 - userLevelEnabled, [79](#)
 - userLevelEngineerStyle, [79](#)
 - UserLevels, [78](#)
 - userLevelScientistStyle, [79](#)
 - userLevelUserStyle, [80](#)
 - userLevelVisibility, [80](#)
 - variable, [80](#)
 - variableAsToolTip, [80](#)
 - variableSubstitutions, [80](#)
 - visible, [80](#)
 - writeOnChange, [78](#)
- QEConfiguredLayout, [81](#)
- QEConfiguredLayoutManager, [82](#)
- QEDragDrop, [83](#)
- QEFileBrowser, [84](#)
- QEFloating, [86](#)
- QEFloatingFormatting, [86](#)
- QEForm, [87](#)
 - setVariableNameAndSubstitutions, [88](#)
- QEFrame, [89](#)
 - allowDrop, [90](#)
 - enabled, [91](#)
 - Engineer, [90](#)
 - int, [91](#)
 - requestEnabled, [90](#)
 - Scientist, [90](#)
 - User, [90](#)
 - userLevelEnabled, [91](#)
 - userLevelEngineerStyle, [91](#)
 - UserLevels, [90](#)
 - userLevelScientistStyle, [91](#)
 - userLevelUserStyle, [91](#)
 - userLevelVisibility, [92](#)
 - variableAsToolTip, [92](#)
 - visible, [92](#)
- QEGenericButton, [92](#)

- QEGroupBox, 94
 - allowDrop, 96
 - enabled, 96
 - Engineer, 95
 - int, 96
 - requestEnabled, 96
 - Scientist, 95
 - User, 95
 - userLevelEnabled, 96
 - userLevelEngineerStyle, 96
 - UserLevels, 95
 - userLevelScientistStyle, 97
 - userLevelUserStyle, 97
 - userLevelVisibility, 97
 - variableAsToolTip, 97
 - visible, 97
- QEImage, 98
 - allowDrop, 103
 - beamXVariable, 103
 - beamYVariable, 103
 - clippingHighVariable, 103
 - clippingLowVariable, 103
 - clippingOnOffVariable, 103
 - dbValueChanged, 103
 - enabled, 104
 - Engineer, 102
 - heightVariable, 104
 - imageVariable, 104
 - int, 104
 - regionOfInterestHVariable, 104
 - regionOfInterestWVariable, 104
 - regionOfInterestXVariable, 104
 - regionOfInterestYVariable, 104
 - requestEnabled, 103
 - Scientist, 102
 - targetTriggerVariable, 105
 - targetXVariable, 105
 - targetYVariable, 105
 - User, 102
 - userLevelEnabled, 105
 - userLevelEngineerStyle, 105
 - UserLevels, 102
 - userLevelScientistStyle, 105
 - userLevelUserStyle, 106
 - userLevelVisibility, 106
 - variableAsToolTip, 106
 - variableSubstitutions, 106
 - visible, 106
 - widthVariable, 106
- QEInteger, 107
 - formatInteger, 108
 - formatIntegerArray, 108
 - formatValue, 108
- QEIntegerFormatting, 107
 - formatInteger, 108
 - formatIntegerArray, 108
 - formatValue, 108
- QELabel, 109
 - addUnits, 114
 - allowDrop, 114
 - Append, 113
 - arrayAction, 115
 - ArrayActions, 113
 - Ascii, 113
 - Automatic, 113
 - dbValueChanged, 114
 - Default, 113
 - enabled, 115
 - Engineer, 114
 - Fixed, 113
 - Floating, 113
 - format, 115
 - Formats, 113
 - Index, 113
 - int, 115
 - Integer, 113
 - leadingZero, 115
 - LocalEnumeration, 113
 - localEnumeration, 116
 - notation, 116
 - Notations, 113
 - pixmap0, 116
 - pixmap1, 116
 - pixmap2, 116
 - pixmap3, 116
 - pixmap4, 116
 - pixmap5, 116
 - pixmap6, 116
 - pixmap7, 117
 - precision, 117
 - QELabel, 114
 - requestEnabled, 114
 - Scientific, 113
 - Scientist, 114
 - Time, 113
 - trailingZeros, 117
 - UnsignedInteger, 113
 - UPDATE_PIXMAP, 113
 - UPDATE_TEXT, 113
 - updateOption, 117
 - updateOptions, 113
 - useDbPrecision, 117
 - User, 114

- userLevelEnabled, [117](#)
- userLevelEngineerStyle, [117](#)
- UserLevels, [113](#)
- userLevelScientistStyle, [118](#)
- userLevelUserStyle, [118](#)
- userLevelVisibility, [118](#)
- variable, [118](#)
- variableAsToolTip, [118](#)
- variableSubstitutions, [118](#)
- visible, [119](#)
- QELineEdit, [119](#)
 - addUnits, [124](#)
 - allowDrop, [124](#)
 - Append, [122](#)
 - arrayAction, [125](#)
 - ArrayActions, [122](#)
 - Ascii, [122](#)
 - Automatic, [122](#)
 - confirmWrite, [125](#)
 - dbValueChanged, [123](#)
 - Default, [122](#)
 - enabled, [125](#)
 - Engineer, [123](#)
 - Fixed, [122](#)
 - Floating, [122](#)
 - format, [125](#)
 - Formats, [122](#)
 - getConfirmWrite, [123](#)
 - getSubscribe, [123](#)
 - getWriteOnEnter, [123](#)
 - getWriteOnFinish, [123](#)
 - getWriteOnLoseFocus, [123](#)
 - Index, [122](#)
 - int, [125](#)
 - Integer, [122](#)
 - leadingZero, [126](#)
 - LocalEnumeration, [122](#)
 - localEnumeration, [126](#)
 - notation, [126](#)
 - Notations, [122](#)
 - precision, [126](#)
 - QELineEdit, [123](#)
 - requestEnabled, [124](#)
 - Scientific, [122](#)
 - Scientist, [123](#)
 - setConfirmWrite, [124](#)
 - setSubscribe, [124](#)
 - setWriteOnEnter, [124](#)
 - setWriteOnFinish, [124](#)
 - setWriteOnLoseFocus, [124](#)
 - subscribe, [126](#)
 - Time, [122](#)
 - trailingZeros, [126](#)
 - UnsignedInteger, [122](#)
 - useDbPrecision, [126](#)
 - User, [123](#)
 - userLevelEnabled, [126](#)
 - userLevelEngineerStyle, [127](#)
 - UserLevels, [122](#)
 - userLevelScientistStyle, [127](#)
 - userLevelUserStyle, [127](#)
 - userLevelVisibility, [127](#)
 - variable, [127](#)
 - variableAsToolTip, [128](#)
 - variableSubstitutions, [128](#)
 - visible, [128](#)
 - writeOnEnter, [128](#)
 - writeOnFinish, [128](#)
 - writeOnLoseFocus, [128](#)
- QELineEditManager, [129](#)
- QELink, [129](#)
- QELog, [131](#)
- QELogin, [133](#)
- QEPERiodic, [134](#)
 - allowDrop, [138](#)
 - dbElementChanged, [138](#)
 - dbValueChanged, [138](#)
 - enabled, [138](#)
 - Engineer, [138](#)
 - int, [139](#)
 - readbackLabelVariable1, [139](#)
 - readbackLabelVariable2, [139](#)
 - requestEnabled, [138](#)
 - Scientist, [138](#)
 - subscribe, [139](#)
 - User, [138](#)
 - userLevelEnabled, [139](#)
 - userLevelEngineerStyle, [139](#)
 - UserLevels, [138](#)
 - userLevelScientistStyle, [140](#)
 - userLevelUserStyle, [140](#)
 - userLevelVisibility, [140](#)
 - variableAsToolTip, [140](#)
 - variableSubstitutions, [140](#)
 - visible, [140](#)
 - writeButtonVariable1, [141](#)
 - writeButtonVariable2, [141](#)
- QEPERiodic::elementInfoStruct, [35](#)
- QEPERiodic::userInfoStructArray, [212](#)
- QEPERiodicComponentData, [141](#)

- QEPeiodicTaskMenu, [141](#)
- QEPeiodicTaskMenuFactory, [142](#)
- QEpicsPV, [142](#)
- QEPlot, [143](#)
 - allowDrop, [147](#)
 - dbValueChanged, [147](#)
 - enabled, [147](#)
 - Engineer, [147](#)
 - int, [147](#)
 - requestEnabled, [147](#)
 - Scientist, [147](#)
 - User, [147](#)
 - userLevelEnabled, [148](#)
 - userLevelEngineerStyle, [148](#)
 - UserLevels, [146](#)
 - userLevelScientistStyle, [148](#)
 - userLevelUserStyle, [148](#)
 - userLevelVisibility, [148](#)
 - variable1, [149](#)
 - variable2, [149](#)
 - variable3, [149](#)
 - variable4, [149](#)
 - variableAsToolTip, [149](#)
 - variableSubstitutions, [149](#)
 - visible, [149](#)
- QEPushButton, [150](#)
 - allowDrop, [153](#)
 - altReadbackVariable, [153](#)
 - dbValueChanged, [153](#)
 - enabled, [153](#)
 - Engineer, [153](#)
 - int, [154](#)
 - requestEnabled, [153](#)
 - Scientist, [153](#)
 - subscribe, [154](#)
 - User, [153](#)
 - userLevelEnabled, [154](#)
 - userLevelEngineerStyle, [154](#)
 - UserLevels, [153](#)
 - userLevelScientistStyle, [154](#)
 - userLevelUserStyle, [154](#)
 - userLevelVisibility, [155](#)
 - variable, [155](#)
 - variableAsToolTip, [155](#)
 - visible, [155](#)
- QEPvProperties, [155](#)
 - allowDrop, [158](#)
 - enabled, [158](#)
 - Engineer, [157](#)
 - int, [158](#)
 - requestEnabled, [158](#)
 - Scientist, [157](#)
 - User, [157](#)
 - userLevelEnabled, [158](#)
 - userLevelEngineerStyle, [158](#)
 - UserLevels, [157](#)
 - userLevelScientistStyle, [159](#)
 - userLevelUserStyle, [159](#)
 - userLevelVisibility, [159](#)
 - variable, [159](#)
 - variableAsToolTip, [159](#)
 - variableSubstitutions, [159](#)
 - visible, [160](#)
- QEPvPropertiesManager, [160](#)
- QERadioButton, [160](#)
 - allowDrop, [164](#)
 - dbValueChanged, [164](#)
 - enabled, [164](#)
 - Engineer, [163](#)
 - int, [164](#)
 - requestEnabled, [164](#)
 - Scientist, [163](#)
 - subscribe, [164](#)
 - User, [163](#)
 - userLevelEnabled, [164](#)
 - userLevelEngineerStyle, [165](#)
 - UserLevels, [163](#)
 - userLevelScientistStyle, [165](#)
 - userLevelUserStyle, [165](#)
 - userLevelVisibility, [165](#)
 - variable, [165](#)
 - variableAsToolTip, [166](#)
 - variableSubstitutions, [166](#)
 - visible, [166](#)
- QERecipe, [166](#)
- QEScript, [168](#)
- QEShape, [170](#)
 - allowDrop, [176](#)
 - animation1, [176](#)
 - animation2, [176](#)
 - animation3, [177](#)
 - animation4, [177](#)
 - animation5, [177](#)
 - animation6, [177](#)
 - animationOptions, [174](#)
 - color1, [177](#)
 - color10, [177](#)
 - color2, [177](#)
 - color3, [177](#)
 - color4, [177](#)

- color5, [178](#)
- color6, [178](#)
- color7, [178](#)
- color8, [178](#)
- color9, [178](#)
- dbValueChanged1, [175](#)
- dbValueChanged2, [175](#)
- dbValueChanged3, [175](#)
- dbValueChanged4, [176](#)
- dbValueChanged5, [176](#)
- dbValueChanged6, [176](#)
- enabled, [178](#)
- Engineer, [175](#)
- int, [178](#)
- offset1, [179](#)
- offset2, [179](#)
- offset3, [179](#)
- offset4, [179](#)
- offset5, [179](#)
- offset6, [179](#)
- point1, [179](#)
- point10, [179](#)
- point2, [179](#)
- point3, [180](#)
- point4, [180](#)
- point5, [180](#)
- point6, [180](#)
- point7, [180](#)
- point8, [180](#)
- point9, [180](#)
- QEShape, [175](#)
- requestEnabled, [176](#)
- scale2, [180](#)
- scale3, [180](#)
- scale4, [181](#)
- scale5, [181](#)
- scale6, [181](#)
- Scientist, [175](#)
- shapeOptions, [174](#)
- User, [175](#)
- userLevelEnabled, [181](#)
- userLevelEngineerStyle, [181](#)
- UserLevels, [175](#)
- userLevelScientistStyle, [181](#)
- userLevelUserStyle, [182](#)
- userLevelVisibility, [182](#)
- variable1, [182](#)
- variable2, [182](#)
- variable3, [182](#)
- variable4, [182](#)
- variable5, [182](#)
- variable6, [183](#)
- variableAsToolTip, [183](#)
- variableSubstitutions, [183](#)
- visible, [183](#)
- QESlider, [183](#)
 - allowDrop, [186](#)
 - dbValueChanged, [186](#)
 - enabled, [186](#)
 - Engineer, [185](#)
 - int, [186](#)
 - requestEnabled, [186](#)
 - Scientist, [185](#)
 - subscribe, [186](#)
 - User, [185](#)
 - userLevelEnabled, [187](#)
 - userLevelEngineerStyle, [187](#)
 - UserLevels, [185](#)
 - userLevelScientistStyle, [187](#)
 - userLevelUserStyle, [187](#)
 - userLevelVisibility, [187](#)
 - variable, [188](#)
 - variableAsToolTip, [188](#)
 - variableSubstitutions, [188](#)
 - visible, [188](#)
 - writeOnChange, [186](#)
- QESpinBox, [188](#)
 - allowDrop, [191](#)
 - dbValueChanged, [191](#)
 - enabled, [191](#)
 - Engineer, [191](#)
 - int, [191](#)
 - requestEnabled, [191](#)
 - Scientist, [191](#)
 - subscribe, [192](#)
 - User, [191](#)
 - userLevelEnabled, [192](#)
 - userLevelEngineerStyle, [192](#)
 - UserLevels, [191](#)
 - userLevelScientistStyle, [192](#)
 - userLevelUserStyle, [192](#)
 - userLevelVisibility, [193](#)
 - variable, [193](#)
 - variableAsToolTip, [193](#)
 - variableSubstitutions, [193](#)
 - visible, [193](#)
- QString, [194](#)
- QStringFormatting, [194](#)
 - APPEND, [195](#)
 - arrayActions, [195](#)

- ASCII, [195](#)
- FORMAT_DEFAULT, [196](#)
- FORMAT_FLOATING, [196](#)
- FORMAT_INTEGER, [196](#)
- FORMAT_LOCAL_ENUMERATE, [196](#)
- FORMAT_STRING, [196](#)
- FORMAT_TIME, [196](#)
- FORMAT_UNSIGNEDINTEGER, [196](#)
- formats, [196](#)
- INDEX, [196](#)
- NOTATION_AUTOMATIC, [196](#)
- NOTATION_FIXED, [196](#)
- NOTATION_SCIENTIFIC, [196](#)
- notations, [196](#)
- QStringFormattingMethods, [196](#)
- QEStripChart, [197](#)
- QEStripChart::PrivateData, [47](#)
- QEStripChartItem, [199](#)
- QEStripChartItem::PrivateData, [47](#)
- QEStripChartItemDialog, [200](#)
- QEStripChartTimeDialog, [200](#)
- QESubstitutedLabel, [200](#)
 - labelText, [201](#)
 - textSubstitutions, [201](#)
- QEToolTip, [201](#)
- QEWidget, [203](#)
 - activate, [205](#)
 - defaultFileLocation, [205](#)
 - getColor, [205](#)
 - getMessageSourceId, [205](#)
 - getQcalItem, [206](#)
 - openQEFile, [206](#)
 - readNow, [206](#)
 - setMessageSourceId, [206](#)
 - setupContextMenu, [206](#)
 - setVariableNameAndSubstitutions, [206](#)
 - writeNow, [206](#)
- QEWidgets, [207](#)
- readbackLabelVariable1
 - QEPeiodic, [139](#)
- readbackLabelVariable2
 - QEPeiodic, [139](#)
- readNow
 - QEWidget, [206](#)
- RecordSpec, [207](#)
- RecordSpecList, [208](#)
- regionOfInterestHVariable
 - QEImage, [104](#)
- regionOfInterestWVariable
 - QEImage, [104](#)
- regionOfInterestXVariable
 - QEImage, [104](#)
- regionOfInterestYVariable
 - QEImage, [104](#)
- requestEnabled
 - QEAnalogProgressBar, [66](#)
 - QEBitStatus, [72](#)
 - QEComboBox, [78](#)
 - QEFrame, [90](#)
 - QEGroupBox, [96](#)
 - QEImage, [103](#)
 - QELabel, [114](#)
 - QELineEdit, [124](#)
 - QEPeriodic, [138](#)
 - QEPlot, [147](#)
 - QEPushButton, [153](#)
 - QEPvProperties, [158](#)
 - QERadioButton, [164](#)
 - QEShape, [176](#)
 - QESlider, [186](#)
 - QESpinBox, [191](#)
- Right_To_Left
 - QEAnalogIndicator, [59](#)
- Scale
 - QEAnalogIndicator, [59](#)
- scale2
 - QEShape, [180](#)
- scale3
 - QEShape, [180](#)
- scale4
 - QEShape, [181](#)
- scale5
 - QEShape, [181](#)
- scale6
 - QEShape, [181](#)
- Scientific
 - QEAnalogProgressBar, [65](#)
 - QELabel, [113](#)
 - QELineEdit, [122](#)
- Scientist
 - QEAnalogProgressBar, [65](#)
 - QEBitStatus, [72](#)
 - QEComboBox, [78](#)
 - QEFrame, [90](#)
 - QEGroupBox, [95](#)
 - QEImage, [102](#)
 - QELabel, [114](#)
 - QELineEdit, [123](#)

- QEPeiodic, [138](#)
- QEPlot, [147](#)
- QEPushButton, [153](#)
- QEPvProperties, [157](#)
- QERadioButton, [163](#)
- QEShape, [175](#)
- QESlider, [185](#)
- QESpinBox, [191](#)
- selectMenu, [208](#)
- setConfirmWrite
 - QELineEdit, [124](#)
- setMessageSourceId
 - QEWidget, [206](#)
- setSubscribe
 - QELineEdit, [124](#)
- setupContextMenu
 - QEWidget, [206](#)
- setVariableNameAndSubstitutions
 - QEAnalogProgressBar, [66](#)
 - QEBitStatus, [72](#)
 - QEFrm, [88](#)
 - QEWidget, [206](#)
- setWriteOnEnter
 - QELineEdit, [124](#)
- setWriteOnFinish
 - QELineEdit, [124](#)
- setWriteOnLoseFocus
 - QELineEdit, [124](#)
- shapeOptions
 - QEShape, [174](#)
- showScale
 - QEAnalogIndicator, [61](#)
- showText
 - QEAnalogIndicator, [61](#)
- spanAngle
 - QEAnalogIndicator, [61](#)
- standardProperties, [208](#)
- StateMachineTemplate, [210](#)
- subscribe
 - QEComboBox, [79](#)
 - QELineEdit, [126](#)
 - QEPeiodic, [139](#)
 - QEPushButton, [154](#)
 - QERadioButton, [164](#)
 - QESlider, [186](#)
 - QESpinBox, [192](#)
- targetTriggerVariable
 - QEImage, [105](#)
- targetXVariable
 - QEImage, [105](#)
- targetYVariable
 - QEImage, [105](#)
- textSubstitutions
 - QESubstitutedLabel, [201](#)
- Time
 - QEAnalogProgressBar, [65](#)
 - QELabel, [113](#)
 - QELineEdit, [122](#)
- Top_To_Bottom
 - QEAnalogIndicator, [59](#)
- trace, [211](#)
- TrackRange, [211](#)
- trailingZeros
 - QEAnalogProgressBar, [68](#)
 - QELabel, [117](#)
 - QELineEdit, [126](#)
- UnsignedInteger
 - QEAnalogProgressBar, [65](#)
 - QELabel, [113](#)
 - QELineEdit, [122](#)
- UPDATE_PIXMAP
 - QELabel, [113](#)
- UPDATE_TEXT
 - QELabel, [113](#)
- updateOption
 - QELabel, [117](#)
- updateOptions
 - QELabel, [113](#)
- useDbDisplayLimits
 - QEAnalogProgressBar, [68](#)
- useDbPrecision
 - QEAnalogProgressBar, [68](#)
 - QELabel, [117](#)
 - QELineEdit, [126](#)
- User
 - QEAnalogProgressBar, [65](#)
 - QEBitStatus, [72](#)
 - QEComboBox, [78](#)
 - QEFrame, [90](#)
 - QEGroupBox, [95](#)
 - QEImage, [102](#)
 - QELabel, [114](#)
 - QELineEdit, [123](#)
 - QEPeiodic, [138](#)
 - QEPlot, [147](#)
 - QEPushButton, [153](#)
 - QEPvProperties, [157](#)
 - QERadioButton, [163](#)

- QEShape, 175
- QESlider, 185
- QESpinBox, 191
- userInfoStruct, 212
- userLevelEnabled
 - QEAnalogProgressBar, 68
 - QEBitStatus, 73
 - QEComboBox, 79
 - QEFrmae, 91
 - QEGroupBox, 96
 - QEImage, 105
 - QELabel, 117
 - QELineEdit, 126
 - QEPeiodic, 139
 - QEPlot, 148
 - QEPushButton, 154
 - QEPvProperties, 158
 - QERadioButton, 164
 - QEShape, 181
 - QESlider, 187
 - QESpinBox, 192
- userLevelEngineerStyle
 - QEAnalogProgressBar, 68
 - QEBitStatus, 73
 - QEComboBox, 79
 - QEFrmae, 91
 - QEGroupBox, 96
 - QEImage, 105
 - QELabel, 117
 - QELineEdit, 127
 - QEPeiodic, 139
 - QEPlot, 148
 - QEPushButton, 154
 - QEPvProperties, 158
 - QERadioButton, 165
 - QEShape, 181
 - QESlider, 187
 - QESpinBox, 192
- UserLevels
 - QEAnalogProgressBar, 65
 - QEBitStatus, 72
 - QEComboBox, 78
 - QEFrmae, 90
 - QEGroupBox, 95
 - QEImage, 102
 - QELabel, 113
 - QELineEdit, 122
 - QEPeiodic, 138
 - QEPlot, 146
 - QEPushButton, 153
- QEPvProperties, 157
- QERadioButton, 163
- QEShape, 175
- QESlider, 185
- QESpinBox, 191
- userLevelScientistStyle
 - QEAnalogProgressBar, 69
 - QEBitStatus, 73
 - QEComboBox, 79
 - QEFrmae, 91
 - QEGroupBox, 97
 - QEImage, 105
 - QELabel, 118
 - QELineEdit, 127
 - QEPeiodic, 140
 - QEPlot, 148
 - QEPushButton, 154
 - QEPvProperties, 159
 - QERadioButton, 165
 - QEShape, 181
 - QESlider, 187
 - QESpinBox, 192
- userLevelSignal, 212
- userLevelSlot, 213
- userLevelUserStyle
 - QEAnalogProgressBar, 69
 - QEBitStatus, 74
 - QEComboBox, 80
 - QEFrmae, 91
 - QEGroupBox, 97
 - QEImage, 106
 - QELabel, 118
 - QELineEdit, 127
 - QEPeiodic, 140
 - QEPlot, 148
 - QEPushButton, 154
 - QEPvProperties, 159
 - QERadioButton, 165
 - QEShape, 182
 - QESlider, 187
 - QESpinBox, 192
- userLevelVisibility
 - QEAnalogProgressBar, 69
 - QEBitStatus, 74
 - QEComboBox, 80
 - QEFrmae, 92
 - QEGroupBox, 97
 - QEImage, 106
 - QELabel, 118
 - QELineEdit, 127

- QEPeiodic, [140](#)
- QEPlot, [148](#)
- QEPushButton, [155](#)
- QEPvProperties, [159](#)
- QERadioButton, [165](#)
- QEShape, [182](#)
- QESlider, [187](#)
- QESpinBox, [193](#)
- UserMessage, [213](#)
- UserMessageSignal, [216](#)
- UserMessageSlot, [217](#)
- value
 - QEAnalogIndicator, [61](#)
- variable
 - QEAnalogProgressBar, [69](#)
 - QEBitStatus, [74](#)
 - QECombobox, [80](#)
 - QELabel, [118](#)
 - QELineEdit, [127](#)
 - QEPushButton, [155](#)
 - QEPvProperties, [159](#)
 - QERadioButton, [165](#)
 - QESlider, [188](#)
 - QESpinBox, [193](#)
- variable1
 - QEPlot, [149](#)
 - QEShape, [182](#)
- variable2
 - QEPlot, [149](#)
 - QEShape, [182](#)
- variable3
 - QEPlot, [149](#)
 - QEShape, [182](#)
- variable4
 - QEPlot, [149](#)
 - QEShape, [182](#)
- variable5
 - QEShape, [182](#)
- variable6
 - QEShape, [183](#)
- variableAsToolTip
 - QEAnalogProgressBar, [69](#)
 - QEBitStatus, [74](#)
 - QECombobox, [80](#)
 - QEFrame, [92](#)
 - QEGroupBox, [97](#)
 - QEImage, [106](#)
 - QELabel, [118](#)
 - QELineEdit, [128](#)
- QEPeiodic, [140](#)
- QEPlot, [149](#)
- QEPushButton, [155](#)
- QEPvProperties, [159](#)
- QERadioButton, [166](#)
- QEShape, [183](#)
- QESlider, [188](#)
- QESpinBox, [193](#)
- variableSubstitutions
 - QEAnalogProgressBar, [69](#)
 - QEBitStatus, [74](#)
 - QECombobox, [80](#)
 - QEImage, [106](#)
 - QELabel, [118](#)
 - QELineEdit, [128](#)
 - QEPeiodic, [140](#)
 - QEPlot, [149](#)
 - QEPvProperties, [159](#)
 - QERadioButton, [166](#)
 - QEShape, [183](#)
 - QESlider, [188](#)
 - QESpinBox, [193](#)
- VideoWidget, [218](#)
- visible
 - QEAnalogProgressBar, [70](#)
 - QEBitStatus, [74](#)
 - QECombobox, [80](#)
 - QEFrame, [92](#)
 - QEGroupBox, [97](#)
 - QEImage, [106](#)
 - QELabel, [119](#)
 - QELineEdit, [128](#)
 - QEPeiodic, [140](#)
 - QEPlot, [149](#)
 - QEPushButton, [155](#)
 - QEPvProperties, [160](#)
 - QERadioButton, [166](#)
 - QEShape, [183](#)
 - QESlider, [188](#)
 - QESpinBox, [193](#)
- WidgetRef, [219](#)
- widthVariable
 - QEImage, [106](#)
- writeButtonVariable1
 - QEPeiodic, [141](#)
- writeButtonVariable2
 - QEPeiodic, [141](#)
- writeNow
 - QEWidget, [206](#)

writeOnChange
 QComboBox, [78](#)
 QSlider, [186](#)
writeOnEnter
 QLineEdit, [128](#)
writeOnFinish
 QLineEdit, [128](#)
writeOnLoseFocus
 QLineEdit, [128](#)

zoomMenu, [219](#)